Torrent SDK Documentation

Release 1

Ion Torrent

January 30, 2015

Contents

The Torrent Suite™ Software Development Kit (SDK) provides a convenient and powerful web services Application Programming Interface (API) for accessing the Torrent Server database and working with analysis results files and plugins. The API implements a well-defined interface based on Representational State Transfer (REST) principles for remote access to resources with programming language independence.

CHAPTER 1

Examples

Below are some Python examples to get you started with the Torrent SDK.

1.1 Torrent Suite™ Software SDK Code Samples

Contents

1.1.1 Get experiment by name (Python)

```
import requests

experiment_name = "b006f48b-27fc-4e93-8a26-cef5bf71b8b0"

ts_api_request = requests.get("http://localhost:10500/rundb/api/v1/experiment/", params={"format": "
ts_api_response = ts_api_request.json()

print ts_api_response
```

1.1.2 Write an experiment note (Python)

```
import json
import requests

metaData = json.dumps({ "notes":"updated note" })

putResp = requests.put('http://myhost/rundb/api/v1/experiment/<pk>/'%base_url, \
data=metaData, headers={'content-type':'application/json'}, \
```

```
auth=('myusername', 'mypassword'))

print resp.status_code
```

1.1.3 Get analysis result metadata and metrics (Python)

```
import requests
import simplejson as json
import sys

if len(sys.argv) == 2:
        [runName] = sys.argv[1:2]
else:
        print '\n\tUsage:   getruninfo.py <runName>'
        print '\n\tExample: getruninfo.py Auto_user_f4--134-br_21'
        sys.exit(1)

resp = requests.get('http://myhost/rundb/api/v1/results?format=json&resultsName=%s'%runName, \
        auth=('myusername', 'mypassword'))
resp_json = resp.json()

try:
        runData = resp_json[u'objects'][0]
        experLoc = runData[u'experiment']
except (KeyError, IndexError):
        print 'ERROR: Invalid name given.'
        sys.exit(1)

expResult = requests.get('http://%s%s'%(myhost, experLoc))
expData = expResult.json()

try:
        print '\nProject:\t\t%s'%expData[u'log'][u'project']
        print 'Experiment Name:\t%s'%expData[u'expName']
        print 'PGM Name:\t\t%s'%expData[u'pgmName']
        print 'Library:\t\t%s'%expData[u'log'][u'library']
        print 'Notes:\t\t\t%s'%expData[u'notes']
except KeyError:
        print 'ERROR: Invalid key in expData.'

try:
        print 'Results:\t\t%s'%runData['resultsName']
        print 'Timestamp:\t\t%s'%runData['timeStamp']
except KeyError:
        print 'ERROR: Invalid key in runData.'

ametricsLoc = runData[u'analysismetrics'][0]
aResult = requests.get('http://%s%s'%(myhost,ametricsLoc))
aData = aResult.json()

print '\n\nAnalysis Metrics:\n==================\n'
for propType, propVal in aData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t= %s'%(propType, propVal)

qmetricsLoc = runData[u'qualitymetrics'][0]
```

```
qResult = requests.get('http://%s%s'%(myhost,qmetricsLoc))
qData = qResult.json()

print '\n\nQuality Metrics:\n==================\n'
for propType, propVal in qData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t=%s'%(propType, propVal)
```

1.1.4 Add a PGM™ or Proton™ Sequencer (Python)

```
import json
import requests

resp = requests.get('http://myhost/rundb/api/v1/rig/<existing_rig>?format=json', \
        auth=('myusername', 'mypassword'))
resp_json = resp.json()

resp_json.update(name='<new_rig_name>')
resp_json.pop('resource_uri')
resp_json['location'].pop('resource_uri')

pdata = json.dumps(resp_json)

status = requests.put('http://myhost/rundb/api/v1/rig/<new_rig_name>/', \
        data=pdata, headers={'content-type':'application/json'}, auth=('myusername', 'mypassword))
```

The same code can be used to update a sequencer with the following changes; replace the 'name' field with whatever needs updating, and direct the put request to the original rig.

1.1.5 Get the status for a PGM™ or Proton™ Sequencer (Python)

```
import requests

ts_api_request = requests.get("http://localhost:10500/rundb/api/v1/rig/", params={"format": "json"})
ts_api_response = ts_api_request.json()

rigs = ts_api_response["objects"]

print "Found %i sequencer(s):" % len(rigs)

for rig in rigs:
        print
        print "Name: %s" % rig["name"]
        print "Status: %s" % rig["state"]
```

1.1.6 Download a FASTQ file (Python)

```
import json
import requests

resp = requests.get('http://myhost/rundb/api/v1/results/<pk>?format=json', \
        auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

```
resp = requests.get('http://myhost/%s'%resp_json['fastqlink'], \
        auth=('myusername', 'mypassword'))

print resp_json['fastqlink']
print resp.content #(The FASTQ file contents.)
```

1.1.7 List file servers (Python)

```
import httplib2
import json

h = httplib2.Http(".cache")
h.add_credentials('myusername', 'mypassword')

resp, content = h.request("http://myhost/rundb/api/v1/fileserver?format=json", "GET")
contentdict = json.loads(content)

objects = contentdict['objects']
for obj in objects:
        print obj['filesPrefix']
```

See also:

See the API Cookbook for information on how to access the API programatically: *Torrent Suite™ Software API Cookbook*.

CHAPTER 2

REST API

The API specifies endpoints with callable methods for each resource and managed entity, such as a plugin or file. To perform an action using the API, you send a request to the endpoint, using a REST method and specifying parameters, data and the data format. The parameters, requests, responses, and error codes for each method are listed in the API reference tables.

See the API reference tables for a listing of all API resources:

2.1 Torrent Server REST API v1 Resources

2.1.1 Activeionchefprepkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/", params
ts_api_response = ts_api_request.json()

activeionchefprepkitinfos = ts_api_response["objects"]

for activeionchefprepkitinfo in activeionchefprepkitinfos:
    print activeionchefprepkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeionchefprepkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "IonChefPrepKit",
            "description": "ION PGM IC 200 KIT",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "100020579",
                    "id": 20085,
                    "resource_uri": "/rundb/api/v1/kitpart/20085/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "100020580",
                    "id": 20086,
                    "resource_uri": "/rundb/api/v1/kitpart/20086/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "01",
                    "id": 20123,
                    "resource_uri": "/rundb/api/v1/kitpart/20123/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "ICPREP0001",
            "resource_uri": "/rundb/api/v1/activeionchefprepkitinfo/20042/",
            "id": 20042,
            "categories": "",
            "name": "ION PGM IC 200 KIT"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.2 Activelibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/", params={"
```

```
ts_api_response = ts_api_request.json()

activelibrarykitinfos = ts_api_response["objects"]

for activelibrarykitinfo in activelibrarykitinfos:
    print activelibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 14,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activelibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activelibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.3 Activepgmlibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/", params=
ts_api_response = ts_api_request.json()

activepgmlibrarykitinfos = ts_api_response["objects"]
```

```
for activepgmlibrarykitinfo in activepgmlibrarykitinfos:
    print activepgmlibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 11,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activepgmlibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activepgmlibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.4 Activepgmsequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/", para
ts_api_response = ts_api_request.json()

activepgmsequencingkitinfos = ts_api_response["objects"]

for activepgmsequencingkitinfo in activepgmsequencingkitinfos:
    print activepgmsequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activepgmsequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion PGM 200 Sequencing Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4474004",
                    "id": 20009,
                    "resource_uri": "/rundb/api/v1/kitpart/20009/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474005",
                    "id": 20010,
                    "resource_uri": "/rundb/api/v1/kitpart/20010/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474006",
                    "id": 20011,
                    "resource_uri": "/rundb/api/v1/kitpart/20011/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474007",
                    "id": 20012,
                    "resource_uri": "/rundb/api/v1/kitpart/20012/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0003",
            "resource_uri": "/rundb/api/v1/activepgmsequencingkitinfo/20003/",
            "id": 20003,
            "categories": "",
            "name": "IonPGM200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.5 Activeprotonlibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/", para
ts_api_response = ts_api_request.json()

activeprotonlibrarykitinfos = ts_api_response["objects"]

for activeprotonlibrarykitinfo in activeprotonlibrarykitinfos:
    print activeprotonlibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 13,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeprotonlibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activeprotonlibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

2.1.6 Activeprotonsequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/?format=json&li`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/", p
ts_api_response = ts_api_request.json()

activeprotonsequencingkitinfos = ts_api_response["objects"]

for activeprotonsequencingkitinfo in activeprotonsequencingkitinfos:
    print activeprotonsequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeprotonsequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "Ion PI Sequencing 200 Kit v2",
            "nucleotideType": "",
            "instrumentType": "proton",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4482282",
                    "id": 20078,
                    "resource_uri": "/rundb/api/v1/kitpart/20078/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4482284",
                    "id": 20079,
                    "resource_uri": "/rundb/api/v1/kitpart/20079/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4485149",
                    "id": 20094,
                    "resource_uri": "/rundb/api/v1/kitpart/20094/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4485521",
                    "id": 20095,
                    "resource_uri": "/rundb/api/v1/kitpart/20095/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4484082",
                    "id": 20096,
```

```
                    "resource_uri": "/rundb/api/v1/kitpart/20096/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0012",
            "resource_uri": "/rundb/api/v1/activeprotonsequencingkitinfo/20044/",
            "id": 20044,
            "categories": "",
            "name": "ProtonI200Kit-v2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.7 Activesequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/", params=
ts_api_response = ts_api_request.json()

activesequencingkitinfos = ts_api_response["objects"]

for activesequencingkitinfo in activesequencingkitinfos:
    print activesequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activesequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion PGM 200 Sequencing Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4474004",
                    "id": 20009,
                    "resource_uri": "/rundb/api/v1/kitpart/20009/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474005",
                    "id": 20010,
                    "resource_uri": "/rundb/api/v1/kitpart/20010/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474006",
                    "id": 20011,
                    "resource_uri": "/rundb/api/v1/kitpart/20011/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474007",
                    "id": 20012,
                    "resource_uri": "/rundb/api/v1/kitpart/20012/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0003",
            "resource_uri": "/rundb/api/v1/activesequencingkitinfo/20003/",
            "id": 20003,
            "categories": "",
            "name": "IonPGM200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.8 Analysisargs Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/analysisargs/`

Schema URL: `http://mytorrentserver/rundb/api/v1/analysisargs/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
chipType	Unicode string data. Ex: "Hello World"		false	false	false	false	string
thumbnailalignmentargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailanalysisargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
samplePrepKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
sequenceKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
analysisargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailcalibrateargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
chip_default	Boolean data. Ex: True	false	false	false	true	false	boolean
beadfindargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
templateKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
prebasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
prethumbnailbasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
alignmentargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailbasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
active	Boolean data. Ex: True	true	false	false	true	false	boolean
thumbnailbeadfindargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
calibrateargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
libraryKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
basecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/analysisargs/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/analysisargs/", params={"format":
ts_api_response = ts_api_request.json()

analysisargss = ts_api_response["objects"]

for analysisargs in analysisargss:
    print analysisargs
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 27,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/analysisargs/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "chipType": "314v2",
            "thumbnailalignmentargs": "",
            "thumbnailanalysisargs": "",
            "samplePrepKitName": "",
            "id": 5,
            "sequenceKitName": "",
            "analysisargs": "Analysis --from-beadfind --use-alternative-etbR-equation",
            "thumbnailcalibrateargs": "",
            "chip_default": true,
            "beadfindargs": "justBeadFind",
            "templateKitName": "",
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 20 --cal
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "",
            "active": true,
            "thumbnailbeadfindargs": "",
            "calibrateargs": "calibrate --skipDroop",
            "libraryKitName": "",
            "name": "default_314v2",
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 20",
            "resource_uri": "/rundb/api/v1/analysisargs/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post

- put
- delete
- patch

2.1.9 Analysismetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/schema/`

Perform read-only operations on `analysismetrics` resources and data elements.

Fields table

field	help text	default	nullable	re
libLive	Integer data. Ex: 2673	n/a	false	fal
ignored	Integer data. Ex: 2673	n/a	false	fal
washout_ambiguous	Integer data. Ex: 2673	n/a	false	fal
tfLive	Integer data. Ex: 2673	n/a	false	fal
sysIE	Floating point numeric data. Ex: 26.73	n/a	false	fal
bead	Integer data. Ex: 2673	n/a	false	fal
tfKp	Integer data. Ex: 2673	n/a	false	fal
washout_live	Integer data. Ex: 2673	n/a	false	fal
id	Integer data. Ex: 2673		false	fal
libFinal	Integer data. Ex: 2673	n/a	false	fal
loading	Floating point numeric data. Ex: 26.73	0.0	false	fal
lib	Integer data. Ex: 2673	n/a	false	fal
keypass_all_beads	Integer data. Ex: 2673	n/a	false	fal
dud	Integer data. Ex: 2673	n/a	false	fal
sysCF	Floating point numeric data. Ex: 26.73	n/a	false	fal
pinned	Integer data. Ex: 2673	n/a	false	fal
live	Integer data. Ex: 2673	n/a	false	fal
excluded	Integer data. Ex: 2673	n/a	false	fal
tf	Integer data. Ex: 2673	n/a	false	fal
empty	Integer data. Ex: 2673	n/a	false	fal
tfFinal	Integer data. Ex: 2673	n/a	false	fal
amb	Integer data. Ex: 2673	n/a	false	fal
lib_pass_basecaller	Integer data. Ex: 2673	n/a	false	fal
lib_pass_cafie	Integer data. Ex: 2673	n/a	false	fal
washout_dud	Integer data. Ex: 2673	n/a	false	fal
libMix	Integer data. Ex: 2673	n/a	false	fal
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	fal
libKp	Integer data. Ex: 2673	n/a	false	fal
adjusted_addressable	Integer data. Ex: 2673	0	false	fal
sysDR	Floating point numeric data. Ex: 26.73	n/a	false	fal
total	Integer data. Ex: 2673	0	false	fal
washout_test_fragment	Integer data. Ex: 2673	n/a	false	fal
washout_library	Integer data. Ex: 2673	n/a	false	fal
washout	Integer data. Ex: 2673	n/a	false	fal

Table 2.1 – continued from previous page

field	help text	default	nullable	rea
tfMix	Integer data. Ex: 2673	n/a	false	fal
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	tru

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/analysismetrics/", params={"format
ts_api_response = ts_api_request.json()

analysismetricss = ts_api_response["objects"]

for analysismetrics in analysismetricss:
    print analysismetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 38886,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/analysismetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "libLive": 0,
            "ignored": 219,
            "washout_ambiguous": 0,
            "tfLive": 0,
            "sysIE": 0.00782,
            "bead": 7461,
            "tfKp": 0,
            "washout_live": 0,
            "id": 1,
            "libFinal": 0,
            "loading": 0.0,
            "lib": 7197,
            "keypass_all_beads": 0,
            "dud": 208,
            "sysCF": 0.00673,
            "pinned": 21,
            "live": 7253,
            "excluded": 0,
            "tf": 56,
            "empty": 2299,
            "tfFinal": 0,
```

```
            "amb": 0,
            "lib_pass_basecaller": 0,
            "lib_pass_cafie": 0,
            "washout_dud": 0,
            "libMix": 0,
            "report": "/rundb/api/v1/results/3/",
            "libKp": 0,
            "adjusted_addressable": 0,
            "sysDR": 0.00274,
            "total": 0,
            "washout_test_fragment": 0,
            "washout_library": 0,
            "washout": 0,
            "tfMix": 0,
            "resource_uri": "/rundb/api/v1/analysismetrics/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.10 Applicationgroup Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/`

Schema URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
applications	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
uid	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/applicationgroup/", params={"forma
ts_api_response = ts_api_request.json()

applicationgroups = ts_api_response["objects"]

for applicationgroup in applicationgroups:
    print applicationgroup
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/applicationgroup/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "DNA",
            "description": "DNA",
            "applications": [
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/",
                        "/rundb/api/v1/applicationgroup/3/",
                        "/rundb/api/v1/applicationgroup/4/"
                    ],
                    "description": "Generic Sequencing",
                    "nucleotideType": "",
                    "barcode": "",
                    "meta": {},
                    "runType": "GENS",
                    "id": 1,
                    "alternate_name": "Other",
                    "resource_uri": "/rundb/api/v1/runtype/1/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "AmpliSeq DNA",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "AMPS",
```

```
                    "id": 2,
                    "alternate_name": "AmpliSeq DNA",
                    "resource_uri": "/rundb/api/v1/runtype/2/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "TargetSeq",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "TARS",
                    "id": 3,
                    "alternate_name": "TargetSeq",
                    "resource_uri": "/rundb/api/v1/runtype/3/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/",
                        "/rundb/api/v1/applicationgroup/4/"
                    ],
                    "description": "Whole Genome",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "WGNM",
                    "id": 4,
                    "alternate_name": "Whole Genome",
                    "resource_uri": "/rundb/api/v1/runtype/4/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "AmpliSeq Exome",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "AMPS_EXOME",
                    "id": 7,
                    "alternate_name": "AmpliSeq Exome",
                    "resource_uri": "/rundb/api/v1/runtype/7/"
                }
            ],
            "uid": "APPLGROUP_0001",
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/applicationgroup/1/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

2.1.11 Applproduct Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/applproduct/`

Schema URL: `http://mytorrentserver/rundb/api/v1/applproduct/schema/`

Fields table

field	help text	defau
isDualNucleotideTypeBySampleSupported	Boolean data. Ex: True	false
defaultHotSpotRegionBedFileName	Unicode string data. Ex: “Hello World”	n/a
isTargetRegionBEDFileSupported	Boolean data. Ex: True	true
isSamplePrepKitSupported	Boolean data. Ex: True	true
defaultPESeqKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultPELibKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultSeqKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultBarcodeKitName	Unicode string data. Ex: “Hello World”	n/a
id	Integer data. Ex: 2673	
productCode	Unicode string data. Ex: “Hello World”	any
isControlSeqTypeBySampleSupported	Boolean data. Ex: True	false
defaultChipType	Unicode string data. Ex: “Hello World”	n/a
isPairedEndSupported	Boolean data. Ex: True	true
appl	A single related resource. Can be either a URI or set of nested resource data.	n/a
instrumentType	Unicode string data. Ex: “Hello World”	
isDefault	Boolean data. Ex: True	false
isTargetTechniqueSelectionSupported	Boolean data. Ex: True	true
description	Unicode string data. Ex: “Hello World”	
isHotspotRegionBEDFileSuppported	Boolean data. Ex: True	true
productName	Unicode string data. Ex: “Hello World”	n/a
isBarcodeKitSelectionRequired	Boolean data. Ex: True	false
isDefaultBarcoded	Boolean data. Ex: True	false
defaultTargetRegionBedFileName	Unicode string data. Ex: “Hello World”	n/a
isActive	Boolean data. Ex: True	true
isReferenceBySampleSupported	Boolean data. Ex: True	false
defaultFlowCount	Integer data. Ex: 2673	0
defaultLibKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
barcodeKitSelectableType	Unicode string data. Ex: “Hello World”	
defaultGenomeRefName	Unicode string data. Ex: “Hello World”	n/a
isVisible	Boolean data. Ex: True	false
isDefaultPairedEnd	Boolean data. Ex: True	false
resource_uri	Unicode string data. Ex: “Hello World”	n/a

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/applproduct/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/applproduct/", params={"format": "
ts_api_response = ts_api_request.json()

applproducts = ts_api_response["objects"]

for applproduct in applproducts:
    print applproduct
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 12,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/applproduct/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isDualNucleotideTypeBySampleSupported": false,
            "defaultHotSpotRegionBedFileName": "",
            "isTargetRegionBEDFileSupported": true,
            "isSamplePrepKitSupported": true,
            "defaultPESeqKit": null,
            "defaultPELibKit": null,
            "defaultSeqKit": {
                "isActive": true,
                "kitType": "SequencingKit",
                "description": "Ion PGM Sequencing 200 Kit v2",
                "nucleotideType": "",
                "instrumentType": "pgm",
                "runMode": "",
                "parts": [
                    {
                        "barcode": "4482006",
                        "id": 20054,
                        "resource_uri": "/rundb/api/v1/kitpart/20054/",
                        "kit": "/rundb/api/v1/kitinfo/20033/"
                    },
                    {
                        "barcode": "4482007",
                        "id": 20055,
                        "resource_uri": "/rundb/api/v1/kitpart/20055/",
                        "kit": "/rundb/api/v1/kitinfo/20033/"
                    },
                    {
```

```
                "barcode": "4482008",
                "id": 20056,
                "resource_uri": "/rundb/api/v1/kitpart/20056/",
                "kit": "/rundb/api/v1/kitinfo/20033/"
            },
            {
                "barcode": "4482009",
                "id": 20057,
                "resource_uri": "/rundb/api/v1/kitpart/20057/",
                "kit": "/rundb/api/v1/kitinfo/20033/"
            }
        ],
        "flowCount": 500,
        "applicationType": "",
        "uid": "SEQ0009",
        "resource_uri": "/rundb/api/v1/kitinfo/20033/",
        "id": 20033,
        "categories": "",
        "name": "IonPGM200Kit-v2"
    },
    "defaultBarcodeKitName": null,
    "id": 20001,
    "productCode": "AMPS_0",
    "isControlSeqTypeBySampleSupported": false,
    "defaultChipType": null,
    "isPairedEndSupported": false,
    "appl": {
        "applicationGroups": [
            "/rundb/api/v1/applicationgroup/1/"
        ],
        "description": "AmpliSeq DNA",
        "nucleotideType": "dna",
        "barcode": "",
        "meta": {},
        "runType": "AMPS",
        "id": 2,
        "alternate_name": "AmpliSeq DNA",
        "resource_uri": "/rundb/api/v1/runtype/2/"
    },
    "instrumentType": "pgm",
    "isDefault": true,
    "isTargetTechniqueSelectionSupported": true,
    "description": "",
    "isHotspotRegionBEDFileSuppported": true,
    "productName": "AMPS_default",
    "isBarcodeKitSelectionRequired": false,
    "isDefaultBarcoded": false,
    "defaultTargetRegionBedFileName": "",
    "isActive": true,
    "isReferenceBySampleSupported": false,
    "defaultFlowCount": 500,
    "defaultLibKit": {
        "isActive": true,
        "kitType": "LibraryKit",
        "description": "Ion AmpliSeq 2.0 Library Kit",
        "nucleotideType": "dna",
        "instrumentType": "",
        "runMode": "",
```

```
                "parts": [
                    {
                        "barcode": "4475345",
                        "id": 20034,
                        "resource_uri": "/rundb/api/v1/kitpart/20034/",
                        "kit": "/rundb/api/v1/kitinfo/20012/"
                    }
                ],
                "flowCount": 0,
                "applicationType": "",
                "uid": "LIB0008",
                "resource_uri": "/rundb/api/v1/kitinfo/20012/",
                "id": 20012,
                "categories": "",
                "name": "Ion AmpliSeq 2.0 Library Kit"
            },
            "barcodeKitSelectableType": "all",
            "defaultGenomeRefName": "hg19",
            "isVisible": true,
            "isDefaultPairedEnd": false,
            "resource_uri": "/rundb/api/v1/applproduct/20001/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.12 Availableionchefplannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”

Table 2.3 – continued from previous page

field	help text
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]

Table 2.3 – continued from previous page

field	help text
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/?format=jso`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/
ts_api_response = ts_api_request.json()

availableionchefplannedexperiments = ts_api_response["objects"]

for availableionchefplannedexperiment in availableionchefplannedexperiments:
    print availableionchefplannedexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 10,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableionchefplannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "C18S2_package1_Thur_psp4",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [
                "Chef_p1v2_dev_IE"
            ],
            "notes": "148420",
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.1.17",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": "00049613",
            "seqKitBarcode": null,
            "barcodeId": "IonXpress",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/23817/",
            "bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
            "isReusable": false,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
            "librarykitname": "Ion AmpliSeq 2.0 Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
```

```
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "pending",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266238,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266238/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266237,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266237/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266236,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266236/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=21
"thumbnailcalibrateargs": "calibrate --skipDroop",
```

```
"templatingKitName": "Ion PI IC 200 Kit",
"runType": "AMPS_EXOME",
"username": "ionuser",
"planName": "C18S2_package1_Thur_psp4",
"sampleDisplayedName": "",
"prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
"controlSequencekitname": "",
"chefMessage": "",
"childPlans": [],
"pairedEndLibraryAdapterName": "",
"runMode": "single",
"irworkflow": "",
"planExecuted": false,
"project": "Chef_p1v2_dev_IE",
"usePostBeadfind": false,
"runname": null,
"planGUID": "994fd167-7d5a-43e5-acca-c8ffa069350e",
"planShortID": "0KXAK",
"sampleSetGroupType": null,
"sample": "",
"planExecutedDate": null,
"reverse_primer": null,
"id": 102054,
"barcodedSamples": {
    "1": {
        "barcodeSampleInfo": {
            "IonXpress_001": {
                "description": "",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                "hotSpotRegionBedFile": "",
                "nucleotideType": "DNA",
                "controlSequenceType": "",
                "externalId": ""
            }
        },
        "barcodes": [
            "IonXpress_001"
        ]
    }
},
"regionfile": "",
"selectedPlugins": {
    "coverageAnalysis": {
        "userInput": "",
        "version": "4.2-r88700",
        "features": [],
        "name": "coverageAnalysis",
        "id": 827
    },
    "validateVariantCaller": {
        "userInput": {
            "variant_caller_name": "variantCaller",
            "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
            "region": "NIST",
            "sample": "NA12878",
            "truth_minor_snp": "None",
            "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
```

```
                "truth_minor_indel": "None"
            },
            "version": "0.2.1",
            "features": [],
            "name": "validateVariantCaller",
            "id": 732
        },
        "variantCaller": {
            "userInput": {
                "torrent_variant_caller": {
                    "snp_min_allele_freq": "0.10000000000000001",
                    "snp_strand_bias": "0.97999999999999998",
                    "hotspot_min_coverage": 6,
                    "hotspot_min_cov_each_strand": 3,
                    "hotspot_min_allele_freq": "0.10000000000000001",
                    "snp_min_variant_score": 15,
                    "hotspot_strand_bias": "0.94999999999999996",
                    "hp_max_length": 8,
                    "filter_insertion_predictions": "0.20000000000000001",
                    "indel_min_variant_score": 20,
                    "indel_min_coverage": 10,
                    "heavy_tailed": 3,
                    "outlier_probability": "0.01",
                    "data_quality_stringency": 5,
                    "snp_min_cov_each_strand": 0,
                    "hotspot_min_variant_score": 10,
                    "indel_strand_bias": "0.90000000000000002",
                    "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                    "downsample_to_coverage": 400,
                    "filter_unusual_predictions": "0.25",
                    "indel_min_allele_freq": "0.14999999999999999",
                    "do_snp_realignment": 1,
                    "prediction_precision": 1,
                    "indel_min_cov_each_strand": 5,
                    "filter_deletion_predictions": "0.20000000000000001",
                    "suppress_recalibration": 0,
                    "snp_min_coverage": 5
                },
                "meta": {
                    "repository_id": "",
                    "ts_version": "4.0",
                    "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.201309
                    "user_selections": {
                        "chip": "proton_p1",
                        "frequency": "germline",
                        "library": "ampliseq",
                        "panel": "/rundb/api/v1/contentupload/53/"
                    },
                    "librarytype": "ampliseq",
                    "trimreads": true,
                    "tooltip": "Retrieved from external file",
                    "tvcargs": "tvc",
                    "built_in": false,
                    "configuration": "",
                    "compatibility": {}
                },
                "long_indel_assembler": {
                    "min_indel_size": 4,
```

```
                            "short_suffix_match": 5,
                            "output_mnv": 0,
                            "min_var_count": 5,
                            "min_var_freq": "0.14999999999999999",
                            "kmer_len": 19,
                            "max_hp_length": 8,
                            "relative_strand_bias": "0.80000000000000004"
                        },
                        "freebayes": {
                            "gen_min_coverage": 5,
                            "allow_mnps": 1,
                            "allow_complex": 0,
                            "read_max_mismatch_fraction": 1,
                            "read_mismatch_limit": 10,
                            "allow_indels": 1,
                            "min_mapping_qv": 4,
                            "gen_min_alt_allele_freq": "0.10000000000000001",
                            "allow_snps": 1,
                            "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                        }
                    },
                    "version": "4.2-r88446",
                    "features": [],
                    "name": "variantCaller",
                    "id": 826
                },
                "AmpliconStats": {
                    "userInput": "",
                    "version": "0.4.5",
                    "features": [],
                    "name": "AmpliconStats",
                    "id": 774
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-06-26T04:47:43.000023+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/availableionchefplannedexperiment/102054/",
            "reverse3primeadapter": ""
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.13 Availableionchefplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/sche`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa

Table 2.4 – continued from previous page

field	help text	default	nullable	readonly	b
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/?for`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiments
ts_api_response = ts_api_request.json()

availableionchefplannedexperimentsummarys = ts_api_response["objects"]

for availableionchefplannedexperimentsummary in availableionchefplannedexperimentsummarys:
    print availableionchefplannedexperimentsummary
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 10,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableionchefplannedexperimentsummary/?offset=1&limit=1&format=jso
    },
    "objects": [
        {
            "isReverseRun": false,
```

```
            "planDisplayedName": "C18S2_package1_Thur_psp4",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "0KXAK",
            "planStatus": "pending",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "00049613",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102054,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI IC 200 Kit",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-26T04:47:43.000023+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "C18S2_package1_Thur_psp4",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "994fd167-7d5a-43e5-acca-c8ffa069350e",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableionchefplannedexperimentsummary/102054/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

2.1.14 Availableonetouchplannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True

Table 2.5 – continued from previous page

field	help text
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”

Table 2.5 – continued from previous page

field	help text
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/?format=js`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment
ts_api_response = ts_api_request.json()

availableonetouchplannedexperiments = ts_api_response["objects"]

for availableonetouchplannedexperiment in availableonetouchplannedexperiments:
    print availableonetouchplannedexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 119,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableonetouchplannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [
                "auto_chip"
```

```
],
"notes": "",
"sequencekitname": "ProtonI200Kit-v3",
"storageHost": null,
"expName": "",
"cycles": null,
"isReverseRun": false,
"storage_options": "A",
"thumbnailalignmentargs": "stage1 map4",
"chipType": "P1.1.17",
"chefProgress": 0.0,
"library": "hg19",
"reverselibrarykey": "",
"sampleTubeLabel": "",
"seqKitBarcode": null,
"barcodeId": "IonXpress",
"chefLogPath": null,
"isPlanGroup": false,
"realign": false,
"sampleGroupingName": "",
"experiment": "/rundb/api/v1/experiment/23946/",
"bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
"isReusable": false,
"isDuplicateReads": false,
"thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
"librarykitname": "Ion AmpliSeq 2.0 Library Kit",
"adapter": null,
"basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
"tfKey": "ATCG",
"parentPlan": null,
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "planned",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
        "id": 266490,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266490/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
```

```
            "id": 266489,
            "qcType": {
                "description": "",
                "minThreshold": 1,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Key Signal (1-100)",
                "id": 2,
                "resource_uri": "/rundb/api/v1/qctype/2/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/266489/"
        },
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
            "id": 266488,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/266488/"
        }
    ],
    "analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=216
    "thumbnailcalibrateargs": "calibrate --skipDroop",
    "templatingKitName": "Ion PI Template OT2 200 Kit v3",
    "runType": "AMPS_EXOME",
    "username": "ionuser",
    "planName": "-R65726-pat25_treatment_dbsa",
    "sampleDisplayedName": "",
    "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
    "controlSequencekitname": "",
    "chefMessage": "",
    "childPlans": [],
    "pairedEndLibraryAdapterName": "",
    "runMode": "single",
    "irworkflow": "",
    "planExecuted": false,
    "project": "auto_chip",
    "usePostBeadfind": false,
    "runname": null,
    "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
    "planShortID": "076B8",
    "sampleSetGroupType": null,
    "sample": "",
    "planExecutedDate": null,
    "reverse_primer": null,
    "id": 102183,
    "barcodedSamples": {
        "148541": {
            "barcodeSampleInfo": {
                "IonXpress_002": {
                    "description": "",
```

```
                    "reference": "hg19",
                    "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                    "hotSpotRegionBedFile": "",
                    "nucleotideType": "dna",
                    "controlSequenceType": "",
                    "externalId": ""
                },
                "IonXpress_001": {
                    "description": "",
                    "reference": "hg19",
                    "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                    "hotSpotRegionBedFile": "",
                    "nucleotideType": "dna",
                    "controlSequenceType": "",
                    "externalId": ""
                }
            },
            "barcodes": [
                "IonXpress_002",
                "IonXpress_001"
            ]
        }
    },
    "regionfile": "",
    "selectedPlugins": {
        "coverageAnalysis": {
            "userInput": "",
            "version": "4.3-r88797",
            "features": [],
            "name": "coverageAnalysis",
            "id": 828
        },
        "validateVariantCaller": {
            "userInput": {
                "variant_caller_name": "variantCaller",
                "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
                "region": "NIST",
                "sample": "NA12878",
                "truth_minor_snp": "None",
                "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
                "truth_minor_indel": "None"
            },
            "version": "0.2.1",
            "features": [],
            "name": "validateVariantCaller",
            "id": 732
        },
        "variantCaller": {
            "userInput": {
                "torrent_variant_caller": {
                    "snp_min_allele_freq": "0.10000000000000001",
                    "snp_strand_bias": "0.97999999999999998",
                    "hotspot_min_coverage": 6,
                    "hotspot_min_cov_each_strand": 3,
                    "hotspot_min_allele_freq": "0.10000000000000001",
                    "snp_min_variant_score": 15,
                    "hotspot_strand_bias": "0.94999999999999996",
                    "hp_max_length": 8,
```

```
            "filter_insertion_predictions": "0.20000000000000001",
            "indel_min_variant_score": 20,
            "indel_min_coverage": 10,
            "heavy_tailed": 3,
            "outlier_probability": "0.01",
            "data_quality_stringency": 5,
            "snp_min_cov_each_strand": 0,
            "hotspot_min_variant_score": 10,
            "indel_strand_bias": "0.90000000000000002",
            "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
            "downsample_to_coverage": 400,
            "filter_unusual_predictions": "0.25",
            "indel_min_allele_freq": "0.14999999999999999",
            "do_snp_realignment": 1,
            "prediction_precision": 1,
            "indel_min_cov_each_strand": 5,
            "filter_deletion_predictions": "0.20000000000000001",
            "suppress_recalibration": 0,
            "snp_min_coverage": 5
        },
        "meta": {
            "repository_id": "",
            "ts_version": "4.0",
            "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.201309
            "user_selections": {
                "chip": "proton_p1",
                "frequency": "germline",
                "library": "ampliseq",
                "panel": "/rundb/api/v1/contentupload/53/"
            },
            "librarytype": "ampliseq",
            "trimreads": true,
            "tooltip": "Retrieved from external file",
            "tvcargs": "tvc",
            "built_in": false,
            "configuration": "",
            "compatibility": {}
        },
        "long_indel_assembler": {
            "min_indel_size": 4,
            "short_suffix_match": 5,
            "output_mnv": 0,
            "min_var_count": 5,
            "min_var_freq": "0.14999999999999999",
            "kmer_len": 19,
            "max_hp_length": 8,
            "relative_strand_bias": "0.80000000000000004"
        },
        "freebayes": {
            "gen_min_coverage": 5,
            "allow_mnps": 1,
            "allow_complex": 0,
            "read_max_mismatch_fraction": 1,
            "read_mismatch_limit": 10,
            "allow_indels": 1,
            "min_mapping_qv": 4,
            "gen_min_alt_allele_freq": "0.10000000000000001",
            "allow_snps": 1,
```

```
                            "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                        }
                    },
                    "version": "4.2-r88446",
                    "features": [],
                    "name": "variantCaller",
                    "id": 826
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/availableonetouchplannedexperiment/102183/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.15 Availableonetouchplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/sch`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr

Table 2.6 – continued from previous page

field	help text	default	nullable	readonly	b
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa

Table 2.6 – continued from previous page

field	help text	default	nullable	readonly	b
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/?fo`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment
ts_api_response = ts_api_request.json()

availableonetouchplannedexperimentsummarys = ts_api_response["objects"]

for availableonetouchplannedexperimentsummary in availableonetouchplannedexperimentsummarys:
    print availableonetouchplannedexperimentsummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 119,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableonetouchplannedexperimentsummary/?offset=1&limit=1&format=jso
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "076B8",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102183,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
```

```
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "-R65726-pat25_treatment_dbsa",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableonetouchplannedexperimentsummary/102183/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.16 Availableplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr

Table 2.7 – continued from previous page

field	help text	default	nullable	readonly	b
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa

Table 2.7 – continued from previous page

field	help text	default	nullable	readonly	b
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/?format=jso`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/
ts_api_response = ts_api_request.json()

availableplannedexperimentsummarys = ts_api_response["objects"]

for availableplannedexperimentsummary in availableplannedexperimentsummarys:
    print availableplannedexperimentsummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 399,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableplannedexperimentsummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "076B8",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102183,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
```

```
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "-R65726-pat25_treatment_dbsa",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableplannedexperimentsummary/102183/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.17 Chip Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/chip/`

Schema URL: `http://mytorrentserver/rundb/api/v1/chip/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”		false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
slots	Integer data. Ex: 2673	n/a	false	false	false	false	integer
instrumentType	Unicode string data. Ex: “Hello World”		false	false	true	false	string
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/chip/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/chip/", params={"format": "json",
ts_api_response = ts_api_request.json()

chips = ts_api_response["objects"]

for chip in chips:
    print chip
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/chip/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "slots": 1,
            "calibrateargs": "calibrate",
            "prebasecallerargs": "BaseCaller",
            "description": "PIv2",
            "prethumbnailbasecallerargs": "BaseCaller",
            "alignmentargs": "",
            "thumbnailbasecallerargs": "BaseCaller",
            "analysisargs": "Analysis",
            "basecallerargs": "BaseCaller",
            "thumbnailbeadfindargs": "justBeadFind",
            "thumbnailalignmentargs": "",
            "thumbnailanalysisargs": "Analysis",
            "instrumentType": "proton",
            "beadfindargs": "justBeadFind",
```

```
            "resource_uri": "/rundb/api/v1/chip/16/",
            "id": 16,
            "isActive": false,
            "name": "900AMPS_EXOME"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.18 Compositedatamanagement Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
base-call_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
in_process	Boolean data. Ex: True	false	false	false	false	false	boolean
misc_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
base-call_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
misc_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
output_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
expName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
output_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
sigproc_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
sigproc_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
diskusage	Integer data. Ex: 2673	n/a	true	false	false	false	integer
expDir	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositedatamanagement/", params=
ts_api_response = ts_api_request.json()

compositedatamanagements = ts_api_response["objects"]

for compositedatamanagement in compositedatamanagements:
    print compositedatamanagement
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43343,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositedatamanagement/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "misc_diskspace": 0.0,
            "expName": "",
            "basecall_state": "Deleting",
            "in_process": true,
            "misc_state": "Deleted",
            "timeStamp": "2011-09-08T18:19:32.000098+00:00",
            "basecall_keep": null,
            "misc_keep": null,
            "output_keep": null,
            "basecall_diskspace": 0.0,
            "resultsName": "Auto__1",
            "output_state": "Deleted",
            "sigproc_state": "Deleted",
            "sigproc_keep": null,
            "sigproc_diskspace": null,
            "diskusage": 0,
            "resource_uri": "/rundb/api/v1/compositedatamanagement/1/",
            "expDir": "/results1/BBDefault/R_2011_08_25_16_44_20_user_BBD-43",
            "id": 1,
            "output_diskspace": 0.0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.19 Compositeexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
ftpStatus	Unicode string data. Ex: "Hello World"		false	false	true	false	string
storage_options	Unicode string data. Ex: "Hello World"	A	false	false	false	false	string
star	Boolean data. Ex: True	false	false	false	true	false	boolean
chipType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
notes	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
resultDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	true	false	false	false	datetime
flows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
repResult	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
expName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
pgmName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	false	false	false	false	datetime
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
plan	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositeexperiment/", params={"fo
ts_api_response = ts_api_request.json()

compositeexperiments = ts_api_response["objects"]

for compositeexperiment in compositeexperiments:
    print compositeexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20366,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositeexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "chipInstrumentType": "proton",
            "chipType": "P1.1.17",
            "results": [
                {
                    "status": "Completed",
                    "processedflows": 0,
                    "analysis_metrics": {
                        "ignored": 1364892,
                        "lib": 118063544,
                        "total_wells": 164699136,
                        "pinned": 463867,
                        "live": 118074386,
                        "excluded": 16095180,
                        "bead": 118525043,
                        "resource_uri": "",
                        "id": 41726,
                        "empty": 28250154,
                        "libFinal": 85604279
                    },
                    "timeStamp": "2014-06-28T07:11:58.000789+00:00",
                    "analysismetrics": {
                        "ignored": 1364892,
                        "lib": 118063544,
                        "total_wells": 164699136,
                        "pinned": 463867,
                        "live": 118074386,
                        "excluded": 16095180,
                        "bead": 118525043,
                        "resource_uri": "",
                        "id": 41726,
                        "empty": 28250154,
                        "libFinal": 85604279
                    },
                    "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
                    "id": 304393,
                    "reportStatus": "Nothing",
                    "quality_metrics": {
                        "q0_mean_read_length": 178.362197969099,
                        "q0_reads": 85604279,
                        "q0_bases": "15268567358",
                        "q20_reads": 85604279,
                        "q20_bases": "13060288783",
                        "q20_mean_read_length": 178,
                        "id": 39683,
                        "resource_uri": ""
                    },
```

```
        "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
        "projects": [
            {
                "resource_uri": "",
                "id": 1385,
                "name": "auto_chip",
                "modified": "2014-06-27T21:21:43.000081+00:00"
            }
        ],
        "qualitymetrics": {
            "q0_mean_read_length": 178.362197969099,
            "q0_reads": 85604279,
            "q0_bases": "15268567358",
            "q20_reads": 85604279,
            "q20_bases": "13060288783",
            "q20_mean_read_length": 178,
            "id": 39683,
            "resource_uri": ""
        },
        "eas": {
            "resource_uri": "",
            "reference": "hg19",
            "barcodeKitName": "IonXpress"
        },
        "resource_uri": "/rundb/api/v1/compositeresult/304393/",
        "libmetrics": {
            "i100Q20_reads": 61016174,
            "aveKeyCounts": 89.0,
            "id": 41874,
            "resource_uri": "",
            "q20_mean_alignment_length": 159
        },
        "autoExempt": false,
        "representative": false
    },
    {
        "status": "Completed",
        "processedflows": 520,
        "analysis_metrics": {
            "ignored": 9939,
            "lib": 726685,
            "total_wells": 960000,
            "pinned": 44080,
            "live": 726723,
            "excluded": 0,
            "bead": 738396,
            "resource_uri": "",
            "id": 41695,
            "empty": 167585,
            "libFinal": 517179
        },
        "timeStamp": "2014-06-28T00:18:42.000351+00:00",
        "analysismetrics": {
            "ignored": 9939,
            "lib": 726685,
            "total_wells": 960000,
            "pinned": 44080,
            "live": 726723,
```

```
                "excluded": 0,
                "bead": 738396,
                "resource_uri": "",
                "id": 41695,
                "empty": 167585,
                "libFinal": 517179
            },
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_tn_304394/"
            "id": 304394,
            "reportStatus": "Nothing",
            "quality_metrics": {
                "q0_mean_read_length": 176.041268110267,
                "q0_reads": 517179,
                "q0_bases": "91044847",
                "q20_reads": 517179,
                "q20_bases": "77321419",
                "q20_mean_read_length": 176,
                "id": 39658,
                "resource_uri": ""
            },
            "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958_tn",
            "projects": [
                {
                    "resource_uri": "",
                    "id": 1385,
                    "name": "auto_chip",
                    "modified": "2014-06-27T21:21:43.000081+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 176.041268110267,
                "q0_reads": 517179,
                "q0_bases": "91044847",
                "q20_reads": 517179,
                "q20_bases": "77321419",
                "q20_mean_read_length": 176,
                "id": 39658,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/compositeresult/304394/",
            "libmetrics": {
                "i100Q20_reads": 358815,
                "aveKeyCounts": 88.0,
                "id": 41849,
                "resource_uri": "",
                "q20_mean_alignment_length": 155
            },
            "autoExempt": false,
            "representative": false
        }
    ],
    "library": "hg19",
    "sample": "148541",
```

```
            "runMode": "single",
            "storage_options": "D",
            "repResult": "/rundb/api/v1/compositeresult/304394/",
            "id": 23958,
            "archived": false,
            "barcodeId": "IonXpress",
            "sampleSetName": "",
            "star": false,
            "resultDate": "2014-06-28T07:11:58.000789+00:00",
            "flows": 520,
            "plan": {
                "runType": "AMPS_EXOME",
                "id": 102195,
                "resource_uri": ""
            },
            "date": "2014-06-27T21:19:01+00:00",
            "ftpStatus": "Complete",
            "notes": "",
            "chipDescription": "PI",
            "pgmName": "Z28",
            "keep": false,
            "expName": "R_2014_06_27_17_13_22_user_Z28-428--r65714-pou4_dbsa",
            "resource_uri": "/rundb/api/v1/compositeexperiment/23958/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.20 Compositeresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositeresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositeresult/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
analysismetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
qualitymetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
libmetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
representative	Boolean data. Ex: True	false	false	false	true	false	boolean

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositeresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositeresult/", params={"format
ts_api_response = ts_api_request.json()

compositeresults = ts_api_response["objects"]

for compositeresult in compositeresults:
    print compositeresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositeresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Completed",
            "processedflows": 0,
            "analysis_metrics": {
                "ignored": 3003404,
                "lib": 134282829,
                "total_wells": 164699136,
                "pinned": 472926,
                "live": 135574135,
                "excluded": 16095180,
                "bead": 135800957,
                "resource_uri": "",
                "id": 31763,
                "empty": 9326669,
                "libFinal": 91521301
            },
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "analysismetrics": {
                "ignored": 3003404,
                "lib": 134282829,
                "total_wells": 164699136,
                "pinned": 472926,
                "live": 135574135,
                "excluded": 16095180,
                "bead": 135800957,
                "resource_uri": "",
                "id": 31763,
                "empty": 9326669,
                "libFinal": 91521301
            },
            "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
            "id": 293943,
            "reportStatus": "Nothing",
            "quality_metrics": {
                "q0_mean_read_length": 168.0,
                "q0_reads": 91521301,
                "q0_bases": "15380233572",
                "q20_reads": 91521301,
                "q20_bases": "12209924742",
                "q20_mean_read_length": 103,
                "id": 31678,
                "resource_uri": ""
            },
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "projects": [
                {
                    "resource_uri": "",
```

```
                    "id": 1080,
                    "name": "chef_827_909_20min_ext",
                    "modified": "2014-01-22T18:50:10.000920+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 168.0,
                "q0_reads": 91521301,
                "q0_bases": "15380233572",
                "q20_reads": 91521301,
                "q20_bases": "12209924742",
                "q20_mean_read_length": 103,
                "id": 31678,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/compositeresult/293943/",
            "libmetrics": {
                "i100Q20_reads": 56284561,
                "aveKeyCounts": 71.0,
                "id": 32368,
                "resource_uri": "",
                "q20_mean_alignment_length": 142
            },
            "autoExempt": false,
            "representative": false
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.21 Content Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/content/`

Schema URL: `http://mytorrentserver/rundb/api/v1/content/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
publisher	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
contentupload	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
meta	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
file	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
path	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/content/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/content/", params={"format": "jso
ts_api_response = ts_api_request.json()

contents = ts_api_response["objects"]

for content in contents:
    print content
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 72,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/content/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "publisher": "/rundb/api/v1/publisher/BED/",
            "contentupload": "/rundb/api/v1/contentupload/16/",
            "meta": {
                "is_ampliseq": false,
                "hotspot_bed": null,
                "reference": "hg19",
```

```
                "primary_bed": "/results/uploads/BED/16/testPanel30.bed",
                "hotspot": false
            },
            "file": "/results/uploads/BED/16/hg19/unmerged/plain/testPanel30.bed",
            "path": "/hg19/unmerged/plain/testPanel30.bed",
            "id": 53,
            "resource_uri": "/rundb/api/v1/content/53/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.22 Contentupload Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/contentupload/`

Schema URL: `http://mytorrentserver/rundb/api/v1/contentupload/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
meta	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	string
file_path	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/contentupload/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/contentupload/", params={"format"
ts_api_response = ts_api_request.json()

contentuploads = ts_api_response["objects"]

for contentupload in contentuploads:
    print contentupload
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 25,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/contentupload/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Successfully Completed",
            "meta": {
                "upload_date": "2014-03-27T00:28:46",
                "description": "Comp Cancer Panel",
                "reference": "hg19",
                "is_ampliseq": true,
                "hotspot": true,
                "choice": "proton",
                "design": {
                    "status": "ORDERABLE",
                    "pipeline": "DNA",
                    "min_number_amplicons_per_pool": 3991,
                    "type": "FIXED_PANEL",
                    "description": "<p>The Ion AmpliSeq&trade; Comprehensive Cancer Panel provides hi
                    "order_number": 90,
                    "design_name": "Comp Cancer Panel",
                    "results_uri": "/ws/tmpldesign/14011153/download/results",
                    "pipeline_version": null,
                    "request_id_and_solution_ordering_id": "CCP",
                    "configuration_choices": [
                        "pgm",
                        "proton"
                    ],
                    "target_size": 1293547,
                    "genome": "HG19",
                    "solution_name": null,
                    "created_date": "2013-10-07T14:21:51.388+0000",
                    "plan": {
                        "missed_bed": null,
                        "hotspot_bed": "CCP.20131001.hotspots.bed",
                        "coverage_summary": null,
                        "designed_bed": "CCP.20131001.designed.bed",
                        "target_mutations": null,
                        "primer_bed": null,
                        "selectedPlugins": {
                            "variantCaller": {
                                "features": [],
                                "ampliSeqVariantCallerConfig": {
                                    "torrent_variant_caller": {
                                        "snp_min_allele_freq": "0.02",
                                        "snp_strand_bias": "0.95",
                                        "hotspot_min_coverage": "6",
                                        "hotspot_min_cov_each_strand": "2",
                                        "hotspot_min_allele_freq": "0.01",
                                        "snp_min_variant_score": "6",
                                        "hotspot_strand_bias": "0.95",
```

```
            "hp_max_length": "8",
            "filter_insertion_predictions": "0.2",
            "indel_min_variant_score": "6",
            "indel_min_coverage": "15",
            "heavy_tailed": "3",
            "outlier_probability": "0.005",
            "data_quality_stringency": "6.5",
            "snp_min_cov_each_strand": "0",
            "hotspot_min_variant_score": "6",
            "indel_strand_bias": "0.9",
            "downsample_to_coverage": "2000",
            "filter_unusual_predictions": "0.3",
            "indel_min_allele_freq": "0.05",
            "do_snp_realignment": "1",
            "prediction_precision": "1.0",
            "indel_min_cov_each_strand": "2",
            "filter_deletion_predictions": "0.2",
            "suppress_recalibration": "0",
            "snp_min_coverage": "6"
        },
        "meta": {
            "repository_id": "",
            "ts_version": "4.0",
            "name": "Panel-optimized - Comp Cancer Panel",
            "user_selections": {
                "chip": "proton_p1",
                "frequency": "germline",
                "library": "ampliseq",
                "panel": "/rundb/api/v1/contentupload/48/"
            },
            "trimreads": true,
            "tooltip": "Panel-optimized parameters from AmpliSeq.com",
            "tvcargs": "tvc",
            "built_in": true,
            "configuration": "",
            "compatibility": {
                "panel": "/rundb/api/v1/contentupload/48/"
            }
        },
        "long_indel_assembler": {
            "min_indel_size": "4",
            "short_suffix_match": "5",
            "output_mnv": "0",
            "min_var_count": "5",
            "min_var_freq": "0.15",
            "kmer_len": "19",
            "max_hp_length": "8",
            "relative_strand_bias": "0.8"
        },
        "freebayes": {
            "gen_min_coverage": "6",
            "allow_mnps": "1",
            "allow_complex": "0",
            "read_max_mismatch_fraction": "1.0",
            "read_mismatch_limit": "10",
            "allow_indels": "1",
            "min_mapping_qv": "4",
            "gen_min_alt_allele_freq": "0.035",
```

```
            "allow_snps": "1",
            "gen_min_indel_alt_allele_freq": "0.1"
        }
    },
    "userInput": {
        "torrent_variant_caller": {
            "snp_min_allele_freq": "0.02",
            "snp_strand_bias": "0.95",
            "hotspot_min_coverage": "6",
            "hotspot_min_cov_each_strand": "2",
            "hotspot_min_allele_freq": "0.01",
            "snp_min_variant_score": "6",
            "hotspot_strand_bias": "0.95",
            "hp_max_length": "8",
            "filter_insertion_predictions": "0.2",
            "indel_min_variant_score": "6",
            "indel_min_coverage": "15",
            "heavy_tailed": "3",
            "outlier_probability": "0.005",
            "data_quality_stringency": "6.5",
            "snp_min_cov_each_strand": "0",
            "hotspot_min_variant_score": "6",
            "indel_strand_bias": "0.9",
            "downsample_to_coverage": "2000",
            "filter_unusual_predictions": "0.3",
            "indel_min_allele_freq": "0.05",
            "do_snp_realignment": "1",
            "prediction_precision": "1.0",
            "indel_min_cov_each_strand": "2",
            "filter_deletion_predictions": "0.2",
            "suppress_recalibration": "0",
            "snp_min_coverage": "6"
        },
        "meta": {
            "repository_id": "",
            "ts_version": "4.0",
            "name": "Panel-optimized - Comp Cancer Panel",
            "user_selections": {
                "chip": "proton_p1",
                "frequency": "germline",
                "library": "ampliseq",
                "panel": "/rundb/api/v1/contentupload/48/"
            },
            "trimreads": true,
            "tooltip": "Panel-optimized parameters from AmpliSeq.com",
            "tvcargs": "tvc",
            "built_in": true,
            "configuration": "",
            "compatibility": {
                "panel": "/rundb/api/v1/contentupload/48/"
            }
        },
        "long_indel_assembler": {
            "min_indel_size": "4",
            "short_suffix_match": "5",
            "output_mnv": "0",
            "min_var_count": "5",
            "min_var_freq": "0.15",
```

```
                                        "kmer_len": "19",
                                        "max_hp_length": "8",
                                        "relative_strand_bias": "0.8"
                                    },
                                    "freebayes": {
                                        "gen_min_coverage": "6",
                                        "allow_mnps": "1",
                                        "allow_complex": "0",
                                        "read_max_mismatch_fraction": "1.0",
                                        "read_mismatch_limit": "10",
                                        "allow_indels": "1",
                                        "min_mapping_qv": "4",
                                        "gen_min_alt_allele_freq": "0.035",
                                        "allow_snps": "1",
                                        "gen_min_indel_alt_allele_freq": "0.1"
                                    }
                                },
                                "version": "4.1-r74477",
                                "id": 698,
                                "name": "variantCaller"
                            }
                        },
                        "coverage_detail": null,
                        "primer_sequences": "CCP.20131001.primerDataSheet.csv",
                        "runType": "AMPS",
                        "submitted_bed": null,
                        "well_plate_data": null
                    },
                    "design_id": "CCP",
                    "number_of_amplicons": 15992,
                    "id": 14011153,
                    "amplicons_coverage_summary": "95.349763093262169",
                    "number_of_amplicon_pools": 4
                }
            },
            "file_path": "/results/uploads/BED/48/CCP.20131001.results.zip",
            "resource_uri": "/rundb/api/v1/contentupload/48/",
            "id": 48
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.23 Datamanagementhistory Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/`

Schema URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
username	Unicode string data. Ex: "Hello World"	ION	false	false	true	false	string
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
text	Unicode string data. Ex: "Hello World"		false	false	false	false	string
object_pk	Integer data. Ex: 2673	n/a	false	false	false	false	integer
resultsName	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/datamanagementhistory/", params={'
ts_api_response = ts_api_request.json()

datamanagementhistorys = ts_api_response["objects"]

for datamanagementhistory in datamanagementhistorys:
    print datamanagementhistory
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 474229,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/datamanagementhistory/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "username": "ION",
            "created": "2013-03-05T15:15:09.000925+00:00",
            "text": "Created DMFileStat (Signal Processing Input)",
            "object_pk": 274692,
```

```
            "resultsName": null,
            "id": 114023,
            "resource_uri": "/rundb/api/v1/datamanagementhistory/114023/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.24 Dnabarcode Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/`

Schema URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/schema/`

Perform CRUD operations on DNABARCODE resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
index	Integer data. Ex: 2673	n/a	false	false	false	false	integer
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
score_cutoff	Floating point numeric data. Ex: 26.73	0	false	false	false	false	float
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
floworder	Unicode string data. Ex: “Hello World”		false	false	true	false	string
adapter	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
length	Integer data. Ex: 2673	0	false	false	true	false	integer
id_str	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
score_mode	Integer data. Ex: 2673	0	false	false	true	false	integer
type	Unicode string data. Ex: “Hello World”		false	false	true	false	string
annotation	Unicode string data. Ex: “Hello World”		false	false	true	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/dnabarcode/", params={"format": "
ts_api_response = ts_api_request.json()

dnabarcodes = ts_api_response["objects"]

for dnabarcode in dnabarcodes:
    print dnabarcode
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9387,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/dnabarcode/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "index": 9,
            "name": "IonXpress",
            "score_cutoff": 2.0,
            "sequence": "TGAGCGGAAC",
            "floworder": "",
            "adapter": "GAT",
            "id": 761,
            "length": 10,
            "id_str": "IonXpress_009",
            "score_mode": 1,
            "type": "",
            "annotation": "",
            "resource_uri": "/rundb/api/v1/dnabarcode/761/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.25 Emailaddress Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/emailaddress/`

Schema URL: `http://mytorrentserver/rundb/api/v1/emailaddress/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
selected	Boolean data. Ex: True		false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
email	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/emailaddress/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/emailaddress/", params={"format":
ts_api_response = ts_api_request.json()

emailaddresss = ts_api_response["objects"]

for emailaddress in emailaddresss:
    print emailaddress
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/emailaddress/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "selected": true,
            "resource_uri": "/rundb/api/v1/emailaddress/2/",
            "email": "bernard.puc@lifetech.com",
            "id": 2
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.26 Eventlog Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/eventlog/`

Schema URL: `http://mytorrentserver/rundb/api/v1/eventlog/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
username	Unicode string data. Ex: "Hello World"	ION	false	false	true	false	string
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	date-time
text	Unicode string data. Ex: "Hello World"		false	false	false	false	string
object_pk	Integer data. Ex: 2673	n/a	false	false	false	false	integer
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/eventlog/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/eventlog/", params={"format": "jso
ts_api_response = ts_api_request.json()

eventlogs = ts_api_response["objects"]

for eventlog in eventlogs:
    print eventlog
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 625153,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/eventlog/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
```

```
            "username": "ION",
            "created": "2012-07-03T15:14:33.000511+00:00",
            "text": "Created during migration from Experiment project label.",
            "object_pk": 1,
            "id": 1,
            "resource_uri": "/rundb/api/v1/eventlog/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.27 Experiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/experiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/experiment/schema/`

Perform CRUD operations on `experiment` resources and data elements.

Fields table

field	help text	default
isReverseRun	Boolean data. Ex: True	false
storage_options	Unicode string data. Ex: “Hello World”	A
chipType	Unicode string data. Ex: “Hello World”	n/a
user_ack	Unicode string data. Ex: “Hello World”	U
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
sample	Unicode string data. Ex: “Hello World”	n/a
runMode	Unicode string data. Ex: “Hello World”	
reverse_primer	Unicode string data. Ex: “Hello World”	n/a
seqKitBarcode	Unicode string data. Ex: “Hello World”	
id	Integer data. Ex: 2673	
metaData	Unicode string data. Ex: “Hello World”	{}
log	Unicode string data. Ex: “Hello World”	{}
sequencekitbarcode	Unicode string data. Ex: “Hello World”	n/a
resource_uri	Unicode string data. Ex: “Hello World”	n/a
eas_set	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
runtype	Unicode string data. Ex: “Hello World”	n/a
platform	Unicode string data. Ex: “Hello World”	
samples	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
pinnedRepResult	Boolean data. Ex: True	false

Table 2.8 – continued from previous page

field	help text	default
reagentBarcode	Unicode string data. Ex: “Hello World”	
star	Boolean data. Ex: True	false
isProton	Unicode string data. Ex: “Hello World”	n/a
expCompInfo	Unicode string data. Ex: “Hello World”	
resultDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true
flows	Integer data. Ex: 2673	n/a
plan	A single related resource. Can be either a URI or set of nested resource data.	n/a
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a
diskusage	Integer data. Ex: 2673	n/a
unique	Unicode string data. Ex: “Hello World”	n/a
expDir	Unicode string data. Ex: “Hello World”	n/a
autoAnalyze	Boolean data. Ex: True	true
ftpStatus	Unicode string data. Ex: “Hello World”	
flowsInOrder	Unicode string data. Ex: “Hello World”	
baselineRun	Boolean data. Ex: True	false
displayName	Unicode string data. Ex: “Hello World”	
notes	Unicode string data. Ex: “Hello World”	n/a
sequencekitname	Unicode string data. Ex: “Hello World”	n/a
chipBarcode	Unicode string data. Ex: “Hello World”	
pgmName	Unicode string data. Ex: “Hello World”	n/a
storageHost	Unicode string data. Ex: “Hello World”	n/a
expName	Unicode string data. Ex: “Hello World”	n/a
status	Unicode string data. Ex: “Hello World”	
usePreBeadfind	Boolean data. Ex: True	true
cycles	Integer data. Ex: 2673	n/a
rawdatastyle	Unicode string data. Ex: “Hello World”	single

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/experiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/experiment/", params={"format": "
ts_api_response = ts_api_request.json()

experiments = ts_api_response["objects"]

for experiment in experiments:
    print experiment
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 21008,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/experiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "storage_options": "A",
            "chipType": "",
            "user_ack": "U",
            "results": [],
            "sample": "E115943-1q204-01-L8095",
            "runMode": "",
            "reverse_primer": null,
            "seqKitBarcode": "",
            "id": 10132,
            "metaData": {},
            "log": {},
            "sequencekitbarcode": "",
            "resource_uri": "/rundb/api/v1/experiment/10132/",
            "eas_set": [
                {
                    "isEditable": true,
                    "hotSpotRegionBedFile": "",
                    "results": [],
                    "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/Ion-TargetSe
                    "thumbnailalignmentargs": "",
                    "thumbnailanalysisargs": "",
                    "id": 10575,
                    "barcodedSamples": {},
                    "reference": "hg19",
                    "isOneTimeOverride": false,
                    "analysisargs": "",
                    "thumbnailcalibrateargs": "",
                    "realign": false,
                    "selectedPlugins": {
                        "pinsPerFlow": {
                            "name": "pinsPerFlow"
                        },
                        "duplicateReads_useZC": {
                            "name": "duplicateReads_useZC"
                        },
                        "libClonality": {
                            "name": "libClonality"
                        },
                        "ProtonErrors": {
                            "name": "ProtonErrors"
                        },
                        "PhasingReport": {
                            "name": "PhasingReport"
                        },
                        "detailedReport": {
                            "name": "detailedReport"
                        },
                        "extended_chip_check": {
                            "name": "extended_chip_check"
                        },
                        "1_Torrent_Accuracy": {
```

```
            "name": "1_Torrent_Accuracy"
        },
        "ConversionRate": {
            "name": "ConversionRate"
        },
        "rawTrace": {
            "name": "rawTrace"
        },
        "filterAndTrim": {
            "name": "filterAndTrim"
        },
        "fsRecalibration": {
            "name": "fsRecalibration"
        },
        "timingPerformance": {
            "name": "timingPerformance"
        },
        "NucRiseParams": {
            "name": "NucRiseParams"
        },
        "AvgTrace": {
            "name": "AvgTrace"
        },
        "autoCal": {
            "name": "autoCal"
        },
        "flowCell": {
            "name": "flowCell"
        },
        "chipDiagnostics": {
            "name": "chipDiagnostics"
        },
        "rawPlots": {
            "name": "rawPlots"
        },
        "spatialPlots": {
            "name": "spatialPlots"
        },
        "RateMapEDA": {
            "name": "RateMapEDA"
        },
        "barcodeMixtureAnalysis": {
            "name": "barcodeMixtureAnalysis"
        },
        "z_homopolymerAnalysis": {
            "name": "z_homopolymerAnalysis"
        },
        "separator": {
            "name": "separator"
        },
        "GC_seq_performance": {
            "name": "GC_seq_performance"
        },
        "flowErr": {
            "name": "flowErr"
        }
    },
    "experiment": "/rundb/api/v1/experiment/10132/",
```

```
            "barcodeKitName": "",
            "beadfindargs": "",
            "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "thumbnailbasecallerargs": "",
            "status": "planned",
            "prebasecallerargs": "",
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "",
            "isDuplicateReads": false,
            "libraryKey": "TCAG",
            "date": "2013-05-15T18:30:24.000115+00:00",
            "libraryKitName": "",
            "thumbnailbeadfindargs": "",
            "calibrateargs": "",
            "tfKey": "",
            "libraryKitBarcode": null,
            "base_recalibrate": true,
            "basecallerargs": "",
            "resource_uri": "/rundb/api/v1/experimentanalysissettings/10575/"
        }
    ],
    "runtype": "GENS",
    "platform": "PGM",
    "samples": [
        {
            "status": "run",
            "sampleSets": [],
            "description": null,
            "displayedName": "E115943-lq204-01-L8095",
            "experiments": [
                "/rundb/api/v1/experiment/10142/",
                "/rundb/api/v1/experiment/10132/"
            ],
            "externalId": "",
            "date": "2013-05-15T18:30:24.000176+00:00",
            "resource_uri": "/rundb/api/v1/sample/2379/",
            "id": 2379,
            "name": "E115943-lq204-01-L8095"
        }
    ],
    "pinnedRepResult": false,
    "reagentBarcode": "",
    "star": false,
    "isProton": "False",
    "expCompInfo": "",
    "resultDate": "2013-05-15T18:30:24.000171+00:00",
    "flows": 400,
    "plan": "/rundb/api/v1/plannedexperiment/88364/",
    "date": "2013-05-15T18:30:24.000167+00:00",
    "diskusage": null,
    "unique": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
    "expDir": "",
    "autoAnalyze": true,
    "ftpStatus": "Complete",
    "flowsInOrder": "",
    "baselineRun": false,
    "displayName": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
    "notes": "OT2 lq204_01 Lib8095 275bp lr2 4B bead 1.2B lib SDS_10mMEDTA break ",
```

```
            "sequencekitname": "",
            "chipBarcode": "",
            "pgmName": "",
            "storageHost": null,
            "expName": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
            "status": "planned",
            "usePreBeadfind": false,
            "cycles": 0,
            "rawdatastyle": "single"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.28 Experimentanalysissettings Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/`

Schema URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/schema/`

Fields table

field	help text	de
isEditable	Boolean data. Ex: True	fal
hotSpotRegionBedFile	Unicode string data. Ex: “Hello World”	n/a
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
targetRegionBedFile	Unicode string data. Ex: “Hello World”	n/a
thumbnailalignmentargs	Unicode string data. Ex: “Hello World”	
thumbnailanalysisargs	Unicode string data. Ex: “Hello World”	
id	Integer data. Ex: 2673	
barcodedSamples	Unicode string data. Ex: “Hello World”	{}
reference	Unicode string data. Ex: “Hello World”	n/a
isOneTimeOverride	Boolean data. Ex: True	fal
analysisargs	Unicode string data. Ex: “Hello World”	
thumbnailcalibrateargs	Unicode string data. Ex: “Hello World”	
realign	Boolean data. Ex: True	fal
selectedPlugins	Unicode string data. Ex: “Hello World”	{}
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a
barcodeKitName	Unicode string data. Ex: “Hello World”	n/a
beadfindargs	Unicode string data. Ex: “Hello World”	
threePrimeAdapter	Unicode string data. Ex: “Hello World”	n/a

Table 2.9 – continued from previous page

field	help text	de
thumbnailbasecallerargs	Unicode string data. Ex: “Hello World”	
status	Unicode string data. Ex: “Hello World”	
prebasecallerargs	Unicode string data. Ex: “Hello World”	
prethumbnailbasecallerargs	Unicode string data. Ex: “Hello World”	
alignmentargs	Unicode string data. Ex: “Hello World”	
isDuplicateReads	Boolean data. Ex: True	fal
libraryKey	Unicode string data. Ex: “Hello World”	
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a
libraryKitName	Unicode string data. Ex: “Hello World”	n/a
thumbnailbeadfindargs	Unicode string data. Ex: “Hello World”	
calibrateargs	Unicode string data. Ex: “Hello World”	
tfKey	Unicode string data. Ex: “Hello World”	
libraryKitBarcode	Unicode string data. Ex: “Hello World”	n/a
base_recalibrate	Boolean data. Ex: True	tru
basecallerargs	Unicode string data. Ex: “Hello World”	
resource_uri	Unicode string data. Ex: “Hello World”	n/a

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/experimentanalysissettings/", para
ts_api_response = ts_api_request.json()

experimentanalysissettingss = ts_api_response["objects"]

for experimentanalysissettings in experimentanalysissettingss:
    print experimentanalysissettings
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 23666,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/experimentanalysissettings/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isEditable": true,
            "hotSpotRegionBedFile": "",
            "results": [],
            "targetRegionBedFile": "/results/uploads/BED/15/hg19/unmerged/detail/Ion-TargetSeq-Exome-
            "thumbnailalignmentargs": "",
```

```
            "thumbnailanalysisargs": "",
            "id": 138,
            "barcodedSamples": {},
            "reference": "hg19",
            "isOneTimeOverride": false,
            "analysisargs": "",
            "thumbnailcalibrateargs": "",
            "realign": false,
            "selectedPlugins": {
                "coverageAnalysis": {
                    "userInput": "",
                    "version": "3.4.47670",
                    "features": [],
                    "name": "coverageAnalysis",
                    "id": "319"
                }
            },
            "experiment": "/rundb/api/v1/experiment/6822/",
            "barcodeKitName": "",
            "beadfindargs": "",
            "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "thumbnailbasecallerargs": "",
            "status": "planned",
            "prebasecallerargs": "",
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "",
            "isDuplicateReads": false,
            "libraryKey": "TCAG",
            "date": "2012-12-04T00:09:21.000461+00:00",
            "libraryKitName": "Ion Xpress Plus Fragment Library Kit",
            "thumbnailbeadfindargs": "",
            "calibrateargs": "",
            "tfKey": "",
            "libraryKitBarcode": null,
            "base_recalibrate": true,
            "basecallerargs": "",
            "resource_uri": "/rundb/api/v1/experimentanalysissettings/138/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.29 Filemonitor Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/filemonitor/`

Schema URL: `http://mytorrentserver/rundb/api/v1/filemonitor/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	false	false	string
updated	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
name	Unicode string data. Ex: “Hello World”		false	false	false	false	string
created	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
url	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
md5sum	Unicode string data. Ex: “Hello World”	None	true	false	false	false	string
celery_task_id	Unicode string data. Ex: “Hello World”		false	false	true	false	string
local_dir	Unicode string data. Ex: “Hello World”		false	false	false	false	string
progress	Unicode string data. Ex: “Hello World”	0	false	false	false	false	string
size	Unicode string data. Ex: “Hello World”	None	true	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
tags	Unicode string data. Ex: “Hello World”		false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/filemonitor/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/filemonitor/", params={"format": '
ts_api_response = ts_api_request.json()

filemonitors = ts_api_response["objects"]

for filemonitor in filemonitors:
    print filemonitor
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 8,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/filemonitor/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Complete",
```

```
            "updated": "2014-05-08T19:25:27.000843+00:00",
            "name": "AmpliSeqExome.20131001.Results.zip",
            "created": "2014-05-08T19:25:01.000513+00:00",
            "url": "https://ampliseq.com/ws/tmpldesign/14035495/download/results",
            "md5sum": null,
            "celery_task_id": "63a36c6f-ccc2-4ce3-8539-18335039f128",
            "local_dir": "/tmp/tmpubvTKY",
            "progress": "24174499",
            "size": "24174499",
            "id": 9,
            "tags": "ampliseq_template",
            "resource_uri": "/rundb/api/v1/filemonitor/9/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.30 Fileserver Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/fileserver/`

Schema URL: `http://mytorrentserver/rundb/api/v1/fileserver/schema/`

Perform read-only operations on `fileserver` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
percentfull	Floating point numeric data. Ex: 26.73	0.0	true	false	false	false	float
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
filesPrefix	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
comments	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/fileserver/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/fileserver/", params={"format": "
ts_api_response = ts_api_request.json()

fileservers = ts_api_response["objects"]

for fileserver in fileservers:
    print fileserver
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/fileserver/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "percentfull": 82.6372068824745,
            "name": "gs1-nfs",
            "filesPrefix": "/ion-data/results/",
            "comments": "gs1-nfs:/gs1/ion-data",
            "id": 5,
            "resource_uri": "/rundb/api/v1/fileserver/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.31 Globalconfig Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/globalconfig/`

Schema URL: `http://mytorrentserver/rundb/api/v1/globalconfig/schema/`

Perform read-only operations on `globalconfig` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	ty
enable_version_lock	Boolean data. Ex: True	false	false	false	true	false	bo
site_name	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_support_upload	Boolean data. Ex: True	false	false	false	true	false	bo
plugin_output_folder	Unicode string data. Ex: “Hello World”		false	false	true	false	st
auto_archive_ack	Boolean data. Ex: True	false	false	false	true	false	bo
default_plugin_script	Unicode string data. Ex: “Hello World”		false	false	true	false	st
id	Integer data. Ex: 2673		false	false	true	true	in
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	st
default_storage_options	Unicode string data. Ex: “Hello World”	D	false	false	true	false	st
selected	Boolean data. Ex: True		false	false	true	false	bo
check_news_posts	Boolean data. Ex: True	true	false	false	true	false	bo
realign	Boolean data. Ex: True	false	false	false	true	false	bo
ts_update_status	Unicode string data. Ex: “Hello World”		false	false	true	false	st
mark_duplicates	Boolean data. Ex: True	false	false	false	true	false	bo
plugin_folder	Unicode string data. Ex: “Hello World”		false	false	true	false	st
auto_archive_enable	Boolean data. Ex: True	false	false	false	true	false	bo
reference_path	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_auto_security	Boolean data. Ex: True	true	false	false	true	false	bo
fasta_path	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_nightly_email	Boolean data. Ex: True	true	false	false	true	false	bo
barcode_args	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	st
sec_update_status	Unicode string data. Ex: “Hello World”		false	false	true	false	st
default_flow_order	Unicode string data. Ex: “Hello World”		false	false	true	false	st
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	st
records_to_display	Integer data. Ex: 2673	20	false	false	true	false	in
base_recalibrate	Boolean data. Ex: True	true	false	false	true	false	bo
default_library_key	Unicode string data. Ex: “Hello World”		false	false	true	false	st
web_root	Unicode string data. Ex: “Hello World”		false	false	true	false	st
default_test_fragment_key	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_auto_pkg_dl	Boolean data. Ex: True	true	false	false	true	false	bo
enable_compendia_OCP	Boolean data. Ex: True	false	false	false	true	false	bo

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/globalconfig/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/globalconfig/", params={"format":
ts_api_response = ts_api_request.json()

globalconfigs = ts_api_response["objects"]

for globalconfig in globalconfigs:
    print globalconfig
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": [
        {
            "enable_version_lock": false,
            "site_name": "blackbird-east",
            "enable_support_upload": false,
            "plugin_output_folder": "plugin_out",
            "auto_archive_ack": true,
            "default_plugin_script": "launch.sh",
            "id": 1,
            "resource_uri": "/rundb/api/v1/globalconfig/1/",
            "default_storage_options": "D",
            "selected": false,
            "check_news_posts": true,
            "realign": false,
            "ts_update_status": "No updates",
            "mark_duplicates": false,
            "plugin_folder": "plugins",
            "auto_archive_enable": true,
            "reference_path": "",
            "enable_auto_security": false,
            "fasta_path": "",
            "enable_nightly_email": true,
            "barcode_args": {
                "filter": "0.01"
            },
            "sec_update_status": "",
            "default_flow_order": "TACG",
            "name": "Config",
            "records_to_display": 50,
            "base_recalibrate": true,
            "default_library_key": "TCAG",
            "web_root": "http://blackbird.ite",
            "default_test_fragment_key": "ATCG",
            "enable_auto_pkg_dl": false,
            "enable_compendia_OCP": true
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.32 Ionchefplantemplate Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”

Table 2.11 – continued from previous page

field	help text
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”

Table 2.11 – continued from previous page

field	help text
reverse3primeadapter	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefplantemplate/", params={"fo
ts_api_response = ts_api_request.json()

ionchefplantemplates = ts_api_response["objects"]

for ionchefplantemplate in ionchefplantemplates:
    print ionchefplantemplate
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 7,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefplantemplate/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "chef_useGUI_Exome Panel_AmpliSeqExome.20131001",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
```

```
"thumbnailalignmentargs": "stage1 map4",
"chipType": "P1.1.17",
"chefProgress": 0.0,
"library": "hg19",
"reverselibrarykey": "",
"sampleTubeLabel": "",
"seqKitBarcode": null,
"barcodeId": "IonXpress",
"chefLogPath": null,
"isPlanGroup": false,
"realign": false,
"sampleGroupingName": "",
"experiment": "/rundb/api/v1/experiment/21899/",
"bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
"isReusable": true,
"isDuplicateReads": false,
"thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
"librarykitname": "Ion AmpliSeq 2.0 Library Kit",
"adapter": null,
"basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
"tfKey": "ATCG",
"parentPlan": null,
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "pending",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": true,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
        "id": 262521,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262521/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
        "id": 262520,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
```

```
                "resource_uri": "/rundb/api/v1/qctype/2/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/262520/"
        },
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
            "id": 262519,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/262519/"
        }
    ],
    "analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=216
    "thumbnailcalibrateargs": "calibrate --skipDroop",
    "templatingKitName": "Ion PI IC 200 Kit",
    "runType": "AMPS_EXOME",
    "username": "ionuser",
    "planName": "chef_useGUI_Exome_Panel_AmpliSeqExome.20131001",
    "sampleDisplayedName": "",
    "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
    "controlSequencekitname": "",
    "chefMessage": "",
    "childPlans": [],
    "pairedEndLibraryAdapterName": "",
    "runMode": "single",
    "irworkflow": "",
    "planExecuted": false,
    "project": "",
    "usePostBeadfind": false,
    "runname": null,
    "planGUID": "83e69cd8-ca3a-4ff2-b4ac-dd637184e28e",
    "planShortID": "YHFWJ",
    "sampleSetGroupType": null,
    "sample": "",
    "planExecutedDate": null,
    "reverse_primer": null,
    "id": 100136,
    "barcodedSamples": {},
    "regionfile": "",
    "selectedPlugins": {
        "IonReporterUploader": {
            "userInput": {
                "accountName": "None",
                "userInputInfo": "",
                "accountId": "0"
            },
            "version": "4.1-r87449",
            "features": [
                "export"
            ],
```

```
            "name": "IonReporterUploader",
            "id": 804
        },
        "coverageAnalysis": {
            "userInput": "",
            "version": "4.2-r86949",
            "features": [],
            "name": "coverageAnalysis",
            "id": 800
        },
        "validateVariantCaller": {
            "userInput": {
                "variant_caller_name": "variantCaller",
                "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
                "region": "NIST",
                "sample": "NA12878",
                "truth_minor_snp": "None",
                "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
                "truth_minor_indel": "None"
            },
            "version": "0.2.1",
            "features": [],
            "name": "validateVariantCaller",
            "id": 732
        },
        "variantCaller": {
            "userInput": {
                "torrent_variant_caller": {
                    "snp_min_allele_freq": "0.10000000000000001",
                    "snp_strand_bias": "0.97999999999999998",
                    "hotspot_min_coverage": 6,
                    "hotspot_min_cov_each_strand": 3,
                    "hotspot_min_allele_freq": "0.10000000000000001",
                    "snp_min_variant_score": 15,
                    "hotspot_strand_bias": "0.94999999999999996",
                    "hp_max_length": 8,
                    "filter_insertion_predictions": "0.20000000000000001",
                    "indel_min_variant_score": 20,
                    "indel_min_coverage": 10,
                    "heavy_tailed": 3,
                    "outlier_probability": "0.01",
                    "data_quality_stringency": 5,
                    "snp_min_cov_each_strand": 0,
                    "hotspot_min_variant_score": 10,
                    "indel_strand_bias": "0.90000000000000002",
                    "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                    "downsample_to_coverage": 400,
                    "filter_unusual_predictions": "0.25",
                    "indel_min_allele_freq": "0.14999999999999999",
                    "do_snp_realignment": 1,
                    "prediction_precision": 1,
                    "indel_min_cov_each_strand": 5,
                    "filter_deletion_predictions": "0.20000000000000001",
                    "suppress_recalibration": 0,
                    "snp_min_coverage": 5
                },
                "meta": {
                    "repository_id": "",
```

```
                    "ts_version": "4.0",
                    "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.201309
                    "user_selections": {
                        "chip": "proton_p1",
                        "frequency": "germline",
                        "library": "ampliseq",
                        "panel": "/rundb/api/v1/contentupload/53/"
                    },
                    "librarytype": "ampliseq",
                    "trimreads": true,
                    "tooltip": "Retrieved from external file",
                    "tvcargs": "tvc",
                    "built_in": false,
                    "configuration": "",
                    "compatibility": {}
                },
                "long_indel_assembler": {
                    "min_indel_size": 4,
                    "short_suffix_match": 5,
                    "output_mnv": 0,
                    "min_var_count": 5,
                    "min_var_freq": "0.14999999999999999",
                    "kmer_len": 19,
                    "max_hp_length": 8,
                    "relative_strand_bias": "0.80000000000000004"
                },
                "freebayes": {
                    "gen_min_coverage": 5,
                    "allow_mnps": 1,
                    "allow_complex": 0,
                    "read_max_mismatch_fraction": 1,
                    "read_mismatch_limit": 10,
                    "allow_indels": 1,
                    "min_mapping_qv": 4,
                    "gen_min_alt_allele_freq": "0.10000000000000001",
                    "allow_snps": 1,
                    "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                }
            },
            "version": "4.1-r74477",
            "features": [],
            "name": "variantCaller",
            "id": 698
        },
        "AmpliconStats": {
            "userInput": "",
            "version": "0.4.5",
            "features": [],
            "name": "AmpliconStats",
            "id": 774
        }
    },
    "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
    "sampleSet": null,
    "isSystemDefault": false,
    "autoName": null,
    "libraryKey": "TCAG",
    "flows": 520,
```

```
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regio
            "date": "2014-05-20T13:56:24.000114+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/ionchefplantemplate/100136/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.33 Ionchefplantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa

Table 2.12 – continued from previous page

field	help text	default	nullable	readonly	b
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/", para
ts_api_response = ts_api_request.json()
```

```
ionchefplantemplatesummarys = ts_api_response["objects"]

for ionchefplantemplatesummary in ionchefplantemplatesummarys:
    print ionchefplantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 7,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefplantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_useGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "YHFWJ",
            "planStatus": "pending",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100136,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI IC 200 Kit",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:56:24.000114+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_useGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
```

```
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "83e69cd8-ca3a-4ff2-b4ac-dd637184e28e",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/ionchefplantemplatesummary/100136/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.34 Ionchefprepkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/", params={"fo
ts_api_response = ts_api_request.json()

ionchefprepkitinfos = ts_api_response["objects"]

for ionchefprepkitinfo in ionchefprepkitinfos:
    print ionchefprepkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefprepkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "IonChefPrepKit",
            "description": "ION PGM IC 200 KIT",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "100020579",
                    "id": 20085,
                    "resource_uri": "/rundb/api/v1/kitpart/20085/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "100020580",
                    "id": 20086,
                    "resource_uri": "/rundb/api/v1/kitpart/20086/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "01",
                    "id": 20123,
                    "resource_uri": "/rundb/api/v1/kitpart/20123/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "ICPREP0001",
            "resource_uri": "/rundb/api/v1/ionchefprepkitinfo/20042/",
            "id": 20042,
            "categories": "",
            "name": "ION PGM IC 200 KIT"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.35 Kitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/kitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/kitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/kitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/kitinfo/", params={"format": "jso
```

```
ts_api_response = ts_api_request.json()

kitinfos = ts_api_response["objects"]

for kitinfo in kitinfos:
    print kitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 67,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/kitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "ControlSequenceKitType",
            "description": "ERCC Mix 1",
            "nucleotideType": "rna",
            "instrumentType": "",
            "runMode": "",
            "parts": [],
            "flowCount": 0,
            "applicationType": "RNA",
            "uid": "CONSEQ0006",
            "resource_uri": "/rundb/api/v1/kitinfo/20061/",
            "id": 20061,
            "categories": "",
            "name": "ERCC Mix 1"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.36 Kitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/kitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/kitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/kitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/kitpart/", params={"format": "jso
ts_api_response = ts_api_request.json()

kitparts = ts_api_response["objects"]

for kitpart in kitparts:
    print kitpart
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 139,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/kitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "4456739",
            "id": 20131,
            "resource_uri": "/rundb/api/v1/kitpart/20131/",
            "kit": "/rundb/api/v1/kitinfo/20060/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

2.1.37 Libmetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/libmetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/libmetrics/schema/`

Perform read-only operations on `libmetrics` resources and data elements.

Fields table

field	help text	default	nullable
i350Q17_reads	Integer data. Ex: 2673	n/a	false
i150Q47_reads	Integer data. Ex: 2673	n/a	false
i300Q47_reads	Integer data. Ex: 2673	n/a	false
i600Q20_reads	Integer data. Ex: 2673	n/a	false
i300Q20_reads	Integer data. Ex: 2673	n/a	false
i250Q17_reads	Integer data. Ex: 2673	n/a	false
q10_longest_alignment	Integer data. Ex: 2673	n/a	false
i50Q10_reads	Integer data. Ex: 2673	n/a	false
aveKeyCounts	Floating point numeric data. Ex: 26.73	n/a	false
i50Q17_reads	Integer data. Ex: 2673	n/a	false
total_mapped_target_bases	Unicode string data. Ex: “Hello World”	n/a	false
i200Q7_reads	Integer data. Ex: 2673	n/a	false
i100Q47_reads	Integer data. Ex: 2673	n/a	false
i50Q20_reads	Integer data. Ex: 2673	n/a	false
i450Q7_reads	Integer data. Ex: 2673	n/a	false
genomesize	Unicode string data. Ex: “Hello World”	n/a	false
i550Q20_reads	Integer data. Ex: 2673	n/a	false
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false
i450Q47_reads	Integer data. Ex: 2673	n/a	false
dr	Floating point numeric data. Ex: 26.73	n/a	false
i150Q17_reads	Integer data. Ex: 2673	n/a	false
q7_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i350Q7_reads	Integer data. Ex: 2673	n/a	false
i500Q20_reads	Integer data. Ex: 2673	n/a	false
q20_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i250Q47_reads	Integer data. Ex: 2673	n/a	false
q47_longest_alignment	Integer data. Ex: 2673	n/a	false
i550Q17_reads	Integer data. Ex: 2673	n/a	false
i50Q47_reads	Integer data. Ex: 2673	n/a	false
i200Q17_reads	Integer data. Ex: 2673	n/a	false
i250Q20_reads	Integer data. Ex: 2673	n/a	false
q47_alignments	Integer data. Ex: 2673	n/a	false
align_sample	Integer data. Ex: 2673	n/a	false

Table 2.13 – continued from previous page

field	help text	default	nullable
i100Q10_reads	Integer data. Ex: 2673	n/a	false
i350Q20_reads	Integer data. Ex: 2673	n/a	false
i100Q7_reads	Integer data. Ex: 2673	n/a	false
i400Q17_reads	Integer data. Ex: 2673	n/a	false
i500Q47_reads	Integer data. Ex: 2673	n/a	false
i450Q20_reads	Integer data. Ex: 2673	n/a	false
q7_mean_alignment_length	Integer data. Ex: 2673	n/a	false
q7_alignments	Integer data. Ex: 2673	n/a	false
total_mapped_reads	Unicode string data. Ex: “Hello World”	n/a	false
i600Q10_reads	Integer data. Ex: 2673	n/a	false
i250Q10_reads	Integer data. Ex: 2673	n/a	false
cf	Floating point numeric data. Ex: 26.73	n/a	false
i500Q7_reads	Integer data. Ex: 2673	n/a	false
q10_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i550Q7_reads	Integer data. Ex: 2673	n/a	false
duplicate_reads	Integer data. Ex: 2673	n/a	true
i350Q47_reads	Integer data. Ex: 2673	n/a	false
totalNumReads	Integer data. Ex: 2673	n/a	false
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false
i350Q10_reads	Integer data. Ex: 2673	n/a	false
i300Q10_reads	Integer data. Ex: 2673	n/a	false
q20_mean_alignment_length	Integer data. Ex: 2673	n/a	false
i250Q7_reads	Integer data. Ex: 2673	n/a	false
i200Q10_reads	Integer data. Ex: 2673	n/a	false
i400Q7_reads	Integer data. Ex: 2673	n/a	false
i200Q47_reads	Integer data. Ex: 2673	n/a	false
q7_longest_alignment	Integer data. Ex: 2673	n/a	false
i500Q10_reads	Integer data. Ex: 2673	n/a	false
Genome_Version	Unicode string data. Ex: “Hello World”	n/a	false
i400Q20_reads	Integer data. Ex: 2673	n/a	false
q10_alignments	Integer data. Ex: 2673	n/a	false
i450Q17_reads	Integer data. Ex: 2673	n/a	false
i100Q20_reads	Integer data. Ex: 2673	n/a	false
i550Q10_reads	Integer data. Ex: 2673	n/a	false
i450Q10_reads	Integer data. Ex: 2673	n/a	false
i400Q47_reads	Integer data. Ex: 2673	n/a	false
q17_longest_alignment	Integer data. Ex: 2673	n/a	false
i150Q7_reads	Integer data. Ex: 2673	n/a	false
i400Q10_reads	Integer data. Ex: 2673	n/a	false
q10_mean_alignment_length	Integer data. Ex: 2673	n/a	false
raw_accuracy	Floating point numeric data. Ex: 26.73	n/a	false
sysSNR	Floating point numeric data. Ex: 26.73	n/a	false
q17_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
Index_Version	Unicode string data. Ex: “Hello World”	n/a	false
i300Q17_reads	Integer data. Ex: 2673	n/a	false
q17_mean_alignment_length	Integer data. Ex: 2673	n/a	false
ie	Floating point numeric data. Ex: 26.73	n/a	false
id	Integer data. Ex: 2673		false
q20_alignments	Integer data. Ex: 2673	n/a	false

Table 2.13 – continued from previous page

field	help text	default	nullable
q47_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
genome	Unicode string data. Ex: “Hello World”	n/a	false
i300Q7_reads	Integer data. Ex: 2673	n/a	false
i150Q20_reads	Integer data. Ex: 2673	n/a	false
i550Q47_reads	Integer data. Ex: 2673	n/a	false
i600Q47_reads	Integer data. Ex: 2673	n/a	false
i100Q17_reads	Integer data. Ex: 2673	n/a	false
q47_mean_alignment_length	Integer data. Ex: 2673	n/a	false
i50Q7_reads	Integer data. Ex: 2673	n/a	false
i600Q7_reads	Integer data. Ex: 2673	n/a	false
i600Q17_reads	Integer data. Ex: 2673	n/a	false
q17_alignments	Integer data. Ex: 2673	n/a	false
i500Q17_reads	Integer data. Ex: 2673	n/a	false
i150Q10_reads	Integer data. Ex: 2673	n/a	false
q20_longest_alignment	Integer data. Ex: 2673	n/a	false
i200Q20_reads	Integer data. Ex: 2673	n/a	false

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/libmetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/libmetrics/", params={"format": "
ts_api_response = ts_api_request.json()

libmetricss = ts_api_response["objects"]

for libmetrics in libmetricss:
    print libmetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 39455,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/libmetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "i350Q17_reads": 0,
            "i150Q47_reads": 0,
            "i300Q47_reads": 0,
            "i600Q20_reads": 0,
            "i300Q20_reads": 0,
```

```
"i250Q17_reads": 0,
"q10_longest_alignment": 137,
"i50Q10_reads": 5244,
"aveKeyCounts": 82.0,
"i50Q17_reads": 5050,
"total_mapped_target_bases": "0",
"i200Q7_reads": 0,
"i100Q47_reads": 2641,
"i50Q20_reads": 4738,
"i450Q7_reads": 0,
"genomesize": "4686137",
"i550Q20_reads": 0,
"report": "/rundb/api/v1/results/3/",
"i450Q47_reads": 0,
"dr": 0.310014492754,
"i150Q17_reads": 0,
"q7_mapped_bases": "552185",
"i350Q7_reads": 0,
"i500Q20_reads": 0,
"q20_mapped_bases": "493269",
"i250Q47_reads": 0,
"q47_longest_alignment": 133,
"i550Q17_reads": 0,
"i50Q47_reads": 4527,
"i200Q17_reads": 0,
"i250Q20_reads": 0,
"q47_alignments": 4942,
"align_sample": 0,
"i100Q10_reads": 3990,
"i350Q20_reads": 0,
"i100Q7_reads": 3991,
"i400Q17_reads": 0,
"i500Q47_reads": 0,
"i450Q20_reads": 0,
"q7_mean_alignment_length": 104,
"q7_alignments": 5311,
"total_mapped_reads": "0",
"i600Q10_reads": 0,
"i250Q10_reads": 0,
"cf": 0.760992753623,
"i500Q7_reads": 0,
"q10_mapped_bases": "551679",
"i550Q7_reads": 0,
"duplicate_reads": null,
"i350Q47_reads": 0,
"totalNumReads": 5381,
"resource_uri": "/rundb/api/v1/libmetrics/1/",
"i350Q10_reads": 0,
"i300Q10_reads": 0,
"q20_mean_alignment_length": 98,
"i250Q7_reads": 0,
"i200Q10_reads": 0,
"i400Q7_reads": 0,
"i200Q47_reads": 0,
"q7_longest_alignment": 137,
"i500Q10_reads": 0,
"Genome_Version": "1",
"i400Q20_reads": 0,
```

```
            "q10_alignments": 5306,
            "i450Q17_reads": 0,
            "i100Q20_reads": 3443,
            "i550Q10_reads": 0,
            "i450Q10_reads": 0,
            "i400Q47_reads": 0,
            "q17_longest_alignment": 137,
            "i150Q7_reads": 0,
            "i400Q10_reads": 0,
            "q10_mean_alignment_length": 104,
            "raw_accuracy": 0.0,
            "sysSNR": 17.32,
            "q17_mapped_bases": "524626",
            "Index_Version": "tmap-f2",
            "i300Q17_reads": 0,
            "q17_mean_alignment_length": 102,
            "ie": 0.884253623188,
            "id": 1,
            "q20_alignments": 5030,
            "q47_mapped_bases": "457712",
            "genome": "E. coli DH10B",
            "i300Q7_reads": 0,
            "i150Q20_reads": 0,
            "i550Q47_reads": 0,
            "i600Q47_reads": 0,
            "i100Q17_reads": 3714,
            "q47_mean_alignment_length": 93,
            "i50Q7_reads": 5250,
            "i600Q7_reads": 0,
            "i600Q17_reads": 0,
            "q17_alignments": 5156,
            "i500Q17_reads": 0,
            "i150Q10_reads": 0,
            "q20_longest_alignment": 137,
            "i200Q20_reads": 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.38 Librarykey Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykey/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykey/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
direction	Unicode string data. Ex: “Hello World”	Forward	false	false	false	false	string
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”		false	false	true	false	string
runMode	Unicode string data. Ex: “Hello World”	single	false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isDefault	Boolean data. Ex: True	false	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykey/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykey/", params={"format": "
ts_api_response = ts_api_request.json()

librarykeys = ts_api_response["objects"]

for librarykey in librarykeys:
    print librarykey
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykey/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "direction": "Forward",
            "name": "Ion TCAG",
            "sequence": "TCAG",
            "description": "Default forward library key",
            "runMode": "single",
            "id": 3,
            "isDefault": true,
            "resource_uri": "/rundb/api/v1/librarykey/3/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.39 Librarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykitinfo/", params={"format"
ts_api_response = ts_api_request.json()

librarykitinfos = ts_api_response["objects"]

for librarykitinfo in librarykitinfos:
    print librarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/librarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

2.1.40 Librarykitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykitpart/", params={"format"
ts_api_response = ts_api_request.json()

librarykitparts = ts_api_response["objects"]

for librarykitpart in librarykitparts:
    print librarykitpart
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 17,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "A25908",
```

```
            "id": 20140,
            "resource_uri": "/rundb/api/v1/librarykitpart/20140/",
            "kit": "/rundb/api/v1/kitinfo/20065/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.41 Location Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/location/`

Schema URL: `http://mytorrentserver/rundb/api/v1/location/schema/`

Perform read-only operations on `location` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
defaultlocation	Only one location can be the default	false	false	false	true	false	boolean
comments	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/location/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/location/", params={"format": "jso
ts_api_response = ts_api_request.json()

locations = ts_api_response["objects"]

for location in locations:
    print location
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/location/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "Local",
            "resource_uri": "/rundb/api/v1/location/3/",
            "defaultlocation": false,
            "comments": "",
            "id": 3
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.42 Log Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/log/`

Schema URL: `http://mytorrentserver/rundb/api/v1/log/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
text	Unicode string data. Ex: “Hello World”		false	false	true	false	string
timeStamp	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
upload	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/log/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/log/", params={"format": "json", "
ts_api_response = ts_api_request.json()

logs = ts_api_response["objects"]

for log in logs:
    print log
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 858,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/log/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "text": "FATAL ERROR: '/.fasta.fai' does not exist.",
            "timeStamp": "2013-05-30T15:09:03.000306+00:00",
            "upload": "/rundb/api/v1/contentupload/26/",
            "id": 885,
            "resource_uri": "/rundb/api/v1/log/885/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.43 Message Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/message/`

Schema URL: `http://mytorrentserver/rundb/api/v1/message/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
body	Unicode string data. Ex: "Hello World"		false	false	true	false	string
status	Unicode string data. Ex: "Hello World"	unread	false	false	true	false	string
level	Integer data. Ex: 2673	20	false	false	false	false	integer
route	Unicode string data. Ex: "Hello World"		false	false	true	false	string
expires	Unicode string data. Ex: "Hello World"	read	false	false	true	false	string
time	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
tags	Unicode string data. Ex: "Hello World"		false	false	true	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/message/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/message/", params={"format": "jso
ts_api_response = ts_api_request.json()

messages = ts_api_response["objects"]

for message in messages:
    print message
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get

- post
- put
- delete
- patch

2.1.44 Monitordata Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/monitordata/`

Schema URL: `http://mytorrentserver/rundb/api/v1/monitordata/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
treeDat	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
name	Unicode string data. Ex: “Hello World”		false	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/monitordata/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/monitordata/", params={"format": '
ts_api_response = ts_api_request.json()

monitordatas = ts_api_response["objects"]

for monitordata in monitordatas:
    print monitordata
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": [
        {
            "resource_uri": "/rundb/api/v1/monitordata/1/",
            "treeDat": {},
```

```
            "id": 1,
            "name": "Debug"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.45 Monitorresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/monitorresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/monitorresult/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
libmetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
analysismetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
library	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
qualitymetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
barcodeId	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
representative	Boolean data. Ex: True	false	false	false	true	false	boolean

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/monitorresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/monitorresult/", params={"format"
ts_api_response = ts_api_request.json()
```

```
monitorresults = ts_api_response["objects"]

for monitorresult in monitorresults:
    print monitorresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 159,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/monitorresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Completed",
            "processedflows": 0,
            "libmetrics": {
                "i100Q20_reads": 61016174,
                "aveKeyCounts": 89.0,
                "id": 41874,
                "resource_uri": "",
                "q20_mean_alignment_length": 159
            },
            "timeStamp": "2014-06-28T07:11:58.000789+00:00",
            "analysismetrics": {
                "ignored": 1364892,
                "lib": 118063544,
                "total_wells": 164699136,
                "pinned": 463867,
                "live": 118074386,
                "excluded": 16095180,
                "bead": 118525043,
                "resource_uri": "",
                "id": 41726,
                "empty": 28250154,
                "libFinal": 85604279
            },
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
            "library": "hg19",
            "id": 304393,
            "reportStatus": "Nothing",
            "experiment": {
                "ftpStatus": "Complete",
                "chipInstrumentType": "proton",
                "displayName": "user Z28-428--r65714-pou4 dbsa",
                "chipType": "P1.1.17",
                "notes": "",
                "chipDescription": "PI",
                "resultDate": "2014-06-28T07:11:58.000789+00:00",
                "flows": 520,
                "runMode": "single",
                "expName": "R_2014_06_27_17_13_22_user_Z28-428--r65714-pou4_dbsa",
                "storage_options": "D",
```

```
                "pgmName": "Z28",
                "date": "2014-06-27T21:19:01+00:00",
                "star": false,
                "resource_uri": "",
                "qcThresholds": {
                    "Key Signal (1-100)": 30,
                    "Usable Sequence (%)": 30,
                    "Bead Loading (%)": 30
                },
                "id": 23958,
                "plan": {
                    "runType": "AMPS_EXOME",
                    "id": 102195,
                    "resource_uri": ""
                }
            },
            "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
            "projects": [
                {
                    "resource_uri": "",
                    "id": 1385,
                    "name": "auto_chip",
                    "modified": "2014-06-27T21:21:43.000081+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 178.362197969099,
                "q0_reads": 85604279,
                "q0_bases": "15268567358",
                "q20_reads": 85604279,
                "q20_bases": "13060288783",
                "q20_mean_read_length": 178,
                "id": 39683,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/monitorresult/304393/",
            "barcodeId": "IonXpress",
            "autoExempt": false,
            "representative": false
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.46 Obsoletereferencegenome Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/`

Schema URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
reference_path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
short_name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
index_version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
notes	Unicode string data. Ex: "Hello World"		false	false	true	false	string
enabled	Boolean data. Ex: True	true	false	false	true	false	boolean
species	Unicode string data. Ex: "Hello World"		false	false	true	false	string
identity_hash	Unicode string data. Ex: "Hello World"	None	true	false	false	false	string
source	Unicode string data. Ex: "Hello World"		false	false	true	false	string
version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
celery_task_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	2014-06-28T14:03:44.000226+00:00	false	false	false	false	datetime
verbose_error	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/", params=
ts_api_response = ts_api_request.json()

obsoletereferencegenomes = ts_api_response["objects"]

for obsoletereferencegenome in obsoletereferencegenomes:
    print obsoletereferencegenome
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.47 Onetouchplantemplate Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/`

Schema URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.

Table 2.14 – continued from previous page

field	help text
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”

Table 2.14 – continued from previous page

field	help text
childPlans	A list of data. Ex: ['abc', 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: "Hello World"
runMode	Unicode string data. Ex: "Hello World"
irworkflow	Unicode string data. Ex: "Hello World"
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: "Hello World"
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: "Hello World"
planGUID	Unicode string data. Ex: "Hello World"
planShortID	Unicode string data. Ex: "Hello World"
sampleSetGroupType	Unicode string data. Ex: "Hello World"
sample	Unicode string data. Ex: "Hello World"
planExecutedDate	A date & time as a string. Ex: "2010-11-10T03:07:43"
reverse_primer	Unicode string data. Ex: "Hello World"
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: "Hello World"
regionfile	Unicode string data. Ex: "Hello World"
selectedPlugins	Unicode string data. Ex: "Hello World"
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: "Hello World"
libraryKey	Unicode string data. Ex: "Hello World"
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: "2010-11-10T03:07:43"
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: "Hello World"
sampleSetDisplayedName	Unicode string data. Ex: "Hello World"
flowsInOrder	Unicode string data. Ex: "Hello World"
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: "Hello World"
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: "Hello World"
reverse3primeadapter	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/onetouchplantemplate/", params={"
ts_api_response = ts_api_request.json()

onetouchplantemplates = ts_api_response["objects"]

for onetouchplantemplate in onetouchplantemplates:
    print onetouchplantemplate
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 58,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/onetouchplantemplate/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.1.17",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": "",
            "seqKitBarcode": null,
            "barcodeId": "IonXpress",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/21987/",
            "bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
            "isReusable": true,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
            "librarykitname": "Ion AmpliSeq 2.0 Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
            "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "planStatus": "planned",
```

```
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": true,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262674,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262674/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262673,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262673/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262672,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262672/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=21
"thumbnailcalibrateargs": "calibrate --skipDroop",
"templatingKitName": "Ion PI Template OT2 200 Kit v3",
"runType": "AMPS_EXOME",
```

```
"username": "ionuser",
"planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
"sampleDisplayedName": "",
"prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
"controlSequencekitname": "",
"chefMessage": "",
"childPlans": [],
"pairedEndLibraryAdapterName": "",
"runMode": "single",
"irworkflow": "",
"planExecuted": false,
"project": "",
"usePostBeadfind": false,
"runname": null,
"planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
"planShortID": "2OXYZ",
"sampleSetGroupType": null,
"sample": "",
"planExecutedDate": null,
"reverse_primer": null,
"id": 100224,
"barcodedSamples": {},
"regionfile": "",
"selectedPlugins": {
    "IonReporterUploader": {
        "userInput": {
            "accountName": "None",
            "userInputInfo": "",
            "accountId": "0"
        },
        "version": "4.1-r87449",
        "features": [
            "export"
        ],
        "name": "IonReporterUploader",
        "id": 804
    },
    "coverageAnalysis": {
        "userInput": "",
        "version": "4.2-r86949",
        "features": [],
        "name": "coverageAnalysis",
        "id": 800
    },
    "variantCaller": {
        "userInput": {
            "torrent_variant_caller": {
                "snp_min_allele_freq": "0.10000000000000001",
                "snp_strand_bias": "0.97999999999999998",
                "hotspot_min_coverage": 6,
                "hotspot_min_cov_each_strand": 3,
                "hotspot_min_allele_freq": "0.10000000000000001",
                "snp_min_variant_score": 15,
                "hotspot_strand_bias": "0.94999999999999996",
                "hp_max_length": 8,
                "filter_insertion_predictions": "0.20000000000000001",
                "indel_min_variant_score": 20,
                "indel_min_coverage": 10,
```

```
                "heavy_tailed": 3,
                "outlier_probability": "0.01",
                "data_quality_stringency": 5,
                "snp_min_cov_each_strand": 0,
                "hotspot_min_variant_score": 10,
                "indel_strand_bias": "0.90000000000000002",
                "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                "downsample_to_coverage": 400,
                "filter_unusual_predictions": "0.25",
                "indel_min_allele_freq": "0.14999999999999999",
                "do_snp_realignment": 1,
                "prediction_precision": 1,
                "indel_min_cov_each_strand": 5,
                "filter_deletion_predictions": "0.20000000000000001",
                "suppress_recalibration": 0,
                "snp_min_coverage": 5
            },
            "meta": {
                "repository_id": "",
                "ts_version": "4.0",
                "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.20130
                "user_selections": {
                    "chip": "proton_p1",
                    "frequency": "germline",
                    "library": "ampliseq",
                    "panel": "/rundb/api/v1/contentupload/53/"
                },
                "librarytype": "ampliseq",
                "trimreads": true,
                "tooltip": "Retrieved from external file",
                "tvcargs": "tvc",
                "built_in": false,
                "configuration": "",
                "compatibility": {}
            },
            "long_indel_assembler": {
                "min_indel_size": 4,
                "short_suffix_match": 5,
                "output_mnv": 0,
                "min_var_count": 5,
                "min_var_freq": "0.14999999999999999",
                "kmer_len": 19,
                "max_hp_length": 8,
                "relative_strand_bias": "0.80000000000000004"
            },
            "freebayes": {
                "gen_min_coverage": 5,
                "allow_mnps": 1,
                "allow_complex": 0,
                "read_max_mismatch_fraction": 1,
                "read_mismatch_limit": 10,
                "allow_indels": 1,
                "min_mapping_qv": 4,
                "gen_min_alt_allele_freq": "0.10000000000000001",
                "allow_snps": 1,
                "gen_min_indel_alt_allele_freq": "0.14999999999999999"
            }
        },
```

```
                    "version": "4.1-r74477",
                    "features": [],
                    "name": "variantCaller",
                    "id": 698
                },
                "validateVariantCaller": {
                    "userInput": {
                        "variant_caller_name": "variantCaller",
                        "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
                        "region": "NIST",
                        "sample": "NA12878",
                        "truth_minor_snp": "None",
                        "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
                        "truth_minor_indel": "None"
                    },
                    "version": "0.2.1",
                    "features": [],
                    "name": "validateVariantCaller",
                    "id": 732
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/onetouchplantemplate/100224/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.48 Onetouchplantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr

Table 2.15 – continued from previous page

field	help text	default	nullable	readonly	b
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/?format=json&limi`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/", par
ts_api_response = ts_api_request.json()

onetouchplantemplatesummarys = ts_api_response["objects"]

for onetouchplantemplatesummary in onetouchplantemplatesummarys:
    print onetouchplantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 58,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/onetouchplantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
```

```
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/onetouchplantemplatesummary/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.49 Plannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/schema/`

Perform CRUD operations on `plannedexperiment` resources and data elements.

Even though plannedExperiment db schema has changed dramatically in TSS 3.6 as part of the "plan data decentralization" (aka PDD) effort. A facade is provided so if you are already familiar with using the plan REST API, changes under the hood are abstracted from the REST API users. However, note that "selectedPlugins" and "barcodedSamples" are JSON fields and their data structures tend to change from release to release.

What has changed in TSS 4.2

- The JSON data structure in barcodedSamples has been changed with the following added
- controlSequenceType
- hotSpotRegionBedFile
- nucleotideType
- reference
- targetRegionBedFile
- The JSON data structure in selectedPlugins for IonReporter has been changed with the following added
- NucleotideType
- cancerType
- cellularityPct
- New VariantCaller parameters have been added and some parameters have been obsolete (persisted in selectedPlugins)
- New values for runType, applicationGroup and sampleGrouping have been added to support DNA and Fusions
- Some new attributes intended for internal use only have been added to plannedExperiment.
- We have started enforcing validation during REST API posting for
- barcodeId
- chipType
- flows
- notes
- planName
- project or projects
- runType
- sampleTubeLabel
- sample or sample in barcodedSamples
- sampleGroupingName
- sequencekitname
- templateKitName
- Posting that fails validation will receive an error code.
- Until stringent validation is fully in place during non-GUI REST API posting, please do your due diligence to ensure the data and data format posted are valid.

Moreover, some attributes require "internal" value instead of the "customer-facing" value to be persisted (e.g., sequencekitname, chipType). Please refer to the Comment/Expected Value column more details.

Validation Rules

RULE-1: Valid characters: letters, numbers, dashes, underscores, dots

RULE-2: Valid characters: letters, numbers, spaces, dashes, underscores, dots

RULE-3: Invalid leading characters: dashes, underscores, dots

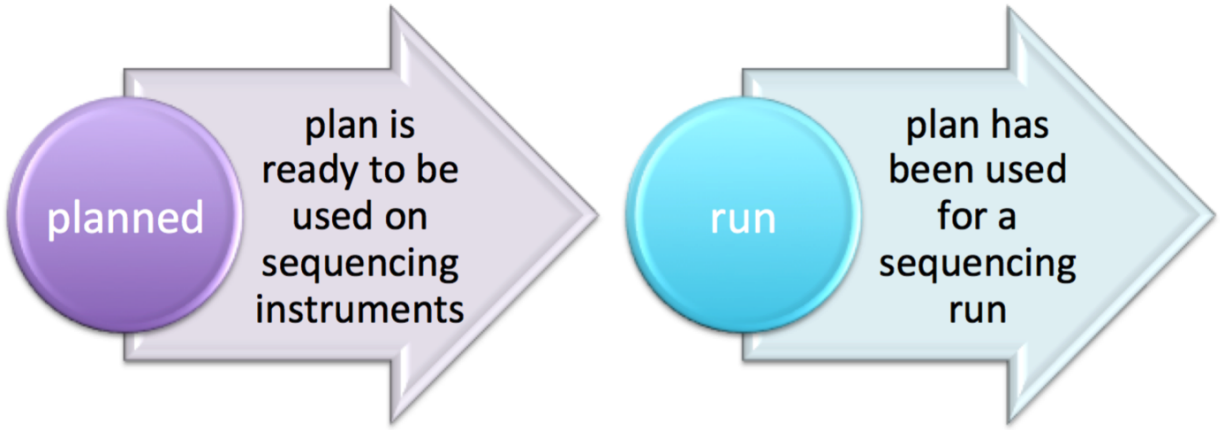
Field Notes

Attribute Name	Required/Optional/Nullable	Data type	Default value	Valid v
adapter	Opt/Nullable	varchar(256)		
applicationGroupDisplayedNamne	Opt/Nullable			DNA, I
autoAnalyze		Boolean	TRUE	
autoName	Opt/Nullable	varchar(512)		
barcodeId	Opt/Nullable	varchar(128)		
barcodedSamples	Opt/Nullable	json		
base_recalibrate	Opt	Boolean		
bedfile	Opt/Nullable	varchar(1024)		
chipBarcode	Opt/Nullable	varchar(64)		
chipType	Opt	varchar(32)		
controlSequencekitname	Opt/Nullable	varchar(512)		
cycles	Opt/Nullable	int		
date	Opt/Nullable	DateTimeField		
expName	Opt	varchar(128)		
flows	Req	int	0	
flowsInOrder	Opt/Nullable	varchar(512)		
forward3primeadapter	Req	varchar(512)		
id	Opt	int		
irworkflow	Opt	varchar(1024)		
isDuplicateReads	Opt	Boolean		
isFavorite	Opt	Boolean	FALSE	
isPlanGroup	Opt	Boolean	FALSE	
isReusable	Opt	Boolean	FALSE	
isReverseRun	Req	Boolean	FALSE	True,Fa
isSystem	Opt	Boolean	FALSE	
isSystemDefault	Opt	Boolean	FALSE	
libkit	Opt/Nullable	varchar(512)		
library	Opt/Nullable	varchar(512)		
libraryKey	Req	varchar(64)		
librarykitname	Opt/Nullable	varchar(512)		
metaData	Opt	json		
notes	Opt/Nullable	varchar(1024)		
pairedEndLibraryAdapterName	Opt/Nullable	varchar(512)		
parentPlan	Opt/Nullable	FK		
planDisplayedName		varchar(512)		
planExecuted	Opt	Boolean	FALSE	True,Fa

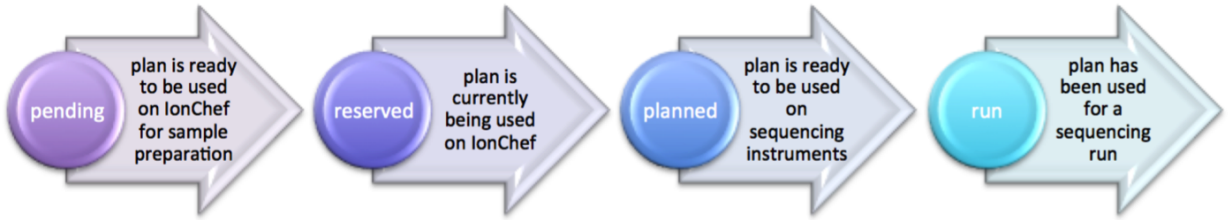
Attribute Name	Required/Optional/Nullable	Data type	Default value	Valid v
planExecutedDate	Opt/Nullable	DateTimeField		
planGUID	Opt/Nullable	varchar(512)		
planName		varchar(512)		
planPGM	Opt/Nullable	varchar(128)		
platform	Opt	varchar(128)	“”	“”, PGM
planShortID	Opt/Nullable			
planStatus		varchar(512)	planned	“”, pend
preAnalysis	Opt	Boolean		
projects	Opt	varchar(64) for each project name		
realign	Opt	Boolean		
regionfile	Opt/Nullable	varchar(1024)		
reverse_primer	Opt/Nullable	varchar(128)		
runMode	Opt	varchar(64)		“”,”sing
runType	Req	varchar(512)	GENS	“AMPS
runName	Opt/Nullable	varchar(255)		
sample	Required for plan	varchar(127)		
sampleDisplayedName	Opt/Nullable	varchar(127)		
sampleGroupingName	Opt/Nullable			DNA_F
samplePrepKitName	Opt/Nullable	varchar(512)		
sampleTubeLabel	Opt/Nullable	varchar(512)		
selectedPlugins	Opt/Nullable	json		
seqKitBarcode	Opt/Nullable	varchar(64)		
sequencekitname	Recommend to set	varchar(512)		
storageHost	Opt/Nullable	varchar(128)		
storage_options	Opt	varchar(200)	A	“KI”,”A
templatingKitName	Opt/Nullable	varchar(512)		
usePostBeadfind	Opt	Boolean		
usePreBeadfind	Opt	Boolean	TRUE	
username	Opt/Nullable	varchar(128)		

PlanStatus state transition

OneTouch



IonChef



barcodedSamples JSON Examples

Generic sequencing plan

```
"barcodedSamples": {
    "s 1": {
        "barcodeSampleInfo": {
            "IonSet1_16": {
                "controlSequenceType": "",
                "description": "desc 1",
                "externalId": "accession 101",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_des
            }
        },
        "barcodes": [
            "IonSet1_16"
        ]
    },
    "s 2": {
        "barcodeSampleInfo": {
            "IonSet1_12": {
                "controlSequenceType": "",
                "description": "desc 2",
                "externalId": "accession 80",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_des
            }
        },
        "barcodes": [
            "IonSet1_12"
        ]
    },
    "s 3": {
        "barcodeSampleInfo": {
            "IonSet1_15": {
                "controlSequenceType": "",
                "description": "desc 3",
```

```
                "externalId": "accession 280",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_desi
            }
        },
        "barcodes": [
            "IonSet1_15"
        ]
    }
},
```

Onconet DNA plan

```
"barcodedSamples": {
    "example 1": {
        "barcodeSampleInfo": {
            "IonXpress_010": {
                "controlSequenceType": "",
                "description": "example here",
                "externalId": "id 1",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            }
        },
        "barcodes": [
            "IonXpress_010"
        ]
    },
    "example 2": {
        "barcodeSampleInfo": {
            "IonXpress_005": {
                "controlSequenceType": "",
                "description": "another example here",
                "externalId": "id 2",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            }
        },
        "barcodes": [
            "IonXpress_005"
        ]
    }
},
```

Onconet DNA and Fusions plan

```
"barcodedSamples": {
    "s 1": {
        "barcodeSampleInfo": {
```

```
            "IonXpress_001": {
                "controlSequenceType": "",
                "description": "description here",
                "externalId": "ext 1",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            },
            "IonXpress_002": {
                "controlSequenceType": "",
                "description": "description here",
                "externalId": "ext 1",
                "hotSpotRegionBedFile": "",
                "nucleotideType": "RNA",
                "reference": "hg19_rna",
                "targetRegionBedFile": ""
            }
        },
        "barcodes": [
            "IonXpress_001",
            "IonXpress_002"
        ]
    }
}
```

selectedPlugins JSON Examples

IonReporterUploader, coverageAnalysis, sampleId and variantCaller

```
"selectedPlugins": {
    "IonReporterUploader": {
        "features": [
            "export"
        ],
        "id": 700,
        "name": "IonReporterUploader",
        "userInput": {
            "accountId": "1234567890abcde",
            "accountName": " demo IonReporter (Version: 4.2 | User: Ion User | Org: IR Org)",
            "userInputInfo": [{
                "ApplicationType": "Low-Coverage Whole Genome Sequencing",
                "Gender": "Female",
                "NucleotideType": "DNA",
                "Relation": "Self",
                "RelationRole": "Self",
                "Workflow": "Test_WK_1",
                "barcodeId": "IonXpress_010",
                "cancerType": "Breast Cancer",
                "cellularityPct": "23",
                "sample": "example 1",
                "sampleDescription": "example here",
                "sampleExternalId": "id 1",
                "sampleName": "example_1",
                "setid": "1__4c310e03-d188-4702-b82a-f9043bc04350"
            }, {
```

```
                "ApplicationType": "Low-Coverage Whole Genome Sequencing",
                "Gender": "Male",
                "NucleotideType": "DNA",
                "Relation": "",
                "RelationRole": "Self",
                "Workflow": "Test_WK_1",
                "barcodeId": "IonXpress_005",
                "cancerType": "Liver Cancer",
                "cellularityPct": "27",
                "sample": "example 2",
                "sampleDescription": "another example here",
                "sampleExternalId": "id 2",
                "sampleName": "example_2",
                "setid": "2__4c310e03-d188-4702-b82a-f9043bc04350"
            }]
        },
        "version": "4.2-r88003"
    },
    "coverageAnalysis": {
        "features": [],
        "id": 696,
        "name": "coverageAnalysis",
        "userInput": "",
        "version": "4.2-r87890"
    },
    "sampleID": {
        "features": [],
        "id": 701,
        "name": "sampleID",
        "userInput": "",
        "version": "4.2-r87942"
    },
    "variantCaller": {
        "features": [],
        "id": 699,
        "name": "variantCaller",
        "userInput": {
            "freebayes": {
                "allow_complex": "0",
                "allow_indels": "1",
                "allow_mnps": "0",
                "allow_snps": "1",
                "gen_min_alt_allele_freq": "0.03",
                "gen_min_coverage": "6",
                "gen_min_indel_alt_allele_freq": "0.1",
                "min_base_qv": "2",
                "min_mapping_qv": "4",
                "read_max_mismatch_fraction": "1.0",
                "read_mismatch_limit": "10"
            },
            "long_indel_assembler": {
                "kmer_len": "19",
                "max_hp_length": "8",
                "min_indel_size": "4",
                "min_var_count": "5",
                "min_var_freq": "0.15",
                "relative_strand_bias": "0.8",
                "short_suffix_match": "5"
```

```
    },
    "meta": {
        "built_in": true,
        "compatibility": {
            "chip": [
                "pgm",
                "proton_p1"
            ],
            "library": [
                "ampliseq"
            ],
            "panel": "/rundb/api/v1/contentupload/22/"
        },
        "configuration": "",
        "librarytype": "ampliseq",
        "name": "Panel-optimized - Colon and Lung Panel - 10/7/2013",
        "repository_id": "",
        "tooltip": "Panel-optimized parameters from AmpliSeq.com",
        "trimreads": true,
        "ts_version": "4.0",
        "tvcargs": "tvc",
        "user_selections": {
            "chip": "pgm",
            "frequency": "germline",
            "library": "ampliseq",
            "panel": "/rundb/api/v1/contentupload/22/"
        }
    },
    "torrent_variant_caller": {
        "data_quality_stringency": "6.5",
        "downsample_to_coverage": "10000",
        "filter_deletion_predictions": "0.2",
        "filter_insertion_predictions": "0.2",
        "filter_unusual_predictions": "0.3",
        "heavy_tailed": "3",
        "hotspot_beta_bias": "100.0",
        "hotspot_min_allele_freq": "0.01",
        "hotspot_min_cov_each_strand": "2",
        "hotspot_min_coverage": "6",
        "hotspot_min_variant_score": "6",
        "hotspot_strand_bias": "0.95",
        "hp_max_length": "8",
        "indel_beta_bias": "10.0",
        "indel_min_allele_freq": "0.05",
        "indel_min_cov_each_strand": "2",
        "indel_min_coverage": "15",
        "indel_min_variant_score": "6",
        "indel_strand_bias": "0.9",
        "outlier_probability": "0.01",
        "prediction_precision": "1.0",
        "snp_beta_bias": "100.0",
        "snp_min_allele_freq": "0.02",
        "snp_min_cov_each_strand": "0",
        "snp_min_coverage": "6",
        "snp_min_variant_score": "6",
        "snp_strand_bias": "0.95"
    }
    bbb
```

```
        },
        "version": "4.2-r87667"
    }
},
"seqKitBarcode": null,
"sequencekitname": "IonPGM200Kit-v2",
"storageHost": null,
"storage_options": "A",
"templatingKitBarcode": null,
"templatingKitName": "Ion PGM Template OT2 200 Kit",
"tfKey": "ATCG",
"thumbnailalignmentargs": "",
"thumbnailanalysisargs": "",
"thumbnailbasecallerargs": "",
"thumbnailbeadfindargs": "",
"thumbnailcalibrateargs": "",
"usePostBeadfind": true,
"usePreBeadfind": true,
"username": "ionadmin",
"variantfrequency": ""
},
```

IonReporterUploader selected for a Onconet DNA and Fusions plan

```
"selectedPlugins": {
    "IonReporterUploader": {
        "features": [
            "export"
        ],
        "id": 700,
        "name": "IonReporterUploader",
        "userInput": {
            "accountId": "1234567890abcde ",
            "accountName": "demo IonReporter (Version: 4.2 | User: Ion User | Org: IR Org)",
            "userInputInfo": [{
                "ApplicationType": "Oncomine_DNA_RNA_Fusion",
                "Gender": "Male",
                "NucleotideType": "DNA",
                "Relation": "DNA_RNA",
                "RelationRole": "Self",
                "Workflow": "AmpliSeq Colon Lung v2 with RNA Lung Fusion single sample",
                "barcodeId": "IonXpress_001",
                "cancerType": "Colorectal Cancer",
                "cellularityPct": "17",
                "sample": "s 1",
                "sampleDescription": "description here",
                "sampleExternalId": "ext 1",
                "sampleName": "s_1",
                "setid": "1__381a5a84-5af0-40ff-84c1-b31720fea6ca"
            }, {
                "ApplicationType": "Oncomine_DNA_RNA_Fusion",
                "Gender": "Male",
                "NucleotideType": "RNA",
                "Relation": "DNA_RNA",
                "RelationRole": "Self",
                "Workflow": "AmpliSeq Colon Lung v2 with RNA Lung Fusion single sample",
```

```
                "barcodeId": "IonXpress_002",
                "cancerType": "Colorectal Cancer",
                "cellularityPct": "17",
                "sample": "s 1",
                "sampleDescription": "description here",
                "sampleExternalId": "ext 1",
                "sampleName": "s_1",
                "setid": "1__381a5a84-5af0-40ff-84c1-b31720fea6ca"
            }]
        },
        "version": "4.2-r88003"
    }
},
```

Creating a plan

Non-barcoded PGM

Post a non-barcoded Target Sequencing PGM plan and to associate results with 2 projects with sampleGrouping and applicationGroup specified:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forw ard3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 500,
    "library": "hg19",
    "bedfile": "/results/uploads/BED/71/hg19/unmerged/detail/CFTRexon.20131001.designed.bed",
    "regionfile": "/results/uploads/BED/71/hg19/unmerged/detail/CFTRexon.20131125.hotspots.bed",
    "planName": "DEMO-TS4_2_x-REST- API_TARS_plan1",
    "sample": "my_sample",
    "notes": "this is a REST test plan",
    "username": "ionuser",
    "preAnalysis": "on",
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "TARS",
    "chipType": "318v2",
    "sequencekitname": "IonPGM200Kit",
    "librarykitname": "Ion Xpress Plus Fragment Library Kit",
    "templatingKitName": "Ion PGM Template OT2 200 Kit",
    "samplePrepKitName": "Ion TargetSeq(tm) Custom Enrichment Kit (100kb-500kb)",
    "projects": ["myProject1", "myProject2"],
    "sampleGroupingName": "Self",
    "applicationGroupDisplayedName": "DNA"
}
```

Non-Barcoded PI

Post a non-barcoded Target Sequencing Proton plan with PI chip, with sample tube label, chip barcode and the QC thresholds specified:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 440,
    "library": "hg19",
    "bedfile": "/results/uploads/BED/14/hg19/unmerged/detail/BRCA1_2.20131001.designed.bed",
    "regionfile": "/results/uploads/BED/14/hg19/unmerged/detail/BRCA1_2.20131001.hotspots.bed",
    "planName": "DEMO-TS4_2_x-REST-API_TARS_Proton_plan2",
    "sample": "my_sample",
    "notes": "here are my notes",
    "username": "ionuser",
    "preAnalysis": "on",
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "TARS",
    "chipType": "P1.1.17",
    "sequencekitname": "ProtonI200Kit-v3",
    "librarykitname": "Ion Xpress Plus Fragment Library Kit",
    "templatingKitName": "Ion PI Template OT2 200 Kit v3",
    "samplePrepKitName": "Ion TargetSeq(tm) Exome Kit (4 rxn)",
    "projects": ["myProject1"],
    "sampleTubeLabel": "abcX254",
    "chipBarcode": "AA02314571",
    "Bead Loading (%)": 33,
    "Key Signal (1-100)": 35,
    "Usable Sequence (%)": 37
}
```

Barcoded RNA PGM

Post a barcoded RNA Sequencing PGM plan:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 160,
    "library": "hg19_rna",
    "planName": "DEMO-TS4_2_x-REST- API_barcoded_RNA_plan3",
    "notes": "test notes here ",
    "username": "ionuser",
    "preAnalysis": "on",
```

```
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "RNA",
    "chipType": "318v2",
    "sequencekitname": "IonPGM200Kit-v2",
    "librarykitname": "Ion Total RNA Seq Kit v2",
    "templatingKitName": "Ion PGM Template OT2 200 Kit",
    "samplePrepKitName": "",
    "projects": ["myProject1", "myProject2"],
    "barcodedSamples": "{'demo sample 1':{'barcodeSampleInfo':{'IonXpressRNA_003':{'controlSequenceTy
    "applicationGroupDisplayedName": "RNA",
    "barcodeId": "IonXpressRNA",
    "sampleTubeLabel": "2554abc",
    "Bead Loading (%)": 30,
    "Key Signal (1-100)": 30,
    "Usable Sequence (%)": 30
}
```


Using POST to update a plan

If you are to update a plan via REST API, please perform a GET first so you'll have all the internally created values for the plan to perform the update with a POST.

To update with a POST, just include "id": <plan PK> in your data packet (e.g., "id":1234)

About using PUT or PATCH to update a plan

Update a plan for its chipBarcode value

`http://<hostname>/rundb/api/v1/plannedexperiment/<plan pk>/?format=json`

```
{
    "chipBarcode": "AA323323"
}
```

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: "Hello World"
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: "Hello World"
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: "Hello World"
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: "Hello World"
platform	Unicode string data. Ex: "Hello World"
categories	Unicode string data. Ex: "Hello World"
planPGM	Unicode string data. Ex: "Hello World"
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.

Table 2.17 – continued from previous page

field	help text
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”

Table 2.17 – continued from previous page

field	help text
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperiment/", params={"form
ts_api_response = ts_api_request.json()

plannedexperiments = ts_api_response["objects"]

for plannedexperiment in plannedexperiments:
    print plannedexperiment
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 18369,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "PROTON",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "",
            "sequencekitname": "ProtonI200Kit-v3",
            "storageHost": null,
            "expName": "R_2014_06_27_14_19_01_user_P19-606",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.0.20",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": null,
            "seqKitBarcode": null,
            "barcodeId": "",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/23975/",
            "bedfile": "",
            "isReusable": false,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=50,50 --bead
            "librarykitname": "Ion Xpress Plus Fragment Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
            "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "planStatus": "run",
            "chefLastUpdate": null,
            "samplePrepKitName": null,
```

```
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266571,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266571/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266570,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266570/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266569,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266569/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=80,
"thumbnailcalibrateargs": "calibrate --skipDroop",
"templatingKitName": "Ion PI Template OT2 200 Kit v3",
"runType": "GENS",
"username": null,
"planName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
```

```
            "sampleDisplayedName": "",
            "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
            "controlSequencekitname": null,
            "chefMessage": "",
            "childPlans": [],
            "pairedEndLibraryAdapterName": null,
            "runMode": "single",
            "irworkflow": "",
            "planExecuted": true,
            "project": "",
            "usePostBeadfind": false,
            "runname": null,
            "planGUID": "80384c15-e0e2-4909-9d4c-950731bf1cf4",
            "planShortID": "G5B50",
            "sampleSetGroupType": null,
            "sample": "",
            "planExecutedDate": null,
            "reverse_primer": null,
            "id": 102212,
            "barcodedSamples": {},
            "regionfile": "",
            "selectedPlugins": {},
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=80,112 --total-timeou
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 260,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regio
            "date": "2014-06-27T23:36:06.000229+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "TACGTACGTCTGAGCATCGATCGATGTACAGC",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/plannedexperiment/102212/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.50 Plannedexperimentdb Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/schema/`

Fields table

field	help text
isReverseRun	Boolean data. Ex: True
planDisplayedName	Unicode string data. Ex: “Hello World”
storage_options	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
isSystemDefault	Boolean data. Ex: True
username	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
planStatus	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
templatingKitBarcode	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
chefStatus	Unicode string data. Ex: “Hello World”
samplePrepKitName	Unicode string data. Ex: “Hello World”
reverse_primer	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
seqKitBarcode	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
sampleSet_planTotal	Integer data. Ex: 2673
experiment	A single related resource. Can be either a URI or set of nested resource data.
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
runType	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
planPGM	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
autoName	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
controlSequencekitname	Unicode string data. Ex: “Hello World”
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
libkit	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”

Table 2.18 – continued from previous page

field	help text
childPlans	A list of data. Ex: ['abc', 26.73, 8]
templatingKitName	Unicode string data. Ex: "Hello World"
pairedEndLibraryAdapterName	Unicode string data. Ex: "Hello World"
runMode	Unicode string data. Ex: "Hello World"
adapter	Unicode string data. Ex: "Hello World"
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
irworkflow	Unicode string data. Ex: "Hello World"
chipBarcode	Unicode string data. Ex: "Hello World"
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: "Hello World"
usePostBeadfind	Boolean data. Ex: True
storageHost	Unicode string data. Ex: "Hello World"
expName	Unicode string data. Ex: "Hello World"
runname	Unicode string data. Ex: "Hello World"
usePreBeadfind	Boolean data. Ex: True
planGUID	Unicode string data. Ex: "Hello World"
cycles	Integer data. Ex: 2673
resource_uri	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperimentdb/", params={"fo
ts_api_response = ts_api_request.json()

plannedexperimentdbs = ts_api_response["objects"]

for plannedexperimentdb in plannedexperimentdbs:
    print plannedexperimentdb
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18369,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperimentdb/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
            "storage_options": "A",
```

```
"preAnalysis": true,
"isSystemDefault": false,
"username": null,
"planShortID": "G5B50",
"planStatus": "run",
"chefLastUpdate": null,
"templatingKitBarcode": null,
"sampleTubeLabel": null,
"planExecutedDate": null,
"chefStatus": "",
"samplePrepKitName": null,
"reverse_primer": null,
"applicationGroup": "/rundb/api/v1/applicationgroup/1/",
"seqKitBarcode": null,
"id": 102212,
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266571,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266571/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266570,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266570/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266569,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
```

```
                    "defaultThreshold": 30,
                    "qcName": "Usable Sequence (%)",
                    "id": 3,
                    "resource_uri": "/rundb/api/v1/qctype/3/"
                },
                "resource_uri": "/rundb/api/v1/plannedexperimentqc/266569/"
            }
        ],
        "chefLogPath": null,
        "isPlanGroup": false,
        "sampleSet_planTotal": 0,
        "experiment": "/rundb/api/v1/experiment/23975/",
        "projects": [],
        "runType": "GENS",
        "sampleSet": null,
        "planPGM": null,
        "chefProgress": 0.0,
        "autoName": null,
        "isReusable": false,
        "controlSequencekitname": null,
        "date": "2014-06-27T23:36:06.000229+00:00",
        "isSystem": false,
        "libkit": null,
        "categories": "",
        "planName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
        "chefMessage": "",
        "parentPlan": null,
        "childPlans": [],
        "templatingKitName": "Ion PI Template OT2 200 Kit v3",
        "pairedEndLibraryAdapterName": null,
        "runMode": "single",
        "adapter": null,
        "sampleGrouping": null,
        "irworkflow": "",
        "chipBarcode": null,
        "planExecuted": true,
        "project": "",
        "usePostBeadfind": false,
        "storageHost": null,
        "expName": "R_2014_06_27_14_19_01_user_P19-606",
        "runname": null,
        "usePreBeadfind": true,
        "planGUID": "80384c15-e0e2-4909-9d4c-950731bf1cf4",
        "cycles": null,
        "resource_uri": "/rundb/api/v1/plannedexperimentdb/102212/"
    }
]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.51 Plannedexperimentqc Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
threshold	Integer data. Ex: 2673	0	false	false	false	false	integer
plannedExperiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
qcType	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperimentqc/", params={"fo
ts_api_response = ts_api_request.json()

plannedexperimentqcs = ts_api_response["objects"]

for plannedexperimentqc in plannedexperimentqcs:
    print plannedexperimentqc
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 22524,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperimentqc/?offset=1&limit=1&format=json"
```

```
    },
    "objects": [
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/86049/",
            "id": 247513,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/247513/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.52 Plantemplatebasicinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/schema/`

Fields table

field	help text	default	null
isReverseRun	Boolean data. Ex: True	false	false
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true
storage_options	Unicode string data. Ex: “Hello World”	A	false
preAnalysis	Boolean data. Ex: True		false
reference	Unicode string data. Ex: “Hello World”		true
isSystemDefault	Boolean data. Ex: True	false	false
hotSpotRegionBedFile	Unicode string data. Ex: “Hello World”		true
planShortID	Unicode string data. Ex: “Hello World”	n/a	true
planStatus	Unicode string data. Ex: “Hello World”		false
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true

Table 2.19 – continued from previous page

field	help text	default	null
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
chefStatus	Unicode string data. Ex: “Hello World”		false
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true
id	Integer data. Ex: 2673		false
metaData	Unicode string data. Ex: “Hello World”	{}	false
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true
isFavorite	Boolean data. Ex: True	false	false
sampleSet_planIndex	Integer data. Ex: 2673	0	false
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true
isPlanGroup	Boolean data. Ex: True	false	false
sampleGroupName	Unicode string data. Ex: “Hello World”	n/a	true
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true
barcodeKitName	Unicode string data. Ex: “Hello World”	n/a	true
runType	Unicode string data. Ex: “Hello World”	GENS	false
planPGM	Unicode string data. Ex: “Hello World”	n/a	true
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false
autoName	Unicode string data. Ex: “Hello World”	n/a	true
isReusable	Boolean data. Ex: True	false	false
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true
irAccountName	Unicode string data. Ex: “Hello World”	n/a	true
sequencingInstrumentType	Unicode string data. Ex: “Hello World”	n/a	true
targetRegionBedFile	Unicode string data. Ex: “Hello World”		true
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
isSystem	Boolean data. Ex: True	false	false
libkit	Unicode string data. Ex: “Hello World”	n/a	true
categories	Unicode string data. Ex: “Hello World”		true
planName	Unicode string data. Ex: “Hello World”	n/a	true
chefMessage	Unicode string data. Ex: “Hello World”		false
templatePrepInstrumentType	Unicode string data. Ex: “Hello World”	n/a	true
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true
runMode	Unicode string data. Ex: “Hello World”		false
adapter	Unicode string data. Ex: “Hello World”	n/a	true
irworkflow	Unicode string data. Ex: “Hello World”		false
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true
planExecuted	Boolean data. Ex: True	false	false
username	Unicode string data. Ex: “Hello World”	n/a	true
usePostBeadfind	Boolean data. Ex: True		false
storageHost	Unicode string data. Ex: “Hello World”	n/a	true
expName	Unicode string data. Ex: “Hello World”		false
runname	Unicode string data. Ex: “Hello World”	n/a	true
usePreBeadfind	Boolean data. Ex: True		false
planGUID	Unicode string data. Ex: “Hello World”	n/a	true
cycles	Integer data. Ex: 2673	n/a	true
notes	Unicode string data. Ex: “Hello World”		true
sampleSet_planTotal	Integer data. Ex: 2673	0	false

Table 2.19 – continued from previous page

field	help text	default	null
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/", params={'
ts_api_response = ts_api_request.json()

plantemplatebasicinfos = ts_api_response["objects"]

for plantemplatebasicinfo in plantemplatebasicinfos:
    print plantemplatebasicinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 66,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plantemplatebasicinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "reference": "hg19",
            "isSystemDefault": false,
            "hotSpotRegionBedFile": "",
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "applicationGroupDisplayedName": "DNA",
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
```

```
            "sampleSet_planIndex": 0,
            "seqKitBarcode": null,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleGroupName": "",
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "barcodeKitName": "IonXpress",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "irAccountName": "None",
            "sequencingInstrumentType": "PROTON",
            "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "templatePrepInstrumentType": "OneTouch",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sampleSet_planTotal": 0,
            "resource_uri": "/rundb/api/v1/plantemplatebasicinfo/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get

2.1.53 Plantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr

Table 2.20 – continued from previous page

field	help text	default	nullable	readonly	b
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plantemplatesummary/", params={"fo
ts_api_response = ts_api_request.json()

plantemplatesummarys = ts_api_response["objects"]

for plantemplatesummary in plantemplatesummarys:
    print plantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 66,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
```

```
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/plantemplatesummary/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.54 Plugin Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plugin/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plugin/schema/`

Perform read-only operations on `plugin` resources and data elements

Fields table

field	help text	default	nullable	read-only	blank	unique	type
active	Boolean data. Ex: True	true	false	false	true	false	boolean
id	Integer data. Ex: 2673		false	false	true	true	integer
isPlanConfig	Boolean data. Ex: True	n/a	false	true	false	false	boolean
autorunMutable	Boolean data. Ex: True	true	false	false	true	false	boolean
script	Unicode string data. Ex: "Hello World"		false	false	true	false	string
selected	Boolean data. Ex: True	false	false	false	true	false	boolean
version	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
hasAbout	Boolean data. Ex: True	n/a	false	true	false	false	boolean
input	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
majorBlock	Boolean data. Ex: True	false	false	false	true	false	boolean
status	Unicode string data. Ex: "Hello World"		true	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
autorun	Boolean data. Ex: True	false	false	false	true	false	boolean
pluginsettings	Unicode string data. Ex: "Hello World"		true	false	false	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
isConfig	Boolean data. Ex: True	n/a	false	true	false	false	boolean
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
userinputfields	Unicode string data. Ex: "Hello World"	{}	true	false	false	false	string
url	Unicode string data. Ex: "Hello World"		false	false	true	false	string
config	Unicode string data. Ex: "Hello World"		true	false	false	false	string
versionedName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
isInstance	Boolean data. Ex: True	n/a	false	true	false	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plugin/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plugin/", params={"format": "json"
ts_api_response = ts_api_request.json()

plugins = ts_api_response["objects"]

for plugin in plugins:
    print plugin
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 102,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plugin/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "active": true,
            "id": 493,
            "isPlanConfig": false,
            "autorunMutable": true,
            "script": "launch.sh",
            "selected": true,
            "version": "0.2.0",
            "hasAbout": false,
            "input": "False",
            "majorBlock": false,
            "status": {},
            "description": "Ion Torrent Plugin - 'BarcodeAlignStats' v0.2.0",
            "autorun": false,
            "pluginsettings": {
                "runtype": [
                    "wholechip",
                    "thumbnail"
                ],
                "depends": [],
                "features": [],
                "runlevel": []
            },
            "date": "2013-05-30T21:32:15.000437+00:00",
            "path": "/results/plugins/BarcodeAlignStats",
            "isConfig": false,
            "name": "BarcodeAlignStats",
            "userinputfields": {},
            "url": "",
            "config": {},
            "versionedName": "BarcodeAlignStats--v0.2.0",
            "isInstance": false,
            "resource_uri": "/rundb/api/v1/plugin/493/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

2.1.55 Pluginresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/pluginresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/pluginresult/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
size	Unicode string data. Ex: "Hello World"	-1	false	false	false	false	string
apikey	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
plugin	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resultName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
pluginVersion	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
jobid	Integer data. Ex: 2673	n/a	true	false	false	false	integer
owner	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
pluginName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
state	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
result	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
starttime	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	true	false	false	false	datetime
duration	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
path	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
store	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
endtime	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	true	false	false	false	datetime
config	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
inodes	Unicode string data. Ex: "Hello World"	-1	false	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/pluginresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/pluginresult/", params={"format":
ts_api_response = ts_api_request.json()

pluginresults = ts_api_response["objects"]

for pluginresult in pluginresults:
    print pluginresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 788680,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/pluginresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "size": "3484344",
            "apikey": null,
            "plugin": "/rundb/api/v1/plugin/823/",
            "resultName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
            "pluginVersion": "4.2-r88266",
            "jobid": 3145172,
            "owner": "/rundb/api/v1/user/1/",
            "pluginName": "AssemblerSPAdes",
            "state": "Completed",
            "result": "/rundb/api/v1/results/304393/",
            "starttime": "2014-06-28T13:49:00.000133+00:00",
            "duration": "0:01:54.098431",
            "path": "/results/analysis/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/p
            "store": {},
            "endtime": "2014-06-28T13:50:54.000232+00:00",
            "config": {
                "only_barcodes": "",
                "spadesOptions": "-k 21,33,55,77,99",
                "spadesversion": "3.1.0",
                "RAM": "32G",
                "min_reads": "500",
                "bgenome": "None",
                "runSpades": "1",
                "fraction_of_reads": "1"
            },
            "id": 815369,
            "inodes": "396",
            "resource_uri": "/rundb/api/v1/pluginresult/815369/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.56 Project Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/project/`

Schema URL: `http://mytorrentserver/rundb/api/v1/project/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
creator	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
modified	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resultsCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
public	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/project/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/project/", params={"format": "jso
ts_api_response = ts_api_request.json()

projects = ts_api_response["objects"]
```

```
for project in projects:
    print project
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 1457,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/project/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "3gb_snapps",
            "creator": "/rundb/api/v1/user/2/",
            "created": "2012-07-03T15:14:33.000509+00:00",
            "results": [
                "/rundb/api/v1/results/2305/",
                "/rundb/api/v1/results/1499/",
                "/rundb/api/v1/results/2304/",
                "/rundb/api/v1/results/1511/",
                "/rundb/api/v1/results/30467/",
                "/rundb/api/v1/results/30458/",
                "/rundb/api/v1/results/30457/",
                "/rundb/api/v1/results/30490/",
                "/rundb/api/v1/results/30464/",
                "/rundb/api/v1/results/30459/",
                "/rundb/api/v1/results/30460/",
                "/rundb/api/v1/results/30496/",
                "/rundb/api/v1/results/3298/",
                "/rundb/api/v1/results/30463/",
                "/rundb/api/v1/results/30446/",
                "/rundb/api/v1/results/30489/",
                "/rundb/api/v1/results/30466/",
                "/rundb/api/v1/results/30456/",
                "/rundb/api/v1/results/30491/",
                "/rundb/api/v1/results/2609/",
                "/rundb/api/v1/results/1466/",
                "/rundb/api/v1/results/1498/",
                "/rundb/api/v1/results/2300/",
                "/rundb/api/v1/results/1497/"
            ],
            "modified": "2012-07-03T15:14:33.000509+00:00",
            "id": 1,
            "resultsCount": 24,
            "public": true,
            "resource_uri": "/rundb/api/v1/project/1/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.57 Projectresults Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/projectresults/`

Schema URL: `http://mytorrentserver/rundb/api/v1/projectresults/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
reference	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
runid	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
metaData	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
log	Unicode string data. Ex: "Hello World"		false	false	true	false	string
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
processedCycles	Integer data. Ex: 2673	n/a	false	false	false	false	integer
sffLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
representative	Boolean data. Ex: True	false	false	false	true	false	boolean
tfSffLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
diskusage	Integer data. Ex: 2673	n/a	true	false	false	false	integer
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
resultsType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
tfFastq	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
parentIDs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
timeToComplete	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
fastqLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
framesProcessed	Integer data. Ex: 2673	n/a	false	false	false	false	integer
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
analysisVersion	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/projectresults/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/projectresults/", params={"format"
ts_api_response = ts_api_request.json()

projectresultss = ts_api_response["objects"]

for projectresults in projectresultss:
    print projectresults
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/projectresults/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "reference": "hg19",
            "reportStatus": "Nothing",
            "runid": "DGMU8",
            "id": 293943,
            "metaData": {},
            "log": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/log.htm
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "status": "Completed",
            "processedflows": 0,
            "processedCycles": 0,
            "sffLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R_2
            "representative": false,
            "tfSffLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R
            "diskusage": 151,
            "projects": [
                {
                    "name": "chef_827_909_20min_ext",
                    "creator": "/rundb/api/v1/user/1/",
                    "created": "2014-01-22T18:50:10.000920+00:00",
                    "results": [
                        "/rundb/api/v1/results/294145/",
                        "/rundb/api/v1/results/293993/",
                        "/rundb/api/v1/results/293992/",
                        "/rundb/api/v1/results/293991/",
                        "/rundb/api/v1/results/293990/",
                        "/rundb/api/v1/results/293946/",
```

```
                        "/rundb/api/v1/results/293945/",
                        "/rundb/api/v1/results/293944/",
                        "/rundb/api/v1/results/293943/",
                        "/rundb/api/v1/results/293934/",
                        "/rundb/api/v1/results/293933/",
                        "/rundb/api/v1/results/293930/",
                        "/rundb/api/v1/results/293929/",
                        "/rundb/api/v1/results/293928/",
                        "/rundb/api/v1/results/293927/",
                        "/rundb/api/v1/results/293917/",
                        "/rundb/api/v1/results/293916/",
                        "/rundb/api/v1/results/293915/",
                        "/rundb/api/v1/results/293914/",
                        "/rundb/api/v1/results/293913/",
                        "/rundb/api/v1/results/293912/",
                        "/rundb/api/v1/results/293908/",
                        "/rundb/api/v1/results/293907/",
                        "/rundb/api/v1/results/293906/",
                        "/rundb/api/v1/results/293905/",
                        "/rundb/api/v1/results/293904/",
                        "/rundb/api/v1/results/293903/",
                        "/rundb/api/v1/results/293902/",
                        "/rundb/api/v1/results/293901/"
                    ],
                    "modified": "2014-01-22T18:50:10.000920+00:00",
                    "id": 1080,
                    "resultsCount": 29,
                    "public": true,
                    "resource_uri": "/rundb/api/v1/project/1080/"
                }
            ],
            "resultsType": "",
            "tfFastq": "_",
            "parentIDs": "",
            "timeToComplete": "0",
            "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
            "fastqLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/k
            "resource_uri": "/rundb/api/v1/projectresults/293943/",
            "framesProcessed": 0,
            "autoExempt": false,
            "analysisVersion": "db:4.1.21+2-1,an:4.1.24+0-1,"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.58 Qctype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/qctype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/qctype/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
minThreshold	Integer data. Ex: 2673	0	false	false	false	false	integer
maxThreshold	Integer data. Ex: 2673	100	false	false	false	false	integer
defaultThreshold	Integer data. Ex: 2673	0	false	false	false	false	integer
qcName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/qctype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/qctype/", params={"format": "json"
ts_api_response = ts_api_request.json()

qctypes = ts_api_response["objects"]

for qctype in qctypes:
    print qctype
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 3,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/qctype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
```

```
            }
        ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.59 Qualitymetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/schema/`

Perform read-only operations on `qualitymetrics` resources and data elements.

Fields table

field	help text	default	nullable
q0_reads	Integer data. Ex: 2673	n/a	false
q17_max_read_length	Integer data. Ex: 2673	n/a	false
q20_median_read_length	Integer data. Ex: 2673	0	false
q20_reads	Integer data. Ex: 2673	n/a	false
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false
q17_mean_read_length	Floating point numeric data. Ex: 26.73	n/a	false
q17_100bp_reads	Integer data. Ex: 2673	n/a	false
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false
q0_max_read_length	Integer data. Ex: 2673	n/a	false
q20_100bp_reads	Integer data. Ex: 2673	n/a	false
id	Integer data. Ex: 2673		false
q20_mean_read_length	Integer data. Ex: 2673	n/a	false
q20_150bp_reads	Integer data. Ex: 2673	n/a	false
q0_bases	Unicode string data. Ex: “Hello World”	n/a	false
q20_50bp_reads	Integer data. Ex: 2673	n/a	false
q17_reads	Integer data. Ex: 2673	n/a	false
q17_50bp_reads	Integer data. Ex: 2673	n/a	false
q17_median_read_length	Integer data. Ex: 2673	0	false
q0_50bp_reads	Integer data. Ex: 2673	n/a	false
q17_150bp_reads	Integer data. Ex: 2673	n/a	false
q0_150bp_reads	Integer data. Ex: 2673	0	false
q0_mean_read_length	Floating point numeric data. Ex: 26.73	n/a	false
q17_bases	Unicode string data. Ex: “Hello World”	n/a	false
q0_mode_read_length	Integer data. Ex: 2673	0	false
q20_mode_read_length	Integer data. Ex: 2673	0	false

Table 2.21 – continued from previous page

field	help text	default	nullable	
q20_max_read_length	Floating point numeric data. Ex: 26.73	n/a	false	f
q20_bases	Unicode string data. Ex: “Hello World”	n/a	false	f
q0_median_read_length	Integer data. Ex: 2673	0	false	f
q0_100bp_reads	Integer data. Ex: 2673	n/a	false	f
q17_mode_read_length	Integer data. Ex: 2673	0	false	f

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/qualitymetrics/", params={"format"
ts_api_response = ts_api_request.json()

qualitymetricss = ts_api_response["objects"]

for qualitymetrics in qualitymetricss:
    print qualitymetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 37351,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/qualitymetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "q0_reads": 0,
            "q17_max_read_length": 0,
            "q20_median_read_length": 0,
            "q20_reads": 0,
            "report": "/rundb/api/v1/results/269819/",
            "q17_mean_read_length": 0.0,
            "q17_100bp_reads": 0,
            "resource_uri": "/rundb/api/v1/qualitymetrics/9943/",
            "q0_max_read_length": 0,
            "q20_100bp_reads": 0,
            "id": 9943,
            "q20_mean_read_length": 0,
            "q20_150bp_reads": 0,
            "q0_bases": "0",
            "q20_50bp_reads": 0,
            "q17_reads": 0,
            "q17_50bp_reads": 0,
```

```
            "q17_median_read_length": 0,
            "q0_50bp_reads": 0,
            "q17_150bp_reads": 0,
            "q0_150bp_reads": 0,
            "q0_mean_read_length": 0.0,
            "q17_bases": "0",
            "q0_mode_read_length": 0,
            "q20_mode_read_length": 0,
            "q20_max_read_length": 0.0,
            "q20_bases": "0",
            "q0_median_read_length": 0,
            "q0_100bp_reads": 0,
            "q17_mode_read_length": 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.60 Referencegenome Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/referencegenome/`

Schema URL: `http://mytorrentserver/rundb/api/v1/referencegenome/schema/`

Perform read-only operations on `referencegenome` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
reference_path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
short_name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
index_version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
notes	Unicode string data. Ex: "Hello World"		false	false	true	false	string
enabled	Boolean data. Ex: True	true	false	false	true	false	boolean
species	Unicode string data. Ex: "Hello World"		false	false	true	false	string
identity_hash	Unicode string data. Ex: "Hello World"	None	true	false	false	false	string
source	Unicode string data. Ex: "Hello World"		false	false	true	false	string
version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
celery_task_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	2014-06-28T14:05:04.000158+00:00	false	false	false	false	datetime
verbose_error	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/referencegenome/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/referencegenome/", params={"format
ts_api_response = ts_api_request.json()

referencegenomes = ts_api_response["objects"]

for referencegenome in referencegenomes:
    print referencegenome
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/referencegenome/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "complete",
            "reference_path": "/results/referenceLibrary/tmap-f3/CFTR_38amp_v2",
            "name": "CFTR_38amp_v2",
            "short_name": "CFTR_38amp_v2",
            "index_version": "tmap-f3",
            "notes": "",
            "enabled": true,
            "species": "",
            "identity_hash": null,
            "source": "",
            "version": "CFTR_38amp_v2",
            "celery_task_id": "",
            "date": "2012-03-21T12:32:00.000382+00:00",
            "verbose_error": "[\"\", \"\\nSequence name 'CFTR.13.120s' contains a non-alphanumeric ch
            "id": 7,
            "resource_uri": "/rundb/api/v1/referencegenome/7/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.61 Results Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/results/`

Schema URL: `http://mytorrentserver/rundb/api/v1/results/schema/`

Perform CRUD operations on `results` resources and data elements.

Fields table

field	help text	default	nul
reference	Unicode string data. Ex: “Hello World”	n/a	true
processedflows	Integer data. Ex: 2673	n/a	fals
reportStatus	Unicode string data. Ex: “Hello World”	Nothing	true
reportstorage	A single related resource. Can be either a URI or set of nested resource data.	n/a	fals
analysisVersion	Unicode string data. Ex: “Hello World”	n/a	fals
runid	Unicode string data. Ex: “Hello World”		fals
id	Integer data. Ex: 2673		fals
filesystempath	Unicode string data. Ex: “Hello World”	n/a	fals
metaData	Unicode string data. Ex: “Hello World”	{}	fals
log	Unicode string data. Ex: “Hello World”		fals
timeStamp	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	fals
libmetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
resultsName	Unicode string data. Ex: “Hello World”	n/a	fals
status	Unicode string data. Ex: “Hello World”	n/a	fals
planShortID	Unicode string data. Ex: “Hello World”	n/a	fals
processedCycles	Integer data. Ex: 2673	n/a	fals
bamLink	Unicode string data. Ex: “Hello World”	n/a	fals
sffLink	Unicode string data. Ex: “Hello World”	n/a	true
representative	Boolean data. Ex: True	false	fals
pluginState	A dictionary of data. Ex: {‘price’: 26.73, ‘name’: ‘Daniel’}	n/a	fals
qualitymetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
diskusage	Integer data. Ex: 2673	n/a	true
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
tfSffLink	Unicode string data. Ex: “Hello World”	n/a	true
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
pluginStore	A dictionary of data. Ex: {‘price’: 26.73, ‘name’: ‘Daniel’}	n/a	fals
resultsType	Unicode string data. Ex: “Hello World”		fals
tfFastq	Unicode string data. Ex: “Hello World”	n/a	fals
tfmetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
parentIDs	Unicode string data. Ex: “Hello World”		fals
analysismetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
timeToComplete	Unicode string data. Ex: “Hello World”	n/a	fals
reportLink	Unicode string data. Ex: “Hello World”	n/a	fals
fastqLink	Unicode string data. Ex: “Hello World”	n/a	fals
pluginresults	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
framesProcessed	Integer data. Ex: 2673	n/a	fals
autoExempt	Boolean data. Ex: True	false	fals
resource_uri	Unicode string data. Ex: “Hello World”	n/a	fals

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/results/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/results/", params={"format": "jso
ts_api_response = ts_api_request.json()
```

```
resultss = ts_api_response["objects"]

for results in resultss:
    print results
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/results/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "reference": "hg19",
            "processedflows": 0,
            "reportStatus": "Nothing",
            "reportstorage": {
                "name": "Home",
                "default": true,
                "webServerPath": "/output",
                "dirPath": "/results/analysis/output",
                "id": 1,
                "resource_uri": ""
            },
            "analysisVersion": "db:4.1.21+2-1,an:4.1.24+0-1,",
            "runid": "DGMU8",
            "id": 293943,
            "filesystempath": "/results/analysis/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m
            "metaData": {},
            "log": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/log.htm
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "libmetrics": [
                "/rundb/api/v1/libmetrics/32368/"
            ],
            "experiment": "/rundb/api/v1/experiment/17446/",
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "status": "Completed",
            "planShortID": "ONPK8",
            "processedCycles": 0,
            "bamLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R_2
            "sffLink": null,
            "representative": false,
            "pluginState": {
                "coverageAnalysisLite": "Completed",
                "SystematicErrorAnalysis": "Completed",
                "1_Torrent_Accuracy": "Error",
                "duplicateReads_useZC": "Completed",
                "autoCal": "Completed",
                "variantCaller": "Completed",
                "ConversionRate": "Completed",
                "validateVariantCaller": "Completed",
                "timingPerformance": "Completed",
```

```
        "coverageAnalysis": "Completed",
        "VariantQC": "Completed"
    },
    "qualitymetrics": [
        "/rundb/api/v1/qualitymetrics/31678/"
    ],
    "diskusage": 151,
    "eas": "/rundb/api/v1/experimentanalysissettings/18714/",
    "tfSffLink": null,
    "projects": [
        "/rundb/api/v1/project/1080/"
    ],
    "pluginStore": {
        "coverageAnalysisLite": {
            "Non-duplicate": "",
            "barcoded": "true",
            "Uniquely mapped": "No",
            "Targetted regions": "/results/uploads/BED/46/hg19/merged/plain/AmpliSeqExome.201
            "Target padding": "0",
            "barcodes": {
                "IonXpress_033": {
                    "Bases in target regions": "57742646",
                    "Number of mapped reads": "41517304",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Percent reads on target": "94.39%",
                    "Average base coverage depth": "112.4",
                    "Reference (File)": "hg19",
                    "Coverage Analysis Lite Report": "N/A",
                    "Target base coverage at 100x": "51.05%",
                    "Target base coverage at 20x": "94.26%",
                    "Uniformity of base coverage": "93.56%",
                    "Target base coverage at 1x": "98.53%",
                    "Using": "All Mapped Reads",
                    "Target base coverage at 500x": "0.10%",
                    "Alignments": "IonXpress_033_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Total base reads on target": "6490060189"
                },
                "IonXpress_034": {
                    "Bases in target regions": "57742646",
                    "Number of mapped reads": "48026110",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Percent reads on target": "94.01%",
                    "Average base coverage depth": "130.8",
                    "Reference (File)": "hg19",
                    "Coverage Analysis Lite Report": "N/A",
                    "Target base coverage at 100x": "61.01%",
                    "Target base coverage at 20x": "94.93%",
                    "Uniformity of base coverage": "93.55%",
                    "Target base coverage at 1x": "98.55%",
                    "Using": "All Mapped Reads",
                    "Target base coverage at 500x": "0.22%",
                    "Alignments": "IonXpress_034_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Total base reads on target": "7552568443"
                }
            }
        },
        "SystematicErrorAnalysis": {
            "qts_peak": "NaN",
```

```
        "indel-5-per-mb": "10626.700000000001",
        "positions-with-sse": "0.0080719999999999993",
        "qts_base": "NaN",
        "barcoded": "true",
        "positions-with-sse-d15": "0.0080719999999999993",
        "Target-regions_file": "/results/uploads/BED/46/hg19/merged/plain/AmpliSeqExome.2
        "stb-95-per-mb": "67945.399999999994",
        "positions-with-sse-i5": "0.0073000000000000001",
        "positions-with-sse-d5": "0.023935000000000001",
        "positions-with-sse-i15": "0.0018799999999999999",
        "indel-15-per-mb": "3118.8000000000002",
        "barcodes": {
            "IonXpress_033": {
                "indel-5-per-mb": "10600.3",
                "positions-with-sse": "0.007770",
                "positions-with-sse-d15": "0.007770",
                "stb-95-per-mb": "68851.0",
                "positions-with-sse-i5": "0.007519",
                "positions-with-sse-d5": "0.022925",
                "positions-with-sse-i15": "0.001964",
                "indel-15-per-mb": "3126.1"
            },
            "IonXpress_034": {
                "indel-5-per-mb": "10653.1",
                "positions-with-sse": "0.008373",
                "positions-with-sse-d15": "0.008373",
                "stb-95-per-mb": "67039.8",
                "positions-with-sse-i5": "0.007080",
                "positions-with-sse-d5": "0.024945",
                "positions-with-sse-i15": "0.001796",
                "indel-15-per-mb": "3111.5"
            }
        }
    },
    "1_Torrent_Accuracy": {},
    "duplicateReads_useZC": {
        "adapter_found_rate_chr1": "0.89400000000000002",
        "duplicate_reads_chr1": 673650,
        "duprate_at_725k_chr1": "0.89300000000000002",
        "duplicate_rate_chr1": "0.89200000000000002",
        "total_reads_chr1": 754826
    },
    "autoCal": {
        "dc_range": 0
    },
    "variantCaller": {
        "barcodes": {
            "IonXpress_033": {
                "hotspots": {},
                "variants": {
                    "no_call": 0,
                    "homo_snps": 18047,
                    "het_snps": 31409,
                    "other": 1321,
                    "variants": 54343,
                    "het_indels": 2444,
                    "homo_indels": 1122
                }
```

```
            },
            "IonXpress_034": {
                "hotspots": {},
                "variants": {
                    "no_call": 0,
                    "homo_snps": 18134,
                    "het_snps": 31524,
                    "other": 1308,
                    "variants": 54522,
                    "het_indels": 2422,
                    "homo_indels": 1134
                }
            }
        },
        "barcoded": "true",
        "targets_bed": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131
        "Target Regions": "AmpliSeqExome.20131001.designed",
        "Trim Reads": true,
        "Target Loci": "Not using",
        "Configuration": "Germ Line - Proton - Low Stringency",
        "Aligned Reads": "R_2014_01_22_16_30_23_user_D1--632--R54651-p8s2_827b2_20m_man-
        "Library Type": "AmpliSeq"
    },
    "ConversionRate": {},
    "validateVariantCaller": {
        "SNP_FP-ConfidentPos": 5410,
        "InDel_AmbPos-AllPos": 0,
        "SNP_PPV>=30x": "96.7228784857032",
        "InDel_FN>=100x": 930,
        "SNP_FN>=100x": 896,
        "InDel_TP-AllPos": 2591,
        "InDel_FP>=30x": 1689,
        "SNP_FN>=30x": 2004,
        "SNP_FP>=1000x": 0,
        "SNP_ConsensusAccuracy-AllPos": "0.999914181450164",
        "InDel_NoCalls-AllPos": 894775,
        "InDel_FP-AllPos": 2681,
        "InDel_FP_50x-100x": 695,
        "InDel_Sensitivity>=20x": "47.7212806026365",
        "SNP_TP>=500x": 132,
        "SNP_Sensitivity>=100x": "98.2144280589877",
        "SNP_Sensitivity>=500x": "99.2481203007519",
        "InDel_ConsensusAccuracy-AllPos": "0.999949885886992",
        "SNP_ConsensusAccuracy>=50x": "0.999971537775062",
        "InDel_FN>=20x": 2776,
        "SNP_FP>=50x": 1762,
        "InDel_PPV-AllPos": "49.1464339908953",
        "InDel_ConsensusAccuracy>=30x": "0.999964918459836",
        "InDel_FP-ncRNA": 18,
        "InDel_ReferenceCalls-AllPos": 0,
        "Target-regions_file": "/results/analysis/output/Home/Auto_user_D1--632--R54651-p
        "SNP_TP-AllPos": 87058,
        "InDel_Sensitivity>=30x": "49.0349819059107",
        "InDel_FN>=500x": 3,
        "InDel_FP_20x-50x": 827,
        "InDel_FP-ConfidentPos": 2681,
        "InDel_FN-AllPos": 3353,
        "SNP_HP11-AllPos": 0,
```

```
"InDel_FN>=50x": 2041,
"SNP_Sensitivity-AllPos": "94.6478076994162",
"Truth-major_SNP_file": "NA12878_NIST_NoChrY_SNP.bed",
"InDel_Sensitivity-AllAnnotations": "87.1800314804519",
"SNP_FP_200x-300x": 84,
"InDel_FP>=20x": 2020,
"SNP_FN>=500x": 1,
"Combined Variant Positive Predictive Value for All Bases at >=20x": "94.0001",
"SNP_Sensitivity>=1000x": 100,
"InDel_FP_700x-1000x": 0,
"SNP_FP>=30x": 2777,
"SNP_FP_700x-1000x": 0,
"SNP_ConsensusAccuracy>=20x": "0.999952352557784",
"InDel_TP>=30x": 2439,
"SNP_AmbPos-AllPos": 0,
"Truth-minor_InDel_file": "None",
"SNP_FP_500x-700x": 1,
"SNP_FP_50x-100x": 1095,
"InDel_ConsensusAccuracy>=20x": "0.999960167834606",
"InDel_FP-Exons": 1062,
"SNP_TP>=30x": 81962,
"SNP_ReferenceCalls-AllPos": 0,
"InDel_FP_500x-700x": 1,
"InDel_FP_200x-300x": 65,
"barcoded": "true",
"SNP_FP_20x-50x": 1773,
"SNP_FP>=100x": 667,
"SNP_Sensitivity>=20x": "97.4601197273262",
"InDel_FP_300x-400x": 9,
"SNP_FP_100x-200x": 565,
"barcodes": {
    "IonXpress_033": {
        "SNP_FP-ConfidentPos": "2733",
        "InDel_AmbPos-AllPos": "0",
        "SNP_PPV>=30x": "96.7970258096804",
        "InDel_FN>=100x": "411",
        "SNP_FN>=100x": "406",
        "InDel_TP-AllPos": "1281",
        "InDel_FP>=30x": "838",
        "SNP_FN>=30x": "1006",
        "SNP_FP>=1000x": "0",
        "SNP_ConsensusAccuracy-AllPos": "0.999912578529413",
        "InDel_NoCalls-AllPos": "450359",
        "InDel_FP-AllPos": "1375",
        "InDel_FP_50x-100x": "369",
        "InDel_Sensitivity>=20x": "47.4402730375427",
        "SNP_TP>=500x": "44",
        "SNP_Sensitivity>=100x": "98.2229614391386",
        "SNP_Sensitivity>=500x": "100",
        "InDel_ConsensusAccuracy-AllPos": "0.999949088579261",
        "SNP_ConsensusAccuracy>=50x": "0.999972775263102",
        "InDel_FN>=20x": "1386",
        "SNP_FP>=50x": "829",
        "InDel_PPV-AllPos": "48.230421686747",
        "InDel_ConsensusAccuracy>=30x": "0.99996518422908",
        "InDel_FP-ncRNA": "9",
        "InDel_ReferenceCalls-AllPos": "0",
        "SNP_TP-AllPos": "43461",
```

```
"InDel_Sensitivity>=30x": "48.7158581328985",
"InDel_FN>=500x": "1",
"InDel_FP_20x-50x": "437",
"InDel_FP-ConfidentPos": "1375",
"InDel_FN-AllPos": "1690",
"SNP_HP11-AllPos": "0",
"InDel_FN>=50x": "977",
"SNP_Sensitivity-AllPos": "94.4989237024635",
"InDel_Sensitivity-AllAnnotations": "43.1167956916863",
"SNP_FP_200x-300x": "34",
"InDel_FP>=20x": "1013",
"SNP_FN>=500x": "0",
"Combined Variant Positive Predictive Value for All Bases at >=20x": "94
"SNP_Sensitivity>=1000x": "0",
"InDel_FP_700x-1000x": "0",
"SNP_FP>=30x": "1344",
"SNP_FP_700x-1000x": "0",
"SNP_ConsensusAccuracy>=20x": "0.999952742906361",
"InDel_TP>=30x": "1195",
"SNP_AmbPos-AllPos": "0",
"SNP_FP_500x-700x": "0",
"SNP_FP_50x-100x": "536",
"InDel_ConsensusAccuracy>=20x": "0.999960151224028",
"InDel_FP-Exons": "537",
"SNP_TP>=30x": "40617",
"SNP_ReferenceCalls-AllPos": "0",
"InDel_FP_500x-700x": "0",
"InDel_FP_200x-300x": "29",
"SNP_FP_20x-50x": "911",
"SNP_FP>=100x": "293",
"SNP_Sensitivity>=20x": "97.439640391121",
"InDel_FP_300x-400x": "0",
"SNP_FP_100x-200x": "255",
"SNP_Sensitivity-AllAnnotations": "94.4989237024635",
"InDel_FP_400x-500x": "2",
"InDel_FP>=50x": "576",
"SNP_FN-AllAnnotations": "2530",
"SNP_FP-AllPos": "2733",
"SNP_PPV-AllPos": "94.0836472269126",
"SNP_FP>=20x": "1740",
"InDel_Sensitivity>=1000x": "0",
"InDel_ConsensusAccuracy>=50x": "0.999974203772787",
"InDel_TP>=100x": "616",
"SNP_TP>=50x": "36440",
"SNP_ConsensusAccuracy>=30x": "0.999960965142337",
"InDel_Sensitivity-AllPos": "43.1167956916863",
"InDel_TP>=20x": "1251",
"InDel_AmbNotDetected-AllPos": "0",
"SNP_FP>=500x": "0",
"% Callable Bases": "99.161500",
"SNP_FN>=20x": "1105",
"InDel_FP>=1000x": "0",
"SNP_Sensitivity>=30x": "97.5830670542729",
"InDel_FN>=1000x": "0",
"InDel_FP>=500x": "0",
"SNP_FP_400x-500x": "2",
"InDel_HP11-AllPos": "0",
"Combined Variant Positive Predictive Value for All Bases at all coverage
```

```
            "SNP_FN>=1000x": "0",
            "SNP_FN>=50x": "810",
            "InDel_TP>=500x": "2",
            "InDel_PPV>=30x": "58.780127889818",
            "InDel_TP>=1000x": "0",
            "SNP_FP_300x-400x": "2",
            "SNP_Sensitivity>=50x": "97.8255033557047",
            "Combined Variant Sensitivity for All Bases at >= 20x": "94.560500",
            "SNP_TP>=100x": "22441",
            "InDel_FP>=100x": "207",
            "SNP_NoCalls-AllPos": "56910",
            "InDel_Sensitivity>=500x": "66.6666666666667",
            "SNP_TP>=20x": "42053",
            "InDel_Sensitivity>=50x": "52.3182040019522",
            "InDel_FN-AllAnnotations": "1690",
            "InDel_TP>=50x": "1072",
            "SNP_TP>=1000x": "0",
            "InDel_FN>=30x": "1258",
            "SNP_AmbNotDetected-AllPos": "0",
            "SNP_FP-ncRNA": "28",
            "SNP_FN-AllPos": "2530",
            "InDel_Sensitivity>=100x": "59.9805258033106",
            "InDel_FP_100x-200x": "176",
            "SNP_FP-Exons": "993",
            "Combined Variant Sensitivity for All Bases at all coverages": "91.381000
        },
        "IonXpress_034": {
            "SNP_FP-ConfidentPos": "2677",
            "InDel_AmbPos-AllPos": "0",
            "SNP_PPV>=30x": "96.6501472719622",
            "InDel_FN>=100x": "519",
            "SNP_FN>=100x": "490",
            "InDel_TP-AllPos": "1310",
            "InDel_FP>=30x": "851",
            "SNP_FN>=30x": "998",
            "SNP_FP>=1000x": "0",
            "SNP_ConsensusAccuracy-AllPos": "0.999915784370915",
            "InDel_NoCalls-AllPos": "444416",
            "InDel_FP-AllPos": "1306",
            "InDel_FP_50x-100x": "326",
            "InDel_Sensitivity>=20x": "47.9985035540591",
            "SNP_TP>=500x": "88",
            "SNP_Sensitivity>=100x": "98.2072952109172",
            "SNP_Sensitivity>=500x": "98.876404494382",
            "InDel_ConsensusAccuracy-AllPos": "0.999950683194723",
            "SNP_ConsensusAccuracy>=50x": "0.999970300287021",
            "InDel_FN>=20x": "1390",
            "SNP_FP>=50x": "933",
            "InDel_PPV-AllPos": "50.0764525993884",
            "InDel_ConsensusAccuracy>=30x": "0.999964652690593",
            "InDel_FP-ncRNA": "9",
            "InDel_ReferenceCalls-AllPos": "0",
            "SNP_TP-AllPos": "43597",
            "InDel_Sensitivity>=30x": "49.3454978183261",
            "InDel_FN>=500x": "2",
            "InDel_FP_20x-50x": "390",
            "InDel_FP-ConfidentPos": "1306",
            "InDel_FN-AllPos": "1663",
```

```
"SNP_HP11-AllPos": "0",
"InDel_FN>=50x": "1064",
"SNP_Sensitivity-AllPos": "94.7966949336812",
"InDel_Sensitivity-AllAnnotations": "44.0632357887656",
"SNP_FP_200x-300x": "50",
"InDel_FP>=20x": "1007",
"SNP_FN>=500x": "1",
"Combined Variant Positive Predictive Value for All Bases at >=20x": "93
"SNP_Sensitivity>=1000x": "100",
"InDel_FP_700x-1000x": "0",
"SNP_FP>=30x": "1433",
"SNP_FP_700x-1000x": "0",
"SNP_ConsensusAccuracy>=20x": "0.999951962209208",
"InDel_TP>=30x": "1244",
"SNP_AmbPos-AllPos": "0",
"SNP_FP_500x-700x": "1",
"SNP_FP_50x-100x": "559",
"InDel_ConsensusAccuracy>=20x": "0.999960184445184",
"InDel_FP-Exons": "525",
"SNP_TP>=30x": "41345",
"SNP_ReferenceCalls-AllPos": "0",
"InDel_FP_500x-700x": "1",
"InDel_FP_200x-300x": "36",
"SNP_FP_20x-50x": "862",
"SNP_FP>=100x": "374",
"SNP_Sensitivity>=20x": "97.4804198534647",
"InDel_FP_300x-400x": "9",
"SNP_FP_100x-200x": "310",
"SNP_Sensitivity-AllAnnotations": "94.7966949336812",
"InDel_FP_400x-500x": "0",
"InDel_FP>=50x": "617",
"SNP_FN-AllAnnotations": "2393",
"SNP_FP-AllPos": "2677",
"SNP_PPV-AllPos": "94.2148938928988",
"SNP_FP>=20x": "1795",
"InDel_Sensitivity>=1000x": "0",
"InDel_ConsensusAccuracy>=50x": "0.999972077618838",
"InDel_TP>=100x": "771",
"SNP_TP>=50x": "38295",
"SNP_ConsensusAccuracy>=30x": "0.999959619685541",
"InDel_Sensitivity-AllPos": "44.0632357887656",
"InDel_TP>=20x": "1283",
"InDel_AmbNotDetected-AllPos": "0",
"SNP_FP>=500x": "1",
"% Callable Bases": "99.176600",
"SNP_FN>=20x": "1097",
"InDel_FP>=1000x": "0",
"SNP_Sensitivity>=30x": "97.6430578844201",
"InDel_FN>=1000x": "0",
"InDel_FP>=500x": "1",
"SNP_FP_400x-500x": "2",
"InDel_HP11-AllPos": "0",
"Combined Variant Positive Predictive Value for All Bases at all coverage
"SNP_FN>=1000x": "0",
"SNP_FN>=50x": "855",
"InDel_TP>=500x": "2",
"InDel_PPV>=30x": "59.3794749403341",
"InDel_TP>=1000x": "0",
```

```
            "SNP_FP_300x-400x": "11",
            "SNP_Sensitivity>=50x": "97.816091954023",
            "Combined Variant Sensitivity for All Bases at >= 20x": "94.618200",
            "SNP_TP>=100x": "26843",
            "InDel_FP>=100x": "291",
            "SNP_NoCalls-AllPos": "53768",
            "InDel_Sensitivity>=500x": "50",
            "SNP_TP>=20x": "42442",
            "InDel_Sensitivity>=50x": "51.5041020966272",
            "InDel_FN-AllAnnotations": "1663",
            "InDel_TP>=50x": "1130",
            "SNP_TP>=1000x": "3",
            "InDel_FN>=30x": "1277",
            "SNP_AmbNotDetected-AllPos": "0",
            "SNP_FP-ncRNA": "32",
            "SNP_FN-AllPos": "2393",
            "InDel_Sensitivity>=100x": "59.7674418604651",
            "InDel_FP_100x-200x": "245",
            "SNP_FP-Exons": "958",
            "Combined Variant Sensitivity for All Bases at all coverages": "91.71610(
        }
    },
    "SNP_Sensitivity-AllAnnotations": "189.295618636145",
    "InDel_FP_400x-500x": 2,
    "InDel_FP>=50x": 1193,
    "SNP_FN-AllAnnotations": 4923,
    "SNP_FP-AllPos": 5410,
    "SNP_PPV-AllPos": "94.1493273348618",
    "SNP_FP>=20x": 3535,
    "InDel_Sensitivity>=1000x": 0,
    "InDel_ConsensusAccuracy>=50x": "0.999973140695813",
    "InDel_TP>=100x": 1387,
    "Region_selected": "NIST",
    "SNP_ConsensusAccuracy>=30x": "0.999960292413939",
    "InDel_Sensitivity-AllPos": "43.5901749663526",
    "InDel_TP>=20x": 2534,
    "InDel_AmbNotDetected-AllPos": 0,
    "SNP_FP>=500x": 1,
    "% Callable Bases": "99.16905",
    "SNP_FN>=20x": 2202,
    "InDel_FP>=1000x": 0,
    "SNP_Sensitivity>=30x": "97.6133196770121",
    "InDel_FN>=1000x": 0,
    "InDel_FP>=500x": 1,
    "SNP_FP_400x-500x": 4,
    "InDel_HP11-AllPos": 0,
    "Combined Variant Positive Predictive Value for All Bases at all coverages": "91
    "SNP_FN>=1000x": 0,
    "SNP_FN>=50x": 1665,
    "InDel_TP>=500x": 4,
    "InDel_PPV>=30x": "59.0843023255814",
    "InDel_TP>=1000x": 0,
    "Truth-minor_SNP_file": "None",
    "SNP_FP_300x-400x": 13,
    "SNP_Sensitivity>=50x": "97.8206806282722",
    "Combined Variant Sensitivity for All Bases at >= 20x": "94.58935",
    "SNP_TP>=100x": 49284,
    "Sample_selected": "NA12878",
```

```
            "InDel_FP>=100x": 498,
            "SNP_NoCalls-AllPos": 110678,
            "Truth-major_InDel_file": "NA12878_NIST_NoChrY_indel.bed",
            "InDel_Sensitivity>=500x": "57.1428571428571",
            "SNP_TP>=20x": 84495,
            "InDel_Sensitivity>=50x": "51.897242517087",
            "InDel_FN-AllAnnotations": 3353,
            "InDel_TP>=50x": 2202,
            "SNP_TP>=1000x": 3,
            "InDel_FN>=30x": 2535,
            "Variant-caller_name": "variantCaller",
            "SNP_AmbNotDetected-AllPos": 0,
            "SNP_TP>=50x": 74735,
            "SNP_FP-ncRNA": 60,
            "SNP_FN-AllPos": 4923,
            "InDel_Sensitivity>=100x": "59.8618903754855",
            "InDel_FP_100x-200x": 421,
            "SNP_FP-Exons": 1951,
            "Combined Variant Sensitivity for All Bases at all coverages": "91.54855"
        },
        "timingPerformance": {
            "runtime": {
                "analysis": "334.17000000000002"
            },
            "threadinfo": {
                "bkgmodel Gpu": 1,
                "fileaccess": 4,
                "beadfind": 6,
                "basecalling": 24,
                "bkgmodel Cpu": 6
            },
            "chipinfo": {
                "oia": 1,
                "flows": 500,
                "chiptype": "900"
            }
        },
        "coverageAnalysis": {
            "Non-duplicate": "No",
            "barcoded": "true",
            "Uniquely mapped": "No",
            "Amplicons reading end-to-end": "26.72%",
            "Targetted regions": "/results/uploads/BED/46/hg19/merged/detail/AmpliSeqExome.2(
            "Target padding": "0",
            "barcodes": {
                "IonXpress_033": {
                    "Bases in target regions": "57742646",
                    "Amplicons with at least 1 read": "99.21%",
                    "Target base coverage at 100x": "51.05%",
                    "Amplicons with at least 500 reads": "0.13%",
                    "Total assigned amplicon reads": "39187438",
                    "Reference (File)": "hg19",
                    "Total base reads on target": "6490060189",
                    "Target base coverage at 20x": "94.26%",
                    "Number of amplicons": "293903",
                    "Target bases with no strand bias": "76.79%",
                    "Percent reads on target": "94.39%",
                    "Amplicons with at least 100 reads": "64.34%",
```

```
                    "Average base coverage depth": "112.4",
                    "Average reads per amplicon": "133.3",
                    "Using": "All Mapped Reads",
                    "Amplicons reading end-to-end": "25.70%",
                    "Sample Name": "None",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Uniformity of base coverage": "93.56%",
                    "Alignments": "IonXpress_033_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Amplicons with at least 20 reads": "95.84%",
                    "Number of mapped reads": "41517304",
                    "Percent assigned amplicon reads": "94.39%",
                    "Amplicons with no strand bias": "92.84%",
                    "Total aligned base reads": "6846723653",
                    "Target base coverage at 1x": "98.53%",
                    "Target base coverage at 500x": "0.10%",
                    "Percent base reads on target": "94.79%",
                    "Uniformity of amplicon coverage": "94.62%"
                },
                "IonXpress_034": {
                    "Bases in target regions": "57742646",
                    "Amplicons with at least 1 read": "99.24%",
                    "Target base coverage at 100x": "61.01%",
                    "Amplicons with at least 500 reads": "0.24%",
                    "Total assigned amplicon reads": "45147738",
                    "Reference (File)": "hg19",
                    "Total base reads on target": "7552568443",
                    "Target base coverage at 20x": "94.93%",
                    "Number of amplicons": "293903",
                    "Target bases with no strand bias": "77.82%",
                    "Percent reads on target": "94.01%",
                    "Amplicons with at least 100 reads": "72.27%",
                    "Average base coverage depth": "130.8",
                    "Average reads per amplicon": "153.6",
                    "Using": "All Mapped Reads",
                    "Amplicons reading end-to-end": "27.74%",
                    "Sample Name": "None",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Uniformity of base coverage": "93.55%",
                    "Alignments": "IonXpress_034_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Amplicons with at least 20 reads": "96.17%",
                    "Number of mapped reads": "48026110",
                    "Percent assigned amplicon reads": "94.01%",
                    "Amplicons with no strand bias": "92.98%",
                    "Total aligned base reads": "8002544816",
                    "Target base coverage at 1x": "98.55%",
                    "Target base coverage at 500x": "0.22%",
                    "Percent base reads on target": "94.38%",
                    "Uniformity of amplicon coverage": "94.48%"
                }
            }
        },
        "VariantQC": {
            "IonXpress_033": {
                "reason": {
                    "filtered": {
                        "HPLEN": 6386,
                        "REJECTION": 610,
                        "Cov": 72787,
```

```
                "REF": 472111,
                "HEALED": 348,
                "SHIFT": 458087,
                "SSE": 78820,
                "STRINGENCY": 1500,
                ".": 538589,
                "Quality": 103314,
                "STDBIAS": 12975
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
                "Cov": 0,
                "REF": 0,
                "HEALED": 1036,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54299,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
            "filtered": {
                "11": 432,
                "10": 972,
                "13": 178,
                "12": 193,
                "15": 160,
                "14": 143,
                "1": 74456,
                "0": 57194,
                "3": 92797,
                "2": 124391,
                "5": 52748,
                "4": 62684,
                "7": 11651,
                "6": 25769,
                "9": 2199,
                "8": 4196
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
```

```
                "8": 8
            },
            "unfiltered": {
                "11": 43,
                "10": 45,
                "13": 20,
                "12": 38,
                "15": 27,
                "14": 27,
                "1": 29948,
                "0": 561,
                "3": 5112,
                "2": 12151,
                "5": 1238,
                "4": 2386,
                "7": 248,
                "6": 567,
                "9": 73,
                "8": 145
            }
        },
        "type": {
            "filtered": {
                "other": 37034,
                "del": 278917,
                "snp": 37539,
                "ins": 185099
            },
            "unfiltered": {
                "other": 2479,
                "del": 1093,
                "snp": 49277,
                "ins": 1450
            }
        },
        "basic": {
            "filtered": 538589,
            "unfiltered": 54299
        }
    },
    "IonXpress_034": {
        "reason": {
            "filtered": {
                "HPLEN": 6511,
                "REJECTION": 545,
                "Cov": 65871,
                "REF": 462781,
                "HEALED": 355,
                "SHIFT": 454012,
                "SSE": 83658,
                "STRINGENCY": 1396,
                ".": 529663,
                "Quality": 83379,
                "STDBIAS": 12531
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
```

```
                "Cov": 0,
                "REF": 0,
                "HEALED": 1082,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54469,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
            "filtered": {
                "11": 407,
                "10": 968,
                "13": 156,
                "12": 186,
                "15": 145,
                "14": 154,
                "1": 67025,
                "0": 51375,
                "3": 92736,
                "2": 124890,
                "5": 54756,
                "4": 63630,
                "7": 12138,
                "6": 26845,
                "9": 2248,
                "8": 4436
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
                "8": 8
            },
            "unfiltered": {
                "11": 39,
                "10": 49,
                "13": 35,
                "12": 32,
                "15": 18,
                "14": 33,
                "1": 30034,
                "0": 522,
                "3": 5116,
```

```
                "2": 12152,
                "5": 1261,
                "4": 2435,
                "7": 267,
                "6": 573,
                "9": 65,
                "8": 139
            }
        },
        "type": {
            "filtered": {
                "other": 35770,
                "del": 302522,
                "snp": 35483,
                "ins": 155888
            },
            "unfiltered": {
                "other": 2501,
                "del": 1095,
                "snp": 49486,
                "ins": 1387
            }
        },
        "basic": {
            "filtered": 529663,
            "unfiltered": 54469
        }
    },
    "summary": {
        "reason": {
            "filtered": {
                "HPLEN": 6386,
                "REJECTION": 610,
                "Cov": 72787,
                "REF": 472111,
                "HEALED": 348,
                "SHIFT": 458087,
                "SSE": 78820,
                "STRINGENCY": 1500,
                ".": 538589,
                "Quality": 103314,
                "STDBIAS": 12975
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
                "Cov": 0,
                "REF": 0,
                "HEALED": 1036,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54299,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
```

```
            "filtered": {
                "11": 432,
                "10": 972,
                "13": 178,
                "12": 193,
                "15": 160,
                "14": 143,
                "1": 74456,
                "0": 57194,
                "3": 92797,
                "2": 124391,
                "5": 52748,
                "4": 62684,
                "7": 11651,
                "6": 25769,
                "9": 2199,
                "8": 4196
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
                "8": 8
            },
            "unfiltered": {
                "11": 43,
                "10": 45,
                "13": 20,
                "12": 38,
                "15": 27,
                "14": 27,
                "1": 29948,
                "0": 561,
                "3": 5112,
                "2": 12151,
                "5": 1238,
                "4": 2386,
                "7": 248,
                "6": 567,
                "9": 73,
                "8": 145
            }
        },
        "type": {
            "filtered": {
                "other": 37034,
```

```
                            "del": 278917,
                            "snp": 37539,
                            "ins": 185099
                        },
                        "unfiltered": {
                            "other": 2479,
                            "del": 1093,
                            "snp": 49277,
                            "ins": 1450
                        }
                    },
                    "basic": {
                        "filtered": 538589,
                        "unfiltered": 54299
                    }
                }
            }
        },
        "resultsType": "",
        "tfFastq": "_",
        "tfmetrics": [
            "/rundb/api/v1/tfmetrics/3119/",
            "/rundb/api/v1/tfmetrics/3118/",
            "/rundb/api/v1/tfmetrics/3117/"
        ],
        "parentIDs": "",
        "analysismetrics": [
            "/rundb/api/v1/analysismetrics/31763/"
        ],
        "timeToComplete": "0",
        "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
        "fastqLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/h
        "pluginresults": [
            "/rundb/api/v1/pluginresult/570190/",
            "/rundb/api/v1/pluginresult/564696/",
            "/rundb/api/v1/pluginresult/564695/",
            "/rundb/api/v1/pluginresult/564694/",
            "/rundb/api/v1/pluginresult/564692/",
            "/rundb/api/v1/pluginresult/564691/",
            "/rundb/api/v1/pluginresult/564690/",
            "/rundb/api/v1/pluginresult/564689/",
            "/rundb/api/v1/pluginresult/564688/",
            "/rundb/api/v1/pluginresult/564687/",
            "/rundb/api/v1/pluginresult/564686/"
        ],
        "framesProcessed": 0,
        "autoExempt": false,
        "resource_uri": "/rundb/api/v1/results/293943/"
    }
  ]
}
```


Allowed HTTP methods

- get
- post

- put
- delete
- patch

2.1.62 Rig Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/rig/`

Schema URL: `http://mytorrentserver/rundb/api/v1/rig/schema/`

Perform CRUD operations on `rig` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
version	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
state	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftproot-dir	Unicode string data. Ex: "Hello World"	results	false	false	false	false	string
last_clean_date	Unicode string data. Ex: "Hello World"		false	false	true	false	string
update-home	Unicode string data. Ex: "Hello World"	192.168.201.1	false	false	false	false	string
ftpserver	Unicode string data. Ex: "Hello World"	192.168.201.1	false	false	false	false	string
com-ments	Unicode string data. Ex: "Hello World"		false	false	true	false	string
last_experiment	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftppass-word	Unicode string data. Ex: "Hello World"	ionguest	false	false	false	false	string
update-flag	Boolean data. Ex: True	false	false	false	true	false	boolean
location	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	re-lated
last_init_date	Unicode string data. Ex: "Hello World"		false	false	true	false	string
alarms	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
serial	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
host_address	Unicode string data. Ex: "Hello World"		false	false	true	false	string
type	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftpuser-name	Unicode string data. Ex: "Hello World"	ionguest	false	false	false	false	string
re-source_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/rig/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/rig/", params={"format": "json", "
ts_api_response = ts_api_request.json()

rigs = ts_api_response["objects"]

for rig in rigs:
    print rig
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 175,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/rig/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "version": {},
            "name": "PGM_test",
            "state": "",
            "ftprootdir": "results",
            "last_clean_date": "",
            "updatehome": "192.168.201.1",
            "ftpserver": "192.168.201.1",
            "comments": "",
            "last_experiment": "",
            "ftppassword": "ionguest",
            "updateflag": false,
            "location": {
                "name": "Home",
                "resource_uri": "/rundb/api/v1/location/1/",
                "defaultlocation": true,
                "comments": "",
                "id": 1
            },
            "last_init_date": "",
            "alarms": {},
            "serial": "",
            "host_address": "",
            "type": "",
            "ftpusername": "ionguest",
            "resource_uri": "/rundb/api/v1/rig/PGM_test/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.63 Runtype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/runtype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/runtype/schema/`

Perform CRUD operations on `runtype` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
applicationGroups	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"	dna	false	false	true	false	string
barcode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
meta	Unicode string data. Ex: "Hello World"		true	false	false	false	string
runType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
alternate_name	Unicode string data. Ex: "Hello World"		true	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/runtype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/runtype/", params={"format": "jso
ts_api_response = ts_api_request.json()
```

```
runtypes = ts_api_response["objects"]

for runtype in runtypes:
    print runtype
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/runtype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "applicationGroups": [
                "/rundb/api/v1/applicationgroup/1/",
                "/rundb/api/v1/applicationgroup/3/",
                "/rundb/api/v1/applicationgroup/4/"
            ],
            "description": "Generic Sequencing",
            "nucleotideType": "",
            "barcode": "",
            "meta": {},
            "runType": "GENS",
            "id": 1,
            "alternate_name": "Other",
            "resource_uri": "/rundb/api/v1/runtype/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.64 Sample Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sample/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sample/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
sampleSets	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
displayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
experiments	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
externalId	Unicode string data. Ex: “Hello World”		true	false	false	false	string
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	true	false	false	false	datetime
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
name	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sample/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sample/", params={"format": "json"
ts_api_response = ts_api_request.json()

samples = ts_api_response["objects"]

for sample in samples:
    print sample
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 7765,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sample/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
```

```
            "status": "run",
            "sampleSets": [],
            "description": "",
            "displayedName": "E122627-lq405-L8095",
            "experiments": [
                "/rundb/api/v1/experiment/11750/"
            ],
            "externalId": "",
            "date": "2013-07-23T17:11:31.000986+00:00",
            "resource_uri": "/rundb/api/v1/sample/3265/",
            "id": 3265,
            "name": "E122627-lq405-L8095"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.65 Sampleannotation_Cv Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/schema/`

Perform read-only operations on `sampleannotation_cv` resources. This resource corresponds to the supported sample relationships (Self | Proband, Tumor, Normal, Mother, Father, etc) in Ion Reporter™ Software.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
annotationType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
isIRCompatible	Boolean data. Ex: True	false	false	false	true	false	boolean
sampleGroupType_CV	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
value	Unicode string data. Ex: "Hello World"		false	false	true	false	string
iRValue	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
iRAnnotationType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleannotation_cv/", params={"fo
ts_api_response = ts_api_request.json()

sampleannotation_cvs = ts_api_response["objects"]

for sampleannotation_cv in sampleannotation_cvs:
    print sampleannotation_cv
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 34,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sampleannotation_cv/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "annotationType": "relationshipRole",
            "uid": "SAMPLEANNOTATE_CV_0001",
            "isIRCompatible": true,
```

```
            "sampleGroupType_CV": "/rundb/api/v1/samplegrouptype_cv/1/",
            "value": "Sample",
            "iRValue": "Sample",
            "iRAnnotationType": "Relation",
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/sampleannotation_cv/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.66 Sampleattribute Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
dataType_name	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
dataType	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
displayedName	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
isMandatory	Boolean data. Ex: True	false	false	false	true	false	boolean
sampleCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleattribute/", params={"format
ts_api_response = ts_api_request.json()

sampleattributes = ts_api_response["objects"]

for sampleattribute in sampleattributes:
    print sampleattribute
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.67 Sampleattributedatatype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
dataType	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleattributedatatype/", params=
ts_api_response = ts_api_request.json()

sampleattributedatatypes = ts_api_response["objects"]

for sampleattributedatatype in sampleattributedatatypes:
    print sampleattributedatatype
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sampleattributedatatype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "dataType": "Text",
            "resource_uri": "/rundb/api/v1/sampleattributedatatype/1/",
            "description": "Up to 1024 characters",
            "isActive": true,
            "id": 1
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

2.1.68 Samplegrouptype_Cv Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/schema/`

Perform read-only operations on `samplegrouptype_cv` resources. This resource corresponds to the supported relationship types (Single, Paired, Trio, etc) in Ion Reporter™ Software and to the sample set Grouping column in the Torrent Suite™ Software.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isIRCompatible	Boolean data. Ex: True	false	false	false	true	false	boolean
description	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
sampleAnnotation_set	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
displayedName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
iRValue	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
iRAnnotationType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
sampleSets	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/", params={"fo
ts_api_response = ts_api_request.json()

samplegrouptype_cvs = ts_api_response["objects"]
```

```
for samplegrouptype_cv in samplegrouptype_cvs:
    print samplegrouptype_cv
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/samplegrouptype_cv/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isIRCompatible": true,
            "description": "",
            "sampleAnnotation_set": [
                "/rundb/api/v1/sampleannotation_cv/1/",
                "/rundb/api/v1/sampleannotation_cv/2/"
            ],
            "displayedName": "Sample_Control",
            "iRValue": "Paired_Sample|Sample_Control",
            "iRAnnotationType": "RelationshipType",
            "uid": "SAMPLEGROUP_CV_0001",
            "sampleSets": [],
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/samplegrouptype_cv/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.69 Sampleset Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleset/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleset/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
sampleCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
displayedName	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
SampleGroupType_CV	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
samples	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
sampleGroupTypeName	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleset/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleset/", params={"format": "j
ts_api_response = ts_api_request.json()

samplesets = ts_api_response["objects"]

for sampleset in samplesets:
    print sampleset
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
```

```
    },
    "objects": [
        {
            "status": "created",
            "description": "",
            "sampleCount": 4,
            "displayedName": "Set",
            "SampleGroupType_CV": "/rundb/api/v1/samplegrouptype_cv/3/",
            "samples": [
                "/rundb/api/v1/samplesetitem/14/",
                "/rundb/api/v1/samplesetitem/15/",
                "/rundb/api/v1/samplesetitem/16/",
                "/rundb/api/v1/samplesetitem/17/"
            ],
            "lastModifiedDate": "2013-10-07T12:04:51.000432+00:00",
            "sampleGroupTypeName": "Tumor_Normal",
            "creationDate": "2013-10-07T12:04:51.000432+00:00",
            "id": 5,
            "resource_uri": "/rundb/api/v1/sampleset/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.70 Samplesetitem Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
sample	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
gender	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
relationshipGroup	Integer data. Ex: 2673	n/a	false	false	false	false	integer
cellularityPct	Integer data. Ex: 2673	n/a	true	false	false	false	integer
relationshipRole	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
cancerType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
sampleSet	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
lastModifiedDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
dnabarcode	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
creationDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplesetitem/", params={"format"
ts_api_response = ts_api_request.json()

samplesetitems = ts_api_response["objects"]

for samplesetitem in samplesetitems:
    print samplesetitem
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
```

```
        "next": "/rundb/api/v1/samplesetitem/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "sample": "/rundb/api/v1/sample/4877/",
            "gender": "Female",
            "relationshipGroup": 1,
            "cellularityPct": null,
            "relationshipRole": "Self",
            "cancerType": null,
            "sampleSet": "/rundb/api/v1/sampleset/5/",
            "lastModifiedDate": "2013-10-07T12:04:51.000440+00:00",
            "dnabarcode": null,
            "creationDate": "2013-10-07T12:04:51.000440+00:00",
            "id": 14,
            "resource_uri": "/rundb/api/v1/samplesetitem/14/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.71 Samplesetiteminfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
sample	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
sampleSetPk	Integer data. Ex: 2673	n/a	true	true	true	false	integer
sampleExternalId	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
sampleDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
gender	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
relationshipGroup	Integer data. Ex: 2673	n/a	true	true	true	false	integer
cellularityPct	Integer data. Ex: 2673	n/a	true	false	false	false	integer
dnabarcodeKit	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
sampleDescription	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
relationshipRole	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
cancerType	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
samplePk	Integer data. Ex: 2673	n/a	true	true	true	false	integer
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
dnabarcode	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	true	true	false	related
sampleSet	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
sampleSetStatus	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplesetiteminfo/", params={"form
ts_api_response = ts_api_request.json()
```

```
samplesetiteminfos = ts_api_response["objects"]

for samplesetiteminfo in samplesetiteminfos:
    print samplesetiteminfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/samplesetiteminfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "sample": "/rundb/api/v1/sample/4877/",
            "samplePk": 4877,
            "sampleExternalId": "NA10859",
            "sampleDisplayedName": "1347-02",
            "gender": "Female",
            "relationshipGroup": 1,
            "cellularityPct": null,
            "dnabarcodeKit": "",
            "sampleDescription": "mother",
            "relationshipRole": "Self",
            "cancerType": null,
            "attribute_dict": {},
            "lastModifiedDate": "2013-10-07T12:04:51.000440+00:00",
            "dnabarcode": "",
            "sampleSetPk": 5,
            "sampleSet": "/rundb/api/v1/sampleset/5/",
            "creationDate": "2013-10-07T12:04:51.000440+00:00",
            "id": 14,
            "sampleSetStatus": "created",
            "resource_uri": "/rundb/api/v1/samplesetiteminfo/14/"
        }
    ]
}
```


Allowed HTTP methods

- get

2.1.72 Sequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sequencingkitinfo/", params={"form
ts_api_response = ts_api_request.json()

sequencingkitinfos = ts_api_response["objects"]

for sequencingkitinfo in sequencingkitinfos:
    print sequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": false,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion Sequencing 200 Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4471258",
                    "id": 20005,
                    "resource_uri": "/rundb/api/v1/kitpart/20005/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471257",
                    "id": 20006,
                    "resource_uri": "/rundb/api/v1/kitpart/20006/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471259",
                    "id": 20007,
                    "resource_uri": "/rundb/api/v1/kitpart/20007/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471260",
                    "id": 20008,
                    "resource_uri": "/rundb/api/v1/kitpart/20008/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                }
            ],
            "flowCount": 520,
            "applicationType": "",
            "uid": "SEQ0002",
            "resource_uri": "/rundb/api/v1/sequencingkitinfo/20002/",
            "id": 20002,
            "categories": "",
            "name": "IonSeq200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.73 Sequencingkitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sequencingkitpart/", params={"form
ts_api_response = ts_api_request.json()

sequencingkitparts = ts_api_response["objects"]

for sequencingkitpart in sequencingkitparts:
    print sequencingkitpart
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 64,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sequencingkitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "A25592",
            "id": 20132,
            "resource_uri": "/rundb/api/v1/sequencingkitpart/20132/",
            "kit": "/rundb/api/v1/kitinfo/20063/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.74 Supportupload Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/supportupload/`

Schema URL: `http://mytorrentserver/rundb/api/v1/supportupload/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
ticket_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
updated	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	date-time
local_message	Unicode string data. Ex: "Hello World"		false	false	true	false	string
description	Unicode string data. Ex: "Hello World"		false	false	false	false	string
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	date-time
ticket_status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
contact_email	Unicode string data. Ex: "Hello World"		false	false	false	false	string
result	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
file	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
celery_task_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ticket_message	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
local_status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/supportupload/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/supportupload/", params={"format"
ts_api_response = ts_api_request.json()

supportuploads = ts_api_response["objects"]

for supportupload in supportuploads:
    print supportupload
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
```

```
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.75 Template Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/template/`

Schema URL: `http://mytorrentserver/rundb/api/v1/template/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isofficial	Boolean data. Ex: True	true	false	false	true	false	boolean
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
sequence	Unicode string data. Ex: “Hello World”		false	false	true	false	string
comments	Unicode string data. Ex: “Hello World”		false	false	true	false	string
key	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/template/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/template/", params={"format": "jso
ts_api_response = ts_api_request.json()

templates = ts_api_response["objects"]

for template in templates:
    print template
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/template/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isofficial": true,
            "name": "TF1.2(tA-tB30') Hyb extend",
            "sequence": "GTTTTAGGGTCCCCGGGGTTAAAAGGTTTCGAACACGATGTCGGAGACACGCAGGGATGAGATGG",
            "comments": "",
            "key": "ATCGT",
            "id": 7,
            "resource_uri": "/rundb/api/v1/template/7/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.76 Tfmetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/schema/`

Perform read-only operations on `tfmetrics` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
corrHPSNR	Unicode string data. Ex: "Hello World"		false	false	true	false	string
Q10Mean	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
SysSNR	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
HPAccuracy	Unicode string data. Ex: "Hello World"		false	false	true	false	string
Q17ReadCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
sequence	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
Q17Histo	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
aveKeyCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
number	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
id	Integer data. Ex: 2673		false	false	true	true	integer
keypass	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
Q10ReadCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
Q17Mean	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
Q10Histo	Unicode string data. Ex: "Hello World"		false	false	true	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/tfmetrics/", params={"format": "js
ts_api_response = ts_api_request.json()

tfmetricss = ts_api_response["objects"]

for tfmetrics in tfmetricss:
    print tfmetrics
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 7320,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/tfmetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "corrHPSNR": "",
            "Q10Mean": 79.7,
            "SysSNR": 20.19,
            "HPAccuracy": "0 : 560244/582614, 1 : 356550/377171, 2 : 35115/45374, 3 : 0/0, 4 : 521/43
            "Q17ReadCount": 3992.0,
            "sequence": "TTGCGCGCGCTGTGAATGCGCTGCTGTCGAATCGCGCTGCGCTGAACGTCGCGTGCGCGAACGATCTGAGACTGCG
            "Q17Histo": "951 0 0 1 9 5 2 7 6 277 5 0 1 2 3 0 2 1 6 1 7 2 3 3 0 0 10 1 0 26 0 2 0 1 2
            "name": "TF_D",
            "aveKeyCount": 71.0,
            "number": 4119.0,
            "id": 1,
            "keypass": 5368.0,
            "Q10ReadCount": 4586.0,
            "report": "/rundb/api/v1/results/89/",
            "resource_uri": "/rundb/api/v1/tfmetrics/1/",
            "Q17Mean": 66.56,
            "Q10Histo": "40 0 0 1 8 3 0 4 2 1 587 5 3 5 1 8 0 2 6 1 5 1 3 3 2 1 5 9 0 0 2 0 2 1 5 1 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.77 Threeprimeadapter Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/`

Schema URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
direction	Unicode string data. Ex: “Hello World”	Forward	false	false	false	false	string
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
chemistryType	Unicode string data. Ex: “Hello World”		false	false	true	false	string
runMode	Unicode string data. Ex: “Hello World”	single	false	false	true	false	string
uid	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isDefault	Boolean data. Ex: True	false	false	false	true	false	boolean
description	Unicode string data. Ex: “Hello World”		false	false	true	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/threeprimeadapter/", params={"form
ts_api_response = ts_api_request.json()

threeprimeadapters = ts_api_response["objects"]

for threeprimeadapter in threeprimeadapters:
    print threeprimeadapter
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 17,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/threeprimeadapter/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "direction": "Forward",
            "name": "Ion P1B",
            "sequence": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "chemistryType": "",
            "runMode": "single",
            "uid": "FWD_0001",
            "resource_uri": "/rundb/api/v1/threeprimeadapter/1/",
            "id": 1,
            "isDefault": true,
            "description": "Default forward adapter"
```

```
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

2.1.78 User Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/user/`

Schema URL: `http://mytorrentserver/rundb/api/v1/user/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
profile	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
username	Required. 30 characters or fewer. Letters, numbers and @/./+/-/_ characters	n/a	false	false	false	true	string
first_name	Unicode string data. Ex: “Hello World”		false	false	true	false	string
last_name	Unicode string data. Ex: “Hello World”		false	false	true	false	string
is_active	Designates whether this user should be treated as active. Unselect this instead of deleting accounts.	true	false	false	true	false	boolean
email	Unicode string data. Ex: “Hello World”		false	false	true	false	string
last_login	A date & time as a string. Ex: “2010-11-10T03:07:43”	2014-06-28T14:04:05.000090+00:00	false	false	false	false	datetime
full_name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
date_joined	A date & time as a string. Ex: “2010-11-10T03:07:43”	2014-06-28T14:04:05.000090+00:00	false	false	false	false	datetime

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/user/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/user/", params={"format": "json",
ts_api_response = ts_api_request.json()

users = ts_api_response["objects"]

for user in users:
    print user
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/user/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "profile": {
                "phone_number": "",
                "name": "",
                "title": "user",
                "last_read_news_post": "1984-11-05T05:00:00+00:00",
                "note": "",
                "id": 6,
                "resource_uri": ""
            },
            "username": "dm_contact",
            "first_name": "",
            "last_name": "",
            "is_active": true,
            "email": "ts-admin@mailman.itw",
            "last_login": "2013-03-13T17:28:05.000596+00:00",
            "full_name": "",
            "resource_uri": "/rundb/api/v1/user/6/",
            "id": 6,
            "date_joined": "2013-03-13T17:28:05.000596+00:00"
        }
    ]
}
```


Allowed HTTP methods

- get

See the API Cookbook for a tutorial on how to programatically access the API:

2.2 Torrent Suite™ Software API Cookbook

2.2.1 Introduction

This "cookbook" introduces you to the basic capabilities of the Torrent Suite™ Software API, using the learn-by-doing method.

About the Examples

- Cookbook examples are discussed in snippets to elaborate on important interface details. Refer to Torrent Suite™ Software SDK Source Code Samples for full example listings.
- Where the server name and authorization credentials are shown, the following convention is used:

Entity	**Placeholder**
Server name	myhost
Username	myusername
Password	mypassword

To run the examples, replace these strings with the host name and credentials required for your server.

- You can interactively explore the REST interface using either the cURL command line utility, a REST client, or a web browser. These tools require less infrastructure than program development and providing a more convenient way to learn the interface.
  - cURL
  - Firefox REST client
  - Chrome REST client
  - Generic REST client

  (!) Each example shows the equivalent URI used with these tools before describing the programming language implementation.
- If you run the examples in your browser using either the browser address window or a REST client, you must include the `?format=json` parameter. This is because the browser requests XML-formatted data before JSON-formatted data but the current implementation does not support XML. This requirement does not apply to your programs, although, the programming examples in this document include the format parameter.
- For examples that demonstrate the API using the Python programming language, one of the following REST libraries is used. You may need to modify the example code for your preferred REST library.
  - httplib2
  - restful_lib (deprecated)
  - requests

- Currently, only the JSON data format is supported. The examples use the simplejson library to encode and decode JSON data into Python objects.
- JavaScript examples use the jQuery framework.

Before Reading This Document

To facilitate your learning the API, we recommend that you run the examples as you work through the cookbook. You can install the necessary programming languages and libraries and run the examples as shown, or modify the examples for your particular language and programming environment. These simple examples are shown using an easy-to-read programming language, requiring minimal glue logic, so they should be easily portable to other languages.

To run the examples or to interactively work with the API using cURL or a REST client, you need a Torrent Server available. Remember to change the server name and credentials shown in the example to those required by your server.

Writing applications using the REST API involves basic web programming. You should have some previous experience developing service-oriented web applications.

You may find the documentation in the following list helpful to provide more insight into the topics presented in the cookbook. This content might prove most useful when you extend the examples and create your own applications:

- The cookbook is intended to provide only the essential information need to get started developing applications. To this end, the examples typically involve resources having a small number of data fields to reduce the amount of non-essential information in the presentation. Refer to the Torrent Suite™ Software Database Tables for a complete, detailed description of each resource. Notice that in the current API version, not all of the resources are exposed by the API.
- For a complete description of the API syntax and functionality for all resources, refer to the API references tables document Torrent Suite™ Software API Reference.
- Once you have gained a basic familiarity with API programming, use the Torrent Suite™ Software API Quick Reference to help recall details about the API that may take some time to remember.

API topics are presented in the form of examples that are "recipes" for mastering each of the various topics. The examples are arranged in order, from the simplest "hello, world" type of application to increasingly complex functionality, where each depends on the understanding gained in previous examples. Each example demonstrates a real-world application that can be easily extended. From the basic operations involved in retrieving data through selecting and sorting data and, finally, updating data and creating new resource objects, you gain the knowledge needed to begin your own application development.

See the API reference tables for the full syntax needed to extend the applications presented in this document:

- Torrent Suite™ Software REST API v1 Resources

Connect with the Server

To connect to a resource, you first authenticate with the server.

Topics on this page:

The connection and authentication is currently as simple as logging into the server and providing your username and password.

The following examples show:

- The general form of authentication using a browser or REST client. You are prompted for your username and password, if they are not provided in the request.
- The cURL command line form.
- Programmatic methods using various Python libraries, PHP, and JavaScript.

General form

```
http://myusername:mypassword@myhost/rundb/api/v1/experiment
```

cURL command

```
curl --user mysername:mypassword
     --header "Content-Type: application/json"
     --location 'http://myhost/rundb/api/v1/experiment'
```

Programatically

Python libraries *restful_lib*

NOTE: restful_lib has not been updated in over 5 years and is considered deprecated.

```
from restful_lib import Connection
base_url = 'http://myhost/rundb/api/v1'
conn = Connection(base_url, username="myusername", password="mypassword")
```

httplib2

```
import httplib2
h = httplib2.Http()
h.add_credentials('myusername', 'mypassword')
```

requests (recommended)

```
import requests
resp = requests.get('http://myhost/rundb/api/v1?format=json', auth=('myusername', 'mypassword'))
```

PHP

```
<?php
$context = stream_context_create(array(
'http' => array(
'header'  =>
    "Authorization: Basic " . base64_encode("myusername:mypassword")
)
));

$url = "http://myhost/rundb/api/v1?format=json";
$feed = file_get_contents($url, false, $context);
?>
```

JavaScript jQuery AJAX call

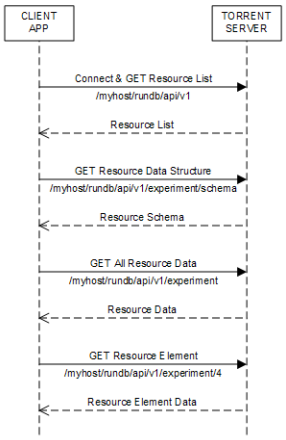
```
$.ajax({
    url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment",
    dataType: 'jsonp',
    success: handleResponse(json_results)
});
```

Cookbook - Your First Request and More

Read this topic for a hands-on introduction to using the REST API. Through a logical progression, using Python examples, you learn the fundamentals of how to make a REST call and how to use the API to access and manipulate a particular data element.

Before you access a resource, you need to get the list of resources exposed by the API, and the mechanism for accessing your particular resource of interest. At each step, you use the API to traverse the relational database or functional component (file or plugin), to obtain your specific data item of interest.

The following diagram shows the request message sequence for to discover a particular data element:



1. Begin with the highest-level request, /myhost/rundb/api/v1, without specifying a resource so all supported resources are returned. The first request sent to the server always requires authentication, which is your username and password.
2. To find the data fields contained by a resource, request the resource schema: /myhost/rundb/api/v1/experiment/schema. You can use these fields to qualify your searches for specific resource elements or data sets.
3. To find all of the resource elements, or objects, send a request that includes only the resource name, or list_endpoint: /myhost/rundb/api/v1/experiment.
4. Once you locate the particular resource element of interest, pass the primary key for the resource, usually the id field, to retrieve only the data for that element: /myhost/rundb/api/v1/experiment/4.

The rest of this document shows how to build on this basic message sequence to refine your searches.

> (!) The interface mechanism demonstrated here is the same for all REST operations, and subsequent more involved examples differ only in particular resource and access request parameters.

Get the list of resources

URIs The REST interface uses a Uniform Resource Identifier (URI) to name and locate a resource. This is the same as an address you commonly type in your browser to retrieve a Web page.

For example:

```
http://myhost/rundb/api/v1/experiment
```

You can see that REST uses the same HTTP protocol as the Web and the rest of the address specifies the location of your resource. This example requests the experiment resource, which is located on the host named myhost in the resource directory /rundb/api/v1, where v1 is the API version.

List available resources If you do not already know what the API name for the resource containing your data item is, you need to get a list of resources:

```
http://myhost/rundb/api/v1?format=json
```

This is the basic URI without specifying a resource after the API version.:

```
(!) The format=json parameter name:value pair is appended to the request to specify the format, JavaS
```

Enter the URI in your browser window or REST client, replacing myhost with your host name. This step also verifies connectivity before continuing with the tutorial. If you are using a REST client and the request is successful, an HTTP status code of 200 is returned. Otherwise, the request failed.

The examples in this tutorial use the Python programming language, because the API is intended to be used programmatically and because Python has both a low barrier to entry and also syntax similar to numerous other commonly used languages.

Also, the examples depend on the json and requests libraries (note that restful_lib is depreciated), so each example assumes the following statements are included:

```
import json
import requests
```

You can use any equivalent libraries and modify the code snippets as needed.

Now, you can programmatically make the same request for a list of resources using the following code snippet:

```
resp = requests.get('http://myhost/rundb/api/v1, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

(!) The examples in this tutorial do not include error handling and assume an HTTP status code of 2xx is always returned.

- On the first API call, it is necessary to connect to the server, providing basepath and authentication parameters. (To run the example, replace myhost with your host name and replace myusername and mypassword with your username and password.)

The connection request returns a connection handle, which is used for subsequent requests.

- The second statement performs a simple GET request, without specifying a resource. The response from the server is returned in the result variable:


```
{
        "account": {
                "list_endpoint": "/rundb/api/v1/account/",
                "schema": "/rundb/api/v1/account/schema/"
        },
        "activeionchefprepkitinfo": {
                "list_endpoint": "/rundb/api/v1/activeionchefprepkitinfo/",
                "schema": "/rundb/api/v1/activeionchefprepkitinfo/schema/"
```

```
    },
    "activelibrarykitinfo": {
            "list_endpoint": "/rundb/api/v1/activelibrarykitinfo/",
            "schema": "/rundb/api/v1/activelibrarykitinfo/schema/"
    },
    "activepgmlibrarykitinfo": {
            "list_endpoint": "/rundb/api/v1/activepgmlibrarykitinfo/",
            "schema": "/rundb/api/v1/activepgmlibrarykitinfo/schema/"
    },
    "activepgmsequencingkitinfo": {
            "list_endpoint": "/rundb/api/v1/activepgmsequencingkitinfo/",
            "schema": "/rundb/api/v1/activepgmsequencingkitinfo/schema/"
    },
    "activeprotonlibrarykitinfo": {
            "list_endpoint": "/rundb/api/v1/activeprotonlibrarykitinfo/",
            "schema": "/rundb/api/v1/activeprotonlibrarykitinfo/schema/"
    },
    "activeprotonsequencingkitinfo": {
            "list_endpoint": "/rundb/api/v1/activeprotonsequencingkitinfo/",
            "schema": "/rundb/api/v1/activeprotonsequencingkitinfo/schema/"
    },
    "activesequencingkitinfo": {
            "list_endpoint": "/rundb/api/v1/activesequencingkitinfo/",
            "schema": "/rundb/api/v1/activesequencingkitinfo/schema/"
    },
    "analysismetrics": {
            "list_endpoint": "/rundb/api/v1/analysismetrics/",
            "schema": "/rundb/api/v1/analysismetrics/schema/"
    },
    "applproduct": {
            "list_endpoint": "/rundb/api/v1/applproduct/",
            "schema": "/rundb/api/v1/applproduct/schema/"
    },
    "availableionchefplannedexperiment": {
            "list_endpoint": "/rundb/api/v1/availableionchefplannedexperiment/",
            "schema": "/rundb/api/v1/availableionchefplannedexperiment/schema/"
    },
    "availableionchefplannedexperimentsummary": {
            "list_endpoint": "/rundb/api/v1/availableionchefplannedexperimentsummary/",
            "schema": "/rundb/api/v1/availableionchefplannedexperimentsummary/schema/"
    },
    "availableonetouchplannedexperiment": {
            "list_endpoint": "/rundb/api/v1/availableonetouchplannedexperiment/",
            "schema": "/rundb/api/v1/availableonetouchplannedexperiment/schema/"
    },
    "availableonetouchplannedexperimentsummary": {
            "list_endpoint": "/rundb/api/v1/availableonetouchplannedexperimentsummary/",
            "schema": "/rundb/api/v1/availableonetouchplannedexperimentsummary/schema/"
    },
    "availableplannedexperimentsummary": {
            "list_endpoint": "/rundb/api/v1/availableplannedexperimentsummary/",
            "schema": "/rundb/api/v1/availableplannedexperimentsummary/schema/"
    },
    "chip": {
            "list_endpoint": "/rundb/api/v1/chip/",
            "schema": "/rundb/api/v1/chip/schema/"
    },
    "compositedatamanagement": {
```

```
        "list_endpoint": "/rundb/api/v1/compositedatamanagement/",
        "schema": "/rundb/api/v1/compositedatamanagement/schema/"
},
"compositeexperiment": {
        "list_endpoint": "/rundb/api/v1/compositeexperiment/",
        "schema": "/rundb/api/v1/compositeexperiment/schema/"
},
"compositeresult": {
        "list_endpoint": "/rundb/api/v1/compositeresult/",
        "schema": "/rundb/api/v1/compositeresult/schema/"
},
"content": {
        "list_endpoint": "/rundb/api/v1/content/",
        "schema": "/rundb/api/v1/content/schema/"
},
"contentupload": {
        "list_endpoint": "/rundb/api/v1/contentupload/",
        "schema": "/rundb/api/v1/contentupload/schema/"
},
"datamanagementhistory": {
        "list_endpoint": "/rundb/api/v1/datamanagementhistory/",
        "schema": "/rundb/api/v1/datamanagementhistory/schema/"
},
"dnabarcode": {
        "list_endpoint": "/rundb/api/v1/dnabarcode/",
        "schema": "/rundb/api/v1/dnabarcode/schema/"
},
"emailaddress": {
        "list_endpoint": "/rundb/api/v1/emailaddress/",
        "schema": "/rundb/api/v1/emailaddress/schema/"
},
"eventlog": {
        "list_endpoint": "/rundb/api/v1/eventlog/",
        "schema": "/rundb/api/v1/eventlog/schema/"
},
"experiment": {
        "list_endpoint": "/rundb/api/v1/experiment/",
        "schema": "/rundb/api/v1/experiment/schema/"
},
"experimentanalysissettings": {
        "list_endpoint": "/rundb/api/v1/experimentanalysissettings/",
        "schema": "/rundb/api/v1/experimentanalysissettings/schema/"
},
"fileserver": {
        "list_endpoint": "/rundb/api/v1/fileserver/",
        "schema": "/rundb/api/v1/fileserver/schema/"
},
"globalconfig": {
        "list_endpoint": "/rundb/api/v1/globalconfig/",
        "schema": "/rundb/api/v1/globalconfig/schema/"
},
"ionchefplantemplate": {
        "list_endpoint": "/rundb/api/v1/ionchefplantemplate/",
        "schema": "/rundb/api/v1/ionchefplantemplate/schema/"
},
"ionchefplantemplatesummary":{
        "list_endpoint": "/rundb/api/v1/ionchefplantemplatesummary/",
        "schema": "/rundb/api/v1/ionchefplantemplatesummary/schema/"
```

```
},
"ionchefprepkitinfo": {
        "list_endpoint": "/rundb/api/v1/ionchefprepkitinfo/",
        "schema": "/rundb/api/v1/ionchefprepkitinfo/schema/"
},
"ionreporter": {
        "list_endpoint": "/rundb/api/v1/ionreporter/",
        "schema": "/rundb/api/v1/ionreporter/schema/"
}, "kitinfo": {
        "list_endpoint": "/rundb/api/v1/kitinfo/",
        "schema": "/rundb/api/v1/kitinfo/schema/"
},
"kitpart": {
        "list_endpoint": "/rundb/api/v1/kitpart/",
        "schema": "/rundb/api/v1/kitpart/schema/"
},
"libmetrics": {
        "list_endpoint": "/rundb/api/v1/libmetrics/",
        "schema": "/rundb/api/v1/libmetrics/schema/"
},
"librarykey": {
        "list_endpoint": "/rundb/api/v1/librarykey/",
        "schema": "/rundb/api/v1/librarykey/schema/"
},
"librarykitinfo": {
        "list_endpoint": "/rundb/api/v1/librarykitinfo/",
        "schema": "/rundb/api/v1/librarykitinfo/schema/"
},
"librarykitpart": {
        "list_endpoint": "/rundb/api/v1/librarykitpart/",
        "schema": "/rundb/api/v1/librarykitpart/schema/"
},
"location": {
        "list_endpoint": "/rundb/api/v1/location/",
        "schema": "/rundb/api/v1/location/schema/"
},
"log": {
        "list_endpoint": "/rundb/api/v1/log/",
        "schema": "/rundb/api/v1/log/schema/"
},
"message": {
        "list_endpoint": "/rundb/api/v1/message/",
        "schema": "/rundb/api/v1/message/schema/"
},
"monitorexperiment": {
        "list_endpoint": "/rundb/api/v1/monitorexperiment/",
        "schema": "/rundb/api/v1/monitorexperiment/schema/"
},
"obsoletereferencegenome": {
        "list_endpoint": "/rundb/api/v1/obsoletereferencegenome/",
        "schema": "/rundb/api/v1/obsoletereferencegenome/schema/"
},
"onetouchplantemplate": {
        "list_endpoint": "/rundb/api/v1/onetouchplantemplate/",
        "schema": "/rundb/api/v1/onetouchplantemplate/schema/"
},
"onetouchplantemplatesummary": {
        "list_endpoint": "/rundb/api/v1/onetouchplantemplatesummary/",
```

```
        "schema": "/rundb/api/v1/onetouchplantemplatesummary/schema/"
},
"plannedexperiment": {
        "list_endpoint": "/rundb/api/v1/plannedexperiment/",
        "schema": "/rundb/api/v1/plannedexperiment/schema/"
},
"plannedexperimentdb": {
        "list_endpoint": "/rundb/api/v1/plannedexperimentdb/",
        "schema": "/rundb/api/v1/plannedexperimentdb/schema/"
},
"plannedexperimentqc": {
        "list_endpoint": "/rundb/api/v1/plannedexperimentqc/",
        "schema": "/rundb/api/v1/plannedexperimentqc/schema/"
},
"plantemplatesummary": {
        "list_endpoint": "/rundb/api/v1/plantemplatesummary/",
        "schema": "/rundb/api/v1/plantemplatesummary/schema/"
},
"plugin": {
        "list_endpoint": "/rundb/api/v1/plugin/",
        "schema": "/rundb/api/v1/plugin/schema/"
},
"pluginresult": {
        "list_endpoint": "/rundb/api/v1/pluginresult/",
        "schema": "/rundb/api/v1/pluginresult/schema/"
},
"project": {
        "list_endpoint": "/rundb/api/v1/project/",
        "schema": "/rundb/api/v1/project/schema/"
},
"publisher": {
        "list_endpoint": "/rundb/api/v1/publisher/",
        "schema": "/rundb/api/v1/publisher/schema/"
},
"qctype": {
        "list_endpoint": "/rundb/api/v1/qctype/",
        "schema": "/rundb/api/v1/qctype/schema/"
},
"qualitymetrics": {
        "list_endpoint": "/rundb/api/v1/qualitymetrics/",
        "schema": "/rundb/api/v1/qualitymetrics/schema/"
},
"referencegenome": {
        "list_endpoint": "/rundb/api/v1/referencegenome/",
        "schema": "/rundb/api/v1/referencegenome/schema/"
},
"results": {
        "list_endpoint": "/rundb/api/v1/results/",
        "schema": "/rundb/api/v1/results/schema/"
},
"rig": {
        "list_endpoint": "/rundb/api/v1/rig/",
        "schema": "/rundb/api/v1/rig/schema/"
},
"runtype": {
        "list_endpoint": "/rundb/api/v1/runtype/",
        "schema": "/rundb/api/v1/runtype/schema/"
},
```

```
        "sample": {
                "list_endpoint": "/rundb/api/v1/sample/",
                "schema": "/rundb/api/v1/sample/schema/"
        },
        "sequencingkitinfo": {
                "list_endpoint": "/rundb/api/v1/sequencingkitinfo/",
                "schema": "/rundb/api/v1/sequencingkitinfo/schema/"
        },
        "sequencingkitpart": {
                "list_endpoint": "/rundb/api/v1/sequencingkitpart/",
                "schema": "/rundb/api/v1/sequencingkitpart/schema/"
        },
        "template": {
                "list_endpoint": "/rundb/api/v1/template/",
                "schema": "/rundb/api/v1/template/schema/"
        },
        "tfmetrics": {
                "list_endpoint": "/rundb/api/v1/tfmetrics/",
                "schema": "/rundb/api/v1/tfmetrics/schema/"
        },
        "threeprimeadapter": {
                "list_endpoint": "/rundb/api/v1/threeprimeadapter/",
                "schema": "/rundb/api/v1/threeprimeadapter/schema/"
        },
        "torrentsuite": {
                "list_endpoint": "/rundb/api/v1/torrentsuite/",
                "schema": "/rundb/api/v1/torrentsuite/schema/"
        },
        "user": {
                "list_endpoint": "/rundb/api/v1/user/",
                "schema": "/rundb/api/v1/user/schema/"
        }
}
```


You now have a listing of all resources available through the REST API. Notice that each resource is described by a list_endpoint and a schema, which are partial URIs. Use these URIs in subsequent calls to read and write resource data.

Get the resource data structure

Use the schema URI to get the resource data structure, which limits the names and fields of all resource data elements.

Example:

```
http://myhost/rundb/api/v1/experiment/schema?format=json
```

Get a list of experiments

This section shows how to get the experiment resource data. This example uses the experiment resource, but the experiment field in the URI could be replaced by any resource name.

> (!) By default, a maximum of 20 resource objects are returned. Add the limit parameter, as shown, to return all objects for a resource (for some resources, this may result in a large amount of data):

This request uses the experiment `list_endpoint` URI and has the following general form:

```
http://myhost/rundb/api/v1/experiment?format=json&limit=0
```

These steps show how to get experiment resource data programmatically:

1. Connect to the resource.
2. Post a request for data using the GET method.

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

In the code snippet, a base_url variable is used to define the part of the URI that is common for all requests. For the experiment resource, the following example data are returned:


```
{
    "meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
    },
    "objects": [
        {
                "autoAnalyze": true,
                "baselineRun": false,
                "chipBarcode": "",
                "chipType": "318",
                "cycles": 0,
                "date": "2013-02-08T21:00:52.000449+00:00",
                "diskusage": 0,
                "displayName": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
                "eas_set": [
                        {
                                "barcodeKitName": "",
                                "barcodedSamples": {},
                                "date": "2012-12-15T00:53:29.000428+00:00",
                                "experiment": "/rundb/api/v1/experiment/7/",
                                "hotSpotRegionBedFile": "",
                                "id": 3,
                                "isDuplicateReads": false,
                                "isEditable": true,
                                "isOneTimeOverride": false,
                                "libraryKey": "TCAG",
                                "libraryKitBarcode": null,
                                "libraryKitName": "Ion Xpress Plus Fragment Library Kit",
                                "reference": "",
                                "resource_uri": "/rundb/api/v1/experimentanalysissettings/3/",
                                "results": [
                                        "/rundb/api/v1/results/55/",
                                        "/rundb/api/v1/results/26/"
                                ],
                                "selectedPlugins": {
                                        "IonReporterUploader": {"features": ["export"], "id": 167, "
                                        "SFFCreator": {"features": [], "id": 157, "name": "SFFCreato
                                        "variantCaller": {"features": [], "id": 165, "name": "variant
                                "status": "planned",
                                "targetRegionBedFile": "",
```

```
                        "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC"
                }
        ],
        "expCompInfo": "",
        "expDir": "",
        "expName": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
        "flows": 500,
        "flowsInOrder": "",
        "ftpStatus": "Complete",
        "id": 7,
        "isReverseRun": false,
        "log": { ... },
        "metaData": {},
        "notes": "",
        "pgmName": "",
        "plan": "/rundb/api/v1/plannedexperiment/41/",
        "rawdatastyle": "single",
        "reagentBarcode": "",
        "resource_uri": "/rundb/api/v1/experiment/7/",
        "resultDate": "2013-02-08T21:00:52.000450+00:00",
        "results": [],
        "reverse_primer": null,
        "runMode": "single",
        "runtype": "GENS",
        "sample": "Example_2",
        "samples": [
                {"date": "2012-12-15T00:53:29.000428+00:00",
                "description": null,
                "displayedName": "Example_2",
                "experiments": ["/rundb/api/v1/experiment/7/"],
                "externalId": null,
                "id": 2,
                "name": "Example_2",
                "resource_uri": "/rundb/api/v1/sample/2/",
                "status": "planned"}
        ],
        "seqKitBarcode": "",
        "sequencekitbarcode": "",
        "sequencekitname": "IonPGM200Kit",
        "star": false,
        "status": "planned",
        "storageHost": null,
        "storage_options": "A",
        "unique": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
        "usePreBeadfind": false,
        "user_ack": "U"
    } ]
}
```


`meta` field The `meta` field contains data about the object data. The metadata of interest for the experiment resource is that the resource currently contains 1 experiment.

`object` field The `object` field is a list containing actual experiment data, or properties. Two elements are listed, which is also indicated by the metadata `total_count` field.

Refer to the database schema for a description of each data item.

Notice that the results data item is another URI list, containing the locations of results data for the experiment.

Get data for a specific experiment

You can get the data for a specific experiment by specifying the experiment resource primary key value in the URI, for the desired experiment. For most resources, the primary key is the id field. The exception is the rig resource, which has the name field as the primary key.

A request for the experiment whose id field is 4 has the following form:

General form

```
http://myhost/rundb/api/v1/experiment/4?format=json
```

Python snippet

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/4?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

The response includes the data for the single experiment whose id is 4. The output is similar to the output shown in *Get a list of experiments*.

Alternate method Using the primary key, you can request data simply by encoding the key value in the main part of the URI (as seen in the previous examples). If instead only one of the field properties of a resource is known, you can use an alternative method of requesting a particular resource element by passing a filtering parameter in the request.

In the following example, instead of specifying a primary key value, the experiment name field, expName, is passed as a parameter to get the same experiment resource element:

General form

```
http://myhost/rundb/api/v1/experiment?format=json&expName=5c2f8551-ac03-4c99-a9a7-83a5c0b41842
```

Python snippet

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName=5c2f8551-ac03-4c99-a9a7-83a5c0b4184'%base_url,
resp_json = resp.json()
```

Try both of these requests to verify that the same experiment data are returned.

Next

This section assumed that no errors occurred in the request-response transaction between your application and the server. In the next section, you learn about the kinds of errors that can occur as you develop more complex applications.

This section finished with a brief mention of filtering as a way of selecting a particular resource element. Following sections show the flexibility provided by filtering in selecting single or groups of resource elements.

Building on selecting the desired resource elements returned by the server, you also learn how to order, or sort, results.

Handle Errors Reported by the Server

Now that you have successfully sent API requests and processed the responses, the topic of error handling is introduced to help overcome problems that might be introduced with more complex applications.

API errors can be classified as either communication or internal server processing errors:

- Communication errors can be caused by connectivity issues, failure to authenticate or message protocol errors, which are listed in section 6 of the RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 standard.
- Internal processing errors are those detected by the Django framework and include software bugs, database anomalies and invalid request formats.

The *HTTP Response Codes* section of the *Torrent Server REST API v1 Resources* document lists errors that might typically occur when using the Torrent Suite™ Software API, and possible causes.

Topics on this page:

- *HTTP status codes*
- *Debug internal errors*

Errors detected by the server are reported in the status code returned with any create, read, update or delete request. To help determine the cause of the error, the returned message body contains additional information about the error. For internal, 500-series server errors, in particular, Django returns very detailed information about the error cause and location. Debugging must be enable to receive the more extensive Django error reports.

Using a REST client with your browser to interactively exercise the API provides an easy way to investigate error conditions, by examining the status code and message body returned with the request. When writing an application program, it is helpful to include exception handling around API calls to catch possible errors, and to always check the returned status code before continuing to process a response.

HTTP status codes

Successful API requests return a status code of 200 or 201.

All other status codes indicate some kind of error condition, and after some experience using HTTP the cause of the error can often readily be determined. To demonstrate an error condition, the following example omits the question mark (?) symbol preceding request parameters, effectively making a request on an undefined resource:

```
http://myhost/rundb/api/v1/rigformat=json
```

If you try sending this request, you will see that the server returns a 404 status code, indicating the resource was Not Found. Additionally, the response message body contains a server-specific HTML page for the 404-type error.

Debug internal errors

Turn on debugging to receive additional detailed information in the response message body for internal, 500-series errors, which are detected by the Django framework:

1. On your server, open the `settings.py` file for editing, found at the following location:

```
/opt/ion/iondb/settings.py
```

2. Set the DEBUG environment variable to True.

```
DEBUG = True
```

3. Restart Apache:

```
sudo /etc/init.d/apache2 restart
```

Thereafter, whenever a 500-series error occurs, a message similar to the following example is provided in the response message with detailed information about the type of error and the source code location where the error was detected:

```
<Response [500]>
{"error_message": "The format indicated 'application/x-www-form-urlencoded' had no available deserial
```

Filter and Sort

You can select a particular element or a group of resource elements by specifying filtering criteria. A filter may specify an exact match or a partial match using a filter qualifier.

Query results can be sorted in either ascending or descending order, using the order_by parameter and specifying the field on which to sort.

Select a Subset of Resources

All resource elements If you specify only the resource in the URI, all of the resource elements are returned. For example:

```
http://myhost/rundb/api/v1/dnabarcode/?format=json
```

A single resource element Similarly, you can select a specific resource element by providing the primary key value of the element, usually the id field:

```
http://myhost/rundb/api/v1/dnabarcode/34?format=json
```

Multiple resource elements To request multiple elements, use the set keyword following the resource name in the URI, then separate each desired element using a semicolon:

```
http://myhost/rundb/api/v1/dnabarcode/set/34;35?format=json
```

This example returns only elements with id 34 and 35.

Basic Filters

Topics on this page:

Get the resource schema and filter list When you request the resource schema, the response includes a filtering field, which is a dictionary of fields you can filter on.

Filters are used in subsequent requests by adding the filter as a request parameter and assigning the filter a value, and possibly a value qualifier. All elements that match the filter criteria are returned for the request.

General form of a schema request

```
http://myhost/rundb/api/v1/location/schema?format=json
```

Python implementation

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location/schema?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Schema request response

```
{
        "default_format": "application/json",
        "fields": {
        "comments": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "id": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "name": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "resource_uri": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": true,
                                "type": "string"
                        }
        },
        "filtering": {
                        "backupconfig": 2,
                        "comments": 2,
                        "cruncher": 2,
                        "fileserver": 2,
                        "id": 2,
                        "name": 2,
                        "rig": 2
                },
        "ordering": \[
        "backupconfig",
        "comments",
        "cruncher",
        "fileserver",
        "id",
```

```
        "name",
        "rig"
        \]
}
```

Select by filter value The location resource is used here as an example, where location contains two elements:

```
"objects": [
{
        "comments": "",
        "id": "1",
        "name": "Home",
"resource_uri": "/rundb/api/v1/location/1/"
},
{
        "comments": "Test comment.",
        "id": "2",
        "name": "testDir",
        "resource_uri": "/rundb/api/v1/location/2/"
}
]
```

Using the name field, a valid filter according to the schema, a request is made to get all elements matching the value (Home) assigned to the name parameter. Only one element is expected to match.

General form of a URI with a filter parameter

```
http://myhost/rundb/api/v1/location?format=json&name=Home
```

Python implementation of a request with a filter parameter

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location?formatjson&name=Home'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Only one location element is returned, having a name field with a value of Home:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 1
        },
        "objects": [
                {
                        "comments": "",
                        "id": "1",
                        "name": "Home",
                        "resource_uri": "/rundb/api/v1/location/1/"
                }
        ]
}
```

Specify multiple filters You can use more than one filter to select resource elements by using multiple request parameters.

General form to specify multiple filters

```
http://myhost/rundb/api/v1/experiment?format=json&cycles=0&rawdatastyle=single
```

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&cycles=0&rawdatastyle=single'%base_url,
        auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

The request to return only those elements whose cycles are 0 and whose rawdatastyle is single returns a single element:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 1
        },
        ...
}
```

Non-matching filter response Where no resource elements match your filter criteria, an empty object list is returned.

For multiple filters, all filters must match.

The following example is similar to the previous one, except that the comments fiilter is assigned a value of Test.

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location?format=json&name=Nothing could possibly have this name'%base_url,
        auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

No elements match both filter values so no elements are returned for the request, confirmed by "total_count": 0.

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 0
        },
        "objects": [ ]
}
```

Qualifying Filters

Topics on this page:

Select all experiments with a common `expName` field Filter values can be qualified so the value does not need to be an exact match to select an element.

> (!) The full set of filter qualifiers is listed in the *Filter Qualifiers* section of the *Torrent Suite™ Software API User Guide*.

Use the following syntax to specify a filter qualifier, where two underscore characters (__) separate the filter name from the filter qualifier name:

```
<filterName>__<filterQualifierName>=<value>
```

For some qualifiers, the behavior is similar to using a wildcard. The names of most qualifiers is self-explanatory, describing how it matches on a value.

In the following example, the `startswith` qualifier is used so any element whose field value "starts with" the specified value is returned, for the specified field.

General form of a URI request with a filter qualifier

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013
```

Python implementation of a filter qualifier

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName__startswith=R_2013'%base_url, auth=('myusernar
resp_json = resp.json()
```

The example returns two elements whose experiment name, expName, starts with R_2013.


```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
```

```
                .
                .
            "unique": "/results/PGM_test/sample",
            "usePreBeadfind": true
        },
        {
            "autoAnalyze": true,
            "barcodeId": "",
            "baselineRun": false,
            "chipBarcode": "AA0011641",
            "chipType": "\"314R\"",
            "cycles": 55,
            "date": "2013-11-05T18:32:00",
            "expCompInfo": "",
            "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
            "expName": "R_2013_11_05_18_32_00_user_B6--237",
                .
                .
                .
            "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
            "usePreBeadfind": true
        }
    ]
}
```

A more restrictive example This example is the same as the previous example, except that the stricter criteria are applied by specifying that the experiment name must start with R_2013_11_05. From the results of the previous example, you can see that only one element is expected to meet this qualification.

General form of a more restrictive filter qualifier

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013_11_05
```

Python implementation of a stricter filter qualifier

```
import json
import requests
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName__startswith=R_2013_11_05'%base_url, auth=('myu
resp_json = resp.json()
```

The response shows that only one element matches the `expName` filter:

```
{
    "meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
    },
    "objects": [
        {
            "autoAnalyze": true,
            "barcodeId": "",
```

```
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                }
        ]
}
```


Select experiments with a given date field The filter qualifier shown in this example works, as most qualifiers do, similar to the previous examples. Here, instead of searching for an element that "starts with" a particular value, you are searching for elements that "contain" a particular value. This example looks for elements whose date field contains the string value 2013-03.

General form of a URI with a filter qualifier on the date field

```
http://myhost/rundb/api/v1/experiment?format=json&date__icontains=2013-03
```

Python implementation of applying a filter qualifier on the date field

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&date__icontains=2013-03'%base_url,
        auth=('myusername', 'mypassword'))
```

One experiment is returned whose date field contains the string `2013-03`. Notice that the `startswith` filter qualifier could also have been used. Considerable flexibility is available to you in choosing a qualifier and the best choice depends on the application and the data set.


```
{
"meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
},
"objects": [
        {
                "autoAnalyze": true,
                "barcodeId": "",
                "baselineRun": false,
                "chipBarcode": "AA0000000",
                "chipType": "\"314R\"",
                "cycles": 27,
                "date": "2011-03-07T17:48:53",
```

```
                "expCompInfo": "",
                "expDir": "/results/PGM_test/sample",
                "expName": "R_2010_11_08_22_30_04_user_B15-45",
                        .
                        .
                        .
                "unique": "/results/PGM_test/sample",
                "usePreBeadfind": true
        }]
}
```

Sort Response Output demonstrates how to sort the returned experiment data by date.

Sort Response Output

Topics on this page:

- *Sort by date*
- *Sort in reverse order*

Sort by date To sort multiple elements, add a sort parameter to your request. Otherwise, elements are returned in the order they occur in the database.

Sorting is specified by using the keyword order_by, which works the same way as the SQL ordering statement. You must also specify the field you want to sort on. The sort parameter has the following syntax:

```
order_by=<field>
```

You can request sorting in ascending or descending alphanumeric order, as these example will demonstrate.

> (!) Elements with, for example, field values of 1, 2, 10 are returned in 1, 10, 2 order.

The first example requests elements to be sorted by the date field. This is the default form of the order_by parameter and returns elements in ascending order. (You should already be familiar with the startswith filter qualifier used in previous examples.)

General form of a sort request

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013&order_by=date
```

Python implementation of a sort request

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&order_by=date'%base_url, auth=('myusername', 'mypassw
resp_json = resp.json()
```

Two matching elements are returned, sorted in ascending order, by date:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
```

```
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                },
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
                                .
                                .
                        "unique": "/results/PGM_test/sample",
                        "usePreBeadfind": true
                }
        ]
}
```


Sort in reverse order To reverse the order of the elements returned in the previous example, add a minus symbol before the name of the field you are sorting on. This returns elements in descending order, for the specified field.

General form of a descending-order request

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013&order_by=-date
```

Python implementation of a descending-order request

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&order_by=-date'%base_url, auth=('myusername', 'mypassw
resp_json = resp.json()
```

You can see that the elements are returned in inverse order of the previous example:


```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
                                .
                                .
                        "unique": "/results/PGM_test/sample",
                        "usePreBeadfind": true
                },
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                }
        ]
}
```


The selection semantics are the same as those of the SQL SELECT statement, where a selection filter equates to the SELECT column name parameter. The API provides that any resource field, with the exception of the resource_uri, can be used as a filter. This gives considerable flexibility in selecting only the desired resource elements. Filters are specified as URI parameters, a filter=value pair, and any resource elements matching the filter criteria are returned in the response. Filter values can be further qualified using certain keywords that act as wildcards or logical operators.

Any of the fields in the resource schema ordering list can be used to sort responses. Ordering is alphanumeric, so elements with, for example, name field values of 1, 2, 10 are returned in the order 1, 10, 2.

You can use filters to retrieve metadata and analysis metrics for runs, for instance, with a given project name or genome name, or within a specific date range.

Work with the Database

Most REST operations involve reading data from the database or updating the database with data from your own applications. The following example applications use more advanced programming procedures than those already presented:

Get Run Metadata and Metrics

This section describes a more involved programming example that begins to approach an actual application. From the previous sections, you gained the basic knowledge needed to begin to write simple applications.

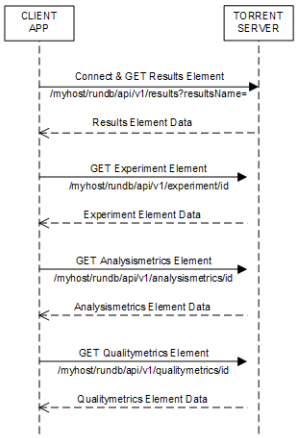
This example gets basic information about a particular run by requesting the results resource element for the run. To demonstrate getting data using links, the run gets experiment data and metrics by traversing results links to the experiment, analysismetrics and qualitymetrics resources. The program outputs experiment metadata and run metrics.

Use the following command to run the program in your Python environment:

```
getruninfo.py <runName>
```

Example: `getruninfo.py Auto_B6--237_3`

The following sequence diagram shows the request-response flow:



The program

(!) You can view the full source code at Torrent Suite API Source Code Samples.

The example uses the Python libraries requests to make REST requests to the server and simplejson to parse JSON data into Python objects. You can modify the program as needed for your libraries.

```
import requests
import simplejson as json
import sys
```

Handle the command line and save the run name input parameter. The run name is used in formatting the request to the server to return results for the particular run.

```
if len(sys.argv) == 2:
        [runName] = sys.argv[1:2]
else:
        print '\n\tUsage:   getruninfo.py <runName>'
```

```
        print '\n\tExample: getruninfo.py Auto_user_f4--134-br_21'
        sys.exit(1)
```

Connect to the server on sending the first request and GET the results element associated with the desired run name.

The requests KeyError and IndexError exceptions are also handled.

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/results?format=json&resultsName=%s'%(base_url, runName), auth=('myusername',
resp_json = resp.json()

try:
        runData = resp_json[u'objects'][0]
        experLoc = runData[u'experiment']
except (KeyError, IndexError):
        print 'ERROR: Invalid name given.'
        sys.exit(1)
```

Connect to the server on sending the first request and GET the results element associated with the desired run name.

Decode the JSON data received from the server into Python objects.

The objects field contains the run data. Of these data, the following fields are of interest for output display or for linking to additional data:

- resultsName
- timeStamp
- experiment
- analysismetrics
- qualitymetrics

Use the experiment field to get the URI of the experiment data associated with this run, and parse the JSON response data.

Display the experiment metadata from the following experiment element fields:

- project
- expName
- pgmName
- library
- notes

The dictionary KeyError exception is also handled.

```
expResult = requests.get('http://%s%s'%(myhost, experLoc))
expData = expResult.json()

try:
        print '\nProject:\t\t%s'%expData[u'log'][u'project']
        print 'Experiment Name:\t%s'%expData[u'expName']
        print 'PGM Name:\t\t%s'%expData[u'pgmName']
        print 'Library:\t\t%s'%expData[u'log'][u'library']
        print 'Notes:\t\t\t%s'%expData[u'notes']
except KeyError:
        print 'ERROR: Invalid key in expData.'
```

Display the results element data previously parsed.

```
try:
        print 'Results:\t\t%s'%runData['resultsName']
        print 'Timestamp:\t\t%s'%runData['timeStamp']
except KeyError:
        print 'ERROR: Invalid key in runData.'
```

Similar to the way you retrieved experiment data, use the analysismetrics and qualitymetrics fields to get the respective URIs for retrieving metrics data from the server. Parse the JSON response data for both elements and display the results. All returned metrics data are displayed.

```
ametricsLoc = runData[u'analysismetrics'][0]
aResult = requests.get('http://%s%s'%(myhost,ametricsLoc))
aData = aResult.json()

print '\n\nAnalysis Metrics:\n==================\n'
for propType, propVal in aData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t= %s'%(propType, propVal)

qmetricsLoc = runData[u'qualitymetrics'][0]
qResult = requests.get('http://%s%s'%(myhost,qmetricsLoc))
qData = qResult.json()

print '\n\nQuality Metrics:\n==================\n'
for propType, propVal in qData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t=%s'%(propType, propVal)
```

The output Run on a sample database, the program described above produces the following output. You should get similar results running the program against your database.

```
Project:                test
Experiment Name:        R_2012_12_05_19_34_18_user_F4--134-br
PGM Name:               f4
Library:                hg19
Notes:
Results:                barcode_test_large
Timestamp:              2013-06-06T15:28:15.000486+00:00


Analysis Metrics:
==================

libLive         = 0
ignored         = 30065
washout_ambiguous               = 0
sysIE           = 0.600278610364
bead            = 736200
tfKp            = 0
washout_live            = 0
id              = 15
libFinal                = 452234
lib             = 720367
keypass_all_beads               = 0
dud             = 15616
sysCF           = 0.877433363348
```

```
pinned          = 56051
live            = 720584
excluded                = 0
tf              = 217
empty           = 137684
tfFinal         = 200
amb             = 0
lib_pass_basecaller             = 0
lib_pass_cafie          = 0
washout_dud             = 0
libMix          = 0
report          = /rundb/api/v1/results/17/
libKp           = 0
tfLive          = 0
sysDR           = 0.0382701400667
washout_test_fragment           = 0
washout_library         = 0
washout         = 0
tfMix           = 0


Quality Metrics:
==================

q0_reads                =451883
q17_max_read_length             =173
q20_reads               =451883
report          =/rundb/api/v1/results/17/
q17_mean_read_length            =87.0
q17_100bp_reads         =263410
q0_max_read_length              =181
q20_100bp_reads         =105246
id              =15
q20_mean_read_length            =49
q17_bases               =39133239
q0_bases                =47709033
q20_150bp_reads         =6
q17_reads               =451883
q17_50bp_reads          =346855
q20_50bp_reads          =198227
q0_50bp_reads           =414922
q17_150bp_reads         =89
q0_150bp_reads          =298
q0_mean_read_length             =105.0
q20_max_read_length             =156.0
q0_100bp_reads          =333009
q20_bases               =35345630
```

Update Experiment Notes

So far, all of the examples have involved getting data from the server. This example shows you how to modify resource data by sending a PUT request to add a note to an experiment.

Get the current notes First, see what is currently stored for the experiment with id=3:

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/3/'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Among other data, the response data shows there are no notes in the notes field of experiment 3:

```
        .
        .
        .
"notes": "",
        .
        .
        .
```

Add a Note Construct a JSON notes string, using the json library dumps method.

```
metaData = json.dumps({ "notes" : "This is a sample note." })
```

For PUT and POST reqeusts, data are passed in the message body instead of as a parameter. Set the message body to the notes JSON string created, above.

Also, the JSON data format must be specified in the message header, using the form: 'content-type':'application/json'.

```
putResp = requests.put('%s/experiment/3/'%base_url,
                                                    data=metaData,
                                                    headers={'content-type':'application/json'},
                                                    auth=('myusername', 'mypassword'))
```

Now send a GET request for the same experiment to verify that the text was added to the notes field:

```
resp = requests.get('%s/experiment/3/'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Typically, you would also test the response status code to verify the action was performed successfully:

```
resp.status_code
```

The notes field now contains the string sent with the PUT request:

```
        .
        .
        .
"notes": "This is a sample note.",
        .
        .
        .
```

Add a PGM™ or Proton™ Sequencer

Topics on this page:

- *View the rig resource before adding an element*
- *Analyze the rig schema*
- *Add a rig element*

- *View the result*

In the previous example, you modified the notes field of an experiment element. In this example, you add an element to the rig resource, which is another name for the PGM™ or Proton™ Sequencer. This example also includes the added complexity of updating a resource that includes a link to another resource.

View the rig resource before adding an element First, use the `cURL` command line program or your REST client to view the rigs defined for your system. Using these tools is a convenient way to view the database while developing and debugging your program. For example:

```
http://myhost/rundb/api/v1/rig?format=json
```

This rig resource contains three PGM™ Sequencers:


```
{
"meta": {
                                .
                                .
                                .
                "total_count": 3
        },
"objects": [
{
                                        .
                                        .
                                        .
                        "name": "B6",
                        "resource_uri": "/rundb/api/v1/rig/B6/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "default",
                        "resource_uri": "/rundb/api/v1/rig/default/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "PGM_test",
                        "resource_uri": "/rundb/api/v1/rig/PGM_test/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
]
}
```


Analyze the rig schema The following example shows the JSON structure of a rig:


```
{
        "comments": "",
        "ftppassword": "ionguest",
```

```
        "ftpserver": "ts",
        "ftpusername": "ionguest",
        "location": {"comments": "", "id": "1", "name": "Home"},
        "name": "PGM_test",
        "updateflag": false,
        "updatehome": "ts"
}
```

What makes this more interesting is that the structure includes a nested dictionary for the location field, with the location schema.

When creating or modifying the rig structure, you also need to provide the location structure, either an existing location or by adding a location resource to the database before adding a rig.

In the programming example, a copy of one of the existing rigs is used but the example shows how to reference a nested dictionary.

Add a `rig` element Because the intention is to copy an existing rig data structure, modifying the desired fields, a GET request is sent to get the `rig` element `PGM_test`, to be copied.

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/rig/PGM_test?format=json'%base_url, auth=('myusername', 'mypassword'))
```

The JSON data structure of the existing rig is returned in the resp variable. Use the .json() method to get a Python json object that can be manipulated as needed.

```
resp_json = resp.json()
```

Only the program name field is changed in the copied rig data. It is changed from `PGM_test` to `myNewPgm`.

Remember that, for almost all resources, all fields must be included in the JSON string when making a PUT or POST request, not only the field you modified. An exception is the `resource_uri` field contained in all resources. The `resource_uri` field is removed in the example using the simplejson pop method.

This example demonstrates the added complication of also removing the `resource_uri` field from the nested location data structure, showing how to access nested data in the process.

```
resp_json.update(name='myNewPgm')
resp_json.pop('resource_uri')
resp_json['location'].pop('resource_uri')
```

Use the json dumps method to encode the Python object into a json string.

```
pdata = json.dumps(resp_json)
```

Use the PUT request to add the new PGM™ or Proton™ Sequencer to the database, passing the URI and message body, pdata, of the new element as parameters. You must also provide the message header and specify the content data type: {'content-type':'application/json'}.

```
status = requests.put('%s/rig/myNewPgm/'%base_url, data=pdata, headers={'content-type':'application/
```

View the result If you again use cURL or a REST client to view the `rig` resource, you can see that a PGM™ or Proton™ Sequencer named `myNewPgm` is added:

```
{
"meta": {
                                .
                                .
                                .
                "total_count": 4
        },
"objects": [
{
                                        .
                                        .
                                        .
                        "name": "B6",
                        "resource_uri": "/rundb/api/v1/rig/B6/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "default",
                        "resource_uri": "/rundb/api/v1/rig/default/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "PGM_test",
                        "resource_uri": "/rundb/api/v1/rig/PGM_test/",
                        "updateflag": false,
                        "updatehome": "ts"
                },{
                                        .
                                        .
                                        .
                        "name": "myNewPgm",
                        "resource_uri": "/rundb/api/v1/rig/myNewPgm/",
                        "updateflag": false,
                        "updatehome": "ts"
                }
]
}
```


Work with PGM™ or Proton™ Status

The `rig` resource API has a unique syntax that permits you to update individual fields. All other resources require that you provide all fields when updating the resource.

By using the status keyword, following the key or sequencer name in the URI, you can update the following `rig` resource fields, individually:

- state
- last_init_date

- last_clean_date
- last_experiment
- version
- alarms

General form of the rig status request

```
http://myhost/rundb/api/v1/PGM_test/status?format=json
```

When you update rig status, you can provide either one or all of the status items as data in the request body.

A rig status update example This example formats a status request, updating all of the status fields, and displays the results.

In the example, a simple cURL or REST client request to GET the PGM_test resource element returns the following results:

```
http://myhost/rundb/api/v1/rig/PGM_test?format=json

{"alarms": {}, "comments": "", "ftppassword": "ionguest",
"ftprootdir": "results", "ftpserver": "192.168.201.1",
"ftpusername": "anonymous", "last_clean_date": "", "last_experiment": "",
"last_init_date": "", "location": {"comments": "", "id": "1",
"name": "Home", "resource_uri": "/rundb/api/v1/location/1/"},
"name": "PGM_test", "resource_uri": "/rundb/api/v1/rig/PGM_test/",
"state": "", "updateflag": false, "updatehome": "192.168.201.1",
"version": {}}
```

You can refine the GET request to only retrieve the status fields, returning the following result:

```
http://myhost/rundb/api/v1/rig/PGM_test/status?format=json

{"alarms": {}, "last_clean_date": "", "last_experiment": "",
"last_init_date": "", "state": "", "version": {}}
```

You can see in this example that all of the field values are empty.

The programming example to update these fields uses the requests and simplejson Python libraries:

```
import requests
import simplejson as json
```

A local status variable is initialized to assign a value to each of the status fields:

```
status = {}

status["last_init_date"] = "rig.last_init_date"
status["state"] = "rig.state"
status["last_clean_date"] = "rig.last_clean_date"
status["last_experiment"] = "rig.last_experiment"
status["version"] = {"version":"test"}
status["alarms"] = {"rig.alarms":"test"}
```

And the Python status object is encoded into a JSON string:

```
pdata = json.dumps(status)
print pdata
```

The program displays the JSON string to be sent to the server in the request body:

```
{"last_clean_date": "rig.last_clean_date",
"last_experiment": "rig.last_experiment",
"state": "rig.state", "version": {"version": "test"},
"last_init_date": "just this", "alarms": {"rig.alarms": "test"}}
```

Now, send the PUT request to the server to update the status fields, providing the JSON string as data:

```
status = requests.put('http://myhost/rundb/api/v1/rig/PGM_test/status/',
                      data=pdata,
                      headers={'content-type':'application/json'},
                      auth=('myusername', 'mypassword'))
print status
```

The server returns an HTTP status code of 204, indicating a successful PUT request.

To verify that the status fields have been updated, a GET request is sent, and the response is displayed:

```
resp1 = requests.get('http://myhost/rundb/api/v1/rig/PGM_test/status/',
                      auth=('myusername', 'mypassword'))
print resp1.content
```

You can see the status fields now contain the data sent with the PUT request:

```
{"alarms": {"rig.alarms": "test"}, "last_clean_date": "rig.last_clean_date",
"last_experiment": "rig.last_experiment",
"last_init_date": "rig.last_init_date", "state": rig.state",
"version": {"version": "test"}}
```

These examples show more complex and involved database query sequences than the basic operations used to introduce REST API programming. They get run metadata then use linked fields to navigate to analysis and quality metrics associated with a run.

Some examples demonstrate how to use the PUT and POST methods to update data resource fields and to create new resource elements.

Although simple resources are shown, having a limited number of fields, the procedures demonstrated in this section apply for any of the resources exposed by the REST API.

Work with the File System

Using the API, you can find and download analysis results files.

Download a FASTQ File

This example shows the REST API facilities for working with the file system.

Making the following request on the results resource,

```
http://myhost/rundb/api/v1/results/13?format=json
```

shows the path of the associated FASTQ file. The database schema includes a number of file path entries, which can all be accessed in the same way.

```
{
                    .
                    .
                    .
```

```
        "fastqLink": "/output/Home/Auto_B15-45_4_013/R_2010_11 ... B15-45_Au...",
                .
                .
                .
}
```

You can get the file contents by copying the path to the URI, following the host name.

```
http://myhost/output/Home/Auto_B15-45_4_013/R_2010_11 ... B15-45_4.fastq
```

The entire sequence is shown in the following programming example.

The GET request on the results resource returns the FASTQ file path in the fastqLink field.

```
import requests
import simplejson as json

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/results/13?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

To GET the file contents, append the fastqLink value to the URI, following the host name.

```
resp = requests.get('http://myhost/%s'%resp_json['fastqLink'], auth=('myusername', 'mypassword'))
```

Display the FASTQ file path and the contents of the file.

```
print resp_json['fastqlink']
print resp.content
```

List File Servers

This example accesses the fileserver resource to find all file server directories.

This example demonstrates the use of the `httplib2` Python REST library:

```
import httplib2
import json
```

On the first request, perform authentication:

```
h = httplib2.Http(".cache")
h.add_credentials('myusername', 'mypassword')
```

Request all fileserver elements using the GET method:

```
resp, content = h.request("http://localhost/rundb/api/v1/fileserver?format=json", "GET")
```

Parse the JSON string response into Python objects:

```
contentdict = json.loads(content)
```

Loop through each object in the list and display the directory name:

```
objects = contentdict['objects']

for obj in objects:
        print obj['filesPrefix']
```

In addition to the FASTQ example, other files included in the results resource include the BAM, test fragment BAM, and default report files. The reference genome file can be also located and downloaded using the link in the `referencegenome` resource.

This example demonstrates the unique way files are referenced using the REST API.

Run a Plugin

You can use the API to run plugins programmatically, including passing parameters to plugins.

Topics on this page:

- *Get a list of plugins*
- *Start a plugin without parameters*
- *Start a plugin with parameters*

For more information about the plugin execution environment and plugins development, see the following documents on the Ion Community:

- Plugin SDK Documentation
- Introduction to Python Plugins

Get a list of plugins

Enter the plugin resource name in the URI to get a list of all plugins. Use the parameter `active=True` to restrict the list to the currently installed plugins.

```
http://myhost/rundb/api/v1/plugin/?format=json&active=True
```

The response includes plugin metadata and the data for each plugin in the system. Notice that, by default, the response only included 20 elements but the total_count meta field indicates there are 28 plugins. (Append `limit=0` to show all the results in a single response, or use `offset=20` to get the next 20 entries.)


```
{
        "meta": {
                "limit": 20,
                "next": "/rundb/api/v1/plugin/?offset=20&limit=20&format=json",
                "offset": 0,
                "previous": null,
                "total_count": 28
        },
        "objects": [
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.438365",
                        "id": "23",
                        "libraryName": "",
                        "name": "top100Ionogram",
                        "path": "/results/plugins/top100Ionogram",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/23/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                },
```

```
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.477418",
                        "id": "24",
                        "libraryName": "",
                        "name": "AmpliconRep",
                        "path": "/results/plugins/AmpliconRep",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/24/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                },
                        .
                        .
                        .
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.760567",
                        "id": "42",
                        "libraryName": "",
                        "name": "igv",
                        "path": "/results/plugins/igv",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/42/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                }
        ]
}
```


Specify the filtering criteria or the plugin id to retrieve the data for a single plugin.

Start a plugin without parameters

The following code snippet shows how to start a plugin that requires no parameters. (The requests and simplejson Python libraries are used, as in previous examples.)

Use a dictionary that has the plugin keyword and the plugin name as the value:

```
myPlugin = json.dumps( {"plugin": ["AmpliconRep"]} )
```

Send a POST request to run the plugin with the plugin name in the request body:

```
status = requests.post('http://myhost/rundb/api/v1/plugin/84/',
                data=myPlugin,
                headers={'content-type':'application/json'},
                auth=('myusername', 'mypassword'))
```

Start a plugin with parameters

To run a plugin requiring runtime parameters, simply add the parameters to the dictionary, as in the following code snippet, and include the plugin name and parameters in the request body:

```
myPlugin = json.dumps(
{
        "plugin": ["AmpliconRep"],
        "pluginconfig" : { "user_variables" : "foo" }
})
```

Again, send a POST request to run the plugin.

Here is a complete example using httplib2. (The shebang #! just allows for easy execution.)

```
#!/usr/bin/python
import httplib2
import json
#the primary key for the report
reportPrimaryKey = "1234"
#the name of the plugin to run
pluginName = "YOUR_PLUGIN"
h = httplib2.Http()
h.add_credentials('ionadmin', 'ionadmin')
headers = {"Content-type": "application/json","Accept": "application/json"}
url = 'http://ionwest' + '/rundb/api/v1/results/' + reportPrimaryKey + "/plugin/"
pluginUpdate  = {"plugin": [pluginName]}
resp, content = h.request(url, "POST", body=json.dumps(pluginUpdate),headers=headers )
print resp
print content
```

Write a Web Service Client

You can use the REST API to write a web service client.

Topics on this page:

- *Get a list of experiments*
- *Display results data*
- *CSS used in these examples*

Some application environments have timing complexities that make a simple request-response communication paradigm undesirable. A non-deterministic, blocking protocol, like HTTP, may take a long time to complete, causing connections to time out or degraded application performance. Using AJAX, you can achieve the asynchronous behavior needed for such applications.

These examples uses the jQuery library to show how to set up and make a REST API call and handle the pending response. The first example simply lists all experiments on the server, sorted by date. The second example uses the experiment resource link to the results data for the experiment to also display all analysis results for the experiment. (The CSS code is only provided to show the UI presentation mechanism used in the examples.)

Because JavaScript prevents data requests to servers in a different domain, the JSONP data format is used handle this limitation.

Building on the fundamental procedures shown in these examples, you might easily modify the application to also monitor the status of a run and report when analysis processing has completed. Such an application could be implemented on a mobile device to allow remote monitoring and real-time notification.

(!) The sample JavaScript code has been tested with the Chrome 11 and Firefox 3.6 browsers.

Get a list of experiments

The first example is to display the list of experiments, including the run date and the PGM™ or Proton™ Sequencer name.

Experiment	Date	PGM
R_2010_11_05_18_32_00_user_B6--237	2010-11-05T18:32:00	B6
R_2010_11_08_22_30_04_user_B15-45	2011-03-07T17:48:53	PGM_test

The HTML is simply a tag encapsulating the display. The JavaScript functions write data to the element whose id is mainPage.

```
<div id="mainPage"></div>
```

The JavaScript uses the jQuery library.

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.6.min.js"></script>
```

On loading the HTML page, a jQuery AJAX function sets up the request for experiment data:

- The URL is set to the URI for the experiment resource.
- The data type is set to jsonp to handle JavaScript cross-domain restrictions.
- A function is defined to handle the response from the server.

```
$(function(){
        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment \
                        ?format=jsonp&order_by=date",
                dataType: 'jsonp',
                success: function(json_results){
                        $('#mainPage').append('<table class="expTable">
                                                        <tr><th>Experiment</th>
                                                             <th>Date</th>
                                                             <th>PGM</th></table>');
                        listItems = $('#mainPage').find('table');
                        $.each(json_results.objects, function(key) {
                                html =  '<td>' + json_results.objects[key].expName + '</td>';
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].date + '</td>' ;
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].pgmName + '</td>' ;
                                listItems.append('<tr class="expTableRow">'+html+'</tr>');
                        });
                }
        });
});
```

Within the `id=mainPage` element, the response handling function constructs an HTML table and populates the cells with the following experiment resource fields, for each element returned in the response:

- `expName`, the experiment name.
- `date`, the date of the experiment.
- `pgmName`, the name of the PGM sequencer where the experiment was run.

Display results data

The previous example is now extended to include the location of results data associated with the experiment. This might be useful for monitoring run status.

Experiment	Date	PGM	Results
R_2010_11_05_18_32_00_user_B6--237	2010-11-05T18:32:00	B6	/rundb/api/v1/results/20/ /rundb/api/v1/results/12/
R_2010_11_08_22_30_04_user_B15-45	2011-03-07T17:48:53	PGM_test	/rundb/api/v1/results/13/

Further, by clicking on the results link in the table, results metadata are displayed:

Results Name	B6--237-TEST2
Date	2011-05-13T14:51:58.518491
Frames Processed	0
Cycles Processed	55
Status	Completed
Time to Complete	0

The JavaScript code snippet is the same as the previous example with the addition of a loop to get all of the results for an experiment and display them in the *results* column.

Notice that the results table entry includes an onClick event handler, which call the showResult function to display results metadata, passing the results location from the results field as a parameter.

```
$(function(){
        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment \
                        ?format=jsonp&order_by=date",
                dataType: 'jsonp',
                success: function(json_results){
                        $('#mainPage').append('<table class="expTable">
                                                        <tr><th>Experiment</th>
                                                                <th>Date</th><th>PGM</th>
                                                                <th>Results</th></table>')
                        listItems = $('#mainPage').find('table');
                        $.each(json_results.objects, function(key) {
                                html =  '<td>' + json_results.objects[key].expName + '</td>';
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].date + '</td>' ;
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].pgmName + '</td>' ;
                                html += '<td class="expTableCol">' ;
                for (var result in json_results.objects[key].results)
                {
                        resultPath = json_results.objects[key].results[result];
                        resultId = resultPath.split('/');
                        html += '<a href="#" onClick= \
                                                        "showResult(' + resultId[5] + '); \
                                                        return false;">' + resultPath + '</a
                }
                                html += '</td>' ;
                                listItems.append('<tr class="expTableRow">'+html+'</tr>');
                        });
                }
        });
});
```

The call to `showResult` constructs an AJAX GET request, including the `results` field parameter in the URI and,

again, specifying a jsonp data type.

```
function showResult(resultLocation) {
/\*     alert("resultLocation: " + resultLocation)  \*/

        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/results/" + \
                          resultLocation + "?format=jsonp",
                dataType: 'jsonp',
```

When a successful response is received from the server, the response handler constructs a table of metadata in the mainPage element.

The following results resource fields are appended to the table:

- `resultsName`, the name assigned to the analysis.
- `timeStamp`, the time of the analysis.
- `framesProcessed`, the number of frames processed.
- `processedCycles`, the number of cycles processed.
- `status`, the analysis status.
- `timeToComplete`, the time remaining to complete the analysis.

```
success: function(json_results){
                $('#mainPage').replaceWith('<div id="mainPage"> \
                                                        <table class="expTable"></table></div>');
                listItems = $('#mainPage').find('table');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Results Name</td><td>' + \
                                                                json_results.resultsName + '</td></
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Date</td> \
                                                                <td>' + json_results.timeStamp + '<
                                                                </tr>');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Frames Processed</td><td>' + \
                                                                json_results.framesProcessed + '</t
                                                                </tr>');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Cycles Processed</td><td>' + \
                                                                json_results.processedCycles + '</t
                                                                </tr>');
                listItems.append('<tr class="expTableRow"><td>Status</td> \
                                                                <td>' + json_results.status + '</td
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Time to Complete</td><td>' + \
                                                                json_results.timeToComplete + '</td
                }
        });

}
```

CSS used in these examples

```
#mainPage
{
```

```
        background-color: #f9f1cd;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}

ul
{
        list-style-type: none;
}

table.expTable
{
        border:none;
        border-spacing:0px;
        margin-left:25px;
        vertical-align:top;
        padding:0px;
}

table.expTable th
{
        border-color: #dddddd;
        border-width: 1px 1px 1px 1px;
        border-style: solid;
        background-color: #333333;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
        color:#f9f1cd;
        font-size:8pt;
        margin-left:25px;
        padding-right:10px;
        vertical-align:top;
}

table.expTable td
{
        border-color: #dddddd;
        border-width: 1px 1px 1px 1px;
        border-style: solid;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
        color:#666666;
        font-size:8pt;
        margin-left:25px;
        padding-right:10px;
        vertical-align:top;
}
```

The API Quick Reference provides a summary of select APIs:

Ion Torrent™ Server API Quick Reference

REST request format

Syntax

```
http://[<username>:<password>@]
        <host>/rundb/api/<version>/<resource>?format=json
        [[&<filter>{=<value> | __<qualifier>=<value>}]...][&order_by=[-]<filter>]
```

Examples

```
http://ionuser:ionuser@ionwest.itw/rundb/api/v1/experiment
        ?format=json&expName__startswith=R_2010_11&order_by=date

curl --user ionuser:ionuser --header "Content-Type: application/json"
        --location 'http://ionwest.itw/rundb/api/v1/results'
```

REST methods

Method	Function	Element URI	List URI
POST	Create	Create new entry in element	Create new entry in list
GET	Read	Get URI content	List URI members
PUT	Update	Replace URI content	Replace URI members
DELETE	Delete	Delete URI element	Delete list members

HTTP status codes

Method	Code	Meaning
	200 301	Resource exists Permanently moved
GET	401	Authorization err or
	404	Not found
	410	No longer exists
	200 201	Resource replaced Resource created
	204	No response
	301	Redirect
PUT/POST	400	Invalid data
	401	Authorization error
	409	Resource state conflict
	500	Internal error
	501	Method not implemented
DELETE	200 400	Resource deleted Resource not deleted
	401	Authorization error

Top-level requests

URI	Scope
/rundb/api/v1/	Resource list
/rundb/api/v1/ <resource>/	Resource element list (default limit: 20)
/rundb/api/v1/ <resource>?limit =0	Resource element list; all elements
/rundb/api/v1/ <resource>/<key>/	Resrouce element
/rundb/api/v1/ <resource>/set/ <key>;<key>;.../	Multiple resource elements
/rundb/api/v1/ <resource>/schema	Resource schema

Resources

- ++ *indicates PUT/POST/DELETE permitted*

- *Bold type indicates KEY field*

analysismetrics amb, bead, dud, empty, excluded, **id**, ignored, keypass_all_beads, lib, libFinal, libKp, libLive, libMix, lib_pass_basecaller, lib_pass_cafie, live, pinned, report, sysCF, sysDR, sysIE, tf, tfFinal, tfKp, tfLive, tfMix, washout, washout_ambiguous, washout_dud, washout_library, washout_live, washout_test_fragment

dnabarcode ++ adapter, annotation, floworder, **id**, index, length, name, sequence, type

experiment ++ autoAnalyze, backup, barcodeId, baselineRun, chipBarcode, chipType, cycles, date, expCompInfo, expDir, expName, flows, flowsInOrder, ftpStatus, **id**, library, libraryKey, log, metaData, notes, pgmName, project, reagentBarcode, results, sample, seqKitBarcode, star, storageHost, storage_options, unique, usePreBeadfind

fileserver comments, filesPrefix, **id**, location, name

globalconfig default_command_line, default_flow_order, default_library_key, default_plugin_script, default_storage_options, default_test_fragment_key, fasta_path, **id**, name, plugin_folder, plugin_output_folder, records_to_display, reference_path, selected, sfftrim, sfftrim_args, site_name, web_root

libmetrics See Filters (continued)

location comments, **id**, name

plugin autorun, chipType, date, **id**, libraryName, name, path, project, sample, selected, version

qualitymetrics **id**, q0_100bp_reads, q0_15bp_reads, q0_50bp_reads, q0_bases, q0_max_read_length, q0_mean_read_length, q0_reads, q17_100bp_reads, q17_150bp_reads, q17_50bp_reads, q17_bases, q17_max_read_length, q17_mean_read_length, q17_reads, q20_100bp_reads, q20_150bp_reads, q20_50bp_reads, q20_bases, q20_max_read_length, q20_mean_read_length, q20_reads, report

referencegenome bled, **id**, index_version, name, notes, reference_path, short_name, source, species, status, verbose_error, version

results ++ analysisVersion, analysismetrics, experiment, fastqLink, framesProcessed, **id**, libmetrics, log, metaData, pluginState, pluginStore, processedCycles, qualitymetrics, reportLink, reportstorage, resultsName, sffLink, status, tfFastq, tfSffLink, tfmetrics, timeStamp, timeToComplete

rig ++ alarms, comments, ftppassword, ftprootdir, ftpserver, ftpusername, last_clean_date, last_experiment, last_init_date, location, **name**, <nameValue>/status, state, updateflag, updatehome, version

runtype ++ barcode, description, **id**, runType

tfmetrics See Filters (continued)

Extended resources

Plugins

POST: *Request Header:*

```
Content-Type: application/json
```

Request Body:

```
{"plugin":["<pluginName>"]}
```

or

```
{"plugin":["<pluginName>"], pluginconfig : { json params } }
```

```
http://myhost/rundb/api/v1/results/<key>/plugin?format=json
```

Files Example:

1. From *results* resource response:

```
{"log": "/output/Home/Auto_B15-45_4_013/log.html"}
```

2. Get file:

```
http://myhost/output/Home/Auto_B15-45_4_013/log.html
```

Filter qualifiers

Usage: `<field>__<qualifier>=<value>`

Example: `library__contains=coli`

contains	icontains	istartswith	search
day	iendswith	lt	startswith
endswith	iexact	lte	week_day
exact	in	month	year
gt	iregex	range	
gte	isnull	regex	

Sort parameter

Usage: `order_by=[-]<filter>`

Examples:

(ascending) `order_by=date`
(descending) `order_by=-date`

Data format parameter

Format	Parameter	Note
JSON	?format=json	
XML	?format=xml	Not supported

Supported run types (>= 3.x)

Run Type	Description
RunType.FULLCHIP	Whole chip PGM run.
RunType.THUMB	Thumbnail run.
RunType.COMPOSITE	Proton run.

Supported run levels (>= 3.x)

Run Level	Description
RunLevel.PRE	Runs after all analysis jobs have been submitted, but before any finish.
RunLevel.BLOCK	Runs when an individual block finishes analysis. Occurs once for each block.
RunLevel.POST	Runs after all blocks are done processing.
RunLevel.LAST	Runs after everything, including other plugins. Multiple plugins w/ LAST will run at the same time.
RunLevel.DEFAULT	Default run level, generally PGM runs.

Ion Torrent™ Server API Quick Reference - 2

Return to API Quick Reference main page

Filters (continued)

libmetrics

Genome_Version, Index_Version, align_sample, aveKeyCounts, cf, dr, extrapolated_100q10_reads, extrapolated_100q17_reads, extrapolated_100q20_reads, extrapolated_100q47_reads, extrapolated_100q7_reads, extrapolated_200q10_reads, extrapolated_200q17_reads, extrapolated_200q20_reads, extrapolated_200q47_reads, extrapolated_200q7_reads, extrapolated_50q10_reads, extrapolated_50q17_reads, extrapolated_50q20_reads, extrapolated_50q47_reads, extrapolated_50q7_reads, extrapolated_from_number_of_sampled_reads, extrapolated_mapped_bases_in_q10_alignments, extrapolated_mapped_bases_in_q17_alignments, extrapolated_mapped_bases_in_q20_alignments, extrapolated_mapped_bases_in_q47_alignments, extrapolated_mapped_bases_in_q7_alignments, extrapolated_q10_alignments, extrapolated_q10_coverage_percentage, extrapolated_q10_longest_alignment, extrapolated_q10_mean_alignment_length, extrapolated_q10_mean_coverage_depth, extrapolated_q17_alignments, extrapolated_q17_coverage_percentage, extrapolated_q17_longest_alignment, extrapolated_q17_mean_alignment_length, extrapolated_q17_mean_coverage_depth, extrapolated_q20_alignments, extrapolated_q20_coverage_percentage, extrapolated_q20_longest_alignment, extrapolated_q20_mean_alignment_length, extrapolated_q20_mean_coverage_depth, extrapolated_q47_alignments, extrapolated_q47_coverage_percentage, extrapolated_q47_longest_alignment, extrapolated_q47_mean_alignment_length, extrapolated_q47_mean_coverage_depth, extrapolated_q7_alignments, extrapolated_q7_coverage_percentage, extrapolated_q7_longest_alignment, extrapolated_q7_mean_alignment_length, extrapolated_q7_mean_coverage_depth, genome, genomelength, genomesize, i100Q10_reads, i100Q17_reads, i100Q20_reads, i100Q47_reads, i100Q7_reads, i200Q10_reads, i200Q17_reads, i200Q20_reads, i200Q47_reads,

i200Q7_reads, i50Q10_reads, i50Q17_reads, i50Q20_reads, i50Q47_reads, i50Q7_reads, **id**, ie, q10_alignments, q10_coverage_percentage, q10_longest_alignment, q10_mapped_bases, q10_mean_alignment_length, q10_qscore_bases, q17_alignments, q17_coverage_percentage, q17_longest_alignment, q17_mapped_bases, q17_mean_alignment_length, q17_qscore_bases, q20_alignments, q20_coverage_percentage, q20_longest_alignment, q20_mapped_bases, q20_mean_alignment_length, q20_qscore_bases, q47_alignments, q47_coverage_percentage, q47_longest_alignment, q47_mapped_bases, q47_mean_alignment_length, q47_qscore_bases, q7_alignments, q7_coverage_percentage, q7_longest_alignment, q7_mapped_bases, q7_mean_alignment_length, q7_qscore_bases, r100Q10, r100Q17, r100Q20, r200Q10, r200Q17, r200Q20, r50Q10, r50Q17, r50Q20, rCoverage, rLongestAlign, rMeanAlignLen, rNumAlignments, report, s100Q10, s100Q17, s100Q20, s200Q10, s200Q17, s200Q20, s50Q10, s50Q17, s50Q20, sCoverage, sLongestAlign, sMeanAlignLen, sNumAlignments, sampled_100q10_reads, sampled_100q17_reads, sampled_100q20_reads, sampled_100q47_reads, sampled_100q7_reads, sampled_200q10_reads, sampled_200q17_reads, sampled_200q20_reads, sampled_200q47_reads, sampled_200q7_reads, sampled_50q10_reads, sampled_50q17_reads, sampled_50q20_reads, sampled_50q47_reads, sampled_50q7_reads, sampled_mapped_bases_in_q10_alignments, sampled_mapped_bases_in_q17_alignments, sampled_mapped_bases_in_q20_alignments, sampled_mapped_bases_in_q47_alignments, sampled_mapped_bases_in_q7_alignments, sampled_q10_alignments, sampled_q10_coverage_percentage, sampled_q10_longest_alignment, sampled_q10_mean_alignment_length, sampled_q10_mean_coverage_depth, sampled_q17_alignments, sampled_q17_coverage_percentage, sampled_q17_longest_alignment, sampled_q17_mean_alignment_length, sampled_q17_mean_coverage_depth, sampled_q20_alignments, sampled_q20_coverage_percentage, sampled_q20_longest_alignment, sampled_q20_mean_alignment_length, sampled_q20_mean_coverage_depth, sampled_q47_alignments, sampled_q47_coverage_percentage, sampled_q47_longest_alignment, sampled_q47_mean_alignment_length, sampled_q47_mean_coverage_depth, sampled_q7_alignments, sampled_q7_coverage_percentage, sampled_q7_longest_alignment, sampled_q7_mean_alignment_length, sampled_q7_mean_coverage_depth, sysSNR, totalNumReads, total_number_of_sampled_reads

tfmetrics

CF, DR, HPAccuracy, HPSNR, IE, Q10Histo, Q10Mean, Q10Mode, Q10ReadCount, Q17Histo, Q17Mean, Q17Mode, Q17ReadCount, SysSNR, aveHqReadCount, aveKeyCount, aveQ10ReadCount, aveQ17ReadCount, corOverlap, corrHPSNR, corrIonogram, error, hqReadCount, **id**, keypass, matchMismatchHisto, matchMismatchMean, matchMismatchMode, name, number, postCorrSNR, preCorrSNR, rawIonogram, rawOverlap, report, sequence

Return to API Quick Reference main page

The API Quick Reference provides a summary listing of select APIs:

2.3 Ion Torrent™ Server API Quick Reference

2.3.1 REST request format

Syntax

```
http://[<username>:<password>@]
        <host>/rundb/api/<version>/<resource>?format=json
        [[&<filter>{=<value> | __<qualifier>=<value>}]...][&order_by=[-]<filter>]
```

Examples

```
http://ionuser:ionuser@ionwest.itw/rundb/api/v1/experiment
        ?format=json&expName__startswith=R_2010_11&order_by=date
```

```
curl --user ionuser:ionuser --header "Content-Type: application/json"
        --location 'http://ionwest.itw/rundb/api/v1/results'
```

2.3.2 REST methods

Method	Function	Element URI	List URI
POST	Create	Create new entry in element	Create new entry in list
GET	Read	Get URI content	List URI members
PUT	Update	Replace URI content	Replace URI members
DELETE	Delete	Delete URI element	Delete list members

2.3.3 HTTP status codes

Method	Code	Meaning
	200 301	Resource exists Permanently moved
GET	401	Authorization err or
	404	Not found
	410	No longer exists
	200 201	Resource replaced Resource created
	204	No response
	301	Redirect
PUT/POST	400	Invalid data
	401	Authorization error
	409	Resource state conflict
	500	Internal error
	501	Method not implemented
DELETE	200 400	Resource deleted Resource not deleted
	401	Authorization error

2.3.4 Top-level requests

URI	Scope
/rundb/api/v1/	Resource list
/rundb/api/v1/ <resource>/	Resource element list (default limit: 20)
/rundb/api/v1/ <resource>?limit =0	Resource element list; all elements
/rundb/api/v1/ <resource>/<key>/	Resrouce element
/rundb/api/v1/ <resource>/set/ <key>;<key>;.../	Multiple resource elements
/rundb/api/v1/ <resource>/schema	Resource schema

2.3.5 Resources

- ++ *indicates PUT/POST/DELETE permitted*
- *Bold type indicates KEY field*

analysismetrics

amb, bead, dud, empty, excluded, **id**, ignored, keypass_all_beads, lib, libFinal, libKp, libLive, libMix, lib_pass_basecaller, lib_pass_cafie, live, pinned, report, sysCF, sysDR, sysIE, tf, tfFinal, tfKp, tfLive, tfMix, washout, washout_ambiguous, washout_dud, washout_library, washout_live, washout_test_fragment

dnabarcode ++

adapter, annotation, floworder, **id**, index, length, name, sequence, type

experiment ++

autoAnalyze, backup, barcodeId, baselineRun, chipBarcode, chipType, cycles, date, expCompInfo, expDir, expName, flows, flowsInOrder, ftpStatus, **id**, library, libraryKey, log, metaData, notes, pgmName, project, reagentBarcode, results, sample, seqKitBarcode, star, storageHost, storage_options, unique, usePreBeadfind

fileserver

comments, filesPrefix, **id**, location, name

globalconfig

default_command_line, default_flow_order, default_library_key, default_plugin_script, default_storage_options, default_test_fragment_key, fasta_path, **id**, name, plugin_folder, plugin_output_folder, records_to_display, reference_path, selected, sfftrim, sfftrim_args, site_name, web_root

libmetrics

See Filters (continued)

location

comments, **id**, name

plugin

autorun, chipType, date, **id**, libraryName, name, path, project, sample, selected, version

qualitymetrics

id, q0_100bp_reads, q0_15bp_reads, q0_50bp_reads, q0_bases, q0_max_read_length, q0_mean_read_length, q0_reads, q17_100bp_reads, q17_150bp_reads, q17_50bp_reads, q17_bases, q17_max_read_length, q17_mean_read_length, q17_reads, q20_100bp_reads, q20_150bp_reads, q20_50bp_reads, q20_bases, q20_max_read_length, q20_mean_read_length, q20_reads, report

referencegenome

bled, **id**, index_version, name, notes, reference_path, short_name, source, species, status, verbose_error, version

results ++

analysisVersion, analysismetrics, experiment, fastqLink, framesProcessed, **id**, libmetrics, log, metaData, pluginState, pluginStore, processedCycles, qualitymetrics, reportLink, reportstorage, resultsName, sffLink, status, tfFastq, tfSffLink, tfmetrics, timeStamp, timeToComplete

rig ++

alarms, comments, ftppassword, ftprootdir, ftpserver, ftpusername, last_clean_date, last_experiment, last_init_date, location, **name**, <nameValue>/status, state, updateflag, updatehome, version

runtype ++

barcode, description, **id**, runType

tfmetrics

See Filters (continued)

2.3.6 Extended resources

Plugins

POST: *Request Header:*

```
Content-Type: application/json
```

Request Body:

```
{"plugin":["<pluginName>"]}

or

{"plugin":["<pluginName>"], pluginconfig : { json params } }
```

```
http://myhost/rundb/api/v1/results/<key>/plugin?format=json
```

Files

Example:

1. From *results* resource response:

   ```
   {"log": "/output/Home/Auto_B15-45_4_013/log.html"}
   ```

2. Get file:

   ```
   http://myhost/output/Home/Auto_B15-45_4_013/log.html
   ```

2.3.7 Filter qualifiers

Usage: `<field>__<qualifier>=<value>`

Example: `library__contains=coli`

contains	icontains	istartswith	search
day	iendswith	lt	startswith
endswith	iexact	lte	week_day
exact	in	month	year
gt	iregex	range	
gte	isnull	regex	

2.3.8 Sort parameter

Usage: `order_by=[-]<filter>`

Examples:

(ascending) `order_by=date`
(descending) `order_by=-date`

2.3.9 Data format parameter

Format	Parameter	Note
JSON	?format=json	
XML	?format=xml	Not supported

2.3.10 Supported run types (>= 3.x)

Run Type	Description
RunType.FULLCHIP	Whole chip PGM run.
RunType.THUMB	Thumbnail run.
RunType.COMPOSITE	Proton run.

2.3.11 Supported run levels (>= 3.x)

Run Level	Description
RunLevel.PRE	Runs after all analysis jobs have been submitted, but before any finish.
RunLevel.BLOCK	Runs when an individual block finishes analysis. Occurs once for each block.
RunLevel.POST	Runs after all blocks are done processing.
RunLevel.LAST	Runs after everything, including other plugins. Multiple plugins w/ LAST will run at the same time.
RunLevel.DEFAULT	Default run level, generally PGM runs.

CHAPTER 3

Database

Analysis, report and configuration data are stored in a PostgreSQL database. Database items include:

- Values computed during analysis pipeline processing.
- Configuration parameters accessible using the Torrent Browser UI.

See the schema tables for a listing of all Torrent Suite™ Software database tables:

3.1 Torrent Server Database Tables

3.1.1 Database Table rundb_analysisargs

Postgres database: `iondb`

Postgres table: `rundb_analysisargs`

Analysis arguments data model.

Lifecycle

Updated when an analysis is launched or re-analyzed.

Schema

3.1.2 Database Table rundb_analysismetrics

Postgres database: `iondb`

Postgres table: `rundb_analysismetrics`

Analysis metrics data model.

Lifecycle

The `rundb_analysismetrics` table values are computed during the pipeline analysis phase.

Referenced by

- rundb_results

Schema

3.1.3 Database Table rundb_applicationgroup

Postgres database: `iondb`

Postgres table: `rundb_applicationgroup`

Application group data model.

Referenced by

- rundb_plannedexperiment

Schema

3.1.4 Database Table rundb_applproduct

Postgres database: `iondb`

Postgres table: `rundb_applproduct`

Application product data model.

Schema

3.1.5 Database Table rundb_backup

Postgres database: `iondb`

Postgres table: `rundb_backup`

Backup description data model.

Lifecycle

The `rundb_backup` table elements are updated when a particular experiment is archived.

Schema

3.1.6 Database Table rundb_backupconfig

Postgres database: `iondb`

Postgres table: `rundb_backupconfig`

Backup configuration specification data model.

Lifecycle

In previous releases, these database items are set to default values and are modified using the Torrent Browser Services tab Archive panel. In 3.6 and beyond, these are placed by the data management tables.

Schema

3.1.7 Database Table rundb_chip

Postgres database: `iondb`

Postgres table: `rundb_chip`

Chip description data model. These are the per-chip default analysis arguments.

Lifecycle

These database items are set to default values during Torrent Server installation.

Schema

3.1.8 Database Table rundb_content

Postgres database: `iondb`

Postgres table: `rundb_content`

Content description data model.

Schema

3.1.9 Database Table rundb_contentupload

Postgres database: `iondb`

Postgres table: `rundb_contentupload`

Content upload data model.

Referenced by

- rundb_content
- rundb_usereventlog

Schema

3.1.10 Database Table rundb_cruncher

Postgres database: `iondb`

Postgres table: `rundb_cruncher`

Cruncher data model.

Schema

3.1.11 Database Table rundb_dm_prune_field

Postgres database: `iondb`

Postgres table: `rundb_dm_prune_field`

Data management prune field data model.

Schema

3.1.12 Database Table rundb_dm_prune_group

Postgres database: `iondb`

Postgres table: `rundb_dm_prune_group`

Data management prune group data model.

Schema

3.1.13 Database Table rundb_dm_reports

Postgres database: `iondb`

Postgres table: `rundb_dm_reports`

Data management reports data model.

Schema

3.1.14 Database Table rundb_dmfileset

Postgres database: `iondb`

Postgres table: `rundb_dmfileset`

Data management file set data model.

Referenced by

- rundb_dmfilestat

Schema

3.1.15 Database Table rundb_dmfilestat

Postgres database: `iondb`

Postgres table: `rundb_dmfilestat`

Data management file statistics data model.

Schema

3.1.16 Database Table rundb_dnabarcode

Postgres database: `iondb`

Postgres table: `rundb_dnabarcode`

Barcode data model (for experiments that use a barcoding kit)

Lifecycle

These database elements are populated by user input on the sequencing instrument. The `rundb_experiment.barcodeId` element references this table to create the `/results/barcodeList.txt` file.

Schema

3.1.17 Database Table rundb_emailaddress

Postgres database: `iondb`

Postgres table: `rundb_emailaddress`

Report recipient email address description data model.

Lifecycle

These database items are updated in the Torrent Browser admin Configure tab Email panel.

Schema

3.1.18 Database Table rundb_eventlog

Postgres database: `iondb`

Postgres table: `rundb_eventlog`

Event log data model.

Schema

3.1.19 Database Table rundb_experiment

Postgres database: `iondb`

Postgres table: `rundb_experiment`

Experiment description data model.

Lifecycle

Each sequencing run has a corresponding `rundb_experiment` table record. These database items are set by the Crawler process, which monitors directories containing PGM or Proton experiment data and creates a record for each new experiment it finds.

Referenced by

- rundb_backup
- rundb_results

Related tables

The following tables hold data related to experiments:

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings

Schema

3.1.20 Database Table rundb_experimentanalysissettings

Postgres database: `iondb`

Postgres table: `rundb_experimentanalysissettings`

Experiment analysis settings data model.

Lifecycle

A versioned set of analysis parameter values is created when the user modifies settings for a reanalysis run.

Referenced by

- rundb_results

Related tables

The following tables hold data on experiments.

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings
- experiment

- project

Schema

3.1.21 Database Table rundb_filemonitor

Postgres database: `iondb`

Postgres table: `rundb_filemonitor`

File Monitor data model.

Referenced by

- rundb_referencegenome

Schema

3.1.22 Database Table rundb_fileserver

Postgres database: `iondb`

Postgres table: `rundb_fileserver`

Ion Torrent server file server description data model.

Schema

3.1.23 Database Table rundb_globalconfig

Postgres database: `iondb`

Postgres table: `rundb_globalconfig`

Global configuration parameters data model.

Lifecycle

These configuration items are set to default values at system installation.

Schema

3.1.24 Database Table rundb_kitinfo

Postgres database: `iondb`

Postgres table: `rundb_kitinfo`

Library kit information data model.

Referenced by

- rundb_applproduct
- rundb_kitpart

Schema

3.1.25 Database Table rundb_kitpart

Postgres database: `iondb`

Postgres table: `rundb_kitpart`

Library kit part number data model.

Schema

3.1.26 Database Table rundb_libmetrics

Postgres database: `iondb`

Postgres table: `rundb_libmetrics`

Library metrics data model.

Lifecycle

Library metrics are computed as part of the analysis pipeline.

Referenced by

- rundb_results

Schema

3.1.27 Database Table rundb_librarykey

Postgres database: `iondb`

Postgres table: `rundb_librarykey`

Library Key data model.

Schema

3.1.28 Database Table rundb_librarykit

Postgres database: `iondb`

Postgres table: `rundb_librarykit`

Library kit data model.

Schema

3.1.29 Database Table rundb_location

Postgres database: `iondb`

Postgres table: `rundb_location`

Ion Torrent server description data model.

Referenced by

- rundb_backupconfig
- rundb_cruncher
- rundb_fileserver
- rundb_rig

Schema

3.1.30 Database Table rundb_message

Postgres database: `iondb`

Postgres table: `rundb_message`

Message data model.

Schema

3.1.31 Database Table rundb_monitordata

Postgres database: `iondb`

Postgres table: `rundb_monitordata`

Monitor Data data model.

Schema

3.1.32 Database Table rundb_newspost

Postgres database: `iondb`

Postgres table: `rundb_newspost`

Newspost data model.

Lifecycle

Created when a news message is sent from Ion to Torrent Suite™ Software.

Schema

3.1.33 Database Table rundb_plannedexperiment

Postgres database: `iondb`

Postgres table: `rundb_plannedexperiment`

Planned experiment description data model.

Note: Use *this* table (not other tables in the related tables list) if you are doing LIMS integration.

Lifecycle

Created by user in the Torrent Browser Planning tab.

Referenced by

- rundb_experiment
- rundb_plannedexperimentqc

Related tables

The following other tables also hold data on experiments:

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings
- experiment
- project

Schema

3.1.34 Database Table rundb_plannedexperimentqc

Postgres database: `iondb`

Postgres table: `rundb_plannedexperimentqc`

Planned experiment QC description data model. Holds the QC threshold metrics from a planned experiment.

Related tables

The following tables also hold data on experiments:

- sample
- plannedexperiment
- plannedexperimentqc

- experimentanalysissettings
- experiment
- project

Schema

3.1.35 Database Table rundb_plugin

Postgres database: `iondb`

Postgres table: `rundb_plugin`

Plugin description data model.

Lifecycle

Created when a plugin is installed.

Referenced by

- rundb_pluginresult

Schema

3.1.36 Database Table rundb_pluginresult

Postgres database: `iondb`

Postgres table: `rundb_pluginresult`

Plugin result data model.

Schema

3.1.37 Database Table rundb_project

Postgres database: `iondb`

Postgres table: `rundb_project`

Project data model.

Schema

3.1.38 Database Table rundb_publisher

Postgres database: `iondb`

Postgres table: `rundb_publisher`

Publisher data model.

Referenced by

- rundb_content
- rundb_contentupload

Schema

3.1.39 Database Table rundb_qctype

Postgres database: `iondb`

Postgres table: `rundb_qctype`

QC type data model (the name of the QC metric, such as Bead Loading, Key Signal, or Usable Sequence).

Referenced by

- rundb_plannedexperimentqc

Schema

3.1.40 Database Table rundb_qualitymetrics

Postgres database: `iondb`

Postgres table: `rundb_qualitymetrics`

Quality metrics data model.

Lifecycle

Quality metrics are calculated during the alignment QC stage of the analysis pipeline.

Referenced by

- rundb_results

Schema

3.1.41 Database Table rundb_referencegenome

Postgres database: `iondb`

Postgres table: `rundb_referencegenome`

Reference genome description data model.

Lifecycle

This data is created when a reference genome is uploaded in the Torrent Browser. These data are used by the PGM and Proton to build a list of available genomes.

Schema

3.1.42 Database Table rundb_remoteaccount

Postgres database: `iondb`

Postgres table: `rundb_remoteaccount`

Schema

3.1.43 Database Table rundb_reportstorage

Postgres database: `iondb`

Postgres table: `rundb_reportstorage`

Report output location description data model.

Lifecycle

These data are used to generate the weekly report.

Referenced by

- rundb_results

Schema

3.1.44 Database Table rundb_results

Postgres database: `iondb`

Postgres table: `rundb_results`

Experiment results description data model.

Lifecycle

A `rundb_results` table record is created each time the analysis pipeline is executed.

Referenced by

- rundb_analysismetrics
- rundb_dmfilestat
- rundb_experimentanalysissettings
- rundb_libmetrics
- rundb_qualitymetrics
- rundb_pluginresult
- rundb_tfmetrics

Schema

3.1.45 Database Table rundb_rig

Postgres database: `iondb`

Postgres table: `rundb_rig`

PGM or Proton description data model.

Schema

3.1.46 Database Table rundb_runscript

Postgres database: `iondb`

Postgres table: `rundb_runscript`

Run script data model. The run script is the Python script that runs the analysis pipeline, adds metrics to the database, and generates reports.

Schema

3.1.47 Database Table rundb_runtype

Postgres database: `iondb`

Postgres table: `rundb_runtype`

Run type data model.

Referenced by

- rundb_applproduct

Schema

3.1.48 Database Table rundb_sample

Postgres database: `iondb`

Postgres table: `rundb_sample`

Sample data model.

Referenced by

- rundb_samplesetitem
- rundb_sampleattributevalue

Schema

3.1.49 Database Table rundb_sampleannotation_cv

Postgres database: `iondb`

Postgres table: `rundb_sampleannotation_cv`

Sample Annotation CV data model. This table corresponds to the supported sample relationships (Self | Proband, Tumor, Normal, Mother, Father, etc) in Ion Reporter™ Software.

Schema

3.1.50 Database Table rundb_sampleattribute

Postgres database: `iondb`

Postgres table: `rundb_sampleattribute`

Sample Attribute data model.

Lifecycle

Created when the user creates a sample attribute.

Referenced by

- rundb_sampleattributevalue

Schema

3.1.51 Database Table rundb_sampleattributedatatype

Postgres database: `iondb`

Postgres table: `rundb_sampleattributedatatype`

Sample Attribute Data Type data model.

Referenced by

- rundb_sampleattribute

Schema

3.1.52 Database Table rundb_sampleattributevalue

Postgres database: `iondb`

Postgres table: `rundb_sampleattributevalue`

Sample Attribute Value data model.

Lifecycle

Created when the user assigns a sample attribute value.

Schema

3.1.53 Database Table rundb_samplegrouptype_cv

Postgres database: `iondb`

Postgres table: `rundb_samplegrouptype_cv`

Sample Group Type CV data model. This table corresponds to the supported relationship types (Single, Paired, Trio, etc) in Ion Reporter™ Software and to the sample set Grouping column in the Torrent Suite™ Software.

Referenced by

- rundb_sampleannotation_cv

Schema

3.1.54 Database Table rundb_sampleset

Postgres database: `iondb`

Postgres table: `rundb_sampleset`

Sample Set data model.

Lifecycle

Created when the user creates a sample set in the Torrent Browser.

Referenced by

- rundb_samplesetitem
- rundb_plannedexperiment

Schema

3.1.55 Database Table rundb_samplesetitem

Postgres database: `iondb`

Postgres table: `rundb_samplesetitem`

Sample Set Item data model.

Lifecycle

Created when the user assigns a sample to a sample set.

Schema

3.1.56 Database Table rundb_sequencingkit

Postgres database: `iondb`

Postgres table: `rundb_sequencingkit`

Sequencing kit data model.

Schema

3.1.57 Database Table rundb_supportupload

Postgres database: `iondb`

Postgres table: `rundb_supportupload`

Schema

3.1.58 Database Table rundb_template

Postgres database: `iondb`

Postgres table: `rundb_template`

Test fragment template description data model.

Schema

3.1.59 Database Table rundb_tfmetrics

Postgres database: `iondb`

Postgres table: `rundb_tfmetrics`

Test Fragment (TF) metrics data model.

Lifecycle

TF metrics are calculated during the basecalling phase of the analysis pipeline.

Schema

3.1.60 Database Table rundb_threeprimeadapter

Postgres database: `iondb`

Postgres table: `rundb_threeprimeadapter`

Three prime adapter data model.

Schema

3.1.61 Database Table rundb_usereventlog

Postgres database: `iondb`

Postgres table: `rundb_usereventlog`

User event log data model.

Schema

3.1.62 Database Table rundb_userprofile

Postgres database: `iondb`

Postgres table: `rundb_userprofile`

User profile data model.

Schema

3.1.63 Database Table rundb_variantfrequencies

Postgres database: `iondb`

Postgres table: `rundb_variantfrequencies`

Variant frequencies data model.

Schema

PostgreSQL is an open-source object-relational Database Management System (DBMS) that supports almost all SQL constructs. PostgreSQL APIs are available for the most popular programming languages to build applications using the database for backend data store. The main user interface to PostgreSQL is the `psql` command line program. The `psql` program permits you to enter database queries directly from a terminal or to execute a query sequence from a file. Database queries demonstrated in this guide use `psql`.

CHAPTER 4

API and schema tables

4.1 Torrent Server REST API v1 Resources

4.1.1 Activeionchefprepkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeionchefprepkitinfo/", params
ts_api_response = ts_api_request.json()

activeionchefprepkitinfos = ts_api_response["objects"]

for activeionchefprepkitinfo in activeionchefprepkitinfos:
    print activeionchefprepkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeionchefprepkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "IonChefPrepKit",
            "description": "ION PGM IC 200 KIT",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "100020579",
                    "id": 20085,
                    "resource_uri": "/rundb/api/v1/kitpart/20085/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "100020580",
                    "id": 20086,
                    "resource_uri": "/rundb/api/v1/kitpart/20086/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "01",
                    "id": 20123,
                    "resource_uri": "/rundb/api/v1/kitpart/20123/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "ICPREP0001",
            "resource_uri": "/rundb/api/v1/activeionchefprepkitinfo/20042/",
            "id": 20042,
            "categories": "",
            "name": "ION PGM IC 200 KIT"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.2 Activelibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activelibrarykitinfo/", params={"
```

```
ts_api_response = ts_api_request.json()

activelibrarykitinfos = ts_api_response["objects"]

for activelibrarykitinfo in activelibrarykitinfos:
    print activelibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 14,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activelibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activelibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.3 Activepgmlibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activepgmlibrarykitinfo/", params=
ts_api_response = ts_api_request.json()

activepgmlibrarykitinfos = ts_api_response["objects"]
```

```
for activepgmlibrarykitinfo in activepgmlibrarykitinfos:
    print activepgmlibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 11,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activepgmlibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activepgmlibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.4 Activepgmsequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activepgmsequencingkitinfo/", para
ts_api_response = ts_api_request.json()

activepgmsequencingkitinfos = ts_api_response["objects"]

for activepgmsequencingkitinfo in activepgmsequencingkitinfos:
    print activepgmsequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activepgmsequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion PGM 200 Sequencing Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4474004",
                    "id": 20009,
                    "resource_uri": "/rundb/api/v1/kitpart/20009/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474005",
                    "id": 20010,
                    "resource_uri": "/rundb/api/v1/kitpart/20010/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474006",
                    "id": 20011,
                    "resource_uri": "/rundb/api/v1/kitpart/20011/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474007",
                    "id": 20012,
                    "resource_uri": "/rundb/api/v1/kitpart/20012/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0003",
            "resource_uri": "/rundb/api/v1/activepgmsequencingkitinfo/20003/",
            "id": 20003,
            "categories": "",
            "name": "IonPGM200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.5 Activeprotonlibrarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeprotonlibrarykitinfo/", para
ts_api_response = ts_api_request.json()

activeprotonlibrarykitinfos = ts_api_response["objects"]

for activeprotonlibrarykitinfo in activeprotonlibrarykitinfos:
    print activeprotonlibrarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 13,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeprotonlibrarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/activeprotonlibrarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

4.1.6 Activeprotonsequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/?format=json&li`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activeprotonsequencingkitinfo/", p
ts_api_response = ts_api_request.json()

activeprotonsequencingkitinfos = ts_api_response["objects"]

for activeprotonsequencingkitinfo in activeprotonsequencingkitinfos:
    print activeprotonsequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activeprotonsequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "Ion PI Sequencing 200 Kit v2",
            "nucleotideType": "",
            "instrumentType": "proton",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4482282",
                    "id": 20078,
                    "resource_uri": "/rundb/api/v1/kitpart/20078/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4482284",
                    "id": 20079,
                    "resource_uri": "/rundb/api/v1/kitpart/20079/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4485149",
                    "id": 20094,
                    "resource_uri": "/rundb/api/v1/kitpart/20094/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4485521",
                    "id": 20095,
                    "resource_uri": "/rundb/api/v1/kitpart/20095/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                },
                {
                    "barcode": "4484082",
                    "id": 20096,
```

```
                    "resource_uri": "/rundb/api/v1/kitpart/20096/",
                    "kit": "/rundb/api/v1/kitinfo/20044/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0012",
            "resource_uri": "/rundb/api/v1/activeprotonsequencingkitinfo/20044/",
            "id": 20044,
            "categories": "",
            "name": "ProtonI200Kit-v2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.7 Activesequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/activesequencingkitinfo/", params=
ts_api_response = ts_api_request.json()

activesequencingkitinfos = ts_api_response["objects"]

for activesequencingkitinfo in activesequencingkitinfos:
    print activesequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/activesequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion PGM 200 Sequencing Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4474004",
                    "id": 20009,
                    "resource_uri": "/rundb/api/v1/kitpart/20009/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474005",
                    "id": 20010,
                    "resource_uri": "/rundb/api/v1/kitpart/20010/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474006",
                    "id": 20011,
                    "resource_uri": "/rundb/api/v1/kitpart/20011/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                },
                {
                    "barcode": "4474007",
                    "id": 20012,
                    "resource_uri": "/rundb/api/v1/kitpart/20012/",
                    "kit": "/rundb/api/v1/kitinfo/20003/"
                }
            ],
            "flowCount": 500,
            "applicationType": "",
            "uid": "SEQ0003",
            "resource_uri": "/rundb/api/v1/activesequencingkitinfo/20003/",
            "id": 20003,
            "categories": "",
            "name": "IonPGM200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.8 Analysisargs Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/analysisargs/`

Schema URL: `http://mytorrentserver/rundb/api/v1/analysisargs/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
chipType	Unicode string data. Ex: "Hello World"		false	false	false	false	string
thumbnailalignmentargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailanalysisargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
samplePrepKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
sequenceKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
analysisargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailcalibrateargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
chip_default	Boolean data. Ex: True	false	false	false	true	false	boolean
beadfindargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
templateKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
prebasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
prethumbnailbasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
alignmentargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
thumbnailbasecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
active	Boolean data. Ex: True	true	false	false	true	false	boolean
thumbnailbeadfindargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
calibrateargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
libraryKitName	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
basecallerargs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/analysisargs/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/analysisargs/", params={"format":
ts_api_response = ts_api_request.json()

analysisargss = ts_api_response["objects"]

for analysisargs in analysisargss:
    print analysisargs
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 27,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/analysisargs/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "chipType": "314v2",
            "thumbnailalignmentargs": "",
            "thumbnailanalysisargs": "",
            "samplePrepKitName": "",
            "id": 5,
            "sequenceKitName": "",
            "analysisargs": "Analysis --from-beadfind --use-alternative-etbR-equation",
            "thumbnailcalibrateargs": "",
            "chip_default": true,
            "beadfindargs": "justBeadFind",
            "templateKitName": "",
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 20 --cal
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "",
            "active": true,
            "thumbnailbeadfindargs": "",
            "calibrateargs": "calibrate --skipDroop",
            "libraryKitName": "",
            "name": "default_314v2",
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 20",
            "resource_uri": "/rundb/api/v1/analysisargs/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post

- put
- delete
- patch

4.1.9 Analysismetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/schema/`

Perform read-only operations on `analysismetrics` resources and data elements.

Fields table

field	help text	default	nullable	re
libLive	Integer data. Ex: 2673	n/a	false	fal
ignored	Integer data. Ex: 2673	n/a	false	fal
washout_ambiguous	Integer data. Ex: 2673	n/a	false	fal
tfLive	Integer data. Ex: 2673	n/a	false	fal
sysIE	Floating point numeric data. Ex: 26.73	n/a	false	fal
bead	Integer data. Ex: 2673	n/a	false	fal
tfKp	Integer data. Ex: 2673	n/a	false	fal
washout_live	Integer data. Ex: 2673	n/a	false	fal
id	Integer data. Ex: 2673		false	fal
libFinal	Integer data. Ex: 2673	n/a	false	fal
loading	Floating point numeric data. Ex: 26.73	0.0	false	fal
lib	Integer data. Ex: 2673	n/a	false	fal
keypass_all_beads	Integer data. Ex: 2673	n/a	false	fal
dud	Integer data. Ex: 2673	n/a	false	fal
sysCF	Floating point numeric data. Ex: 26.73	n/a	false	fal
pinned	Integer data. Ex: 2673	n/a	false	fal
live	Integer data. Ex: 2673	n/a	false	fal
excluded	Integer data. Ex: 2673	n/a	false	fal
tf	Integer data. Ex: 2673	n/a	false	fal
empty	Integer data. Ex: 2673	n/a	false	fal
tfFinal	Integer data. Ex: 2673	n/a	false	fal
amb	Integer data. Ex: 2673	n/a	false	fal
lib_pass_basecaller	Integer data. Ex: 2673	n/a	false	fal
lib_pass_cafie	Integer data. Ex: 2673	n/a	false	fal
washout_dud	Integer data. Ex: 2673	n/a	false	fal
libMix	Integer data. Ex: 2673	n/a	false	fal
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	fal
libKp	Integer data. Ex: 2673	n/a	false	fal
adjusted_addressable	Integer data. Ex: 2673	0	false	fal
sysDR	Floating point numeric data. Ex: 26.73	n/a	false	fal
total	Integer data. Ex: 2673	0	false	fal
washout_test_fragment	Integer data. Ex: 2673	n/a	false	fal
washout_library	Integer data. Ex: 2673	n/a	false	fal
washout	Integer data. Ex: 2673	n/a	false	fal

Table 4.1 – continued from previous page

field	help text	default	nullable	rea
tfMix	Integer data. Ex: 2673	n/a	false	fal
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	tru

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/analysismetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/analysismetrics/", params={"format
ts_api_response = ts_api_request.json()

analysismetricss = ts_api_response["objects"]

for analysismetrics in analysismetricss:
    print analysismetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 38886,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/analysismetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "libLive": 0,
            "ignored": 219,
            "washout_ambiguous": 0,
            "tfLive": 0,
            "sysIE": 0.00782,
            "bead": 7461,
            "tfKp": 0,
            "washout_live": 0,
            "id": 1,
            "libFinal": 0,
            "loading": 0.0,
            "lib": 7197,
            "keypass_all_beads": 0,
            "dud": 208,
            "sysCF": 0.00673,
            "pinned": 21,
            "live": 7253,
            "excluded": 0,
            "tf": 56,
            "empty": 2299,
            "tfFinal": 0,
```

```
            "amb": 0,
            "lib_pass_basecaller": 0,
            "lib_pass_cafie": 0,
            "washout_dud": 0,
            "libMix": 0,
            "report": "/rundb/api/v1/results/3/",
            "libKp": 0,
            "adjusted_addressable": 0,
            "sysDR": 0.00274,
            "total": 0,
            "washout_test_fragment": 0,
            "washout_library": 0,
            "washout": 0,
            "tfMix": 0,
            "resource_uri": "/rundb/api/v1/analysismetrics/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.10 Applicationgroup Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/`

Schema URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
applications	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
uid	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/applicationgroup/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/applicationgroup/", params={"forma
ts_api_response = ts_api_request.json()

applicationgroups = ts_api_response["objects"]

for applicationgroup in applicationgroups:
    print applicationgroup
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/applicationgroup/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "DNA",
            "description": "DNA",
            "applications": [
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/",
                        "/rundb/api/v1/applicationgroup/3/",
                        "/rundb/api/v1/applicationgroup/4/"
                    ],
                    "description": "Generic Sequencing",
                    "nucleotideType": "",
                    "barcode": "",
                    "meta": {},
                    "runType": "GENS",
                    "id": 1,
                    "alternate_name": "Other",
                    "resource_uri": "/rundb/api/v1/runtype/1/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "AmpliSeq DNA",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "AMPS",
```

```
                    "id": 2,
                    "alternate_name": "AmpliSeq DNA",
                    "resource_uri": "/rundb/api/v1/runtype/2/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "TargetSeq",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "TARS",
                    "id": 3,
                    "alternate_name": "TargetSeq",
                    "resource_uri": "/rundb/api/v1/runtype/3/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/",
                        "/rundb/api/v1/applicationgroup/4/"
                    ],
                    "description": "Whole Genome",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "WGNM",
                    "id": 4,
                    "alternate_name": "Whole Genome",
                    "resource_uri": "/rundb/api/v1/runtype/4/"
                },
                {
                    "applicationGroups": [
                        "/rundb/api/v1/applicationgroup/1/"
                    ],
                    "description": "AmpliSeq Exome",
                    "nucleotideType": "dna",
                    "barcode": "",
                    "meta": {},
                    "runType": "AMPS_EXOME",
                    "id": 7,
                    "alternate_name": "AmpliSeq Exome",
                    "resource_uri": "/rundb/api/v1/runtype/7/"
                }
            ],
            "uid": "APPLGROUP_0001",
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/applicationgroup/1/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

4.1.11 Applproduct Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/applproduct/`

Schema URL: `http://mytorrentserver/rundb/api/v1/applproduct/schema/`

Fields table

field	help text	defau
isDualNucleotideTypeBySampleSupported	Boolean data. Ex: True	false
defaultHotSpotRegionBedFileName	Unicode string data. Ex: “Hello World”	n/a
isTargetRegionBEDFileSupported	Boolean data. Ex: True	true
isSamplePrepKitSupported	Boolean data. Ex: True	true
defaultPESeqKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultPELibKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultSeqKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
defaultBarcodeKitName	Unicode string data. Ex: “Hello World”	n/a
id	Integer data. Ex: 2673	
productCode	Unicode string data. Ex: “Hello World”	any
isControlSeqTypeBySampleSupported	Boolean data. Ex: True	false
defaultChipType	Unicode string data. Ex: “Hello World”	n/a
isPairedEndSupported	Boolean data. Ex: True	true
appl	A single related resource. Can be either a URI or set of nested resource data.	n/a
instrumentType	Unicode string data. Ex: “Hello World”	
isDefault	Boolean data. Ex: True	false
isTargetTechniqueSelectionSupported	Boolean data. Ex: True	true
description	Unicode string data. Ex: “Hello World”	
isHotspotRegionBEDFileSuppported	Boolean data. Ex: True	true
productName	Unicode string data. Ex: “Hello World”	n/a
isBarcodeKitSelectionRequired	Boolean data. Ex: True	false
isDefaultBarcoded	Boolean data. Ex: True	false
defaultTargetRegionBedFileName	Unicode string data. Ex: “Hello World”	n/a
isActive	Boolean data. Ex: True	true
isReferenceBySampleSupported	Boolean data. Ex: True	false
defaultFlowCount	Integer data. Ex: 2673	0
defaultLibKit	A single related resource. Can be either a URI or set of nested resource data.	n/a
barcodeKitSelectableType	Unicode string data. Ex: “Hello World”	
defaultGenomeRefName	Unicode string data. Ex: “Hello World”	n/a
isVisible	Boolean data. Ex: True	false
isDefaultPairedEnd	Boolean data. Ex: True	false
resource_uri	Unicode string data. Ex: “Hello World”	n/a

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/applproduct/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/applproduct/", params={"format": "
ts_api_response = ts_api_request.json()

applproducts = ts_api_response["objects"]

for applproduct in applproducts:
    print applproduct
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 12,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/applproduct/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isDualNucleotideTypeBySampleSupported": false,
            "defaultHotSpotRegionBedFileName": "",
            "isTargetRegionBEDFileSupported": true,
            "isSamplePrepKitSupported": true,
            "defaultPESeqKit": null,
            "defaultPELibKit": null,
            "defaultSeqKit": {
                "isActive": true,
                "kitType": "SequencingKit",
                "description": "Ion PGM Sequencing 200 Kit v2",
                "nucleotideType": "",
                "instrumentType": "pgm",
                "runMode": "",
                "parts": [
                    {
                        "barcode": "4482006",
                        "id": 20054,
                        "resource_uri": "/rundb/api/v1/kitpart/20054/",
                        "kit": "/rundb/api/v1/kitinfo/20033/"
                    },
                    {
                        "barcode": "4482007",
                        "id": 20055,
                        "resource_uri": "/rundb/api/v1/kitpart/20055/",
                        "kit": "/rundb/api/v1/kitinfo/20033/"
                    },
                    {
```

```
                "barcode": "4482008",
                "id": 20056,
                "resource_uri": "/rundb/api/v1/kitpart/20056/",
                "kit": "/rundb/api/v1/kitinfo/20033/"
            },
            {
                "barcode": "4482009",
                "id": 20057,
                "resource_uri": "/rundb/api/v1/kitpart/20057/",
                "kit": "/rundb/api/v1/kitinfo/20033/"
            }
        ],
        "flowCount": 500,
        "applicationType": "",
        "uid": "SEQ0009",
        "resource_uri": "/rundb/api/v1/kitinfo/20033/",
        "id": 20033,
        "categories": "",
        "name": "IonPGM200Kit-v2"
    },
    "defaultBarcodeKitName": null,
    "id": 20001,
    "productCode": "AMPS_0",
    "isControlSeqTypeBySampleSupported": false,
    "defaultChipType": null,
    "isPairedEndSupported": false,
    "appl": {
        "applicationGroups": [
            "/rundb/api/v1/applicationgroup/1/"
        ],
        "description": "AmpliSeq DNA",
        "nucleotideType": "dna",
        "barcode": "",
        "meta": {},
        "runType": "AMPS",
        "id": 2,
        "alternate_name": "AmpliSeq DNA",
        "resource_uri": "/rundb/api/v1/runtype/2/"
    },
    "instrumentType": "pgm",
    "isDefault": true,
    "isTargetTechniqueSelectionSupported": true,
    "description": "",
    "isHotspotRegionBEDFileSuppported": true,
    "productName": "AMPS_default",
    "isBarcodeKitSelectionRequired": false,
    "isDefaultBarcoded": false,
    "defaultTargetRegionBedFileName": "",
    "isActive": true,
    "isReferenceBySampleSupported": false,
    "defaultFlowCount": 500,
    "defaultLibKit": {
        "isActive": true,
        "kitType": "LibraryKit",
        "description": "Ion AmpliSeq 2.0 Library Kit",
        "nucleotideType": "dna",
        "instrumentType": "",
        "runMode": "",
```

```
                "parts": [
                    {
                        "barcode": "4475345",
                        "id": 20034,
                        "resource_uri": "/rundb/api/v1/kitpart/20034/",
                        "kit": "/rundb/api/v1/kitinfo/20012/"
                    }
                ],
                "flowCount": 0,
                "applicationType": "",
                "uid": "LIB0008",
                "resource_uri": "/rundb/api/v1/kitinfo/20012/",
                "id": 20012,
                "categories": "",
                "name": "Ion AmpliSeq 2.0 Library Kit"
            },
            "barcodeKitSelectableType": "all",
            "defaultGenomeRefName": "hg19",
            "isVisible": true,
            "isDefaultPairedEnd": false,
            "resource_uri": "/rundb/api/v1/applproduct/20001/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.12 Availableionchefplannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”

Table 4.3 – continued from previous page

field	help text
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]

Table 4.3 – continued from previous page

field	help text
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/?format=jso`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiment/
ts_api_response = ts_api_request.json()

availableionchefplannedexperiments = ts_api_response["objects"]

for availableionchefplannedexperiment in availableionchefplannedexperiments:
    print availableionchefplannedexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 10,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableionchefplannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "C18S2_package1_Thur_psp4",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [
                "Chef_p1v2_dev_IE"
            ],
            "notes": "148420",
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.1.17",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": "00049613",
            "seqKitBarcode": null,
            "barcodeId": "IonXpress",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/23817/",
            "bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
            "isReusable": false,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
            "librarykitname": "Ion AmpliSeq 2.0 Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
```

```
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "pending",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266238,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266238/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266237,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266237/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102054/",
        "id": 266236,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266236/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=21
"thumbnailcalibrateargs": "calibrate --skipDroop",
```

```
"templatingKitName": "Ion PI IC 200 Kit",
"runType": "AMPS_EXOME",
"username": "ionuser",
"planName": "C18S2_package1_Thur_psp4",
"sampleDisplayedName": "",
"prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
"controlSequencekitname": "",
"chefMessage": "",
"childPlans": [],
"pairedEndLibraryAdapterName": "",
"runMode": "single",
"irworkflow": "",
"planExecuted": false,
"project": "Chef_p1v2_dev_IE",
"usePostBeadfind": false,
"runname": null,
"planGUID": "994fd167-7d5a-43e5-acca-c8ffa069350e",
"planShortID": "0KXAK",
"sampleSetGroupType": null,
"sample": "",
"planExecutedDate": null,
"reverse_primer": null,
"id": 102054,
"barcodedSamples": {
    "1": {
        "barcodeSampleInfo": {
            "IonXpress_001": {
                "description": "",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                "hotSpotRegionBedFile": "",
                "nucleotideType": "DNA",
                "controlSequenceType": "",
                "externalId": ""
            }
        },
        "barcodes": [
            "IonXpress_001"
        ]
    }
},
"regionfile": "",
"selectedPlugins": {
    "coverageAnalysis": {
        "userInput": "",
        "version": "4.2-r88700",
        "features": [],
        "name": "coverageAnalysis",
        "id": 827
    },
    "validateVariantCaller": {
        "userInput": {
            "variant_caller_name": "variantCaller",
            "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
            "region": "NIST",
            "sample": "NA12878",
            "truth_minor_snp": "None",
            "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
```

```
            "truth_minor_indel": "None"
        },
        "version": "0.2.1",
        "features": [],
        "name": "validateVariantCaller",
        "id": 732
    },
    "variantCaller": {
        "userInput": {
            "torrent_variant_caller": {
                "snp_min_allele_freq": "0.10000000000000001",
                "snp_strand_bias": "0.97999999999999998",
                "hotspot_min_coverage": 6,
                "hotspot_min_cov_each_strand": 3,
                "hotspot_min_allele_freq": "0.10000000000000001",
                "snp_min_variant_score": 15,
                "hotspot_strand_bias": "0.94999999999999996",
                "hp_max_length": 8,
                "filter_insertion_predictions": "0.20000000000000001",
                "indel_min_variant_score": 20,
                "indel_min_coverage": 10,
                "heavy_tailed": 3,
                "outlier_probability": "0.01",
                "data_quality_stringency": 5,
                "snp_min_cov_each_strand": 0,
                "hotspot_min_variant_score": 10,
                "indel_strand_bias": "0.90000000000000002",
                "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                "downsample_to_coverage": 400,
                "filter_unusual_predictions": "0.25",
                "indel_min_allele_freq": "0.14999999999999999",
                "do_snp_realignment": 1,
                "prediction_precision": 1,
                "indel_min_cov_each_strand": 5,
                "filter_deletion_predictions": "0.20000000000000001",
                "suppress_recalibration": 0,
                "snp_min_coverage": 5
            },
            "meta": {
                "repository_id": "",
                "ts_version": "4.0",
                "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.20130
                "user_selections": {
                    "chip": "proton_p1",
                    "frequency": "germline",
                    "library": "ampliseq",
                    "panel": "/rundb/api/v1/contentupload/53/"
                },
                "librarytype": "ampliseq",
                "trimreads": true,
                "tooltip": "Retrieved from external file",
                "tvcargs": "tvc",
                "built_in": false,
                "configuration": "",
                "compatibility": {}
            },
            "long_indel_assembler": {
                "min_indel_size": 4,
```

```
                            "short_suffix_match": 5,
                            "output_mnv": 0,
                            "min_var_count": 5,
                            "min_var_freq": "0.14999999999999999",
                            "kmer_len": 19,
                            "max_hp_length": 8,
                            "relative_strand_bias": "0.80000000000000004"
                        },
                        "freebayes": {
                            "gen_min_coverage": 5,
                            "allow_mnps": 1,
                            "allow_complex": 0,
                            "read_max_mismatch_fraction": 1,
                            "read_mismatch_limit": 10,
                            "allow_indels": 1,
                            "min_mapping_qv": 4,
                            "gen_min_alt_allele_freq": "0.10000000000000001",
                            "allow_snps": 1,
                            "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                        }
                    },
                    "version": "4.2-r88446",
                    "features": [],
                    "name": "variantCaller",
                    "id": 826
                },
                "AmpliconStats": {
                    "userInput": "",
                    "version": "0.4.5",
                    "features": [],
                    "name": "AmpliconStats",
                    "id": 774
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-06-26T04:47:43.000023+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/availableionchefplannedexperiment/102054/",
            "reverse3primeadapter": ""
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.13 Availableionchefplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/sche`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa

Table 4.4 – continued from previous page

field	help text	default	nullable	readonly	b
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableionchefplannedexperimentsummary/?for`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableionchefplannedexperiments
ts_api_response = ts_api_request.json()

availableionchefplannedexperimentsummarys = ts_api_response["objects"]

for availableionchefplannedexperimentsummary in availableionchefplannedexperimentsummarys:
    print availableionchefplannedexperimentsummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 10,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableionchefplannedexperimentsummary/?offset=1&limit=1&format=jso
    },
    "objects": [
        {
            "isReverseRun": false,
```

```
            "planDisplayedName": "C18S2_package1_Thur_psp4",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "0KXAK",
            "planStatus": "pending",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "00049613",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102054,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI IC 200 Kit",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-26T04:47:43.000023+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "C18S2_package1_Thur_psp4",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "994fd167-7d5a-43e5-acca-c8ffa069350e",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableionchefplannedexperimentsummary/102054/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

4.1.14 Availableonetouchplannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True

Table 4.5 – continued from previous page

field	help text
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”

Table 4.5 – continued from previous page

field	help text
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment/?format=js`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment
ts_api_response = ts_api_request.json()

availableonetouchplannedexperiments = ts_api_response["objects"]

for availableonetouchplannedexperiment in availableonetouchplannedexperiments:
    print availableonetouchplannedexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 119,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableonetouchplannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [
                "auto_chip"
```

```
],
"notes": "",
"sequencekitname": "ProtonI200Kit-v3",
"storageHost": null,
"expName": "",
"cycles": null,
"isReverseRun": false,
"storage_options": "A",
"thumbnailalignmentargs": "stage1 map4",
"chipType": "P1.1.17",
"chefProgress": 0.0,
"library": "hg19",
"reverselibrarykey": "",
"sampleTubeLabel": "",
"seqKitBarcode": null,
"barcodeId": "IonXpress",
"chefLogPath": null,
"isPlanGroup": false,
"realign": false,
"sampleGroupingName": "",
"experiment": "/rundb/api/v1/experiment/23946/",
"bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
"isReusable": false,
"isDuplicateReads": false,
"thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
"librarykitname": "Ion AmpliSeq 2.0 Library Kit",
"adapter": null,
"basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
"tfKey": "ATCG",
"parentPlan": null,
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "planned",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
        "id": 266490,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266490/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
```

```
            "id": 266489,
            "qcType": {
                "description": "",
                "minThreshold": 1,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Key Signal (1-100)",
                "id": 2,
                "resource_uri": "/rundb/api/v1/qctype/2/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/266489/"
        },
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/102183/",
            "id": 266488,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/266488/"
        }
    ],
    "analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=216
    "thumbnailcalibrateargs": "calibrate --skipDroop",
    "templatingKitName": "Ion PI Template OT2 200 Kit v3",
    "runType": "AMPS_EXOME",
    "username": "ionuser",
    "planName": "-R65726-pat25_treatment_dbsa",
    "sampleDisplayedName": "",
    "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
    "controlSequencekitname": "",
    "chefMessage": "",
    "childPlans": [],
    "pairedEndLibraryAdapterName": "",
    "runMode": "single",
    "irworkflow": "",
    "planExecuted": false,
    "project": "auto_chip",
    "usePostBeadfind": false,
    "runname": null,
    "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
    "planShortID": "076B8",
    "sampleSetGroupType": null,
    "sample": "",
    "planExecutedDate": null,
    "reverse_primer": null,
    "id": 102183,
    "barcodedSamples": {
        "148541": {
            "barcodeSampleInfo": {
                "IonXpress_002": {
                    "description": "",
```

```
                    "reference": "hg19",
                    "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                    "hotSpotRegionBedFile": "",
                    "nucleotideType": "dna",
                    "controlSequenceType": "",
                    "externalId": ""
                },
                "IonXpress_001": {
                    "description": "",
                    "reference": "hg19",
                    "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/Ampl
                    "hotSpotRegionBedFile": "",
                    "nucleotideType": "dna",
                    "controlSequenceType": "",
                    "externalId": ""
                }
            },
            "barcodes": [
                "IonXpress_002",
                "IonXpress_001"
            ]
        }
    },
    "regionfile": "",
    "selectedPlugins": {
        "coverageAnalysis": {
            "userInput": "",
            "version": "4.3-r88797",
            "features": [],
            "name": "coverageAnalysis",
            "id": 828
        },
        "validateVariantCaller": {
            "userInput": {
                "variant_caller_name": "variantCaller",
                "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
                "region": "NIST",
                "sample": "NA12878",
                "truth_minor_snp": "None",
                "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
                "truth_minor_indel": "None"
            },
            "version": "0.2.1",
            "features": [],
            "name": "validateVariantCaller",
            "id": 732
        },
        "variantCaller": {
            "userInput": {
                "torrent_variant_caller": {
                    "snp_min_allele_freq": "0.10000000000000001",
                    "snp_strand_bias": "0.97999999999999998",
                    "hotspot_min_coverage": 6,
                    "hotspot_min_cov_each_strand": 3,
                    "hotspot_min_allele_freq": "0.10000000000000001",
                    "snp_min_variant_score": 15,
                    "hotspot_strand_bias": "0.94999999999999996",
                    "hp_max_length": 8,
```

```
            "filter_insertion_predictions": "0.20000000000000001",
            "indel_min_variant_score": 20,
            "indel_min_coverage": 10,
            "heavy_tailed": 3,
            "outlier_probability": "0.01",
            "data_quality_stringency": 5,
            "snp_min_cov_each_strand": 0,
            "hotspot_min_variant_score": 10,
            "indel_strand_bias": "0.90000000000000002",
            "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
            "downsample_to_coverage": 400,
            "filter_unusual_predictions": "0.25",
            "indel_min_allele_freq": "0.14999999999999999",
            "do_snp_realignment": 1,
            "prediction_precision": 1,
            "indel_min_cov_each_strand": 5,
            "filter_deletion_predictions": "0.20000000000000001",
            "suppress_recalibration": 0,
            "snp_min_coverage": 5
        },
        "meta": {
            "repository_id": "",
            "ts_version": "4.0",
            "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.20130
            "user_selections": {
                "chip": "proton_p1",
                "frequency": "germline",
                "library": "ampliseq",
                "panel": "/rundb/api/v1/contentupload/53/"
            },
            "librarytype": "ampliseq",
            "trimreads": true,
            "tooltip": "Retrieved from external file",
            "tvcargs": "tvc",
            "built_in": false,
            "configuration": "",
            "compatibility": {}
        },
        "long_indel_assembler": {
            "min_indel_size": 4,
            "short_suffix_match": 5,
            "output_mnv": 0,
            "min_var_count": 5,
            "min_var_freq": "0.14999999999999999",
            "kmer_len": 19,
            "max_hp_length": 8,
            "relative_strand_bias": "0.80000000000000004"
        },
        "freebayes": {
            "gen_min_coverage": 5,
            "allow_mnps": 1,
            "allow_complex": 0,
            "read_max_mismatch_fraction": 1,
            "read_mismatch_limit": 10,
            "allow_indels": 1,
            "min_mapping_qv": 4,
            "gen_min_alt_allele_freq": "0.10000000000000001",
            "allow_snps": 1,
```

```
                            "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                        }
                    },
                    "version": "4.2-r88446",
                    "features": [],
                    "name": "variantCaller",
                    "id": 826
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/availableonetouchplannedexperiment/102183/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.15 Availableonetouchplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/sch`

Fields table

field	help text	default	nullable	readonly	h
isReverseRun	Boolean data. Ex: True	false	false	false	tr

Table 4.6 – continued from previous page

field	help text	default	nullable	readonly	b
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa

Table 4.6 – continued from previous page

field	help text	default	nullable	readonly	b
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperimentsummary/?fo`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableonetouchplannedexperiment
ts_api_response = ts_api_request.json()

availableonetouchplannedexperimentsummarys = ts_api_response["objects"]

for availableonetouchplannedexperimentsummary in availableonetouchplannedexperimentsummarys:
    print availableonetouchplannedexperimentsummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 119,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableonetouchplannedexperimentsummary/?offset=1&limit=1&format=jso
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "076B8",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102183,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
```

```
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "-R65726-pat25_treatment_dbsa",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableonetouchplannedexperimentsummary/102183/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.16 Availableplannedexperimentsummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr

Table 4.7 – continued from previous page

field	help text	default	nullable	readonly	b
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa

Table 4.7 – continued from previous page

field	help text	default	nullable	readonly	b
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/?format=jso`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/availableplannedexperimentsummary/
ts_api_response = ts_api_request.json()

availableplannedexperimentsummarys = ts_api_response["objects"]

for availableplannedexperimentsummary in availableplannedexperimentsummarys:
    print availableplannedexperimentsummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 399,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/availableplannedexperimentsummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "-R65726-pat25_treatment_dbsa",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "076B8",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 102183,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": false,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
```

```
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": false,
            "controlSequencekitname": "",
            "date": "2014-06-27T17:51:43.000186+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "-R65726-pat25_treatment_dbsa",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "3e94bf7c-ec86-4474-8884-3ae8c16827b8",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/availableplannedexperimentsummary/102183/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.17 Chip Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/chip/`

Schema URL: `http://mytorrentserver/rundb/api/v1/chip/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”		false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
slots	Integer data. Ex: 2673	n/a	false	false	false	false	integer
instrumentType	Unicode string data. Ex: “Hello World”		false	false	true	false	string
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/chip/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/chip/", params={"format": "json",
ts_api_response = ts_api_request.json()

chips = ts_api_response["objects"]

for chip in chips:
    print chip
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/chip/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "slots": 1,
            "calibrateargs": "calibrate",
            "prebasecallerargs": "BaseCaller",
            "description": "PIv2",
            "prethumbnailbasecallerargs": "BaseCaller",
            "alignmentargs": "",
            "thumbnailbasecallerargs": "BaseCaller",
            "analysisargs": "Analysis",
            "basecallerargs": "BaseCaller",
            "thumbnailbeadfindargs": "justBeadFind",
            "thumbnailalignmentargs": "",
            "thumbnailanalysisargs": "Analysis",
            "instrumentType": "proton",
            "beadfindargs": "justBeadFind",
```

```
            "resource_uri": "/rundb/api/v1/chip/16/",
            "id": 16,
            "isActive": false,
            "name": "900AMPS_EXOME"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.18 Compositedatamanagement Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
base-call_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
in_process	Boolean data. Ex: True	false	false	false	false	false	boolean
misc_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
base-call_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
misc_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
output_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
expName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
output_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
sigproc_state	Unicode string data. Ex: "Hello World"	Unknown	false	true	false	false	string
sigproc_keep	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
diskusage	Integer data. Ex: 2673	n/a	true	false	false	false	integer
expDir	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositedatamanagement/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositedatamanagement/", params=
ts_api_response = ts_api_request.json()

compositedatamanagements = ts_api_response["objects"]

for compositedatamanagement in compositedatamanagements:
    print compositedatamanagement
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43343,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositedatamanagement/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "misc_diskspace": 0.0,
            "expName": "",
            "basecall_state": "Deleting",
            "in_process": true,
            "misc_state": "Deleted",
            "timeStamp": "2011-09-08T18:19:32.000098+00:00",
            "basecall_keep": null,
            "misc_keep": null,
            "output_keep": null,
            "basecall_diskspace": 0.0,
            "resultsName": "Auto__1",
            "output_state": "Deleted",
            "sigproc_state": "Deleted",
            "sigproc_keep": null,
            "sigproc_diskspace": null,
            "diskusage": 0,
            "resource_uri": "/rundb/api/v1/compositedatamanagement/1/",
            "expDir": "/results1/BBDefault/R_2011_08_25_16_44_20_user_BBD-43",
            "id": 1,
            "output_diskspace": 0.0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.19 Compositeexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
ftpStatus	Unicode string data. Ex: "Hello World"		false	false	true	false	string
storage_options	Unicode string data. Ex: "Hello World"	A	false	false	false	false	string
star	Boolean data. Ex: True	false	false	false	true	false	boolean
chipType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
notes	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
resultDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	true	false	false	false	datetime
flows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
repResult	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
expName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
pgmName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	false	false	false	false	datetime
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
plan	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositeexperiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositeexperiment/", params={"fo
ts_api_response = ts_api_request.json()

compositeexperiments = ts_api_response["objects"]

for compositeexperiment in compositeexperiments:
    print compositeexperiment
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20366,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositeexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "chipInstrumentType": "proton",
            "chipType": "P1.1.17",
            "results": [
                {
                    "status": "Completed",
                    "processedflows": 0,
                    "analysis_metrics": {
                        "ignored": 1364892,
                        "lib": 118063544,
                        "total_wells": 164699136,
                        "pinned": 463867,
                        "live": 118074386,
                        "excluded": 16095180,
                        "bead": 118525043,
                        "resource_uri": "",
                        "id": 41726,
                        "empty": 28250154,
                        "libFinal": 85604279
                    },
                    "timeStamp": "2014-06-28T07:11:58.000789+00:00",
                    "analysismetrics": {
                        "ignored": 1364892,
                        "lib": 118063544,
                        "total_wells": 164699136,
                        "pinned": 463867,
                        "live": 118074386,
                        "excluded": 16095180,
                        "bead": 118525043,
                        "resource_uri": "",
                        "id": 41726,
                        "empty": 28250154,
                        "libFinal": 85604279
                    },
                    "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
                    "id": 304393,
                    "reportStatus": "Nothing",
                    "quality_metrics": {
                        "q0_mean_read_length": 178.362197969099,
                        "q0_reads": 85604279,
                        "q0_bases": "15268567358",
                        "q20_reads": 85604279,
                        "q20_bases": "13060288783",
                        "q20_mean_read_length": 178,
                        "id": 39683,
                        "resource_uri": ""
                    },
```

```
            "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
            "projects": [
                {
                    "resource_uri": "",
                    "id": 1385,
                    "name": "auto_chip",
                    "modified": "2014-06-27T21:21:43.000081+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 178.362197969099,
                "q0_reads": 85604279,
                "q0_bases": "15268567358",
                "q20_reads": 85604279,
                "q20_bases": "13060288783",
                "q20_mean_read_length": 178,
                "id": 39683,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/compositeresult/304393/",
            "libmetrics": {
                "i100Q20_reads": 61016174,
                "aveKeyCounts": 89.0,
                "id": 41874,
                "resource_uri": "",
                "q20_mean_alignment_length": 159
            },
            "autoExempt": false,
            "representative": false
        },
        {
            "status": "Completed",
            "processedflows": 520,
            "analysis_metrics": {
                "ignored": 9939,
                "lib": 726685,
                "total_wells": 960000,
                "pinned": 44080,
                "live": 726723,
                "excluded": 0,
                "bead": 738396,
                "resource_uri": "",
                "id": 41695,
                "empty": 167585,
                "libFinal": 517179
            },
            "timeStamp": "2014-06-28T00:18:42.000351+00:00",
            "analysismetrics": {
                "ignored": 9939,
                "lib": 726685,
                "total_wells": 960000,
                "pinned": 44080,
                "live": 726723,
```

```
                "excluded": 0,
                "bead": 738396,
                "resource_uri": "",
                "id": 41695,
                "empty": 167585,
                "libFinal": 517179
            },
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_tn_304394/"
            "id": 304394,
            "reportStatus": "Nothing",
            "quality_metrics": {
                "q0_mean_read_length": 176.041268110267,
                "q0_reads": 517179,
                "q0_bases": "91044847",
                "q20_reads": 517179,
                "q20_bases": "77321419",
                "q20_mean_read_length": 176,
                "id": 39658,
                "resource_uri": ""
            },
            "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958_tn",
            "projects": [
                {
                    "resource_uri": "",
                    "id": 1385,
                    "name": "auto_chip",
                    "modified": "2014-06-27T21:21:43.000081+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 176.041268110267,
                "q0_reads": 517179,
                "q0_bases": "91044847",
                "q20_reads": 517179,
                "q20_bases": "77321419",
                "q20_mean_read_length": 176,
                "id": 39658,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/compositeresult/304394/",
            "libmetrics": {
                "i100Q20_reads": 358815,
                "aveKeyCounts": 88.0,
                "id": 41849,
                "resource_uri": "",
                "q20_mean_alignment_length": 155
            },
            "autoExempt": false,
            "representative": false
        }
    ],
    "library": "hg19",
    "sample": "148541",
```

```
            "runMode": "single",
            "storage_options": "D",
            "repResult": "/rundb/api/v1/compositeresult/304394/",
            "id": 23958,
            "archived": false,
            "barcodeId": "IonXpress",
            "sampleSetName": "",
            "star": false,
            "resultDate": "2014-06-28T07:11:58.000789+00:00",
            "flows": 520,
            "plan": {
                "runType": "AMPS_EXOME",
                "id": 102195,
                "resource_uri": ""
            },
            "date": "2014-06-27T21:19:01+00:00",
            "ftpStatus": "Complete",
            "notes": "",
            "chipDescription": "PI",
            "pgmName": "Z28",
            "keep": false,
            "expName": "R_2014_06_27_17_13_22_user_Z28-428--r65714-pou4_dbsa",
            "resource_uri": "/rundb/api/v1/compositeexperiment/23958/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.20 Compositeresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/compositeresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/compositeresult/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
analysismetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
qualitymetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
libmetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
representative	Boolean data. Ex: True	false	false	false	true	false	boolean

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/compositeresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/compositeresult/", params={"format
ts_api_response = ts_api_request.json()

compositeresults = ts_api_response["objects"]

for compositeresult in compositeresults:
    print compositeresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/compositeresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Completed",
            "processedflows": 0,
            "analysis_metrics": {
                "ignored": 3003404,
                "lib": 134282829,
                "total_wells": 164699136,
                "pinned": 472926,
                "live": 135574135,
                "excluded": 16095180,
                "bead": 135800957,
                "resource_uri": "",
                "id": 31763,
                "empty": 9326669,
                "libFinal": 91521301
            },
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "analysismetrics": {
                "ignored": 3003404,
                "lib": 134282829,
                "total_wells": 164699136,
                "pinned": 472926,
                "live": 135574135,
                "excluded": 16095180,
                "bead": 135800957,
                "resource_uri": "",
                "id": 31763,
                "empty": 9326669,
                "libFinal": 91521301
            },
            "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
            "id": 293943,
            "reportStatus": "Nothing",
            "quality_metrics": {
                "q0_mean_read_length": 168.0,
                "q0_reads": 91521301,
                "q0_bases": "15380233572",
                "q20_reads": 91521301,
                "q20_bases": "12209924742",
                "q20_mean_read_length": 103,
                "id": 31678,
                "resource_uri": ""
            },
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "projects": [
                {
                    "resource_uri": "",
```

```
                    "id": 1080,
                    "name": "chef_827_909_20min_ext",
                    "modified": "2014-01-22T18:50:10.000920+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 168.0,
                "q0_reads": 91521301,
                "q0_bases": "15380233572",
                "q20_reads": 91521301,
                "q20_bases": "12209924742",
                "q20_mean_read_length": 103,
                "id": 31678,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/compositeresult/293943/",
            "libmetrics": {
                "i100Q20_reads": 56284561,
                "aveKeyCounts": 71.0,
                "id": 32368,
                "resource_uri": "",
                "q20_mean_alignment_length": 142
            },
            "autoExempt": false,
            "representative": false
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.21 Content Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/content/`

Schema URL: `http://mytorrentserver/rundb/api/v1/content/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
publisher	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
contentupload	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
meta	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
file	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
path	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/content/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/content/", params={"format": "jso
ts_api_response = ts_api_request.json()

contents = ts_api_response["objects"]

for content in contents:
    print content
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 72,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/content/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "publisher": "/rundb/api/v1/publisher/BED/",
            "contentupload": "/rundb/api/v1/contentupload/16/",
            "meta": {
                "is_ampliseq": false,
                "hotspot_bed": null,
                "reference": "hg19",
```

```
                "primary_bed": "/results/uploads/BED/16/testPanel30.bed",
                "hotspot": false
            },
            "file": "/results/uploads/BED/16/hg19/unmerged/plain/testPanel30.bed",
            "path": "/hg19/unmerged/plain/testPanel30.bed",
            "id": 53,
            "resource_uri": "/rundb/api/v1/content/53/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.22 Contentupload Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/contentupload/`

Schema URL: `http://mytorrentserver/rundb/api/v1/contentupload/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
meta	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	string
file_path	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/contentupload/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/contentupload/", params={"format"
ts_api_response = ts_api_request.json()

contentuploads = ts_api_response["objects"]

for contentupload in contentuploads:
    print contentupload
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 25,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/contentupload/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Successfully Completed",
            "meta": {
                "upload_date": "2014-03-27T00:28:46",
                "description": "Comp Cancer Panel",
                "reference": "hg19",
                "is_ampliseq": true,
                "hotspot": true,
                "choice": "proton",
                "design": {
                    "status": "ORDERABLE",
                    "pipeline": "DNA",
                    "min_number_amplicons_per_pool": 3991,
                    "type": "FIXED_PANEL",
                    "description": "<p>The Ion AmpliSeq&trade; Comprehensive Cancer Panel provides hi
                    "order_number": 90,
                    "design_name": "Comp Cancer Panel",
                    "results_uri": "/ws/tmpldesign/14011153/download/results",
                    "pipeline_version": null,
                    "request_id_and_solution_ordering_id": "CCP",
                    "configuration_choices": [
                        "pgm",
                        "proton"
                    ],
                    "target_size": 1293547,
                    "genome": "HG19",
                    "solution_name": null,
                    "created_date": "2013-10-07T14:21:51.388+0000",
                    "plan": {
                        "missed_bed": null,
                        "hotspot_bed": "CCP.20131001.hotspots.bed",
                        "coverage_summary": null,
                        "designed_bed": "CCP.20131001.designed.bed",
                        "target_mutations": null,
                        "primer_bed": null,
                        "selectedPlugins": {
                            "variantCaller": {
                                "features": [],
                                "ampliSeqVariantCallerConfig": {
                                    "torrent_variant_caller": {
                                        "snp_min_allele_freq": "0.02",
                                        "snp_strand_bias": "0.95",
                                        "hotspot_min_coverage": "6",
                                        "hotspot_min_cov_each_strand": "2",
                                        "hotspot_min_allele_freq": "0.01",
                                        "snp_min_variant_score": "6",
                                        "hotspot_strand_bias": "0.95",
```

```
            "hp_max_length": "8",
            "filter_insertion_predictions": "0.2",
            "indel_min_variant_score": "6",
            "indel_min_coverage": "15",
            "heavy_tailed": "3",
            "outlier_probability": "0.005",
            "data_quality_stringency": "6.5",
            "snp_min_cov_each_strand": "0",
            "hotspot_min_variant_score": "6",
            "indel_strand_bias": "0.9",
            "downsample_to_coverage": "2000",
            "filter_unusual_predictions": "0.3",
            "indel_min_allele_freq": "0.05",
            "do_snp_realignment": "1",
            "prediction_precision": "1.0",
            "indel_min_cov_each_strand": "2",
            "filter_deletion_predictions": "0.2",
            "suppress_recalibration": "0",
            "snp_min_coverage": "6"
        },
        "meta": {
            "repository_id": "",
            "ts_version": "4.0",
            "name": "Panel-optimized - Comp Cancer Panel",
            "user_selections": {
                "chip": "proton_p1",
                "frequency": "germline",
                "library": "ampliseq",
                "panel": "/rundb/api/v1/contentupload/48/"
            },
            "trimreads": true,
            "tooltip": "Panel-optimized parameters from AmpliSeq.com",
            "tvcargs": "tvc",
            "built_in": true,
            "configuration": "",
            "compatibility": {
                "panel": "/rundb/api/v1/contentupload/48/"
            }
        },
        "long_indel_assembler": {
            "min_indel_size": "4",
            "short_suffix_match": "5",
            "output_mnv": "0",
            "min_var_count": "5",
            "min_var_freq": "0.15",
            "kmer_len": "19",
            "max_hp_length": "8",
            "relative_strand_bias": "0.8"
        },
        "freebayes": {
            "gen_min_coverage": "6",
            "allow_mnps": "1",
            "allow_complex": "0",
            "read_max_mismatch_fraction": "1.0",
            "read_mismatch_limit": "10",
            "allow_indels": "1",
            "min_mapping_qv": "4",
            "gen_min_alt_allele_freq": "0.035",
```

```
                "allow_snps": "1",
                "gen_min_indel_alt_allele_freq": "0.1"
            }
        },
        "userInput": {
            "torrent_variant_caller": {
                "snp_min_allele_freq": "0.02",
                "snp_strand_bias": "0.95",
                "hotspot_min_coverage": "6",
                "hotspot_min_cov_each_strand": "2",
                "hotspot_min_allele_freq": "0.01",
                "snp_min_variant_score": "6",
                "hotspot_strand_bias": "0.95",
                "hp_max_length": "8",
                "filter_insertion_predictions": "0.2",
                "indel_min_variant_score": "6",
                "indel_min_coverage": "15",
                "heavy_tailed": "3",
                "outlier_probability": "0.005",
                "data_quality_stringency": "6.5",
                "snp_min_cov_each_strand": "0",
                "hotspot_min_variant_score": "6",
                "indel_strand_bias": "0.9",
                "downsample_to_coverage": "2000",
                "filter_unusual_predictions": "0.3",
                "indel_min_allele_freq": "0.05",
                "do_snp_realignment": "1",
                "prediction_precision": "1.0",
                "indel_min_cov_each_strand": "2",
                "filter_deletion_predictions": "0.2",
                "suppress_recalibration": "0",
                "snp_min_coverage": "6"
            },
            "meta": {
                "repository_id": "",
                "ts_version": "4.0",
                "name": "Panel-optimized - Comp Cancer Panel",
                "user_selections": {
                    "chip": "proton_p1",
                    "frequency": "germline",
                    "library": "ampliseq",
                    "panel": "/rundb/api/v1/contentupload/48/"
                },
                "trimreads": true,
                "tooltip": "Panel-optimized parameters from AmpliSeq.com",
                "tvcargs": "tvc",
                "built_in": true,
                "configuration": "",
                "compatibility": {
                    "panel": "/rundb/api/v1/contentupload/48/"
                }
            },
            "long_indel_assembler": {
                "min_indel_size": "4",
                "short_suffix_match": "5",
                "output_mnv": "0",
                "min_var_count": "5",
                "min_var_freq": "0.15",
```

```
                                        "kmer_len": "19",
                                        "max_hp_length": "8",
                                        "relative_strand_bias": "0.8"
                                    },
                                    "freebayes": {
                                        "gen_min_coverage": "6",
                                        "allow_mnps": "1",
                                        "allow_complex": "0",
                                        "read_max_mismatch_fraction": "1.0",
                                        "read_mismatch_limit": "10",
                                        "allow_indels": "1",
                                        "min_mapping_qv": "4",
                                        "gen_min_alt_allele_freq": "0.035",
                                        "allow_snps": "1",
                                        "gen_min_indel_alt_allele_freq": "0.1"
                                    }
                                },
                                "version": "4.1-r74477",
                                "id": 698,
                                "name": "variantCaller"
                            }
                        },
                        "coverage_detail": null,
                        "primer_sequences": "CCP.20131001.primerDataSheet.csv",
                        "runType": "AMPS",
                        "submitted_bed": null,
                        "well_plate_data": null
                    },
                    "design_id": "CCP",
                    "number_of_amplicons": 15992,
                    "id": 14011153,
                    "amplicons_coverage_summary": "95.349763093262169",
                    "number_of_amplicon_pools": 4
                }
            },
            "file_path": "/results/uploads/BED/48/CCP.20131001.results.zip",
            "resource_uri": "/rundb/api/v1/contentupload/48/",
            "id": 48
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.23 Datamanagementhistory Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/`

Schema URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
username	Unicode string data. Ex: "Hello World"	ION	false	false	true	false	string
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	date-time
text	Unicode string data. Ex: "Hello World"		false	false	false	false	string
object_pk	Integer data. Ex: 2673	n/a	false	false	false	false	integer
resultsName	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/datamanagementhistory/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/datamanagementhistory/", params={'
ts_api_response = ts_api_request.json()

datamanagementhistorys = ts_api_response["objects"]

for datamanagementhistory in datamanagementhistorys:
    print datamanagementhistory
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 474229,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/datamanagementhistory/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "username": "ION",
            "created": "2013-03-05T15:15:09.000925+00:00",
            "text": "Created DMFileStat (Signal Processing Input)",
            "object_pk": 274692,
```

```
            "resultsName": null,
            "id": 114023,
            "resource_uri": "/rundb/api/v1/datamanagementhistory/114023/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.24 Dnabarcode Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/`

Schema URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/schema/`

Perform CRUD operations on DNABARCODE resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
index	Integer data. Ex: 2673	n/a	false	false	false	false	integer
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
score_cutoff	Floating point numeric data. Ex: 26.73	0	false	false	false	false	float
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
floworder	Unicode string data. Ex: “Hello World”		false	false	true	false	string
adapter	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
length	Integer data. Ex: 2673	0	false	false	true	false	integer
id_str	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
score_mode	Integer data. Ex: 2673	0	false	false	true	false	integer
type	Unicode string data. Ex: “Hello World”		false	false	true	false	string
annotation	Unicode string data. Ex: “Hello World”		false	false	true	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/dnabarcode/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/dnabarcode/", params={"format": "
ts_api_response = ts_api_request.json()

dnabarcodes = ts_api_response["objects"]

for dnabarcode in dnabarcodes:
    print dnabarcode
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9387,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/dnabarcode/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "index": 9,
            "name": "IonXpress",
            "score_cutoff": 2.0,
            "sequence": "TGAGCGGAAC",
            "floworder": "",
            "adapter": "GAT",
            "id": 761,
            "length": 10,
            "id_str": "IonXpress_009",
            "score_mode": 1,
            "type": "",
            "annotation": "",
            "resource_uri": "/rundb/api/v1/dnabarcode/761/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.25 Emailaddress Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/emailaddress/`

Schema URL: `http://mytorrentserver/rundb/api/v1/emailaddress/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
selected	Boolean data. Ex: True		false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
email	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/emailaddress/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/emailaddress/", params={"format":
ts_api_response = ts_api_request.json()

emailaddresss = ts_api_response["objects"]

for emailaddress in emailaddresss:
    print emailaddress
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/emailaddress/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "selected": true,
            "resource_uri": "/rundb/api/v1/emailaddress/2/",
            "email": "bernard.puc@lifetech.com",
            "id": 2
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.26 Eventlog Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/eventlog/`

Schema URL: `http://mytorrentserver/rundb/api/v1/eventlog/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
username	Unicode string data. Ex: "Hello World"	ION	false	false	true	false	string
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	date-time
text	Unicode string data. Ex: "Hello World"		false	false	false	false	string
object_pk	Integer data. Ex: 2673	n/a	false	false	false	false	integer
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/eventlog/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/eventlog/", params={"format": "jso
ts_api_response = ts_api_request.json()

eventlogs = ts_api_response["objects"]

for eventlog in eventlogs:
    print eventlog
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 625153,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/eventlog/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
```

```
            "username": "ION",
            "created": "2012-07-03T15:14:33.000511+00:00",
            "text": "Created during migration from Experiment project label.",
            "object_pk": 1,
            "id": 1,
            "resource_uri": "/rundb/api/v1/eventlog/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.27 Experiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/experiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/experiment/schema/`

Perform CRUD operations on `experiment` resources and data elements.

Fields table

field	help text	default
isReverseRun	Boolean data. Ex: True	false
storage_options	Unicode string data. Ex: “Hello World”	A
chipType	Unicode string data. Ex: “Hello World”	n/a
user_ack	Unicode string data. Ex: “Hello World”	U
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
sample	Unicode string data. Ex: “Hello World”	n/a
runMode	Unicode string data. Ex: “Hello World”	
reverse_primer	Unicode string data. Ex: “Hello World”	n/a
seqKitBarcode	Unicode string data. Ex: “Hello World”	
id	Integer data. Ex: 2673	
metaData	Unicode string data. Ex: “Hello World”	{}
log	Unicode string data. Ex: “Hello World”	{}
sequencekitbarcode	Unicode string data. Ex: “Hello World”	n/a
resource_uri	Unicode string data. Ex: “Hello World”	n/a
eas_set	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
runtype	Unicode string data. Ex: “Hello World”	n/a
platform	Unicode string data. Ex: “Hello World”	
samples	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
pinnedRepResult	Boolean data. Ex: True	false

Table 4.8 – continued from previous page

field	help text	default
reagentBarcode	Unicode string data. Ex: “Hello World”	
star	Boolean data. Ex: True	false
isProton	Unicode string data. Ex: “Hello World”	n/a
expCompInfo	Unicode string data. Ex: “Hello World”	
resultDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true
flows	Integer data. Ex: 2673	n/a
plan	A single related resource. Can be either a URI or set of nested resource data.	n/a
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a
diskusage	Integer data. Ex: 2673	n/a
unique	Unicode string data. Ex: “Hello World”	n/a
expDir	Unicode string data. Ex: “Hello World”	n/a
autoAnalyze	Boolean data. Ex: True	true
ftpStatus	Unicode string data. Ex: “Hello World”	
flowsInOrder	Unicode string data. Ex: “Hello World”	
baselineRun	Boolean data. Ex: True	false
displayName	Unicode string data. Ex: “Hello World”	
notes	Unicode string data. Ex: “Hello World”	n/a
sequencekitname	Unicode string data. Ex: “Hello World”	n/a
chipBarcode	Unicode string data. Ex: “Hello World”	
pgmName	Unicode string data. Ex: “Hello World”	n/a
storageHost	Unicode string data. Ex: “Hello World”	n/a
expName	Unicode string data. Ex: “Hello World”	n/a
status	Unicode string data. Ex: “Hello World”	
usePreBeadfind	Boolean data. Ex: True	true
cycles	Integer data. Ex: 2673	n/a
rawdatastyle	Unicode string data. Ex: “Hello World”	single

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/experiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/experiment/", params={"format": "
ts_api_response = ts_api_request.json()

experiments = ts_api_response["objects"]

for experiment in experiments:
    print experiment
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 21008,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/experiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "storage_options": "A",
            "chipType": "",
            "user_ack": "U",
            "results": [],
            "sample": "E115943-1q204-01-L8095",
            "runMode": "",
            "reverse_primer": null,
            "seqKitBarcode": "",
            "id": 10132,
            "metaData": {},
            "log": {},
            "sequencekitbarcode": "",
            "resource_uri": "/rundb/api/v1/experiment/10132/",
            "eas_set": [
                {
                    "isEditable": true,
                    "hotSpotRegionBedFile": "",
                    "results": [],
                    "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/Ion-TargetSe
                    "thumbnailalignmentargs": "",
                    "thumbnailanalysisargs": "",
                    "id": 10575,
                    "barcodedSamples": {},
                    "reference": "hg19",
                    "isOneTimeOverride": false,
                    "analysisargs": "",
                    "thumbnailcalibrateargs": "",
                    "realign": false,
                    "selectedPlugins": {
                        "pinsPerFlow": {
                            "name": "pinsPerFlow"
                        },
                        "duplicateReads_useZC": {
                            "name": "duplicateReads_useZC"
                        },
                        "libClonality": {
                            "name": "libClonality"
                        },
                        "ProtonErrors": {
                            "name": "ProtonErrors"
                        },
                        "PhasingReport": {
                            "name": "PhasingReport"
                        },
                        "detailedReport": {
                            "name": "detailedReport"
                        },
                        "extended_chip_check": {
                            "name": "extended_chip_check"
                        },
                        "1_Torrent_Accuracy": {
```

```
            "name": "1_Torrent_Accuracy"
        },
        "ConversionRate": {
            "name": "ConversionRate"
        },
        "rawTrace": {
            "name": "rawTrace"
        },
        "filterAndTrim": {
            "name": "filterAndTrim"
        },
        "fsRecalibration": {
            "name": "fsRecalibration"
        },
        "timingPerformance": {
            "name": "timingPerformance"
        },
        "NucRiseParams": {
            "name": "NucRiseParams"
        },
        "AvgTrace": {
            "name": "AvgTrace"
        },
        "autoCal": {
            "name": "autoCal"
        },
        "flowCell": {
            "name": "flowCell"
        },
        "chipDiagnostics": {
            "name": "chipDiagnostics"
        },
        "rawPlots": {
            "name": "rawPlots"
        },
        "spatialPlots": {
            "name": "spatialPlots"
        },
        "RateMapEDA": {
            "name": "RateMapEDA"
        },
        "barcodeMixtureAnalysis": {
            "name": "barcodeMixtureAnalysis"
        },
        "z_homopolymerAnalysis": {
            "name": "z_homopolymerAnalysis"
        },
        "separator": {
            "name": "separator"
        },
        "GC_seq_performance": {
            "name": "GC_seq_performance"
        },
        "flowErr": {
            "name": "flowErr"
        }
    },
    "experiment": "/rundb/api/v1/experiment/10132/",
```

```
            "barcodeKitName": "",
            "beadfindargs": "",
            "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "thumbnailbasecallerargs": "",
            "status": "planned",
            "prebasecallerargs": "",
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "",
            "isDuplicateReads": false,
            "libraryKey": "TCAG",
            "date": "2013-05-15T18:30:24.000115+00:00",
            "libraryKitName": "",
            "thumbnailbeadfindargs": "",
            "calibrateargs": "",
            "tfKey": "",
            "libraryKitBarcode": null,
            "base_recalibrate": true,
            "basecallerargs": "",
            "resource_uri": "/rundb/api/v1/experimentanalysissettings/10575/"
        }
    ],
    "runtype": "GENS",
    "platform": "PGM",
    "samples": [
        {
            "status": "run",
            "sampleSets": [],
            "description": null,
            "displayedName": "E115943-lq204-01-L8095",
            "experiments": [
                "/rundb/api/v1/experiment/10142/",
                "/rundb/api/v1/experiment/10132/"
            ],
            "externalId": "",
            "date": "2013-05-15T18:30:24.000176+00:00",
            "resource_uri": "/rundb/api/v1/sample/2379/",
            "id": 2379,
            "name": "E115943-lq204-01-L8095"
        }
    ],
    "pinnedRepResult": false,
    "reagentBarcode": "",
    "star": false,
    "isProton": "False",
    "expCompInfo": "",
    "resultDate": "2013-05-15T18:30:24.000171+00:00",
    "flows": 400,
    "plan": "/rundb/api/v1/plannedexperiment/88364/",
    "date": "2013-05-15T18:30:24.000167+00:00",
    "diskusage": null,
    "unique": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
    "expDir": "",
    "autoAnalyze": true,
    "ftpStatus": "Complete",
    "flowsInOrder": "",
    "baselineRun": false,
    "displayName": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
    "notes": "OT2 lq204_01 Lib8095 275bp lr2 4B bead 1.2B lib SDS_10mMEDTA break ",
```

```
            "sequencekitname": "",
            "chipBarcode": "",
            "pgmName": "",
            "storageHost": null,
            "expName": "ea5aefc7-e1ec-4c79-9843-b0e299253a9a",
            "status": "planned",
            "usePreBeadfind": false,
            "cycles": 0,
            "rawdatastyle": "single"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.28 Experimentanalysissettings Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/`

Schema URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/schema/`

Fields table

field	help text	de
isEditable	Boolean data. Ex: True	fal
hotSpotRegionBedFile	Unicode string data. Ex: “Hello World”	n/a
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a
targetRegionBedFile	Unicode string data. Ex: “Hello World”	n/a
thumbnailalignmentargs	Unicode string data. Ex: “Hello World”	
thumbnailanalysisargs	Unicode string data. Ex: “Hello World”	
id	Integer data. Ex: 2673	
barcodedSamples	Unicode string data. Ex: “Hello World”	{}
reference	Unicode string data. Ex: “Hello World”	n/a
isOneTimeOverride	Boolean data. Ex: True	fal
analysisargs	Unicode string data. Ex: “Hello World”	
thumbnailcalibrateargs	Unicode string data. Ex: “Hello World”	
realign	Boolean data. Ex: True	fal
selectedPlugins	Unicode string data. Ex: “Hello World”	{}
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a
barcodeKitName	Unicode string data. Ex: “Hello World”	n/a
beadfindargs	Unicode string data. Ex: “Hello World”	
threePrimeAdapter	Unicode string data. Ex: “Hello World”	n/a

Table 4.9 – continued from previous page

field	help text	de
thumbnailbasecallerargs	Unicode string data. Ex: “Hello World”	
status	Unicode string data. Ex: “Hello World”	
prebasecallerargs	Unicode string data. Ex: “Hello World”	
prethumbnailbasecallerargs	Unicode string data. Ex: “Hello World”	
alignmentargs	Unicode string data. Ex: “Hello World”	
isDuplicateReads	Boolean data. Ex: True	fal
libraryKey	Unicode string data. Ex: “Hello World”	
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a
libraryKitName	Unicode string data. Ex: “Hello World”	n/a
thumbnailbeadfindargs	Unicode string data. Ex: “Hello World”	
calibrateargs	Unicode string data. Ex: “Hello World”	
tfKey	Unicode string data. Ex: “Hello World”	
libraryKitBarcode	Unicode string data. Ex: “Hello World”	n/a
base_recalibrate	Boolean data. Ex: True	tru
basecallerargs	Unicode string data. Ex: “Hello World”	
resource_uri	Unicode string data. Ex: “Hello World”	n/a

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/experimentanalysissettings/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/experimentanalysissettings/", para
ts_api_response = ts_api_request.json()

experimentanalysissettingss = ts_api_response["objects"]

for experimentanalysissettings in experimentanalysissettingss:
    print experimentanalysissettings
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 23666,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/experimentanalysissettings/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isEditable": true,
            "hotSpotRegionBedFile": "",
            "results": [],
            "targetRegionBedFile": "/results/uploads/BED/15/hg19/unmerged/detail/Ion-TargetSeq-Exome-
            "thumbnailalignmentargs": "",
```

```
            "thumbnailanalysisargs": "",
            "id": 138,
            "barcodedSamples": {},
            "reference": "hg19",
            "isOneTimeOverride": false,
            "analysisargs": "",
            "thumbnailcalibrateargs": "",
            "realign": false,
            "selectedPlugins": {
                "coverageAnalysis": {
                    "userInput": "",
                    "version": "3.4.47670",
                    "features": [],
                    "name": "coverageAnalysis",
                    "id": "319"
                }
            },
            "experiment": "/rundb/api/v1/experiment/6822/",
            "barcodeKitName": "",
            "beadfindargs": "",
            "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "thumbnailbasecallerargs": "",
            "status": "planned",
            "prebasecallerargs": "",
            "prethumbnailbasecallerargs": "",
            "alignmentargs": "",
            "isDuplicateReads": false,
            "libraryKey": "TCAG",
            "date": "2012-12-04T00:09:21.000461+00:00",
            "libraryKitName": "Ion Xpress Plus Fragment Library Kit",
            "thumbnailbeadfindargs": "",
            "calibrateargs": "",
            "tfKey": "",
            "libraryKitBarcode": null,
            "base_recalibrate": true,
            "basecallerargs": "",
            "resource_uri": "/rundb/api/v1/experimentanalysissettings/138/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.29 Filemonitor Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/filemonitor/`

Schema URL: `http://mytorrentserver/rundb/api/v1/filemonitor/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	false	false	string
updated	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
name	Unicode string data. Ex: “Hello World”		false	false	false	false	string
created	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
url	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
md5sum	Unicode string data. Ex: “Hello World”	None	true	false	false	false	string
celery_task_id	Unicode string data. Ex: “Hello World”		false	false	true	false	string
local_dir	Unicode string data. Ex: “Hello World”		false	false	false	false	string
progress	Unicode string data. Ex: “Hello World”	0	false	false	false	false	string
size	Unicode string data. Ex: “Hello World”	None	true	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
tags	Unicode string data. Ex: “Hello World”		false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/filemonitor/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/filemonitor/", params={"format": '
ts_api_response = ts_api_request.json()

filemonitors = ts_api_response["objects"]

for filemonitor in filemonitors:
    print filemonitor
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 8,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/filemonitor/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Complete",
```

```
            "updated": "2014-05-08T19:25:27.000843+00:00",
            "name": "AmpliSeqExome.20131001.Results.zip",
            "created": "2014-05-08T19:25:01.000513+00:00",
            "url": "https://ampliseq.com/ws/tmpldesign/14035495/download/results",
            "md5sum": null,
            "celery_task_id": "63a36c6f-ccc2-4ce3-8539-18335039f128",
            "local_dir": "/tmp/tmpubvTKY",
            "progress": "24174499",
            "size": "24174499",
            "id": 9,
            "tags": "ampliseq_template",
            "resource_uri": "/rundb/api/v1/filemonitor/9/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.30 Fileserver Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/fileserver/`

Schema URL: `http://mytorrentserver/rundb/api/v1/fileserver/schema/`

Perform read-only operations on `fileserver` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
percentfull	Floating point numeric data. Ex: 26.73	0.0	true	false	false	false	float
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
filesPrefix	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
comments	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/fileserver/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/fileserver/", params={"format": "
ts_api_response = ts_api_request.json()

fileservers = ts_api_response["objects"]

for fileserver in fileservers:
    print fileserver
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/fileserver/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "percentfull": 82.6372068824745,
            "name": "gs1-nfs",
            "filesPrefix": "/ion-data/results/",
            "comments": "gs1-nfs:/gs1/ion-data",
            "id": 5,
            "resource_uri": "/rundb/api/v1/fileserver/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.31 Globalconfig Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/globalconfig/`

Schema URL: `http://mytorrentserver/rundb/api/v1/globalconfig/schema/`

Perform read-only operations on `globalconfig` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	ty
enable_version_lock	Boolean data. Ex: True	false	false	false	true	false	bo
site_name	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_support_upload	Boolean data. Ex: True	false	false	false	true	false	bo
plugin_output_folder	Unicode string data. Ex: “Hello World”		false	false	true	false	st
auto_archive_ack	Boolean data. Ex: True	false	false	false	true	false	bo
default_plugin_script	Unicode string data. Ex: “Hello World”		false	false	true	false	st
id	Integer data. Ex: 2673		false	false	true	true	in
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	st
default_storage_options	Unicode string data. Ex: “Hello World”	D	false	false	true	false	st
selected	Boolean data. Ex: True		false	false	true	false	bo
check_news_posts	Boolean data. Ex: True	true	false	false	true	false	bo
realign	Boolean data. Ex: True	false	false	false	true	false	bo
ts_update_status	Unicode string data. Ex: “Hello World”		false	false	true	false	st
mark_duplicates	Boolean data. Ex: True	false	false	false	true	false	bo
plugin_folder	Unicode string data. Ex: “Hello World”		false	false	true	false	st
auto_archive_enable	Boolean data. Ex: True	false	false	false	true	false	bo
reference_path	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_auto_security	Boolean data. Ex: True	true	false	false	true	false	bo
fasta_path	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_nightly_email	Boolean data. Ex: True	true	false	false	true	false	bo
barcode_args	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	st
sec_update_status	Unicode string data. Ex: “Hello World”		false	false	true	false	st
default_flow_order	Unicode string data. Ex: “Hello World”		false	false	true	false	st
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	st
records_to_display	Integer data. Ex: 2673	20	false	false	true	false	in
base_recalibrate	Boolean data. Ex: True	true	false	false	true	false	bo
default_library_key	Unicode string data. Ex: “Hello World”		false	false	true	false	st
web_root	Unicode string data. Ex: “Hello World”		false	false	true	false	st
default_test_fragment_key	Unicode string data. Ex: “Hello World”		false	false	true	false	st
enable_auto_pkg_dl	Boolean data. Ex: True	true	false	false	true	false	bo
enable_compendia_OCP	Boolean data. Ex: True	false	false	false	true	false	bo

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/globalconfig/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/globalconfig/", params={"format":
ts_api_response = ts_api_request.json()

globalconfigs = ts_api_response["objects"]

for globalconfig in globalconfigs:
    print globalconfig
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": [
        {
            "enable_version_lock": false,
            "site_name": "blackbird-east",
            "enable_support_upload": false,
            "plugin_output_folder": "plugin_out",
            "auto_archive_ack": true,
            "default_plugin_script": "launch.sh",
            "id": 1,
            "resource_uri": "/rundb/api/v1/globalconfig/1/",
            "default_storage_options": "D",
            "selected": false,
            "check_news_posts": true,
            "realign": false,
            "ts_update_status": "No updates",
            "mark_duplicates": false,
            "plugin_folder": "plugins",
            "auto_archive_enable": true,
            "reference_path": "",
            "enable_auto_security": false,
            "fasta_path": "",
            "enable_nightly_email": true,
            "barcode_args": {
                "filter": "0.01"
            },
            "sec_update_status": "",
            "default_flow_order": "TACG",
            "name": "Config",
            "records_to_display": 50,
            "base_recalibrate": true,
            "default_library_key": "TCAG",
            "web_root": "http://blackbird.ite",
            "default_test_fragment_key": "ATCG",
            "enable_auto_pkg_dl": false,
            "enable_compendia_OCP": true
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.32 Ionchefplantemplate Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”

Table 4.11 – continued from previous page

field	help text
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”

Table 4.11 – continued from previous page

field	help text
reverse3primeadapter	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplate/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefplantemplate/", params={"fo
ts_api_response = ts_api_request.json()

ionchefplantemplates = ts_api_response["objects"]

for ionchefplantemplate in ionchefplantemplates:
    print ionchefplantemplate
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 7,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefplantemplate/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "chef_useGUI_Exome Panel_AmpliSeqExome.20131001",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
```

```
"thumbnailalignmentargs": "stage1 map4",
"chipType": "P1.1.17",
"chefProgress": 0.0,
"library": "hg19",
"reverselibrarykey": "",
"sampleTubeLabel": "",
"seqKitBarcode": null,
"barcodeId": "IonXpress",
"chefLogPath": null,
"isPlanGroup": false,
"realign": false,
"sampleGroupingName": "",
"experiment": "/rundb/api/v1/experiment/21899/",
"bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
"isReusable": true,
"isDuplicateReads": false,
"thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
"librarykitname": "Ion AmpliSeq 2.0 Library Kit",
"adapter": null,
"basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
"tfKey": "ATCG",
"parentPlan": null,
"forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
"planStatus": "pending",
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": true,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
        "id": 262521,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262521/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
        "id": 262520,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
```

```
                "resource_uri": "/rundb/api/v1/qctype/2/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/262520/"
        },
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/100136/",
            "id": 262519,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/262519/"
        }
    ],
    "analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=216
    "thumbnailcalibrateargs": "calibrate --skipDroop",
    "templatingKitName": "Ion PI IC 200 Kit",
    "runType": "AMPS_EXOME",
    "username": "ionuser",
    "planName": "chef_useGUI_Exome_Panel_AmpliSeqExome.20131001",
    "sampleDisplayedName": "",
    "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
    "controlSequencekitname": "",
    "chefMessage": "",
    "childPlans": [],
    "pairedEndLibraryAdapterName": "",
    "runMode": "single",
    "irworkflow": "",
    "planExecuted": false,
    "project": "",
    "usePostBeadfind": false,
    "runname": null,
    "planGUID": "83e69cd8-ca3a-4ff2-b4ac-dd637184e28e",
    "planShortID": "YHFWJ",
    "sampleSetGroupType": null,
    "sample": "",
    "planExecutedDate": null,
    "reverse_primer": null,
    "id": 100136,
    "barcodedSamples": {},
    "regionfile": "",
    "selectedPlugins": {
        "IonReporterUploader": {
            "userInput": {
                "accountName": "None",
                "userInputInfo": "",
                "accountId": "0"
            },
            "version": "4.1-r87449",
            "features": [
                "export"
            ],
```

```
        "name": "IonReporterUploader",
        "id": 804
    },
    "coverageAnalysis": {
        "userInput": "",
        "version": "4.2-r86949",
        "features": [],
        "name": "coverageAnalysis",
        "id": 800
    },
    "validateVariantCaller": {
        "userInput": {
            "variant_caller_name": "variantCaller",
            "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
            "region": "NIST",
            "sample": "NA12878",
            "truth_minor_snp": "None",
            "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
            "truth_minor_indel": "None"
        },
        "version": "0.2.1",
        "features": [],
        "name": "validateVariantCaller",
        "id": 732
    },
    "variantCaller": {
        "userInput": {
            "torrent_variant_caller": {
                "snp_min_allele_freq": "0.10000000000000001",
                "snp_strand_bias": "0.97999999999999998",
                "hotspot_min_coverage": 6,
                "hotspot_min_cov_each_strand": 3,
                "hotspot_min_allele_freq": "0.10000000000000001",
                "snp_min_variant_score": 15,
                "hotspot_strand_bias": "0.94999999999999996",
                "hp_max_length": 8,
                "filter_insertion_predictions": "0.20000000000000001",
                "indel_min_variant_score": 20,
                "indel_min_coverage": 10,
                "heavy_tailed": 3,
                "outlier_probability": "0.01",
                "data_quality_stringency": 5,
                "snp_min_cov_each_strand": 0,
                "hotspot_min_variant_score": 10,
                "indel_strand_bias": "0.90000000000000002",
                "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                "downsample_to_coverage": 400,
                "filter_unusual_predictions": "0.25",
                "indel_min_allele_freq": "0.14999999999999999",
                "do_snp_realignment": 1,
                "prediction_precision": 1,
                "indel_min_cov_each_strand": 5,
                "filter_deletion_predictions": "0.20000000000000001",
                "suppress_recalibration": 0,
                "snp_min_coverage": 5
            },
            "meta": {
                "repository_id": "",
```

```
                    "ts_version": "4.0",
                    "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.201309
                    "user_selections": {
                        "chip": "proton_p1",
                        "frequency": "germline",
                        "library": "ampliseq",
                        "panel": "/rundb/api/v1/contentupload/53/"
                    },
                    "librarytype": "ampliseq",
                    "trimreads": true,
                    "tooltip": "Retrieved from external file",
                    "tvcargs": "tvc",
                    "built_in": false,
                    "configuration": "",
                    "compatibility": {}
                },
                "long_indel_assembler": {
                    "min_indel_size": 4,
                    "short_suffix_match": 5,
                    "output_mnv": 0,
                    "min_var_count": 5,
                    "min_var_freq": "0.14999999999999999",
                    "kmer_len": 19,
                    "max_hp_length": 8,
                    "relative_strand_bias": "0.80000000000000004"
                },
                "freebayes": {
                    "gen_min_coverage": 5,
                    "allow_mnps": 1,
                    "allow_complex": 0,
                    "read_max_mismatch_fraction": 1,
                    "read_mismatch_limit": 10,
                    "allow_indels": 1,
                    "min_mapping_qv": 4,
                    "gen_min_alt_allele_freq": "0.10000000000000001",
                    "allow_snps": 1,
                    "gen_min_indel_alt_allele_freq": "0.14999999999999999"
                }
            },
            "version": "4.1-r74477",
            "features": [],
            "name": "variantCaller",
            "id": 698
        },
        "AmpliconStats": {
            "userInput": "",
            "version": "0.4.5",
            "features": [],
            "name": "AmpliconStats",
            "id": 774
        }
    },
    "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
    "sampleSet": null,
    "isSystemDefault": false,
    "autoName": null,
    "libraryKey": "TCAG",
    "flows": 520,
```

```
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regio
            "date": "2014-05-20T13:56:24.000114+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/ionchefplantemplate/100136/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.33 Ionchefplantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa

Table 4.12 – continued from previous page

field	help text	default	nullable	readonly	b
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/?format=json&limit`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefplantemplatesummary/", para
ts_api_response = ts_api_request.json()
```

```
ionchefplantemplatesummarys = ts_api_response["objects"]

for ionchefplantemplatesummary in ionchefplantemplatesummarys:
    print ionchefplantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 7,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefplantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_useGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "YHFWJ",
            "planStatus": "pending",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100136,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI IC 200 Kit",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:56:24.000114+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_useGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
```

```
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "83e69cd8-ca3a-4ff2-b4ac-dd637184e28e",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/ionchefplantemplatesummary/100136/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.34 Ionchefprepkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/ionchefprepkitinfo/", params={"fo
ts_api_response = ts_api_request.json()

ionchefprepkitinfos = ts_api_response["objects"]

for ionchefprepkitinfo in ionchefprepkitinfos:
    print ionchefprepkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/ionchefprepkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "IonChefPrepKit",
            "description": "ION PGM IC 200 KIT",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "100020579",
                    "id": 20085,
                    "resource_uri": "/rundb/api/v1/kitpart/20085/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "100020580",
                    "id": 20086,
                    "resource_uri": "/rundb/api/v1/kitpart/20086/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                },
                {
                    "barcode": "01",
                    "id": 20123,
                    "resource_uri": "/rundb/api/v1/kitpart/20123/",
                    "kit": "/rundb/api/v1/kitinfo/20042/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "ICPREP0001",
            "resource_uri": "/rundb/api/v1/ionchefprepkitinfo/20042/",
            "id": 20042,
            "categories": "",
            "name": "ION PGM IC 200 KIT"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.35 Kitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/kitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/kitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/kitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/kitinfo/", params={"format": "jso
```

```
ts_api_response = ts_api_request.json()

kitinfos = ts_api_response["objects"]

for kitinfo in kitinfos:
    print kitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 67,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/kitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "ControlSequenceKitType",
            "description": "ERCC Mix 1",
            "nucleotideType": "rna",
            "instrumentType": "",
            "runMode": "",
            "parts": [],
            "flowCount": 0,
            "applicationType": "RNA",
            "uid": "CONSEQ0006",
            "resource_uri": "/rundb/api/v1/kitinfo/20061/",
            "id": 20061,
            "categories": "",
            "name": "ERCC Mix 1"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.36 Kitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/kitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/kitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/kitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/kitpart/", params={"format": "jso
ts_api_response = ts_api_request.json()

kitparts = ts_api_response["objects"]

for kitpart in kitparts:
    print kitpart
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 139,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/kitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "4456739",
            "id": 20131,
            "resource_uri": "/rundb/api/v1/kitpart/20131/",
            "kit": "/rundb/api/v1/kitinfo/20060/"
        }
    ]
}
```


Allowed HTTP methods

- get

- post
- put
- delete
- patch

4.1.37 Libmetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/libmetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/libmetrics/schema/`

Perform read-only operations on `libmetrics` resources and data elements.

Fields table

field	help text	default	nullable
i350Q17_reads	Integer data. Ex: 2673	n/a	false
i150Q47_reads	Integer data. Ex: 2673	n/a	false
i300Q47_reads	Integer data. Ex: 2673	n/a	false
i600Q20_reads	Integer data. Ex: 2673	n/a	false
i300Q20_reads	Integer data. Ex: 2673	n/a	false
i250Q17_reads	Integer data. Ex: 2673	n/a	false
q10_longest_alignment	Integer data. Ex: 2673	n/a	false
i50Q10_reads	Integer data. Ex: 2673	n/a	false
aveKeyCounts	Floating point numeric data. Ex: 26.73	n/a	false
i50Q17_reads	Integer data. Ex: 2673	n/a	false
total_mapped_target_bases	Unicode string data. Ex: “Hello World”	n/a	false
i200Q7_reads	Integer data. Ex: 2673	n/a	false
i100Q47_reads	Integer data. Ex: 2673	n/a	false
i50Q20_reads	Integer data. Ex: 2673	n/a	false
i450Q7_reads	Integer data. Ex: 2673	n/a	false
genomesize	Unicode string data. Ex: “Hello World”	n/a	false
i550Q20_reads	Integer data. Ex: 2673	n/a	false
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false
i450Q47_reads	Integer data. Ex: 2673	n/a	false
dr	Floating point numeric data. Ex: 26.73	n/a	false
i150Q17_reads	Integer data. Ex: 2673	n/a	false
q7_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i350Q7_reads	Integer data. Ex: 2673	n/a	false
i500Q20_reads	Integer data. Ex: 2673	n/a	false
q20_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i250Q47_reads	Integer data. Ex: 2673	n/a	false
q47_longest_alignment	Integer data. Ex: 2673	n/a	false
i550Q17_reads	Integer data. Ex: 2673	n/a	false
i50Q47_reads	Integer data. Ex: 2673	n/a	false
i200Q17_reads	Integer data. Ex: 2673	n/a	false
i250Q20_reads	Integer data. Ex: 2673	n/a	false
q47_alignments	Integer data. Ex: 2673	n/a	false
align_sample	Integer data. Ex: 2673	n/a	false

Table 4.13 – continued from previous page

field	help text	default	nullable
i100Q10_reads	Integer data. Ex: 2673	n/a	false
i350Q20_reads	Integer data. Ex: 2673	n/a	false
i100Q7_reads	Integer data. Ex: 2673	n/a	false
i400Q17_reads	Integer data. Ex: 2673	n/a	false
i500Q47_reads	Integer data. Ex: 2673	n/a	false
i450Q20_reads	Integer data. Ex: 2673	n/a	false
q7_mean_alignment_length	Integer data. Ex: 2673	n/a	false
q7_alignments	Integer data. Ex: 2673	n/a	false
total_mapped_reads	Unicode string data. Ex: “Hello World”	n/a	false
i600Q10_reads	Integer data. Ex: 2673	n/a	false
i250Q10_reads	Integer data. Ex: 2673	n/a	false
cf	Floating point numeric data. Ex: 26.73	n/a	false
i500Q7_reads	Integer data. Ex: 2673	n/a	false
q10_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
i550Q7_reads	Integer data. Ex: 2673	n/a	false
duplicate_reads	Integer data. Ex: 2673	n/a	true
i350Q47_reads	Integer data. Ex: 2673	n/a	false
totalNumReads	Integer data. Ex: 2673	n/a	false
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false
i350Q10_reads	Integer data. Ex: 2673	n/a	false
i300Q10_reads	Integer data. Ex: 2673	n/a	false
q20_mean_alignment_length	Integer data. Ex: 2673	n/a	false
i250Q7_reads	Integer data. Ex: 2673	n/a	false
i200Q10_reads	Integer data. Ex: 2673	n/a	false
i400Q7_reads	Integer data. Ex: 2673	n/a	false
i200Q47_reads	Integer data. Ex: 2673	n/a	false
q7_longest_alignment	Integer data. Ex: 2673	n/a	false
i500Q10_reads	Integer data. Ex: 2673	n/a	false
Genome_Version	Unicode string data. Ex: “Hello World”	n/a	false
i400Q20_reads	Integer data. Ex: 2673	n/a	false
q10_alignments	Integer data. Ex: 2673	n/a	false
i450Q17_reads	Integer data. Ex: 2673	n/a	false
i100Q20_reads	Integer data. Ex: 2673	n/a	false
i550Q10_reads	Integer data. Ex: 2673	n/a	false
i450Q10_reads	Integer data. Ex: 2673	n/a	false
i400Q47_reads	Integer data. Ex: 2673	n/a	false
q17_longest_alignment	Integer data. Ex: 2673	n/a	false
i150Q7_reads	Integer data. Ex: 2673	n/a	false
i400Q10_reads	Integer data. Ex: 2673	n/a	false
q10_mean_alignment_length	Integer data. Ex: 2673	n/a	false
raw_accuracy	Floating point numeric data. Ex: 26.73	n/a	false
sysSNR	Floating point numeric data. Ex: 26.73	n/a	false
q17_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
Index_Version	Unicode string data. Ex: “Hello World”	n/a	false
i300Q17_reads	Integer data. Ex: 2673	n/a	false
q17_mean_alignment_length	Integer data. Ex: 2673	n/a	false
ie	Floating point numeric data. Ex: 26.73	n/a	false
id	Integer data. Ex: 2673		false
q20_alignments	Integer data. Ex: 2673	n/a	false

Table 4.13 – continued from previous page

field	help text	default	nullable
q47_mapped_bases	Unicode string data. Ex: “Hello World”	n/a	false
genome	Unicode string data. Ex: “Hello World”	n/a	false
i300Q7_reads	Integer data. Ex: 2673	n/a	false
i150Q20_reads	Integer data. Ex: 2673	n/a	false
i550Q47_reads	Integer data. Ex: 2673	n/a	false
i600Q47_reads	Integer data. Ex: 2673	n/a	false
i100Q17_reads	Integer data. Ex: 2673	n/a	false
q47_mean_alignment_length	Integer data. Ex: 2673	n/a	false
i50Q7_reads	Integer data. Ex: 2673	n/a	false
i600Q7_reads	Integer data. Ex: 2673	n/a	false
i600Q17_reads	Integer data. Ex: 2673	n/a	false
q17_alignments	Integer data. Ex: 2673	n/a	false
i500Q17_reads	Integer data. Ex: 2673	n/a	false
i150Q10_reads	Integer data. Ex: 2673	n/a	false
q20_longest_alignment	Integer data. Ex: 2673	n/a	false
i200Q20_reads	Integer data. Ex: 2673	n/a	false

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/libmetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/libmetrics/", params={"format": "
ts_api_response = ts_api_request.json()

libmetricss = ts_api_response["objects"]

for libmetrics in libmetricss:
    print libmetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 39455,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/libmetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "i350Q17_reads": 0,
            "i150Q47_reads": 0,
            "i300Q47_reads": 0,
            "i600Q20_reads": 0,
            "i300Q20_reads": 0,
```

```
"i250Q17_reads": 0,
"q10_longest_alignment": 137,
"i50Q10_reads": 5244,
"aveKeyCounts": 82.0,
"i50Q17_reads": 5050,
"total_mapped_target_bases": "0",
"i200Q7_reads": 0,
"i100Q47_reads": 2641,
"i50Q20_reads": 4738,
"i450Q7_reads": 0,
"genomesize": "4686137",
"i550Q20_reads": 0,
"report": "/rundb/api/v1/results/3/",
"i450Q47_reads": 0,
"dr": 0.310014492754,
"i150Q17_reads": 0,
"q7_mapped_bases": "552185",
"i350Q7_reads": 0,
"i500Q20_reads": 0,
"q20_mapped_bases": "493269",
"i250Q47_reads": 0,
"q47_longest_alignment": 133,
"i550Q17_reads": 0,
"i50Q47_reads": 4527,
"i200Q17_reads": 0,
"i250Q20_reads": 0,
"q47_alignments": 4942,
"align_sample": 0,
"i100Q10_reads": 3990,
"i350Q20_reads": 0,
"i100Q7_reads": 3991,
"i400Q17_reads": 0,
"i500Q47_reads": 0,
"i450Q20_reads": 0,
"q7_mean_alignment_length": 104,
"q7_alignments": 5311,
"total_mapped_reads": "0",
"i600Q10_reads": 0,
"i250Q10_reads": 0,
"cf": 0.760992753623,
"i500Q7_reads": 0,
"q10_mapped_bases": "551679",
"i550Q7_reads": 0,
"duplicate_reads": null,
"i350Q47_reads": 0,
"totalNumReads": 5381,
"resource_uri": "/rundb/api/v1/libmetrics/1/",
"i350Q10_reads": 0,
"i300Q10_reads": 0,
"q20_mean_alignment_length": 98,
"i250Q7_reads": 0,
"i200Q10_reads": 0,
"i400Q7_reads": 0,
"i200Q47_reads": 0,
"q7_longest_alignment": 137,
"i500Q10_reads": 0,
"Genome_Version": "1",
"i400Q20_reads": 0,
```

```
            "q10_alignments": 5306,
            "i450Q17_reads": 0,
            "i100Q20_reads": 3443,
            "i550Q10_reads": 0,
            "i450Q10_reads": 0,
            "i400Q47_reads": 0,
            "q17_longest_alignment": 137,
            "i150Q7_reads": 0,
            "i400Q10_reads": 0,
            "q10_mean_alignment_length": 104,
            "raw_accuracy": 0.0,
            "sysSNR": 17.32,
            "q17_mapped_bases": "524626",
            "Index_Version": "tmap-f2",
            "i300Q17_reads": 0,
            "q17_mean_alignment_length": 102,
            "ie": 0.884253623188,
            "id": 1,
            "q20_alignments": 5030,
            "q47_mapped_bases": "457712",
            "genome": "E. coli DH10B",
            "i300Q7_reads": 0,
            "i150Q20_reads": 0,
            "i550Q47_reads": 0,
            "i600Q47_reads": 0,
            "i100Q17_reads": 3714,
            "q47_mean_alignment_length": 93,
            "i50Q7_reads": 5250,
            "i600Q7_reads": 0,
            "i600Q17_reads": 0,
            "q17_alignments": 5156,
            "i500Q17_reads": 0,
            "i150Q10_reads": 0,
            "q20_longest_alignment": 137,
            "i200Q20_reads": 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.38 Librarykey Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykey/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykey/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
direction	Unicode string data. Ex: “Hello World”	Forward	false	false	false	false	string
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
description	Unicode string data. Ex: “Hello World”		false	false	true	false	string
runMode	Unicode string data. Ex: “Hello World”	single	false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isDefault	Boolean data. Ex: True	false	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykey/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykey/", params={"format": "
ts_api_response = ts_api_request.json()

librarykeys = ts_api_response["objects"]

for librarykey in librarykeys:
    print librarykey
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykey/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "direction": "Forward",
            "name": "Ion TCAG",
            "sequence": "TCAG",
            "description": "Default forward library key",
            "runMode": "single",
            "id": 3,
            "isDefault": true,
            "resource_uri": "/rundb/api/v1/librarykey/3/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.39 Librarykitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykitinfo/", params={"format"
ts_api_response = ts_api_request.json()

librarykitinfos = ts_api_response["objects"]

for librarykitinfo in librarykitinfos:
    print librarykitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": true,
            "kitType": "LibraryKit",
            "description": "Ion Fragment Library Kit",
            "nucleotideType": "dna",
            "instrumentType": "",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4466464",
                    "id": 20014,
                    "resource_uri": "/rundb/api/v1/kitpart/20014/",
                    "kit": "/rundb/api/v1/kitinfo/20005/"
                }
            ],
            "flowCount": 0,
            "applicationType": "",
            "uid": "LIB0002",
            "resource_uri": "/rundb/api/v1/librarykitinfo/20005/",
            "id": 20005,
            "categories": "",
            "name": "IonFragmentLibKit2"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

4.1.40 Librarykitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/librarykitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/librarykitpart/", params={"format"
ts_api_response = ts_api_request.json()

librarykitparts = ts_api_response["objects"]

for librarykitpart in librarykitparts:
    print librarykitpart
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 17,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/librarykitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "A25908",
```

```
            "id": 20140,
            "resource_uri": "/rundb/api/v1/librarykitpart/20140/",
            "kit": "/rundb/api/v1/kitinfo/20065/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.41 Location Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/location/`

Schema URL: `http://mytorrentserver/rundb/api/v1/location/schema/`

Perform read-only operations on `location` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
defaultlocation	Only one location can be the default	false	false	false	true	false	boolean
comments	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/location/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/location/", params={"format": "jso
ts_api_response = ts_api_request.json()

locations = ts_api_response["objects"]

for location in locations:
    print location
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/location/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "Local",
            "resource_uri": "/rundb/api/v1/location/3/",
            "defaultlocation": false,
            "comments": "",
            "id": 3
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.42 Log Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/log/`

Schema URL: `http://mytorrentserver/rundb/api/v1/log/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
text	Unicode string data. Ex: “Hello World”		false	false	true	false	string
timeStamp	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
upload	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/log/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/log/", params={"format": "json", "
ts_api_response = ts_api_request.json()

logs = ts_api_response["objects"]

for log in logs:
    print log
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 858,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/log/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "text": "FATAL ERROR: '/.fasta.fai' does not exist.",
            "timeStamp": "2013-05-30T15:09:03.000306+00:00",
            "upload": "/rundb/api/v1/contentupload/26/",
            "id": 885,
            "resource_uri": "/rundb/api/v1/log/885/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.43 Message Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/message/`

Schema URL: `http://mytorrentserver/rundb/api/v1/message/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
body	Unicode string data. Ex: "Hello World"		false	false	true	false	string
status	Unicode string data. Ex: "Hello World"	unread	false	false	true	false	string
level	Integer data. Ex: 2673	20	false	false	false	false	integer
route	Unicode string data. Ex: "Hello World"		false	false	true	false	string
expires	Unicode string data. Ex: "Hello World"	read	false	false	true	false	string
time	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
tags	Unicode string data. Ex: "Hello World"		false	false	true	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/message/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/message/", params={"format": "jso
ts_api_response = ts_api_request.json()

messages = ts_api_response["objects"]

for message in messages:
    print message
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get

- post
- put
- delete
- patch

4.1.44 Monitordata Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/monitordata/`

Schema URL: `http://mytorrentserver/rundb/api/v1/monitordata/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
treeDat	Unicode string data. Ex: “Hello World”	{}	false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
name	Unicode string data. Ex: “Hello World”		false	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/monitordata/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/monitordata/", params={"format": '
ts_api_response = ts_api_request.json()

monitordatas = ts_api_response["objects"]

for monitordata in monitordatas:
    print monitordata
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": [
        {
            "resource_uri": "/rundb/api/v1/monitordata/1/",
            "treeDat": {},
```

```
            "id": 1,
            "name": "Debug"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.45 Monitorresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/monitorresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/monitorresult/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
libmetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
analysismetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
library	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
qualitymetrics	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
barcodeId	Unicode string data. Ex: "Hello World"	n/a	true	true	false	false	string
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
representative	Boolean data. Ex: True	false	false	false	true	false	boolean

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/monitorresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/monitorresult/", params={"format"
ts_api_response = ts_api_request.json()
```

```
monitorresults = ts_api_response["objects"]

for monitorresult in monitorresults:
    print monitorresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 159,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/monitorresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "Completed",
            "processedflows": 0,
            "libmetrics": {
                "i100Q20_reads": 61016174,
                "aveKeyCounts": 89.0,
                "id": 41874,
                "resource_uri": "",
                "q20_mean_alignment_length": 159
            },
            "timeStamp": "2014-06-28T07:11:58.000789+00:00",
            "analysismetrics": {
                "ignored": 1364892,
                "lib": 118063544,
                "total_wells": 164699136,
                "pinned": 463867,
                "live": 118074386,
                "excluded": 16095180,
                "bead": 118525043,
                "resource_uri": "",
                "id": 41726,
                "empty": 28250154,
                "libFinal": 85604279
            },
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
            "library": "hg19",
            "id": 304393,
            "reportStatus": "Nothing",
            "experiment": {
                "ftpStatus": "Complete",
                "chipInstrumentType": "proton",
                "displayName": "user Z28-428--r65714-pou4 dbsa",
                "chipType": "P1.1.17",
                "notes": "",
                "chipDescription": "PI",
                "resultDate": "2014-06-28T07:11:58.000789+00:00",
                "flows": 520,
                "runMode": "single",
                "expName": "R_2014_06_27_17_13_22_user_Z28-428--r65714-pou4_dbsa",
                "storage_options": "D",
```

```
                "pgmName": "Z28",
                "date": "2014-06-27T21:19:01+00:00",
                "star": false,
                "resource_uri": "",
                "qcThresholds": {
                    "Key Signal (1-100)": 30,
                    "Usable Sequence (%)": 30,
                    "Bead Loading (%)": 30
                },
                "id": 23958,
                "plan": {
                    "runType": "AMPS_EXOME",
                    "id": 102195,
                    "resource_uri": ""
                }
            },
            "resultsName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
            "projects": [
                {
                    "resource_uri": "",
                    "id": 1385,
                    "name": "auto_chip",
                    "modified": "2014-06-27T21:21:43.000081+00:00"
                }
            ],
            "qualitymetrics": {
                "q0_mean_read_length": 178.362197969099,
                "q0_reads": 85604279,
                "q0_bases": "15268567358",
                "q20_reads": 85604279,
                "q20_bases": "13060288783",
                "q20_mean_read_length": 178,
                "id": 39683,
                "resource_uri": ""
            },
            "eas": {
                "resource_uri": "",
                "reference": "hg19",
                "barcodeKitName": "IonXpress"
            },
            "resource_uri": "/rundb/api/v1/monitorresult/304393/",
            "barcodeId": "IonXpress",
            "autoExempt": false,
            "representative": false
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.46 Obsoletereferencegenome Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/`

Schema URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
reference_path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
short_name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
index_version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
notes	Unicode string data. Ex: "Hello World"		false	false	true	false	string
enabled	Boolean data. Ex: True	true	false	false	true	false	boolean
species	Unicode string data. Ex: "Hello World"		false	false	true	false	string
identity_hash	Unicode string data. Ex: "Hello World"	None	true	false	false	false	string
source	Unicode string data. Ex: "Hello World"		false	false	true	false	string
version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
celery_task_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	2014-06-28T14:03:44.000226+00:00	false	false	false	false	datetime
verbose_error	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/obsoletereferencegenome/", params=
ts_api_response = ts_api_request.json()

obsoletereferencegenomes = ts_api_response["objects"]

for obsoletereferencegenome in obsoletereferencegenomes:
    print obsoletereferencegenome
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.47 Onetouchplantemplate Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/`

Schema URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/schema/`

Fields table

field	help text
planDisplayedName	Unicode string data. Ex: “Hello World”
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.

Table 4.14 – continued from previous page

field	help text
libkit	Unicode string data. Ex: “Hello World”
platform	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planPGM	Unicode string data. Ex: “Hello World”
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”

Table 4.14 – continued from previous page

field	help text
childPlans	A list of data. Ex: ['abc', 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: "Hello World"
runMode	Unicode string data. Ex: "Hello World"
irworkflow	Unicode string data. Ex: "Hello World"
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: "Hello World"
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: "Hello World"
planGUID	Unicode string data. Ex: "Hello World"
planShortID	Unicode string data. Ex: "Hello World"
sampleSetGroupType	Unicode string data. Ex: "Hello World"
sample	Unicode string data. Ex: "Hello World"
planExecutedDate	A date & time as a string. Ex: "2010-11-10T03:07:43"
reverse_primer	Unicode string data. Ex: "Hello World"
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: "Hello World"
regionfile	Unicode string data. Ex: "Hello World"
selectedPlugins	Unicode string data. Ex: "Hello World"
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: "Hello World"
libraryKey	Unicode string data. Ex: "Hello World"
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: "2010-11-10T03:07:43"
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: "Hello World"
sampleSetDisplayedName	Unicode string data. Ex: "Hello World"
flowsInOrder	Unicode string data. Ex: "Hello World"
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: "Hello World"
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: "Hello World"
reverse3primeadapter	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplate/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/onetouchplantemplate/", params={"
ts_api_response = ts_api_request.json()

onetouchplantemplates = ts_api_response["objects"]

for onetouchplantemplate in onetouchplantemplates:
    print onetouchplantemplate
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 58,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/onetouchplantemplate/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sequencekitname": "ProtonIIC200Kit",
            "storageHost": null,
            "expName": "",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.1.17",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": "",
            "seqKitBarcode": null,
            "barcodeId": "IonXpress",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/21987/",
            "bedfile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131001.designed
            "isReusable": true,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=100,100 --be
            "librarykitname": "Ion AmpliSeq 2.0 Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
            "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "planStatus": "planned",
```

```
"chefLastUpdate": null,
"samplePrepKitName": "Ion AmpliSeq Exome Kit",
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": true,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262674,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262674/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262673,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262673/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/100224/",
        "id": 262672,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/262672/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=21
"thumbnailcalibrateargs": "calibrate --skipDroop",
"templatingKitName": "Ion PI Template OT2 200 Kit v3",
"runType": "AMPS_EXOME",
```

```
"username": "ionuser",
"planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
"sampleDisplayedName": "",
"prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
"controlSequencekitname": "",
"chefMessage": "",
"childPlans": [],
"pairedEndLibraryAdapterName": "",
"runMode": "single",
"irworkflow": "",
"planExecuted": false,
"project": "",
"usePostBeadfind": false,
"runname": null,
"planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
"planShortID": "2OXYZ",
"sampleSetGroupType": null,
"sample": "",
"planExecutedDate": null,
"reverse_primer": null,
"id": 100224,
"barcodedSamples": {},
"regionfile": "",
"selectedPlugins": {
    "IonReporterUploader": {
        "userInput": {
            "accountName": "None",
            "userInputInfo": "",
            "accountId": "0"
        },
        "version": "4.1-r87449",
        "features": [
            "export"
        ],
        "name": "IonReporterUploader",
        "id": 804
    },
    "coverageAnalysis": {
        "userInput": "",
        "version": "4.2-r86949",
        "features": [],
        "name": "coverageAnalysis",
        "id": 800
    },
    "variantCaller": {
        "userInput": {
            "torrent_variant_caller": {
                "snp_min_allele_freq": "0.10000000000000001",
                "snp_strand_bias": "0.97999999999999998",
                "hotspot_min_coverage": 6,
                "hotspot_min_cov_each_strand": 3,
                "hotspot_min_allele_freq": "0.10000000000000001",
                "snp_min_variant_score": 15,
                "hotspot_strand_bias": "0.94999999999999996",
                "hp_max_length": 8,
                "filter_insertion_predictions": "0.20000000000000001",
                "indel_min_variant_score": 20,
                "indel_min_coverage": 10,
```

```
                "heavy_tailed": 3,
                "outlier_probability": "0.01",
                "data_quality_stringency": 5,
                "snp_min_cov_each_strand": 0,
                "hotspot_min_variant_score": 10,
                "indel_strand_bias": "0.90000000000000002",
                "tvc_parameters_version": "germline_low_stringency_proton-3.6.66827",
                "downsample_to_coverage": 400,
                "filter_unusual_predictions": "0.25",
                "indel_min_allele_freq": "0.14999999999999999",
                "do_snp_realignment": 1,
                "prediction_precision": 1,
                "indel_min_cov_each_strand": 5,
                "filter_deletion_predictions": "0.20000000000000001",
                "suppress_recalibration": 0,
                "snp_min_coverage": 5
            },
            "meta": {
                "repository_id": "",
                "ts_version": "4.0",
                "name": "External file AmpliseqExome.germline_lowstringency_p1.4_0.20130
                "user_selections": {
                    "chip": "proton_p1",
                    "frequency": "germline",
                    "library": "ampliseq",
                    "panel": "/rundb/api/v1/contentupload/53/"
                },
                "librarytype": "ampliseq",
                "trimreads": true,
                "tooltip": "Retrieved from external file",
                "tvcargs": "tvc",
                "built_in": false,
                "configuration": "",
                "compatibility": {}
            },
            "long_indel_assembler": {
                "min_indel_size": 4,
                "short_suffix_match": 5,
                "output_mnv": 0,
                "min_var_count": 5,
                "min_var_freq": "0.14999999999999999",
                "kmer_len": 19,
                "max_hp_length": 8,
                "relative_strand_bias": "0.80000000000000004"
            },
            "freebayes": {
                "gen_min_coverage": 5,
                "allow_mnps": 1,
                "allow_complex": 0,
                "read_max_mismatch_fraction": 1,
                "read_mismatch_limit": 10,
                "allow_indels": 1,
                "min_mapping_qv": 4,
                "gen_min_alt_allele_freq": "0.10000000000000001",
                "allow_snps": 1,
                "gen_min_indel_alt_allele_freq": "0.14999999999999999"
            }
        },
```

```
                    "version": "4.1-r74477",
                    "features": [],
                    "name": "variantCaller",
                    "id": 698
                },
                "validateVariantCaller": {
                    "userInput": {
                        "variant_caller_name": "variantCaller",
                        "truth_major_snp": "NA12878_NIST_NoChrY_SNP.bed",
                        "region": "NIST",
                        "sample": "NA12878",
                        "truth_minor_snp": "None",
                        "truth_major_indel": "NA12878_NIST_NoChrY_indel.bed",
                        "truth_minor_indel": "None"
                    },
                    "version": "0.2.1",
                    "features": [],
                    "name": "validateVariantCaller",
                    "id": 732
                }
            },
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=216,224 --total-timec
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 520,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regior
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/onetouchplantemplate/100224/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.48 Onetouchplantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr

Table 4.15 – continued from previous page

field	help text	default	nullable	readonly	b
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/?format=json&limi`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/onetouchplantemplatesummary/", par
ts_api_response = ts_api_request.json()

onetouchplantemplatesummarys = ts_api_response["objects"]

for onetouchplantemplatesummary in onetouchplantemplatesummarys:
    print onetouchplantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 58,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/onetouchplantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
```

```
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/onetouchplantemplatesummary/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.49 Plannedexperiment Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/schema/`

Perform CRUD operations on `plannedexperiment` resources and data elements.

Even though plannedExperiment db schema has changed dramatically in TSS 3.6 as part of the "plan data decentralization" (aka PDD) effort. A facade is provided so if you are already familiar with using the plan REST API, changes under the hood are abstracted from the REST API users. However, note that "selectedPlugins" and "barcodedSamples" are JSON fields and their data structures tend to change from release to release.

What has changed in TSS 4.2

- The JSON data structure in barcodedSamples has been changed with the following added
- controlSequenceType
- hotSpotRegionBedFile
- nucleotideType
- reference
- targetRegionBedFile
- The JSON data structure in selectedPlugins for IonReporter has been changed with the following added
- NucleotideType
- cancerType
- cellularityPct
- New VariantCaller parameters have been added and some parameters have been obsolete (persisted in selectedPlugins)
- New values for runType, applicationGroup and sampleGrouping have been added to support DNA and Fusions
- Some new attributes intended for internal use only have been added to plannedExperiment.
- We have started enforcing validation during REST API posting for
- barcodeId
- chipType
- flows
- notes
- planName
- project or projects
- runType
- sampleTubeLabel
- sample or sample in barcodedSamples
- sampleGroupingName
- sequencekitname
- templateKitName
- Posting that fails validation will receive an error code.
- Until stringent validation is fully in place during non-GUI REST API posting, please do your due diligence to ensure the data and data format posted are valid.

Moreover, some attributes require "internal" value instead of the "customer-facing" value to be persisted (e.g., sequencekitname, chipType). Please refer to the Comment/Expected Value column more details.

Validation Rules

RULE-1: Valid characters: letters, numbers, dashes, underscores, dots

RULE-2: Valid characters: letters, numbers, spaces, dashes, underscores, dots

RULE-3: Invalid leading characters: dashes, underscores, dots

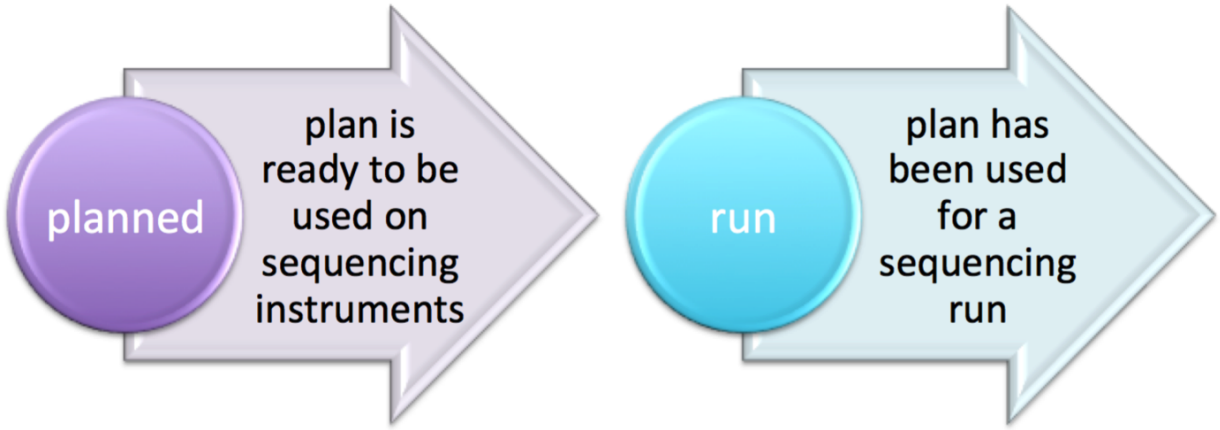
Field Notes

Attribute Name	Required/Optional/Nullable	Data type	Default value	Valid v
adapter	Opt/Nullable	varchar(256)		
applicationGroupDisplayedNamne	Opt/Nullable			DNA, I
autoAnalyze		Boolean	TRUE	
autoName	Opt/Nullable	varchar(512)		
barcodeId	Opt/Nullable	varchar(128)		
barcodedSamples	Opt/Nullable	json		
base_recalibrate	Opt	Boolean		
bedfile	Opt/Nullable	varchar(1024)		
chipBarcode	Opt/Nullable	varchar(64)		
chipType	Opt	varchar(32)		
controlSequencekitname	Opt/Nullable	varchar(512)		
cycles	Opt/Nullable	int		
date	Opt/Nullable	DateTimeField		
expName	Opt	varchar(128)		
flows	Req	int	0	
flowsInOrder	Opt/Nullable	varchar(512)		
forward3primeadapter	Req	varchar(512)		
id	Opt	int		
irworkflow	Opt	varchar(1024)		
isDuplicateReads	Opt	Boolean		
isFavorite	Opt	Boolean	FALSE	
isPlanGroup	Opt	Boolean	FALSE	
isReusable	Opt	Boolean	FALSE	
isReverseRun	Req	Boolean	FALSE	True,Fa
isSystem	Opt	Boolean	FALSE	
isSystemDefault	Opt	Boolean	FALSE	
libkit	Opt/Nullable	varchar(512)		
library	Opt/Nullable	varchar(512)		
libraryKey	Req	varchar(64)		
librarykitname	Opt/Nullable	varchar(512)		
metaData	Opt	json		
notes	Opt/Nullable	varchar(1024)		
pairedEndLibraryAdapterName	Opt/Nullable	varchar(512)		
parentPlan	Opt/Nullable	FK		
planDisplayedName		varchar(512)		
planExecuted	Opt	Boolean	FALSE	True,Fa

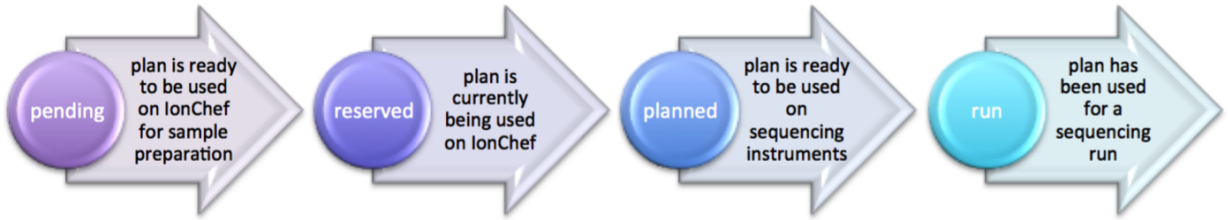
Attribute Name	Required/Optional/Nullable	Data type	Default value	Valid v
planExecutedDate	Opt/Nullable	DateTimeField		
planGUID	Opt/Nullable	varchar(512)		
planName		varchar(512)		
planPGM	Opt/Nullable	varchar(128)		
platform	Opt	varchar(128)	“”	“”, PGM
planShortID	Opt/Nullable			
planStatus		varchar(512)	planned	“”, pend
preAnalysis	Opt	Boolean		
projects	Opt	varchar(64) for each project name		
realign	Opt	Boolean		
regionfile	Opt/Nullable	varchar(1024)		
reverse_primer	Opt/Nullable	varchar(128)		
runMode	Opt	varchar(64)		“”,”sing
runType	Req	varchar(512)	GENS	“AMPS
runName	Opt/Nullable	varchar(255)		
sample	Required for plan	varchar(127)		
sampleDisplayedName	Opt/Nullable	varchar(127)		
sampleGroupingName	Opt/Nullable			DNA_F
samplePrepKitName	Opt/Nullable	varchar(512)		
sampleTubeLabel	Opt/Nullable	varchar(512)		
selectedPlugins	Opt/Nullable	json		
seqKitBarcode	Opt/Nullable	varchar(64)		
sequencekitname	Recommend to set	varchar(512)		
storageHost	Opt/Nullable	varchar(128)		
storage_options	Opt	varchar(200)	A	“KI”,”A
templatingKitName	Opt/Nullable	varchar(512)		
usePostBeadfind	Opt	Boolean		
usePreBeadfind	Opt	Boolean	TRUE	
username	Opt/Nullable	varchar(128)		

PlanStatus state transition

OneTouch



IonChef



barcodedSamples JSON Examples

Generic sequencing plan

```
"barcodedSamples": {
    "s 1": {
        "barcodeSampleInfo": {
            "IonSet1_16": {
                "controlSequenceType": "",
                "description": "desc 1",
                "externalId": "accession 101",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_desi
            }
        },
        "barcodes": [
            "IonSet1_16"
        ]
    },
    "s 2": {
        "barcodeSampleInfo": {
            "IonSet1_12": {
                "controlSequenceType": "",
                "description": "desc 2",
                "externalId": "accession 80",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_desi
            }
        },
        "barcodes": [
            "IonSet1_12"
        ]
    },
    "s 3": {
        "barcodeSampleInfo": {
            "IonSet1_15": {
                "controlSequenceType": "",
                "description": "desc 3",
```

```
                "externalId": "accession 280",
                "hotSpotRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_hot
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/19/hg19/unmerged/detail/4477685_CCP_desi
            }
        },
        "barcodes": [
            "IonSet1_15"
        ]
    }
},
```

Onconet DNA plan

```
"barcodedSamples": {
    "example 1": {
        "barcodeSampleInfo": {
            "IonXpress_010": {
                "controlSequenceType": "",
                "description": "example here",
                "externalId": "id 1",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            }
        },
        "barcodes": [
            "IonXpress_010"
        ]
    },
    "example 2": {
        "barcodeSampleInfo": {
            "IonXpress_005": {
                "controlSequenceType": "",
                "description": "another example here",
                "externalId": "id 2",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            }
        },
        "barcodes": [
            "IonXpress_005"
        ]
    }
},
```

Onconet DNA and Fusions plan

```
"barcodedSamples": {
    "s 1": {
        "barcodeSampleInfo": {
```

```
            "IonXpress_001": {
                "controlSequenceType": "",
                "description": "description here",
                "externalId": "ext 1",
                "hotSpotRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.20131
                "nucleotideType": "DNA",
                "reference": "hg19",
                "targetRegionBedFile": "/results/uploads/BED/22/hg19/unmerged/detail/ColonLung.201310
            },
            "IonXpress_002": {
                "controlSequenceType": "",
                "description": "description here",
                "externalId": "ext 1",
                "hotSpotRegionBedFile": "",
                "nucleotideType": "RNA",
                "reference": "hg19_rna",
                "targetRegionBedFile": ""
            }
        },
        "barcodes": [
            "IonXpress_001",
            "IonXpress_002"
        ]
    }
}
```

selectedPlugins JSON Examples

IonReporterUploader, coverageAnalysis, sampleId and variantCaller

```
"selectedPlugins": {
    "IonReporterUploader": {
        "features": [
            "export"
        ],
        "id": 700,
        "name": "IonReporterUploader",
        "userInput": {
            "accountId": "1234567890abcde",
            "accountName": " demo IonReporter (Version: 4.2 | User: Ion User | Org: IR Org)",
            "userInputInfo": [{
                "ApplicationType": "Low-Coverage Whole Genome Sequencing",
                "Gender": "Female",
                "NucleotideType": "DNA",
                "Relation": "Self",
                "RelationRole": "Self",
                "Workflow": "Test_WK_1",
                "barcodeId": "IonXpress_010",
                "cancerType": "Breast Cancer",
                "cellularityPct": "23",
                "sample": "example 1",
                "sampleDescription": "example here",
                "sampleExternalId": "id 1",
                "sampleName": "example_1",
                "setid": "1__4c310e03-d188-4702-b82a-f9043bc04350"
            }, {
```

```
                "ApplicationType": "Low-Coverage Whole Genome Sequencing",
                "Gender": "Male",
                "NucleotideType": "DNA",
                "Relation": "",
                "RelationRole": "Self",
                "Workflow": "Test_WK_1",
                "barcodeId": "IonXpress_005",
                "cancerType": "Liver Cancer",
                "cellularityPct": "27",
                "sample": "example 2",
                "sampleDescription": "another example here",
                "sampleExternalId": "id 2",
                "sampleName": "example_2",
                "setid": "2__4c310e03-d188-4702-b82a-f9043bc04350"
            }]
        },
        "version": "4.2-r88003"
    },
    "coverageAnalysis": {
        "features": [],
        "id": 696,
        "name": "coverageAnalysis",
        "userInput": "",
        "version": "4.2-r87890"
    },
    "sampleID": {
        "features": [],
        "id": 701,
        "name": "sampleID",
        "userInput": "",
        "version": "4.2-r87942"
    },
    "variantCaller": {
        "features": [],
        "id": 699,
        "name": "variantCaller",
        "userInput": {
            "freebayes": {
                "allow_complex": "0",
                "allow_indels": "1",
                "allow_mnps": "0",
                "allow_snps": "1",
                "gen_min_alt_allele_freq": "0.03",
                "gen_min_coverage": "6",
                "gen_min_indel_alt_allele_freq": "0.1",
                "min_base_qv": "2",
                "min_mapping_qv": "4",
                "read_max_mismatch_fraction": "1.0",
                "read_mismatch_limit": "10"
            },
            "long_indel_assembler": {
                "kmer_len": "19",
                "max_hp_length": "8",
                "min_indel_size": "4",
                "min_var_count": "5",
                "min_var_freq": "0.15",
                "relative_strand_bias": "0.8",
                "short_suffix_match": "5"
```

```
},
"meta": {
    "built_in": true,
    "compatibility": {
        "chip": [
            "pgm",
            "proton_p1"
        ],
        "library": [
            "ampliseq"
        ],
        "panel": "/rundb/api/v1/contentupload/22/"
    },
    "configuration": "",
    "librarytype": "ampliseq",
    "name": "Panel-optimized - Colon and Lung Panel - 10/7/2013",
    "repository_id": "",
    "tooltip": "Panel-optimized parameters from AmpliSeq.com",
    "trimreads": true,
    "ts_version": "4.0",
    "tvcargs": "tvc",
    "user_selections": {
        "chip": "pgm",
        "frequency": "germline",
        "library": "ampliseq",
        "panel": "/rundb/api/v1/contentupload/22/"
    }
},
"torrent_variant_caller": {
    "data_quality_stringency": "6.5",
    "downsample_to_coverage": "10000",
    "filter_deletion_predictions": "0.2",
    "filter_insertion_predictions": "0.2",
    "filter_unusual_predictions": "0.3",
    "heavy_tailed": "3",
    "hotspot_beta_bias": "100.0",
    "hotspot_min_allele_freq": "0.01",
    "hotspot_min_cov_each_strand": "2",
    "hotspot_min_coverage": "6",
    "hotspot_min_variant_score": "6",
    "hotspot_strand_bias": "0.95",
    "hp_max_length": "8",
    "indel_beta_bias": "10.0",
    "indel_min_allele_freq": "0.05",
    "indel_min_cov_each_strand": "2",
    "indel_min_coverage": "15",
    "indel_min_variant_score": "6",
    "indel_strand_bias": "0.9",
    "outlier_probability": "0.01",
    "prediction_precision": "1.0",
    "snp_beta_bias": "100.0",
    "snp_min_allele_freq": "0.02",
    "snp_min_cov_each_strand": "0",
    "snp_min_coverage": "6",
    "snp_min_variant_score": "6",
    "snp_strand_bias": "0.95"
}
bbb
```

```
        },
        "version": "4.2-r87667"
    }
},
"seqKitBarcode": null,
"sequencekitname": "IonPGM200Kit-v2",
"storageHost": null,
"storage_options": "A",
"templatingKitBarcode": null,
"templatingKitName": "Ion PGM Template OT2 200 Kit",
"tfKey": "ATCG",
"thumbnailalignmentargs": "",
"thumbnailanalysisargs": "",
"thumbnailbasecallerargs": "",
"thumbnailbeadfindargs": "",
"thumbnailcalibrateargs": "",
"usePostBeadfind": true,
"usePreBeadfind": true,
"username": "ionadmin",
"variantfrequency": ""
},
```

IonReporterUploader selected for a Onconet DNA and Fusions plan

```
"selectedPlugins": {
    "IonReporterUploader": {
        "features": [
            "export"
        ],
        "id": 700,
        "name": "IonReporterUploader",
        "userInput": {
            "accountId": "1234567890abcde ",
            "accountName": "demo IonReporter (Version: 4.2 | User: Ion User | Org: IR Org)",
            "userInputInfo": [{
                "ApplicationType": "Oncomine_DNA_RNA_Fusion",
                "Gender": "Male",
                "NucleotideType": "DNA",
                "Relation": "DNA_RNA",
                "RelationRole": "Self",
                "Workflow": "AmpliSeq Colon Lung v2 with RNA Lung Fusion single sample",
                "barcodeId": "IonXpress_001",
                "cancerType": "Colorectal Cancer",
                "cellularityPct": "17",
                "sample": "s 1",
                "sampleDescription": "description here",
                "sampleExternalId": "ext 1",
                "sampleName": "s_1",
                "setid": "1__381a5a84-5af0-40ff-84c1-b31720fea6ca"
            }, {
                "ApplicationType": "Oncomine_DNA_RNA_Fusion",
                "Gender": "Male",
                "NucleotideType": "RNA",
                "Relation": "DNA_RNA",
                "RelationRole": "Self",
                "Workflow": "AmpliSeq Colon Lung v2 with RNA Lung Fusion single sample",
```

```
                "barcodeId": "IonXpress_002",
                "cancerType": "Colorectal Cancer",
                "cellularityPct": "17",
                "sample": "s 1",
                "sampleDescription": "description here",
                "sampleExternalId": "ext 1",
                "sampleName": "s_1",
                "setid": "1__381a5a84-5af0-40ff-84c1-b31720fea6ca"
            }]
        },
        "version": "4.2-r88003"
    }
},
```

Creating a plan

Non-barcoded PGM

Post a non-barcoded Target Sequencing PGM plan and to associate results with 2 projects with sampleGrouping and applicationGroup specified:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forw ard3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 500,
    "library": "hg19",
    "bedfile": "/results/uploads/BED/71/hg19/unmerged/detail/CFTRexon.20131001.designed.bed",
    "regionfile": "/results/uploads/BED/71/hg19/unmerged/detail/CFTRexon.20131125.hotspots.bed",
    "planName": "DEMO-TS4_2_x-REST- API_TARS_plan1",
    "sample": "my_sample",
    "notes": "this is a REST test plan",
    "username": "ionuser",
    "preAnalysis": "on",
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "TARS",
    "chipType": "318v2",
    "sequencekitname": "IonPGM200Kit",
    "librarykitname": "Ion Xpress Plus Fragment Library Kit",
    "templatingKitName": "Ion PGM Template OT2 200 Kit",
    "samplePrepKitName": "Ion TargetSeq(tm) Custom Enrichment Kit (100kb-500kb)",
    "projects": ["myProject1", "myProject2"],
    "sampleGroupingName": "Self",
    "applicationGroupDisplayedName": "DNA"
}
```

Non-Barcoded PI

Post a non-barcoded Target Sequencing Proton plan with PI chip, with sample tube label, chip barcode and the QC thresholds specified:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 440,
    "library": "hg19",
    "bedfile": "/results/uploads/BED/14/hg19/unmerged/detail/BRCA1_2.20131001.designed.bed",
    "regionfile": "/results/uploads/BED/14/hg19/unmerged/detail/BRCA1_2.20131001.hotspots.bed",
    "planName": "DEMO-TS4_2_x-REST-API_TARS_Proton_plan2",
    "sample": "my_sample",
    "notes": "here are my notes",
    "username": "ionuser",
    "preAnalysis": "on",
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "TARS",
    "chipType": "P1.1.17",
    "sequencekitname": "ProtonI200Kit-v3",
    "librarykitname": "Ion Xpress Plus Fragment Library Kit",
    "templatingKitName": "Ion PI Template OT2 200 Kit v3",
    "samplePrepKitName": "Ion TargetSeq(tm) Exome Kit (4 rxn)",
    "projects": ["myProject1"],
    "sampleTubeLabel": "abcX254",
    "chipBarcode": "AA02314571",
    "Bead Loading (%)": 33,
    "Key Signal (1-100)": 35,
    "Usable Sequence (%)": 37
}
```

Barcoded RNA PGM

Post a barcoded RNA Sequencing PGM plan:

```
{
    "autoAnalyze": "true",
    "usePreBeadfind": "true",
    "usePostBeadfind": "true",
    "reverselibrarykey": "",
    "reverse3primeadapter": "",
    "libraryKey": "TCAG",
    "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
    "flows": 160,
    "library": "hg19_rna",
    "planName": "DEMO-TS4_2_x-REST- API_barcoded_RNA_plan3",
    "notes": "test notes here ",
    "username": "ionuser",
    "preAnalysis": "on",
```

```
    "isReverseRun": false,
    "isPlanGroup": false,
    "runMode": "single",
    "runType": "RNA",
    "chipType": "318v2",
    "sequencekitname": "IonPGM200Kit-v2",
    "librarykitname": "Ion Total RNA Seq Kit v2",
    "templatingKitName": "Ion PGM Template OT2 200 Kit",
    "samplePrepKitName": "",
    "projects": ["myProject1", "myProject2"],
    "barcodedSamples": "{'demo sample 1':{'barcodeSampleInfo':{'IonXpressRNA_003':{'controlSequenceTy
    "applicationGroupDisplayedName": "RNA",
    "barcodeId": "IonXpressRNA",
    "sampleTubeLabel": "2554abc",
    "Bead Loading (%)": 30,
    "Key Signal (1-100)": 30,
    "Usable Sequence (%)": 30
}
```


Using POST to update a plan

If you are to update a plan via REST API, please perform a GET first so you'll have all the internally created values for the plan to perform the update with a POST.

To update with a POST, just include "id": <plan PK> in your data packet (e.g., "id":1234)

About using PUT or PATCH to update a plan

Update a plan for its chipBarcode value

`http://<hostname>/rundb/api/v1/plannedexperiment/<plan pk>/?format=json`


```
{
    "chipBarcode": "AA323323"
}
```


Fields table

field	help text
planDisplayedName	Unicode string data. Ex: "Hello World"
autoAnalyze	Boolean data. Ex: True
templatingKitBarcode	Unicode string data. Ex: "Hello World"
preAnalysis	Boolean data. Ex: True
chefStatus	Unicode string data. Ex: "Hello World"
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
libkit	Unicode string data. Ex: "Hello World"
platform	Unicode string data. Ex: "Hello World"
categories	Unicode string data. Ex: "Hello World"
planPGM	Unicode string data. Ex: "Hello World"
sampleSet_planTotal	Integer data. Ex: 2673
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.

Table 4.17 – continued from previous page

field	help text
notes	Unicode string data. Ex: “Hello World”
sequencekitname	Unicode string data. Ex: “Hello World”
storageHost	Unicode string data. Ex: “Hello World”
expName	Unicode string data. Ex: “Hello World”
cycles	Integer data. Ex: 2673
isReverseRun	Boolean data. Ex: True
storage_options	Unicode string data. Ex: “Hello World”
chipType	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
library	Unicode string data. Ex: “Hello World”
reverselibrarykey	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
seqKitBarcode	Unicode string data. Ex: “Hello World”
barcodeId	Unicode string data. Ex: “Hello World”
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
realign	Boolean data. Ex: True
sampleGroupingName	Unicode string data. Ex: “Hello World”
experiment	A single related resource. Can be either a URI or set of nested resource data.
bedfile	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
isDuplicateReads	Boolean data. Ex: True
librarykitname	Unicode string data. Ex: “Hello World”
adapter	Unicode string data. Ex: “Hello World”
tfKey	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”
forward3primeadapter	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
samplePrepKitName	Unicode string data. Ex: “Hello World”
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
planStatus	Unicode string data. Ex: “Hello World”
templatingKitName	Unicode string data. Ex: “Hello World”
runType	Unicode string data. Ex: “Hello World”
username	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
sampleDisplayedName	Unicode string data. Ex: “Hello World”
controlSequencekitname	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
childPlans	A list of data. Ex: [’abc’, 26.73, 8]
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”
runMode	Unicode string data. Ex: “Hello World”
irworkflow	Unicode string data. Ex: “Hello World”
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: “Hello World”

Table 4.17 – continued from previous page

field	help text
usePostBeadfind	Boolean data. Ex: True
runname	Unicode string data. Ex: “Hello World”
planGUID	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
sampleSetGroupType	Unicode string data. Ex: “Hello World”
sample	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
reverse_primer	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
barcodedSamples	Unicode string data. Ex: “Hello World”
regionfile	Unicode string data. Ex: “Hello World”
selectedPlugins	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
isSystemDefault	Boolean data. Ex: True
autoName	Unicode string data. Ex: “Hello World”
libraryKey	Unicode string data. Ex: “Hello World”
flows	Integer data. Ex: 2673
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
variantfrequency	Unicode string data. Ex: “Hello World”
sampleSetDisplayedName	Unicode string data. Ex: “Hello World”
flowsInOrder	Unicode string data. Ex: “Hello World”
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
base_recalibrate	Boolean data. Ex: True
chipBarcode	Unicode string data. Ex: “Hello World”
usePreBeadfind	Boolean data. Ex: True
resource_uri	Unicode string data. Ex: “Hello World”
reverse3primeadapter	Unicode string data. Ex: “Hello World”

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperiment/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperiment/", params={"form
ts_api_response = ts_api_request.json()

plannedexperiments = ts_api_response["objects"]

for plannedexperiment in plannedexperiments:
    print plannedexperiment
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 18369,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperiment/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "planDisplayedName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
            "autoAnalyze": true,
            "templatingKitBarcode": null,
            "preAnalysis": true,
            "chefStatus": "",
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "libkit": null,
            "platform": "PROTON",
            "categories": "",
            "planPGM": null,
            "prebasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --key
            "alignmentargs": "stage1 map4",
            "thumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10
            "sampleSet_planTotal": 0,
            "projects": [],
            "notes": "",
            "sequencekitname": "ProtonI200Kit-v3",
            "storageHost": null,
            "expName": "R_2014_06_27_14_19_01_user_P19-606",
            "cycles": null,
            "isReverseRun": false,
            "storage_options": "A",
            "thumbnailalignmentargs": "stage1 map4",
            "chipType": "P1.0.20",
            "chefProgress": 0.0,
            "library": "hg19",
            "reverselibrarykey": "",
            "sampleTubeLabel": null,
            "seqKitBarcode": null,
            "barcodeId": "",
            "chefLogPath": null,
            "isPlanGroup": false,
            "realign": false,
            "sampleGroupingName": "",
            "experiment": "/rundb/api/v1/experiment/23975/",
            "bedfile": "",
            "isReusable": false,
            "isDuplicateReads": false,
            "thumbnailbeadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=50,50 --bead
            "librarykitname": "Ion Xpress Plus Fragment Library Kit",
            "adapter": null,
            "basecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads 10 --keypas
            "tfKey": "ATCG",
            "parentPlan": null,
            "forward3primeadapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "planStatus": "run",
            "chefLastUpdate": null,
            "samplePrepKitName": null,
```

```
"applicationGroupDisplayedName": "DNA",
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266571,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266571/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266570,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266570/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266569,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Usable Sequence (%)",
            "id": 3,
            "resource_uri": "/rundb/api/v1/qctype/3/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266569/"
    }
],
"analysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --region-size=80,
"thumbnailcalibrateargs": "calibrate --skipDroop",
"templatingKitName": "Ion PI Template OT2 200 Kit v3",
"runType": "GENS",
"username": null,
"planName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
```

```
            "sampleDisplayedName": "",
            "prethumbnailbasecallerargs": "BaseCaller --barcode-filter 0.01 --barcode-filter-minreads
            "controlSequencekitname": null,
            "chefMessage": "",
            "childPlans": [],
            "pairedEndLibraryAdapterName": null,
            "runMode": "single",
            "irworkflow": "",
            "planExecuted": true,
            "project": "",
            "usePostBeadfind": false,
            "runname": null,
            "planGUID": "80384c15-e0e2-4909-9d4c-950731bf1cf4",
            "planShortID": "G5B50",
            "sampleSetGroupType": null,
            "sample": "",
            "planExecutedDate": null,
            "reverse_primer": null,
            "id": 102212,
            "barcodedSamples": {},
            "regionfile": "",
            "selectedPlugins": {},
            "beadfindargs": "justBeadFind --beadfind-minlivesnr 3 --region-size=80,112 --total-timeou
            "sampleSet": null,
            "isSystemDefault": false,
            "autoName": null,
            "libraryKey": "TCAG",
            "flows": 260,
            "thumbnailanalysisargs": "Analysis --from-beadfind --clonal-filter-bkgmodel true --regio
            "date": "2014-06-27T23:36:06.000229+00:00",
            "isSystem": false,
            "variantfrequency": "",
            "sampleSetDisplayedName": "",
            "calibrateargs": "calibrate --skipDroop",
            "flowsInOrder": "TACGTACGTCTGAGCATCGATCGATGTACAGC",
            "sampleGrouping": null,
            "base_recalibrate": true,
            "chipBarcode": null,
            "usePreBeadfind": true,
            "resource_uri": "/rundb/api/v1/plannedexperiment/102212/",
            "reverse3primeadapter": ""
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.50 Plannedexperimentdb Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/schema/`

Fields table

field	help text
isReverseRun	Boolean data. Ex: True
planDisplayedName	Unicode string data. Ex: “Hello World”
storage_options	Unicode string data. Ex: “Hello World”
preAnalysis	Boolean data. Ex: True
isSystemDefault	Boolean data. Ex: True
username	Unicode string data. Ex: “Hello World”
planShortID	Unicode string data. Ex: “Hello World”
planStatus	Unicode string data. Ex: “Hello World”
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”
templatingKitBarcode	Unicode string data. Ex: “Hello World”
sampleTubeLabel	Unicode string data. Ex: “Hello World”
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”
chefStatus	Unicode string data. Ex: “Hello World”
samplePrepKitName	Unicode string data. Ex: “Hello World”
reverse_primer	Unicode string data. Ex: “Hello World”
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.
seqKitBarcode	Unicode string data. Ex: “Hello World”
id	Integer data. Ex: 2673
metaData	Unicode string data. Ex: “Hello World”
sampleSet_uid	Unicode string data. Ex: “Hello World”
isFavorite	Boolean data. Ex: True
sampleSet_planIndex	Integer data. Ex: 2673
qcValues	Many related resources. Can be either a list of URIs or list of individually nested resource data.
chefLogPath	Unicode string data. Ex: “Hello World”
isPlanGroup	Boolean data. Ex: True
sampleSet_planTotal	Integer data. Ex: 2673
experiment	A single related resource. Can be either a URI or set of nested resource data.
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.
runType	Unicode string data. Ex: “Hello World”
sampleSet	A single related resource. Can be either a URI or set of nested resource data.
planPGM	Unicode string data. Ex: “Hello World”
chefProgress	Floating point numeric data. Ex: 26.73
autoName	Unicode string data. Ex: “Hello World”
isReusable	Boolean data. Ex: True
controlSequencekitname	Unicode string data. Ex: “Hello World”
date	A date & time as a string. Ex: “2010-11-10T03:07:43”
isSystem	Boolean data. Ex: True
libkit	Unicode string data. Ex: “Hello World”
categories	Unicode string data. Ex: “Hello World”
planName	Unicode string data. Ex: “Hello World”
chefMessage	Unicode string data. Ex: “Hello World”
parentPlan	Unicode string data. Ex: “Hello World”

Table 4.18 – continued from previous page

field	help text
childPlans	A list of data. Ex: ['abc', 26.73, 8]
templatingKitName	Unicode string data. Ex: "Hello World"
pairedEndLibraryAdapterName	Unicode string data. Ex: "Hello World"
runMode	Unicode string data. Ex: "Hello World"
adapter	Unicode string data. Ex: "Hello World"
sampleGrouping	A single related resource. Can be either a URI or set of nested resource data.
irworkflow	Unicode string data. Ex: "Hello World"
chipBarcode	Unicode string data. Ex: "Hello World"
planExecuted	Boolean data. Ex: True
project	Unicode string data. Ex: "Hello World"
usePostBeadfind	Boolean data. Ex: True
storageHost	Unicode string data. Ex: "Hello World"
expName	Unicode string data. Ex: "Hello World"
runname	Unicode string data. Ex: "Hello World"
usePreBeadfind	Boolean data. Ex: True
planGUID	Unicode string data. Ex: "Hello World"
cycles	Integer data. Ex: 2673
resource_uri	Unicode string data. Ex: "Hello World"

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentdb/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperimentdb/", params={"fo
ts_api_response = ts_api_request.json()

plannedexperimentdbs = ts_api_response["objects"]

for plannedexperimentdb in plannedexperimentdbs:
    print plannedexperimentdb
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18369,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperimentdb/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
            "storage_options": "A",
```

```
"preAnalysis": true,
"isSystemDefault": false,
"username": null,
"planShortID": "G5B50",
"planStatus": "run",
"chefLastUpdate": null,
"templatingKitBarcode": null,
"sampleTubeLabel": null,
"planExecutedDate": null,
"chefStatus": "",
"samplePrepKitName": null,
"reverse_primer": null,
"applicationGroup": "/rundb/api/v1/applicationgroup/1/",
"seqKitBarcode": null,
"id": 102212,
"metaData": {},
"sampleSet_uid": null,
"isFavorite": false,
"sampleSet_planIndex": 0,
"qcValues": [
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266571,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266571/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266570,
        "qcType": {
            "description": "",
            "minThreshold": 1,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Key Signal (1-100)",
            "id": 2,
            "resource_uri": "/rundb/api/v1/qctype/2/"
        },
        "resource_uri": "/rundb/api/v1/plannedexperimentqc/266570/"
    },
    {
        "threshold": 30,
        "plannedExperiment": "/rundb/api/v1/plannedexperiment/102212/",
        "id": 266569,
        "qcType": {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
```

```
                    "defaultThreshold": 30,
                    "qcName": "Usable Sequence (%)",
                    "id": 3,
                    "resource_uri": "/rundb/api/v1/qctype/3/"
                },
                "resource_uri": "/rundb/api/v1/plannedexperimentqc/266569/"
            }
        ],
        "chefLogPath": null,
        "isPlanGroup": false,
        "sampleSet_planTotal": 0,
        "experiment": "/rundb/api/v1/experiment/23975/",
        "projects": [],
        "runType": "GENS",
        "sampleSet": null,
        "planPGM": null,
        "chefProgress": 0.0,
        "autoName": null,
        "isReusable": false,
        "controlSequencekitname": null,
        "date": "2014-06-27T23:36:06.000229+00:00",
        "isSystem": false,
        "libkit": null,
        "categories": "",
        "planName": "CopyOfSystemDefault_R_2014_06_27_14_19_01_user_P19-606",
        "chefMessage": "",
        "parentPlan": null,
        "childPlans": [],
        "templatingKitName": "Ion PI Template OT2 200 Kit v3",
        "pairedEndLibraryAdapterName": null,
        "runMode": "single",
        "adapter": null,
        "sampleGrouping": null,
        "irworkflow": "",
        "chipBarcode": null,
        "planExecuted": true,
        "project": "",
        "usePostBeadfind": false,
        "storageHost": null,
        "expName": "R_2014_06_27_14_19_01_user_P19-606",
        "runname": null,
        "usePreBeadfind": true,
        "planGUID": "80384c15-e0e2-4909-9d4c-950731bf1cf4",
        "cycles": null,
        "resource_uri": "/rundb/api/v1/plannedexperimentdb/102212/"
    }
]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.51 Plannedexperimentqc Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
threshold	Integer data. Ex: 2673	0	false	false	false	false	integer
plannedExperiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
qcType	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plannedexperimentqc/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plannedexperimentqc/", params={"fo
ts_api_response = ts_api_request.json()

plannedexperimentqcs = ts_api_response["objects"]

for plannedexperimentqc in plannedexperimentqcs:
    print plannedexperimentqc
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 22524,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plannedexperimentqc/?offset=1&limit=1&format=json"
```

```
    },
    "objects": [
        {
            "threshold": 30,
            "plannedExperiment": "/rundb/api/v1/plannedexperiment/86049/",
            "id": 247513,
            "qcType": {
                "description": "",
                "minThreshold": 0,
                "maxThreshold": 100,
                "defaultThreshold": 30,
                "qcName": "Bead Loading (%)",
                "id": 1,
                "resource_uri": "/rundb/api/v1/qctype/1/"
            },
            "resource_uri": "/rundb/api/v1/plannedexperimentqc/247513/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.52 Plantemplatebasicinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/schema/`

Fields table

field	help text	default	null
isReverseRun	Boolean data. Ex: True	false	false
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true
storage_options	Unicode string data. Ex: “Hello World”	A	false
preAnalysis	Boolean data. Ex: True		false
reference	Unicode string data. Ex: “Hello World”		true
isSystemDefault	Boolean data. Ex: True	false	false
hotSpotRegionBedFile	Unicode string data. Ex: “Hello World”		true
planShortID	Unicode string data. Ex: “Hello World”	n/a	true
planStatus	Unicode string data. Ex: “Hello World”		false
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true

Table 4.19 – continued from previous page

field	help text	default	null
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
chefStatus	Unicode string data. Ex: “Hello World”		false
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true
applicationGroup	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
applicationGroupDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true
id	Integer data. Ex: 2673		false
metaData	Unicode string data. Ex: “Hello World”	{}	false
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true
isFavorite	Boolean data. Ex: True	false	false
sampleSet_planIndex	Integer data. Ex: 2673	0	false
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true
isPlanGroup	Boolean data. Ex: True	false	false
sampleGroupName	Unicode string data. Ex: “Hello World”	n/a	true
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true
barcodeKitName	Unicode string data. Ex: “Hello World”	n/a	true
runType	Unicode string data. Ex: “Hello World”	GENS	false
planPGM	Unicode string data. Ex: “Hello World”	n/a	true
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false
autoName	Unicode string data. Ex: “Hello World”	n/a	true
isReusable	Boolean data. Ex: True	false	false
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true
irAccountName	Unicode string data. Ex: “Hello World”	n/a	true
sequencingInstrumentType	Unicode string data. Ex: “Hello World”	n/a	true
targetRegionBedFile	Unicode string data. Ex: “Hello World”		true
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true
isSystem	Boolean data. Ex: True	false	false
libkit	Unicode string data. Ex: “Hello World”	n/a	true
categories	Unicode string data. Ex: “Hello World”		true
planName	Unicode string data. Ex: “Hello World”	n/a	true
chefMessage	Unicode string data. Ex: “Hello World”		false
templatePrepInstrumentType	Unicode string data. Ex: “Hello World”	n/a	true
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true
runMode	Unicode string data. Ex: “Hello World”		false
adapter	Unicode string data. Ex: “Hello World”	n/a	true
irworkflow	Unicode string data. Ex: “Hello World”		false
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true
planExecuted	Boolean data. Ex: True	false	false
username	Unicode string data. Ex: “Hello World”	n/a	true
usePostBeadfind	Boolean data. Ex: True		false
storageHost	Unicode string data. Ex: “Hello World”	n/a	true
expName	Unicode string data. Ex: “Hello World”		false
runname	Unicode string data. Ex: “Hello World”	n/a	true
usePreBeadfind	Boolean data. Ex: True		false
planGUID	Unicode string data. Ex: “Hello World”	n/a	true
cycles	Integer data. Ex: 2673	n/a	true
notes	Unicode string data. Ex: “Hello World”		true
sampleSet_planTotal	Integer data. Ex: 2673	0	false

Table 4.19 – continued from previous page

field	help text	default	null
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plantemplatebasicinfo/", params={'
ts_api_response = ts_api_request.json()

plantemplatebasicinfos = ts_api_response["objects"]

for plantemplatebasicinfo in plantemplatebasicinfos:
    print plantemplatebasicinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 66,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plantemplatebasicinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "reference": "hg19",
            "isSystemDefault": false,
            "hotSpotRegionBedFile": "",
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "applicationGroup": "/rundb/api/v1/applicationgroup/1/",
            "applicationGroupDisplayedName": "DNA",
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
```

```
            "sampleSet_planIndex": 0,
            "seqKitBarcode": null,
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleGroupName": "",
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "barcodeKitName": "IonXpress",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "irAccountName": "None",
            "sequencingInstrumentType": "PROTON",
            "targetRegionBedFile": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "templatePrepInstrumentType": "OneTouch",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "notes": "Uploaded from amplseq.com and available at-jira.itw_wiki_x_HAHcAg\r\nReplace _
            "sampleSet_planTotal": 0,
            "resource_uri": "/rundb/api/v1/plantemplatebasicinfo/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get

4.1.53 Plantemplatesummary Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/schema/`

Fields table

field	help text	default	nullable	readonly	b
isReverseRun	Boolean data. Ex: True	false	false	false	tr
planDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
storage_options	Unicode string data. Ex: “Hello World”	A	false	false	fa
preAnalysis	Boolean data. Ex: True		false	false	tr
isSystemDefault	Boolean data. Ex: True	false	false	false	tr
planShortID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planStatus	Unicode string data. Ex: “Hello World”		false	false	tr
chefLastUpdate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
templatingKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
sampleTubeLabel	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecutedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
chefStatus	Unicode string data. Ex: “Hello World”		false	false	tr
samplePrepKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
reverse_primer	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
seqKitBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
id	Integer data. Ex: 2673		false	false	tr
metaData	Unicode string data. Ex: “Hello World”	{}	false	false	tr
sampleSet_uid	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isFavorite	Boolean data. Ex: True	false	false	false	tr
sampleSet_planIndex	Integer data. Ex: 2673	0	false	false	fa
chefLogPath	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isPlanGroup	Boolean data. Ex: True	false	false	false	tr
sampleSet_planTotal	Integer data. Ex: 2673	0	false	false	fa
templatingKitName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runType	Unicode string data. Ex: “Hello World”	GENS	false	false	fa
planPGM	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefProgress	Floating point numeric data. Ex: 26.73	0.0	false	false	tr
autoName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
isReusable	Boolean data. Ex: True	false	false	false	tr
controlSequencekitname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	n/a	true	false	fa
isSystem	Boolean data. Ex: True	false	false	false	tr
libkit	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
categories	Unicode string data. Ex: “Hello World”		true	false	fa
planName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
chefMessage	Unicode string data. Ex: “Hello World”		false	false	tr
pairedEndLibraryAdapterName	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
runMode	Unicode string data. Ex: “Hello World”		false	false	tr
adapter	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
irworkflow	Unicode string data. Ex: “Hello World”		false	false	tr
chipBarcode	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
planExecuted	Boolean data. Ex: True	false	false	false	tr
username	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePostBeadfind	Boolean data. Ex: True		false	false	tr
storageHost	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
expName	Unicode string data. Ex: “Hello World”		false	false	tr
runname	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
usePreBeadfind	Boolean data. Ex: True		false	false	tr

Table 4.20 – continued from previous page

field	help text	default	nullable	readonly	b
planGUID	Unicode string data. Ex: “Hello World”	n/a	true	false	fa
cycles	Integer data. Ex: 2673	n/a	true	false	fa
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	fa

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plantemplatesummary/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plantemplatesummary/", params={"fo
ts_api_response = ts_api_request.json()

plantemplatesummarys = ts_api_response["objects"]

for plantemplatesummary in plantemplatesummarys:
    print plantemplatesummary
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 66,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plantemplatesummary/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isReverseRun": false,
            "planDisplayedName": "chef_nonGUI_Exome Panel_AmpliSeqExome.20131001",
            "storage_options": "A",
            "preAnalysis": true,
            "isSystemDefault": false,
            "planShortID": "2OXYZ",
            "planStatus": "planned",
            "chefLastUpdate": null,
            "templatingKitBarcode": null,
            "sampleTubeLabel": "",
            "planExecutedDate": null,
            "chefStatus": "",
            "samplePrepKitName": "Ion AmpliSeq Exome Kit",
            "reverse_primer": null,
            "seqKitBarcode": null,
            "id": 100224,
            "metaData": {},
            "sampleSet_uid": null,
            "isFavorite": true,
            "sampleSet_planIndex": 0,
```

```
            "chefLogPath": null,
            "isPlanGroup": false,
            "sampleSet_planTotal": 0,
            "templatingKitName": "Ion PI Template OT2 200 Kit v3",
            "runType": "AMPS_EXOME",
            "planPGM": null,
            "chefProgress": 0.0,
            "autoName": null,
            "isReusable": true,
            "controlSequencekitname": "",
            "date": "2014-05-20T13:55:02.000398+00:00",
            "isSystem": false,
            "libkit": null,
            "categories": "",
            "planName": "chef_nonGUI_Exome_Panel_AmpliSeqExome.20131001",
            "chefMessage": "",
            "pairedEndLibraryAdapterName": "",
            "runMode": "single",
            "adapter": null,
            "irworkflow": "",
            "chipBarcode": null,
            "planExecuted": false,
            "username": "ionuser",
            "usePostBeadfind": false,
            "storageHost": null,
            "expName": "",
            "runname": null,
            "usePreBeadfind": true,
            "planGUID": "a81c02b6-e8f7-411f-aeeb-302fa16baa51",
            "cycles": null,
            "resource_uri": "/rundb/api/v1/plantemplatesummary/100224/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.54 Plugin Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/plugin/`

Schema URL: `http://mytorrentserver/rundb/api/v1/plugin/schema/`

Perform read-only operations on `plugin` resources and data elements

Fields table

field	help text	default	nullable	read-only	blank	unique	type
active	Boolean data. Ex: True	true	false	false	true	false	boolean
id	Integer data. Ex: 2673		false	false	true	true	integer
isPlanConfig	Boolean data. Ex: True	n/a	false	true	false	false	boolean
autorunMutable	Boolean data. Ex: True	true	false	false	true	false	boolean
script	Unicode string data. Ex: "Hello World"		false	false	true	false	string
selected	Boolean data. Ex: True	false	false	false	true	false	boolean
version	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
hasAbout	Boolean data. Ex: True	n/a	false	true	false	false	boolean
input	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
majorBlock	Boolean data. Ex: True	false	false	false	true	false	boolean
status	Unicode string data. Ex: "Hello World"		true	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
autorun	Boolean data. Ex: True	false	false	false	true	false	boolean
pluginsettings	Unicode string data. Ex: "Hello World"		true	false	false	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
isConfig	Boolean data. Ex: True	n/a	false	true	false	false	boolean
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
userinputfields	Unicode string data. Ex: "Hello World"	{}	true	false	false	false	string
url	Unicode string data. Ex: "Hello World"		false	false	true	false	string
config	Unicode string data. Ex: "Hello World"		true	false	false	false	string
versionedName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
isInstance	Boolean data. Ex: True	n/a	false	true	false	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/plugin/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/plugin/", params={"format": "json"
ts_api_response = ts_api_request.json()

plugins = ts_api_response["objects"]

for plugin in plugins:
    print plugin
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 102,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/plugin/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "active": true,
            "id": 493,
            "isPlanConfig": false,
            "autorunMutable": true,
            "script": "launch.sh",
            "selected": true,
            "version": "0.2.0",
            "hasAbout": false,
            "input": "False",
            "majorBlock": false,
            "status": {},
            "description": "Ion Torrent Plugin - 'BarcodeAlignStats' v0.2.0",
            "autorun": false,
            "pluginsettings": {
                "runtype": [
                    "wholechip",
                    "thumbnail"
                ],
                "depends": [],
                "features": [],
                "runlevel": []
            },
            "date": "2013-05-30T21:32:15.000437+00:00",
            "path": "/results/plugins/BarcodeAlignStats",
            "isConfig": false,
            "name": "BarcodeAlignStats",
            "userinputfields": {},
            "url": "",
            "config": {},
            "versionedName": "BarcodeAlignStats--v0.2.0",
            "isInstance": false,
            "resource_uri": "/rundb/api/v1/plugin/493/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete

- patch

4.1.55 Pluginresult Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/pluginresult/`

Schema URL: `http://mytorrentserver/rundb/api/v1/pluginresult/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
size	Unicode string data. Ex: "Hello World"	-1	false	false	false	false	string
apikey	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
plugin	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resultName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
pluginVersion	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
jobid	Integer data. Ex: 2673	n/a	true	false	false	false	integer
owner	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
pluginName	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
state	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
result	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
starttime	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	true	false	false	false	datetime
duration	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
path	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
store	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
endtime	A date & time as a string. Ex: "2010-11-10T03:07:43"	n/a	true	false	false	false	datetime
config	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
inodes	Unicode string data. Ex: "Hello World"	-1	false	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/pluginresult/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/pluginresult/", params={"format":
ts_api_response = ts_api_request.json()

pluginresults = ts_api_response["objects"]

for pluginresult in pluginresults:
    print pluginresult
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 788680,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/pluginresult/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "size": "3484344",
            "apikey": null,
            "plugin": "/rundb/api/v1/plugin/823/",
            "resultName": "Auto_user_Z28-428--r65714-pou4_dbsa_23958",
            "reportLink": "/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/",
            "pluginVersion": "4.2-r88266",
            "jobid": 3145172,
            "owner": "/rundb/api/v1/user/1/",
            "pluginName": "AssemblerSPAdes",
            "state": "Completed",
            "result": "/rundb/api/v1/results/304393/",
            "starttime": "2014-06-28T13:49:00.000133+00:00",
            "duration": "0:01:54.098431",
            "path": "/results/analysis/output/Home/Auto_user_Z28-428--r65714-pou4_dbsa_23958_304393/p
            "store": {},
            "endtime": "2014-06-28T13:50:54.000232+00:00",
            "config": {
                "only_barcodes": "",
                "spadesOptions": "-k 21,33,55,77,99",
                "spadesversion": "3.1.0",
                "RAM": "32G",
                "min_reads": "500",
                "bgenome": "None",
                "runSpades": "1",
                "fraction_of_reads": "1"
            },
            "id": 815369,
            "inodes": "396",
            "resource_uri": "/rundb/api/v1/pluginresult/815369/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.56 Project Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/project/`

Schema URL: `http://mytorrentserver/rundb/api/v1/project/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
creator	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
created	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
results	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
modified	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resultsCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
public	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/project/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/project/", params={"format": "jso
ts_api_response = ts_api_request.json()

projects = ts_api_response["objects"]
```

```
for project in projects:
    print project
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 1457,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/project/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "name": "3gb_snapps",
            "creator": "/rundb/api/v1/user/2/",
            "created": "2012-07-03T15:14:33.000509+00:00",
            "results": [
                "/rundb/api/v1/results/2305/",
                "/rundb/api/v1/results/1499/",
                "/rundb/api/v1/results/2304/",
                "/rundb/api/v1/results/1511/",
                "/rundb/api/v1/results/30467/",
                "/rundb/api/v1/results/30458/",
                "/rundb/api/v1/results/30457/",
                "/rundb/api/v1/results/30490/",
                "/rundb/api/v1/results/30464/",
                "/rundb/api/v1/results/30459/",
                "/rundb/api/v1/results/30460/",
                "/rundb/api/v1/results/30496/",
                "/rundb/api/v1/results/3298/",
                "/rundb/api/v1/results/30463/",
                "/rundb/api/v1/results/30446/",
                "/rundb/api/v1/results/30489/",
                "/rundb/api/v1/results/30466/",
                "/rundb/api/v1/results/30456/",
                "/rundb/api/v1/results/30491/",
                "/rundb/api/v1/results/2609/",
                "/rundb/api/v1/results/1466/",
                "/rundb/api/v1/results/1498/",
                "/rundb/api/v1/results/2300/",
                "/rundb/api/v1/results/1497/"
            ],
            "modified": "2012-07-03T15:14:33.000509+00:00",
            "id": 1,
            "resultsCount": 24,
            "public": true,
            "resource_uri": "/rundb/api/v1/project/1/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.57 Projectresults Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/projectresults/`

Schema URL: `http://mytorrentserver/rundb/api/v1/projectresults/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
reference	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
reportStatus	Unicode string data. Ex: "Hello World"	Nothing	true	false	false	false	string
runid	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
metaData	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
log	Unicode string data. Ex: "Hello World"		false	false	true	false	string
timeStamp	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
resultsName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
status	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
processedflows	Integer data. Ex: 2673	n/a	false	false	false	false	integer
processedCycles	Integer data. Ex: 2673	n/a	false	false	false	false	integer
sffLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
representative	Boolean data. Ex: True	false	false	false	true	false	boolean
tfSffLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
diskusage	Integer data. Ex: 2673	n/a	true	false	false	false	integer
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
resultsType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
tfFastq	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
parentIDs	Unicode string data. Ex: "Hello World"		false	false	true	false	string
timeToComplete	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
reportLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
fastqLink	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
framesProcessed	Integer data. Ex: 2673	n/a	false	false	false	false	integer
autoExempt	Boolean data. Ex: True	false	false	false	true	false	boolean
analysisVersion	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/projectresults/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/projectresults/", params={"format"
ts_api_response = ts_api_request.json()

projectresultss = ts_api_response["objects"]

for projectresults in projectresultss:
    print projectresults
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/projectresults/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "reference": "hg19",
            "reportStatus": "Nothing",
            "runid": "DGMU8",
            "id": 293943,
            "metaData": {},
            "log": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/log.htr
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "status": "Completed",
            "processedflows": 0,
            "processedCycles": 0,
            "sffLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R_2
            "representative": false,
            "tfSffLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R
            "diskusage": 151,
            "projects": [
                {
                    "name": "chef_827_909_20min_ext",
                    "creator": "/rundb/api/v1/user/1/",
                    "created": "2014-01-22T18:50:10.000920+00:00",
                    "results": [
                        "/rundb/api/v1/results/294145/",
                        "/rundb/api/v1/results/293993/",
                        "/rundb/api/v1/results/293992/",
                        "/rundb/api/v1/results/293991/",
                        "/rundb/api/v1/results/293990/",
                        "/rundb/api/v1/results/293946/",
```

```
                        "/rundb/api/v1/results/293945/",
                        "/rundb/api/v1/results/293944/",
                        "/rundb/api/v1/results/293943/",
                        "/rundb/api/v1/results/293934/",
                        "/rundb/api/v1/results/293933/",
                        "/rundb/api/v1/results/293930/",
                        "/rundb/api/v1/results/293929/",
                        "/rundb/api/v1/results/293928/",
                        "/rundb/api/v1/results/293927/",
                        "/rundb/api/v1/results/293917/",
                        "/rundb/api/v1/results/293916/",
                        "/rundb/api/v1/results/293915/",
                        "/rundb/api/v1/results/293914/",
                        "/rundb/api/v1/results/293913/",
                        "/rundb/api/v1/results/293912/",
                        "/rundb/api/v1/results/293908/",
                        "/rundb/api/v1/results/293907/",
                        "/rundb/api/v1/results/293906/",
                        "/rundb/api/v1/results/293905/",
                        "/rundb/api/v1/results/293904/",
                        "/rundb/api/v1/results/293903/",
                        "/rundb/api/v1/results/293902/",
                        "/rundb/api/v1/results/293901/"
                    ],
                    "modified": "2014-01-22T18:50:10.000920+00:00",
                    "id": 1080,
                    "resultsCount": 29,
                    "public": true,
                    "resource_uri": "/rundb/api/v1/project/1080/"
                }
            ],
            "resultsType": "",
            "tfFastq": "_",
            "parentIDs": "",
            "timeToComplete": "0",
            "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
            "fastqLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/k
            "resource_uri": "/rundb/api/v1/projectresults/293943/",
            "framesProcessed": 0,
            "autoExempt": false,
            "analysisVersion": "db:4.1.21+2-1,an:4.1.24+0-1,"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.58 Qctype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/qctype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/qctype/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
description	Unicode string data. Ex: “Hello World”		false	false	true	false	string
minThreshold	Integer data. Ex: 2673	0	false	false	false	false	integer
maxThreshold	Integer data. Ex: 2673	100	false	false	false	false	integer
defaultThreshold	Integer data. Ex: 2673	0	false	false	false	false	integer
qcName	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/qctype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/qctype/", params={"format": "json"
ts_api_response = ts_api_request.json()

qctypes = ts_api_response["objects"]

for qctype in qctypes:
    print qctype
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 3,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/qctype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "description": "",
            "minThreshold": 0,
            "maxThreshold": 100,
            "defaultThreshold": 30,
            "qcName": "Bead Loading (%)",
            "id": 1,
            "resource_uri": "/rundb/api/v1/qctype/1/"
```

```
            }
        ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.59 Qualitymetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/schema/`

Perform read-only operations on `qualitymetrics` resources and data elements.

Fields table

field	help text	default	nullable	
q0_reads	Integer data. Ex: 2673	n/a	false	
q17_max_read_length	Integer data. Ex: 2673	n/a	false	
q20_median_read_length	Integer data. Ex: 2673	0	false	
q20_reads	Integer data. Ex: 2673	n/a	false	
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	
q17_mean_read_length	Floating point numeric data. Ex: 26.73	n/a	false	
q17_100bp_reads	Integer data. Ex: 2673	n/a	false	
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	
q0_max_read_length	Integer data. Ex: 2673	n/a	false	
q20_100bp_reads	Integer data. Ex: 2673	n/a	false	
id	Integer data. Ex: 2673		false	
q20_mean_read_length	Integer data. Ex: 2673	n/a	false	
q20_150bp_reads	Integer data. Ex: 2673	n/a	false	
q0_bases	Unicode string data. Ex: “Hello World”	n/a	false	
q20_50bp_reads	Integer data. Ex: 2673	n/a	false	
q17_reads	Integer data. Ex: 2673	n/a	false	
q17_50bp_reads	Integer data. Ex: 2673	n/a	false	
q17_median_read_length	Integer data. Ex: 2673	0	false	
q0_50bp_reads	Integer data. Ex: 2673	n/a	false	
q17_150bp_reads	Integer data. Ex: 2673	n/a	false	
q0_150bp_reads	Integer data. Ex: 2673	0	false	
q0_mean_read_length	Floating point numeric data. Ex: 26.73	n/a	false	
q17_bases	Unicode string data. Ex: “Hello World”	n/a	false	
q0_mode_read_length	Integer data. Ex: 2673	0	false	
q20_mode_read_length	Integer data. Ex: 2673	0	false	

Table 4.21 – continued from previous page

field	help text	default	nullable	
q20_max_read_length	Floating point numeric data. Ex: 26.73	n/a	false	f
q20_bases	Unicode string data. Ex: “Hello World”	n/a	false	f
q0_median_read_length	Integer data. Ex: 2673	0	false	f
q0_100bp_reads	Integer data. Ex: 2673	n/a	false	f
q17_mode_read_length	Integer data. Ex: 2673	0	false	f

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/qualitymetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/qualitymetrics/", params={"format"
ts_api_response = ts_api_request.json()

qualitymetricss = ts_api_response["objects"]

for qualitymetrics in qualitymetricss:
    print qualitymetrics
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 37351,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/qualitymetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "q0_reads": 0,
            "q17_max_read_length": 0,
            "q20_median_read_length": 0,
            "q20_reads": 0,
            "report": "/rundb/api/v1/results/269819/",
            "q17_mean_read_length": 0.0,
            "q17_100bp_reads": 0,
            "resource_uri": "/rundb/api/v1/qualitymetrics/9943/",
            "q0_max_read_length": 0,
            "q20_100bp_reads": 0,
            "id": 9943,
            "q20_mean_read_length": 0,
            "q20_150bp_reads": 0,
            "q0_bases": "0",
            "q20_50bp_reads": 0,
            "q17_reads": 0,
            "q17_50bp_reads": 0,
```

```
            "q17_median_read_length": 0,
            "q0_50bp_reads": 0,
            "q17_150bp_reads": 0,
            "q0_150bp_reads": 0,
            "q0_mean_read_length": 0.0,
            "q17_bases": "0",
            "q0_mode_read_length": 0,
            "q20_mode_read_length": 0,
            "q20_max_read_length": 0.0,
            "q20_bases": "0",
            "q0_median_read_length": 0,
            "q0_100bp_reads": 0,
            "q17_mode_read_length": 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.60 Referencegenome Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/referencegenome/`

Schema URL: `http://mytorrentserver/rundb/api/v1/referencegenome/schema/`

Perform read-only operations on `referencegenome` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: "Hello World"		false	false	true	false	string
reference_path	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
short_name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
index_version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
notes	Unicode string data. Ex: "Hello World"		false	false	true	false	string
enabled	Boolean data. Ex: True	true	false	false	true	false	boolean
species	Unicode string data. Ex: "Hello World"		false	false	true	false	string
identity_hash	Unicode string data. Ex: "Hello World"	None	true	false	false	false	string
source	Unicode string data. Ex: "Hello World"		false	false	true	false	string
version	Unicode string data. Ex: "Hello World"		false	false	true	false	string
celery_task_id	Unicode string data. Ex: "Hello World"		false	false	true	false	string
date	A date & time as a string. Ex: "2010-11-10T03:07:43"	2014-06-28T14:05:04.000158+00:00	false	false	false	false	datetime
verbose_error	Unicode string data. Ex: "Hello World"		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/referencegenome/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/referencegenome/", params={"format
ts_api_response = ts_api_request.json()

referencegenomes = ts_api_response["objects"]

for referencegenome in referencegenomes:
    print referencegenome
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/referencegenome/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "status": "complete",
            "reference_path": "/results/referenceLibrary/tmap-f3/CFTR_38amp_v2",
            "name": "CFTR_38amp_v2",
            "short_name": "CFTR_38amp_v2",
            "index_version": "tmap-f3",
            "notes": "",
            "enabled": true,
            "species": "",
            "identity_hash": null,
            "source": "",
            "version": "CFTR_38amp_v2",
            "celery_task_id": "",
            "date": "2012-03-21T12:32:00.000382+00:00",
            "verbose_error": "[\"\", \"\\nSequence name 'CFTR.13.120s' contains a non-alphanumeric ch
            "id": 7,
            "resource_uri": "/rundb/api/v1/referencegenome/7/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.61 Results Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/results/`

Schema URL: `http://mytorrentserver/rundb/api/v1/results/schema/`

Perform CRUD operations on `results` resources and data elements.

Fields table

field	help text	default	nul
reference	Unicode string data. Ex: “Hello World”	n/a	true
processedflows	Integer data. Ex: 2673	n/a	fals
reportStatus	Unicode string data. Ex: “Hello World”	Nothing	true
reportstorage	A single related resource. Can be either a URI or set of nested resource data.	n/a	fals
analysisVersion	Unicode string data. Ex: “Hello World”	n/a	fals
runid	Unicode string data. Ex: “Hello World”		fals
id	Integer data. Ex: 2673		fals
filesystempath	Unicode string data. Ex: “Hello World”	n/a	fals
metaData	Unicode string data. Ex: “Hello World”	{}	fals
log	Unicode string data. Ex: “Hello World”		fals
timeStamp	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	fals
libmetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
experiment	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
resultsName	Unicode string data. Ex: “Hello World”	n/a	fals
status	Unicode string data. Ex: “Hello World”	n/a	fals
planShortID	Unicode string data. Ex: “Hello World”	n/a	fals
processedCycles	Integer data. Ex: 2673	n/a	fals
bamLink	Unicode string data. Ex: “Hello World”	n/a	fals
sffLink	Unicode string data. Ex: “Hello World”	n/a	true
representative	Boolean data. Ex: True	false	fals
pluginState	A dictionary of data. Ex: {‘price’: 26.73, ‘name’: ‘Daniel’}	n/a	fals
qualitymetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
diskusage	Integer data. Ex: 2673	n/a	true
eas	A single related resource. Can be either a URI or set of nested resource data.	n/a	true
tfSffLink	Unicode string data. Ex: “Hello World”	n/a	true
projects	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
pluginStore	A dictionary of data. Ex: {‘price’: 26.73, ‘name’: ‘Daniel’}	n/a	fals
resultsType	Unicode string data. Ex: “Hello World”		fals
tfFastq	Unicode string data. Ex: “Hello World”	n/a	fals
tfmetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
parentIDs	Unicode string data. Ex: “Hello World”		fals
analysismetrics	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
timeToComplete	Unicode string data. Ex: “Hello World”	n/a	fals
reportLink	Unicode string data. Ex: “Hello World”	n/a	fals
fastqLink	Unicode string data. Ex: “Hello World”	n/a	fals
pluginresults	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	fals
framesProcessed	Integer data. Ex: 2673	n/a	fals
autoExempt	Boolean data. Ex: True	false	fals
resource_uri	Unicode string data. Ex: “Hello World”	n/a	fals

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/results/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/results/", params={"format": "jso
ts_api_response = ts_api_request.json()
```

```
resultss = ts_api_response["objects"]

for results in resultss:
    print results
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 43354,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/results/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "reference": "hg19",
            "processedflows": 0,
            "reportStatus": "Nothing",
            "reportstorage": {
                "name": "Home",
                "default": true,
                "webServerPath": "/output",
                "dirPath": "/results/analysis/output",
                "id": 1,
                "resource_uri": ""
            },
            "analysisVersion": "db:4.1.21+2-1,an:4.1.24+0-1,",
            "runid": "DGMU8",
            "id": 293943,
            "filesystempath": "/results/analysis/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m
            "metaData": {},
            "log": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/log.htm
            "timeStamp": "2014-01-23T07:39:52.000803+00:00",
            "libmetrics": [
                "/rundb/api/v1/libmetrics/32368/"
            ],
            "experiment": "/rundb/api/v1/experiment/17446/",
            "resultsName": "Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446",
            "status": "Completed",
            "planShortID": "ONPK8",
            "processedCycles": 0,
            "bamLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/R_2
            "sffLink": null,
            "representative": false,
            "pluginState": {
                "coverageAnalysisLite": "Completed",
                "SystematicErrorAnalysis": "Completed",
                "1_Torrent_Accuracy": "Error",
                "duplicateReads_useZC": "Completed",
                "autoCal": "Completed",
                "variantCaller": "Completed",
                "ConversionRate": "Completed",
                "validateVariantCaller": "Completed",
                "timingPerformance": "Completed",
```

```
        "coverageAnalysis": "Completed",
        "VariantQC": "Completed"
    },
    "qualitymetrics": [
        "/rundb/api/v1/qualitymetrics/31678/"
    ],
    "diskusage": 151,
    "eas": "/rundb/api/v1/experimentanalysissettings/18714/",
    "tfSffLink": null,
    "projects": [
        "/rundb/api/v1/project/1080/"
    ],
    "pluginStore": {
        "coverageAnalysisLite": {
            "Non-duplicate": "",
            "barcoded": "true",
            "Uniquely mapped": "No",
            "Targetted regions": "/results/uploads/BED/46/hg19/merged/plain/AmpliSeqExome.201
            "Target padding": "0",
            "barcodes": {
                "IonXpress_033": {
                    "Bases in target regions": "57742646",
                    "Number of mapped reads": "41517304",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Percent reads on target": "94.39%",
                    "Average base coverage depth": "112.4",
                    "Reference (File)": "hg19",
                    "Coverage Analysis Lite Report": "N/A",
                    "Target base coverage at 100x": "51.05%",
                    "Target base coverage at 20x": "94.26%",
                    "Uniformity of base coverage": "93.56%",
                    "Target base coverage at 1x": "98.53%",
                    "Using": "All Mapped Reads",
                    "Target base coverage at 500x": "0.10%",
                    "Alignments": "IonXpress_033_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Total base reads on target": "6490060189"
                },
                "IonXpress_034": {
                    "Bases in target regions": "57742646",
                    "Number of mapped reads": "48026110",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Percent reads on target": "94.01%",
                    "Average base coverage depth": "130.8",
                    "Reference (File)": "hg19",
                    "Coverage Analysis Lite Report": "N/A",
                    "Target base coverage at 100x": "61.01%",
                    "Target base coverage at 20x": "94.93%",
                    "Uniformity of base coverage": "93.55%",
                    "Target base coverage at 1x": "98.55%",
                    "Using": "All Mapped Reads",
                    "Target base coverage at 500x": "0.22%",
                    "Alignments": "IonXpress_034_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Total base reads on target": "7552568443"
                }
            }
        },
        "SystematicErrorAnalysis": {
            "qts_peak": "NaN",
```

```
        "indel-5-per-mb": "10626.700000000001",
        "positions-with-sse": "0.0080719999999999993",
        "qts_base": "NaN",
        "barcoded": "true",
        "positions-with-sse-d15": "0.0080719999999999993",
        "Target-regions_file": "/results/uploads/BED/46/hg19/merged/plain/AmpliSeqExome.2
        "stb-95-per-mb": "67945.399999999994",
        "positions-with-sse-i5": "0.0073000000000000001",
        "positions-with-sse-d5": "0.023935000000000001",
        "positions-with-sse-i15": "0.0018799999999999999",
        "indel-15-per-mb": "3118.8000000000002",
        "barcodes": {
            "IonXpress_033": {
                "indel-5-per-mb": "10600.3",
                "positions-with-sse": "0.007770",
                "positions-with-sse-d15": "0.007770",
                "stb-95-per-mb": "68851.0",
                "positions-with-sse-i5": "0.007519",
                "positions-with-sse-d5": "0.022925",
                "positions-with-sse-i15": "0.001964",
                "indel-15-per-mb": "3126.1"
            },
            "IonXpress_034": {
                "indel-5-per-mb": "10653.1",
                "positions-with-sse": "0.008373",
                "positions-with-sse-d15": "0.008373",
                "stb-95-per-mb": "67039.8",
                "positions-with-sse-i5": "0.007080",
                "positions-with-sse-d5": "0.024945",
                "positions-with-sse-i15": "0.001796",
                "indel-15-per-mb": "3111.5"
            }
        }
    },
    "1_Torrent_Accuracy": {},
    "duplicateReads_useZC": {
        "adapter_found_rate_chr1": "0.89400000000000002",
        "duplicate_reads_chr1": 673650,
        "duprate_at_725k_chr1": "0.89300000000000002",
        "duplicate_rate_chr1": "0.89200000000000002",
        "total_reads_chr1": 754826
    },
    "autoCal": {
        "dc_range": 0
    },
    "variantCaller": {
        "barcodes": {
            "IonXpress_033": {
                "hotspots": {},
                "variants": {
                    "no_call": 0,
                    "homo_snps": 18047,
                    "het_snps": 31409,
                    "other": 1321,
                    "variants": 54343,
                    "het_indels": 2444,
                    "homo_indels": 1122
                }
```

```
            },
            "IonXpress_034": {
                "hotspots": {},
                "variants": {
                    "no_call": 0,
                    "homo_snps": 18134,
                    "het_snps": 31524,
                    "other": 1308,
                    "variants": 54522,
                    "het_indels": 2422,
                    "homo_indels": 1134
                }
            }
        },
        "barcoded": "true",
        "targets_bed": "/results/uploads/BED/46/hg19/unmerged/detail/AmpliSeqExome.20131
        "Target Regions": "AmpliSeqExome.20131001.designed",
        "Trim Reads": true,
        "Target Loci": "Not using",
        "Configuration": "Germ Line - Proton - Low Stringency",
        "Aligned Reads": "R_2014_01_22_16_30_23_user_D1--632--R54651-p8s2_827b2_20m_man-
        "Library Type": "AmpliSeq"
    },
    "ConversionRate": {},
    "validateVariantCaller": {
        "SNP_FP-ConfidentPos": 5410,
        "InDel_AmbPos-AllPos": 0,
        "SNP_PPV>=30x": "96.7228784857032",
        "InDel_FN>=100x": 930,
        "SNP_FN>=100x": 896,
        "InDel_TP-AllPos": 2591,
        "InDel_FP>=30x": 1689,
        "SNP_FN>=30x": 2004,
        "SNP_FP>=1000x": 0,
        "SNP_ConsensusAccuracy-AllPos": "0.999914181450164",
        "InDel_NoCalls-AllPos": 894775,
        "InDel_FP-AllPos": 2681,
        "InDel_FP_50x-100x": 695,
        "InDel_Sensitivity>=20x": "47.7212806026365",
        "SNP_TP>=500x": 132,
        "SNP_Sensitivity>=100x": "98.2144280589877",
        "SNP_Sensitivity>=500x": "99.2481203007519",
        "InDel_ConsensusAccuracy-AllPos": "0.999949885886992",
        "SNP_ConsensusAccuracy>=50x": "0.999971537775062",
        "InDel_FN>=20x": 2776,
        "SNP_FP>=50x": 1762,
        "InDel_PPV-AllPos": "49.1464339908953",
        "InDel_ConsensusAccuracy>=30x": "0.999964918459836",
        "InDel_FP-ncRNA": 18,
        "InDel_ReferenceCalls-AllPos": 0,
        "Target-regions_file": "/results/analysis/output/Home/Auto_user_D1--632--R54651-p
        "SNP_TP-AllPos": 87058,
        "InDel_Sensitivity>=30x": "49.0349819059107",
        "InDel_FN>=500x": 3,
        "InDel_FP_20x-50x": 827,
        "InDel_FP-ConfidentPos": 2681,
        "InDel_FN-AllPos": 3353,
        "SNP_HP11-AllPos": 0,
```

```
"InDel_FN>=50x": 2041,
"SNP_Sensitivity-AllPos": "94.6478076994162",
"Truth-major_SNP_file": "NA12878_NIST_NoChrY_SNP.bed",
"InDel_Sensitivity-AllAnnotations": "87.1800314804519",
"SNP_FP_200x-300x": 84,
"InDel_FP>=20x": 2020,
"SNP_FN>=500x": 1,
"Combined Variant Positive Predictive Value for All Bases at >=20x": "94.0001",
"SNP_Sensitivity>=1000x": 100,
"InDel_FP_700x-1000x": 0,
"SNP_FP>=30x": 2777,
"SNP_FP_700x-1000x": 0,
"SNP_ConsensusAccuracy>=20x": "0.999952352557784",
"InDel_TP>=30x": 2439,
"SNP_AmbPos-AllPos": 0,
"Truth-minor_InDel_file": "None",
"SNP_FP_500x-700x": 1,
"SNP_FP_50x-100x": 1095,
"InDel_ConsensusAccuracy>=20x": "0.999960167834606",
"InDel_FP-Exons": 1062,
"SNP_TP>=30x": 81962,
"SNP_ReferenceCalls-AllPos": 0,
"InDel_FP_500x-700x": 1,
"InDel_FP_200x-300x": 65,
"barcoded": "true",
"SNP_FP_20x-50x": 1773,
"SNP_FP>=100x": 667,
"SNP_Sensitivity>=20x": "97.4601197273262",
"InDel_FP_300x-400x": 9,
"SNP_FP_100x-200x": 565,
"barcodes": {
    "IonXpress_033": {
        "SNP_FP-ConfidentPos": "2733",
        "InDel_AmbPos-AllPos": "0",
        "SNP_PPV>=30x": "96.7970258096804",
        "InDel_FN>=100x": "411",
        "SNP_FN>=100x": "406",
        "InDel_TP-AllPos": "1281",
        "InDel_FP>=30x": "838",
        "SNP_FN>=30x": "1006",
        "SNP_FP>=1000x": "0",
        "SNP_ConsensusAccuracy-AllPos": "0.999912578529413",
        "InDel_NoCalls-AllPos": "450359",
        "InDel_FP-AllPos": "1375",
        "InDel_FP_50x-100x": "369",
        "InDel_Sensitivity>=20x": "47.4402730375427",
        "SNP_TP>=500x": "44",
        "SNP_Sensitivity>=100x": "98.2229614391386",
        "SNP_Sensitivity>=500x": "100",
        "InDel_ConsensusAccuracy-AllPos": "0.999949088579261",
        "SNP_ConsensusAccuracy>=50x": "0.999972775263102",
        "InDel_FN>=20x": "1386",
        "SNP_FP>=50x": "829",
        "InDel_PPV-AllPos": "48.230421686747",
        "InDel_ConsensusAccuracy>=30x": "0.99996518422908",
        "InDel_FP-ncRNA": "9",
        "InDel_ReferenceCalls-AllPos": "0",
        "SNP_TP-AllPos": "43461",
```

```
"InDel_Sensitivity>=30x": "48.7158581328985",
"InDel_FN>=500x": "1",
"InDel_FP_20x-50x": "437",
"InDel_FP-ConfidentPos": "1375",
"InDel_FN-AllPos": "1690",
"SNP_HP11-AllPos": "0",
"InDel_FN>=50x": "977",
"SNP_Sensitivity-AllPos": "94.4989237024635",
"InDel_Sensitivity-AllAnnotations": "43.1167956916863",
"SNP_FP_200x-300x": "34",
"InDel_FP>=20x": "1013",
"SNP_FN>=500x": "0",
"Combined Variant Positive Predictive Value for All Bases at >=20x": "94
"SNP_Sensitivity>=1000x": "0",
"InDel_FP_700x-1000x": "0",
"SNP_FP>=30x": "1344",
"SNP_FP_700x-1000x": "0",
"SNP_ConsensusAccuracy>=20x": "0.999952742906361",
"InDel_TP>=30x": "1195",
"SNP_AmbPos-AllPos": "0",
"SNP_FP_500x-700x": "0",
"SNP_FP_50x-100x": "536",
"InDel_ConsensusAccuracy>=20x": "0.999960151224028",
"InDel_FP-Exons": "537",
"SNP_TP>=30x": "40617",
"SNP_ReferenceCalls-AllPos": "0",
"InDel_FP_500x-700x": "0",
"InDel_FP_200x-300x": "29",
"SNP_FP_20x-50x": "911",
"SNP_FP>=100x": "293",
"SNP_Sensitivity>=20x": "97.439640391121",
"InDel_FP_300x-400x": "0",
"SNP_FP_100x-200x": "255",
"SNP_Sensitivity-AllAnnotations": "94.4989237024635",
"InDel_FP_400x-500x": "2",
"InDel_FP>=50x": "576",
"SNP_FN-AllAnnotations": "2530",
"SNP_FP-AllPos": "2733",
"SNP_PPV-AllPos": "94.0836472269126",
"SNP_FP>=20x": "1740",
"InDel_Sensitivity>=1000x": "0",
"InDel_ConsensusAccuracy>=50x": "0.999974203772787",
"InDel_TP>=100x": "616",
"SNP_TP>=50x": "36440",
"SNP_ConsensusAccuracy>=30x": "0.999960965142337",
"InDel_Sensitivity-AllPos": "43.1167956916863",
"InDel_TP>=20x": "1251",
"InDel_AmbNotDetected-AllPos": "0",
"SNP_FP>=500x": "0",
"% Callable Bases": "99.161500",
"SNP_FN>=20x": "1105",
"InDel_FP>=1000x": "0",
"SNP_Sensitivity>=30x": "97.5830670542729",
"InDel_FN>=1000x": "0",
"InDel_FP>=500x": "0",
"SNP_FP_400x-500x": "2",
"InDel_HP11-AllPos": "0",
"Combined Variant Positive Predictive Value for All Bases at all coverage
```

```
            "SNP_FN>=1000x": "0",
            "SNP_FN>=50x": "810",
            "InDel_TP>=500x": "2",
            "InDel_PPV>=30x": "58.780127889818",
            "InDel_TP>=1000x": "0",
            "SNP_FP_300x-400x": "2",
            "SNP_Sensitivity>=50x": "97.8255033557047",
            "Combined Variant Sensitivity for All Bases at >= 20x": "94.560500",
            "SNP_TP>=100x": "22441",
            "InDel_FP>=100x": "207",
            "SNP_NoCalls-AllPos": "56910",
            "InDel_Sensitivity>=500x": "66.6666666666667",
            "SNP_TP>=20x": "42053",
            "InDel_Sensitivity>=50x": "52.3182040019522",
            "InDel_FN-AllAnnotations": "1690",
            "InDel_TP>=50x": "1072",
            "SNP_TP>=1000x": "0",
            "InDel_FN>=30x": "1258",
            "SNP_AmbNotDetected-AllPos": "0",
            "SNP_FP-ncRNA": "28",
            "SNP_FN-AllPos": "2530",
            "InDel_Sensitivity>=100x": "59.9805258033106",
            "InDel_FP_100x-200x": "176",
            "SNP_FP-Exons": "993",
            "Combined Variant Sensitivity for All Bases at all coverages": "91.381000
        },
        "IonXpress_034": {
            "SNP_FP-ConfidentPos": "2677",
            "InDel_AmbPos-AllPos": "0",
            "SNP_PPV>=30x": "96.6501472719622",
            "InDel_FN>=100x": "519",
            "SNP_FN>=100x": "490",
            "InDel_TP-AllPos": "1310",
            "InDel_FP>=30x": "851",
            "SNP_FN>=30x": "998",
            "SNP_FP>=1000x": "0",
            "SNP_ConsensusAccuracy-AllPos": "0.999915784370915",
            "InDel_NoCalls-AllPos": "444416",
            "InDel_FP-AllPos": "1306",
            "InDel_FP_50x-100x": "326",
            "InDel_Sensitivity>=20x": "47.9985035540591",
            "SNP_TP>=500x": "88",
            "SNP_Sensitivity>=100x": "98.2072952109172",
            "SNP_Sensitivity>=500x": "98.876404494382",
            "InDel_ConsensusAccuracy-AllPos": "0.999950683194723",
            "SNP_ConsensusAccuracy>=50x": "0.999970300287021",
            "InDel_FN>=20x": "1390",
            "SNP_FP>=50x": "933",
            "InDel_PPV-AllPos": "50.0764525993884",
            "InDel_ConsensusAccuracy>=30x": "0.999964652690593",
            "InDel_FP-ncRNA": "9",
            "InDel_ReferenceCalls-AllPos": "0",
            "SNP_TP-AllPos": "43597",
            "InDel_Sensitivity>=30x": "49.3454978183261",
            "InDel_FN>=500x": "2",
            "InDel_FP_20x-50x": "390",
            "InDel_FP-ConfidentPos": "1306",
            "InDel_FN-AllPos": "1663",
```

```
"SNP_HP11-AllPos": "0",
"InDel_FN>=50x": "1064",
"SNP_Sensitivity-AllPos": "94.7966949336812",
"InDel_Sensitivity-AllAnnotations": "44.0632357887656",
"SNP_FP_200x-300x": "50",
"InDel_FP>=20x": "1007",
"SNP_FN>=500x": "1",
"Combined Variant Positive Predictive Value for All Bases at >=20x": "93
"SNP_Sensitivity>=1000x": "100",
"InDel_FP_700x-1000x": "0",
"SNP_FP>=30x": "1433",
"SNP_FP_700x-1000x": "0",
"SNP_ConsensusAccuracy>=20x": "0.999951962209208",
"InDel_TP>=30x": "1244",
"SNP_AmbPos-AllPos": "0",
"SNP_FP_500x-700x": "1",
"SNP_FP_50x-100x": "559",
"InDel_ConsensusAccuracy>=20x": "0.999960184445184",
"InDel_FP-Exons": "525",
"SNP_TP>=30x": "41345",
"SNP_ReferenceCalls-AllPos": "0",
"InDel_FP_500x-700x": "1",
"InDel_FP_200x-300x": "36",
"SNP_FP_20x-50x": "862",
"SNP_FP>=100x": "374",
"SNP_Sensitivity>=20x": "97.4804198534647",
"InDel_FP_300x-400x": "9",
"SNP_FP_100x-200x": "310",
"SNP_Sensitivity-AllAnnotations": "94.7966949336812",
"InDel_FP_400x-500x": "0",
"InDel_FP>=50x": "617",
"SNP_FN-AllAnnotations": "2393",
"SNP_FP-AllPos": "2677",
"SNP_PPV-AllPos": "94.2148938928988",
"SNP_FP>=20x": "1795",
"InDel_Sensitivity>=1000x": "0",
"InDel_ConsensusAccuracy>=50x": "0.999972077618838",
"InDel_TP>=100x": "771",
"SNP_TP>=50x": "38295",
"SNP_ConsensusAccuracy>=30x": "0.999959619685541",
"InDel_Sensitivity-AllPos": "44.0632357887656",
"InDel_TP>=20x": "1283",
"InDel_AmbNotDetected-AllPos": "0",
"SNP_FP>=500x": "1",
"% Callable Bases": "99.176600",
"SNP_FN>=20x": "1097",
"InDel_FP>=1000x": "0",
"SNP_Sensitivity>=30x": "97.6430578844201",
"InDel_FN>=1000x": "0",
"InDel_FP>=500x": "1",
"SNP_FP_400x-500x": "2",
"InDel_HP11-AllPos": "0",
"Combined Variant Positive Predictive Value for All Bases at all coverage
"SNP_FN>=1000x": "0",
"SNP_FN>=50x": "855",
"InDel_TP>=500x": "2",
"InDel_PPV>=30x": "59.3794749403341",
"InDel_TP>=1000x": "0",
```

```
            "SNP_FP_300x-400x": "11",
            "SNP_Sensitivity>=50x": "97.816091954023",
            "Combined Variant Sensitivity for All Bases at >= 20x": "94.618200",
            "SNP_TP>=100x": "26843",
            "InDel_FP>=100x": "291",
            "SNP_NoCalls-AllPos": "53768",
            "InDel_Sensitivity>=500x": "50",
            "SNP_TP>=20x": "42442",
            "InDel_Sensitivity>=50x": "51.5041020966272",
            "InDel_FN-AllAnnotations": "1663",
            "InDel_TP>=50x": "1130",
            "SNP_TP>=1000x": "3",
            "InDel_FN>=30x": "1277",
            "SNP_AmbNotDetected-AllPos": "0",
            "SNP_FP-ncRNA": "32",
            "SNP_FN-AllPos": "2393",
            "InDel_Sensitivity>=100x": "59.7674418604651",
            "InDel_FP_100x-200x": "245",
            "SNP_FP-Exons": "958",
            "Combined Variant Sensitivity for All Bases at all coverages": "91.71610(
        }
    },
    "SNP_Sensitivity-AllAnnotations": "189.295618636145",
    "InDel_FP_400x-500x": 2,
    "InDel_FP>=50x": 1193,
    "SNP_FN-AllAnnotations": 4923,
    "SNP_FP-AllPos": 5410,
    "SNP_PPV-AllPos": "94.1493273348618",
    "SNP_FP>=20x": 3535,
    "InDel_Sensitivity>=1000x": 0,
    "InDel_ConsensusAccuracy>=50x": "0.999973140695813",
    "InDel_TP>=100x": 1387,
    "Region_selected": "NIST",
    "SNP_ConsensusAccuracy>=30x": "0.999960292413939",
    "InDel_Sensitivity-AllPos": "43.5901749663526",
    "InDel_TP>=20x": 2534,
    "InDel_AmbNotDetected-AllPos": 0,
    "SNP_FP>=500x": 1,
    "% Callable Bases": "99.16905",
    "SNP_FN>=20x": 2202,
    "InDel_FP>=1000x": 0,
    "SNP_Sensitivity>=30x": "97.6133196770121",
    "InDel_FN>=1000x": 0,
    "InDel_FP>=500x": 1,
    "SNP_FP_400x-500x": 4,
    "InDel_HP11-AllPos": 0,
    "Combined Variant Positive Predictive Value for All Bases at all coverages": "91
    "SNP_FN>=1000x": 0,
    "SNP_FN>=50x": 1665,
    "InDel_TP>=500x": 4,
    "InDel_PPV>=30x": "59.0843023255814",
    "InDel_TP>=1000x": 0,
    "Truth-minor_SNP_file": "None",
    "SNP_FP_300x-400x": 13,
    "SNP_Sensitivity>=50x": "97.8206806282722",
    "Combined Variant Sensitivity for All Bases at >= 20x": "94.58935",
    "SNP_TP>=100x": 49284,
    "Sample_selected": "NA12878",
```

```
            "InDel_FP>=100x": 498,
            "SNP_NoCalls-AllPos": 110678,
            "Truth-major_InDel_file": "NA12878_NIST_NoChrY_indel.bed",
            "InDel_Sensitivity>=500x": "57.1428571428571",
            "SNP_TP>=20x": 84495,
            "InDel_Sensitivity>=50x": "51.897242517087",
            "InDel_FN-AllAnnotations": 3353,
            "InDel_TP>=50x": 2202,
            "SNP_TP>=1000x": 3,
            "InDel_FN>=30x": 2535,
            "Variant-caller_name": "variantCaller",
            "SNP_AmbNotDetected-AllPos": 0,
            "SNP_TP>=50x": 74735,
            "SNP_FP-ncRNA": 60,
            "SNP_FN-AllPos": 4923,
            "InDel_Sensitivity>=100x": "59.8618903754855",
            "InDel_FP_100x-200x": 421,
            "SNP_FP-Exons": 1951,
            "Combined Variant Sensitivity for All Bases at all coverages": "91.54855"
        },
        "timingPerformance": {
            "runtime": {
                "analysis": "334.17000000000002"
            },
            "threadinfo": {
                "bkgmodel Gpu": 1,
                "fileaccess": 4,
                "beadfind": 6,
                "basecalling": 24,
                "bkgmodel Cpu": 6
            },
            "chipinfo": {
                "oia": 1,
                "flows": 500,
                "chiptype": "900"
            }
        },
        "coverageAnalysis": {
            "Non-duplicate": "No",
            "barcoded": "true",
            "Uniquely mapped": "No",
            "Amplicons reading end-to-end": "26.72%",
            "Targetted regions": "/results/uploads/BED/46/hg19/merged/detail/AmpliSeqExome.2(
            "Target padding": "0",
            "barcodes": {
                "IonXpress_033": {
                    "Bases in target regions": "57742646",
                    "Amplicons with at least 1 read": "99.21%",
                    "Target base coverage at 100x": "51.05%",
                    "Amplicons with at least 500 reads": "0.13%",
                    "Total assigned amplicon reads": "39187438",
                    "Reference (File)": "hg19",
                    "Total base reads on target": "6490060189",
                    "Target base coverage at 20x": "94.26%",
                    "Number of amplicons": "293903",
                    "Target bases with no strand bias": "76.79%",
                    "Percent reads on target": "94.39%",
                    "Amplicons with at least 100 reads": "64.34%",
```

```
                    "Average base coverage depth": "112.4",
                    "Average reads per amplicon": "133.3",
                    "Using": "All Mapped Reads",
                    "Amplicons reading end-to-end": "25.70%",
                    "Sample Name": "None",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Uniformity of base coverage": "93.56%",
                    "Alignments": "IonXpress_033_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Amplicons with at least 20 reads": "95.84%",
                    "Number of mapped reads": "41517304",
                    "Percent assigned amplicon reads": "94.39%",
                    "Amplicons with no strand bias": "92.84%",
                    "Total aligned base reads": "6846723653",
                    "Target base coverage at 1x": "98.53%",
                    "Target base coverage at 500x": "0.10%",
                    "Percent base reads on target": "94.79%",
                    "Uniformity of amplicon coverage": "94.62%"
                },
                "IonXpress_034": {
                    "Bases in target regions": "57742646",
                    "Amplicons with at least 1 read": "99.24%",
                    "Target base coverage at 100x": "61.01%",
                    "Amplicons with at least 500 reads": "0.24%",
                    "Total assigned amplicon reads": "45147738",
                    "Reference (File)": "hg19",
                    "Total base reads on target": "7552568443",
                    "Target base coverage at 20x": "94.93%",
                    "Number of amplicons": "293903",
                    "Target bases with no strand bias": "77.82%",
                    "Percent reads on target": "94.01%",
                    "Amplicons with at least 100 reads": "72.27%",
                    "Average base coverage depth": "130.8",
                    "Average reads per amplicon": "153.6",
                    "Using": "All Mapped Reads",
                    "Amplicons reading end-to-end": "27.74%",
                    "Sample Name": "None",
                    "Targeted Regions": "AmpliSeqExome.20131001.designed",
                    "Uniformity of base coverage": "93.55%",
                    "Alignments": "IonXpress_034_R_2014_01_22_16_30_23_user_D1--632--R54651-p
                    "Amplicons with at least 20 reads": "96.17%",
                    "Number of mapped reads": "48026110",
                    "Percent assigned amplicon reads": "94.01%",
                    "Amplicons with no strand bias": "92.98%",
                    "Total aligned base reads": "8002544816",
                    "Target base coverage at 1x": "98.55%",
                    "Target base coverage at 500x": "0.22%",
                    "Percent base reads on target": "94.38%",
                    "Uniformity of amplicon coverage": "94.48%"
                }
            }
        },
        "VariantQC": {
            "IonXpress_033": {
                "reason": {
                    "filtered": {
                        "HPLEN": 6386,
                        "REJECTION": 610,
                        "Cov": 72787,
```

```
                "REF": 472111,
                "HEALED": 348,
                "SHIFT": 458087,
                "SSE": 78820,
                "STRINGENCY": 1500,
                ".": 538589,
                "Quality": 103314,
                "STDBIAS": 12975
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
                "Cov": 0,
                "REF": 0,
                "HEALED": 1036,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54299,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
            "filtered": {
                "11": 432,
                "10": 972,
                "13": 178,
                "12": 193,
                "15": 160,
                "14": 143,
                "1": 74456,
                "0": 57194,
                "3": 92797,
                "2": 124391,
                "5": 52748,
                "4": 62684,
                "7": 11651,
                "6": 25769,
                "9": 2199,
                "8": 4196
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
```

```
                "8": 8
            },
            "unfiltered": {
                "11": 43,
                "10": 45,
                "13": 20,
                "12": 38,
                "15": 27,
                "14": 27,
                "1": 29948,
                "0": 561,
                "3": 5112,
                "2": 12151,
                "5": 1238,
                "4": 2386,
                "7": 248,
                "6": 567,
                "9": 73,
                "8": 145
            }
        },
        "type": {
            "filtered": {
                "other": 37034,
                "del": 278917,
                "snp": 37539,
                "ins": 185099
            },
            "unfiltered": {
                "other": 2479,
                "del": 1093,
                "snp": 49277,
                "ins": 1450
            }
        },
        "basic": {
            "filtered": 538589,
            "unfiltered": 54299
        }
    },
    "IonXpress_034": {
        "reason": {
            "filtered": {
                "HPLEN": 6511,
                "REJECTION": 545,
                "Cov": 65871,
                "REF": 462781,
                "HEALED": 355,
                "SHIFT": 454012,
                "SSE": 83658,
                "STRINGENCY": 1396,
                ".": 529663,
                "Quality": 83379,
                "STDBIAS": 12531
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
```

```
                "Cov": 0,
                "REF": 0,
                "HEALED": 1082,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54469,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
            "filtered": {
                "11": 407,
                "10": 968,
                "13": 156,
                "12": 186,
                "15": 145,
                "14": 154,
                "1": 67025,
                "0": 51375,
                "3": 92736,
                "2": 124890,
                "5": 54756,
                "4": 63630,
                "7": 12138,
                "6": 26845,
                "9": 2248,
                "8": 4436
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
                "8": 8
            },
            "unfiltered": {
                "11": 39,
                "10": 49,
                "13": 35,
                "12": 32,
                "15": 18,
                "14": 33,
                "1": 30034,
                "0": 522,
                "3": 5116,
```

```
                "2": 12152,
                "5": 1261,
                "4": 2435,
                "7": 267,
                "6": 573,
                "9": 65,
                "8": 139
            }
        },
        "type": {
            "filtered": {
                "other": 35770,
                "del": 302522,
                "snp": 35483,
                "ins": 155888
            },
            "unfiltered": {
                "other": 2501,
                "del": 1095,
                "snp": 49486,
                "ins": 1387
            }
        },
        "basic": {
            "filtered": 529663,
            "unfiltered": 54469
        }
    },
    "summary": {
        "reason": {
            "filtered": {
                "HPLEN": 6386,
                "REJECTION": 610,
                "Cov": 72787,
                "REF": 472111,
                "HEALED": 348,
                "SHIFT": 458087,
                "SSE": 78820,
                "STRINGENCY": 1500,
                ".": 538589,
                "Quality": 103314,
                "STDBIAS": 12975
            },
            "unfiltered": {
                "HPLEN": 0,
                "REJECTION": 0,
                "Cov": 0,
                "REF": 0,
                "HEALED": 1036,
                "SHIFT": 0,
                "SSE": 0,
                "STRINGENCY": 0,
                ".": 54299,
                "Quality": 0,
                "STDBIAS": 0
            }
        },
        "hrun": {
```

```
            "filtered": {
                "11": 432,
                "10": 972,
                "13": 178,
                "12": 193,
                "15": 160,
                "14": 143,
                "1": 74456,
                "0": 57194,
                "3": 92797,
                "2": 124391,
                "5": 52748,
                "4": 62684,
                "7": 11651,
                "6": 25769,
                "9": 2199,
                "8": 4196
            },
            "run": {
                "11": 11,
                "10": 10,
                "13": 13,
                "12": 12,
                "15": 15,
                "14": 14,
                "1": 1,
                "0": 0,
                "3": 3,
                "2": 2,
                "5": 5,
                "4": 4,
                "7": 7,
                "6": 6,
                "9": 9,
                "8": 8
            },
            "unfiltered": {
                "11": 43,
                "10": 45,
                "13": 20,
                "12": 38,
                "15": 27,
                "14": 27,
                "1": 29948,
                "0": 561,
                "3": 5112,
                "2": 12151,
                "5": 1238,
                "4": 2386,
                "7": 248,
                "6": 567,
                "9": 73,
                "8": 145
            }
        },
        "type": {
            "filtered": {
                "other": 37034,
```

```
                            "del": 278917,
                            "snp": 37539,
                            "ins": 185099
                        },
                        "unfiltered": {
                            "other": 2479,
                            "del": 1093,
                            "snp": 49277,
                            "ins": 1450
                        }
                    },
                    "basic": {
                        "filtered": 538589,
                        "unfiltered": 54299
                    }
                }
            }
        },
        "resultsType": "",
        "tfFastq": "_",
        "tfmetrics": [
            "/rundb/api/v1/tfmetrics/3119/",
            "/rundb/api/v1/tfmetrics/3118/",
            "/rundb/api/v1/tfmetrics/3117/"
        ],
        "parentIDs": "",
        "analysismetrics": [
            "/rundb/api/v1/analysismetrics/31763/"
        ],
        "timeToComplete": "0",
        "reportLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/
        "fastqLink": "/output/Home/Auto_user_D1--632--R54651-p8s2_827b2_20m_man-cf_17446_293943/h
        "pluginresults": [
            "/rundb/api/v1/pluginresult/570190/",
            "/rundb/api/v1/pluginresult/564696/",
            "/rundb/api/v1/pluginresult/564695/",
            "/rundb/api/v1/pluginresult/564694/",
            "/rundb/api/v1/pluginresult/564692/",
            "/rundb/api/v1/pluginresult/564691/",
            "/rundb/api/v1/pluginresult/564690/",
            "/rundb/api/v1/pluginresult/564689/",
            "/rundb/api/v1/pluginresult/564688/",
            "/rundb/api/v1/pluginresult/564687/",
            "/rundb/api/v1/pluginresult/564686/"
        ],
        "framesProcessed": 0,
        "autoExempt": false,
        "resource_uri": "/rundb/api/v1/results/293943/"
    }
  ]
}
```


Allowed HTTP methods

- get
- post

- put
- delete
- patch

4.1.62 Rig Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/rig/`

Schema URL: `http://mytorrentserver/rundb/api/v1/rig/schema/`

Perform CRUD operations on `rig` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
version	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
state	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftproot-dir	Unicode string data. Ex: "Hello World"	results	false	false	false	false	string
last_clean_date	Unicode string data. Ex: "Hello World"		false	false	true	false	string
update-home	Unicode string data. Ex: "Hello World"	192.168.201.1	false	false	false	false	string
ftpserver	Unicode string data. Ex: "Hello World"	192.168.201.1	false	false	false	false	string
com-ments	Unicode string data. Ex: "Hello World"		false	false	true	false	string
last_experiment	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftppass-word	Unicode string data. Ex: "Hello World"	ionguest	false	false	false	false	string
update-flag	Boolean data. Ex: True	false	false	false	true	false	boolean
location	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	re-lated
last_init_date	Unicode string data. Ex: "Hello World"		false	false	true	false	string
alarms	Unicode string data. Ex: "Hello World"	{}	false	false	true	false	string
serial	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
host_address	Unicode string data. Ex: "Hello World"		false	false	true	false	string
type	Unicode string data. Ex: "Hello World"		false	false	true	false	string
ftpuser-name	Unicode string data. Ex: "Hello World"	ionguest	false	false	false	false	string
re-source_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/rig/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/rig/", params={"format": "json", "
ts_api_response = ts_api_request.json()

rigs = ts_api_response["objects"]

for rig in rigs:
    print rig
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 175,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/rig/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "version": {},
            "name": "PGM_test",
            "state": "",
            "ftprootdir": "results",
            "last_clean_date": "",
            "updatehome": "192.168.201.1",
            "ftpserver": "192.168.201.1",
            "comments": "",
            "last_experiment": "",
            "ftppassword": "ionguest",
            "updateflag": false,
            "location": {
                "name": "Home",
                "resource_uri": "/rundb/api/v1/location/1/",
                "defaultlocation": true,
                "comments": "",
                "id": 1
            },
            "last_init_date": "",
            "alarms": {},
            "serial": "",
            "host_address": "",
            "type": "",
            "ftpusername": "ionguest",
            "resource_uri": "/rundb/api/v1/rig/PGM_test/"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.63 Runtype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/runtype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/runtype/schema/`

Perform CRUD operations on `runtype` resources and data elements.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
applicationGroups	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	false	false	related
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"	dna	false	false	true	false	string
barcode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
meta	Unicode string data. Ex: "Hello World"		true	false	false	false	string
runType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
alternate_name	Unicode string data. Ex: "Hello World"		true	false	false	false	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/runtype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/runtype/", params={"format": "jso
ts_api_response = ts_api_request.json()
```

```
runtypes = ts_api_response["objects"]

for runtype in runtypes:
    print runtype
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/runtype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "applicationGroups": [
                "/rundb/api/v1/applicationgroup/1/",
                "/rundb/api/v1/applicationgroup/3/",
                "/rundb/api/v1/applicationgroup/4/"
            ],
            "description": "Generic Sequencing",
            "nucleotideType": "",
            "barcode": "",
            "meta": {},
            "runType": "GENS",
            "id": 1,
            "alternate_name": "Other",
            "resource_uri": "/rundb/api/v1/runtype/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.64 Sample Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sample/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sample/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
sampleSets	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
displayedName	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
experiments	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
externalId	Unicode string data. Ex: “Hello World”		true	false	false	false	string
date	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	true	false	false	false	datetime
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
name	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sample/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sample/", params={"format": "json"
ts_api_response = ts_api_request.json()

samples = ts_api_response["objects"]

for sample in samples:
    print sample
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 7765,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sample/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
```

```
            "status": "run",
            "sampleSets": [],
            "description": "",
            "displayedName": "E122627-lq405-L8095",
            "experiments": [
                "/rundb/api/v1/experiment/11750/"
            ],
            "externalId": "",
            "date": "2013-07-23T17:11:31.000986+00:00",
            "resource_uri": "/rundb/api/v1/sample/3265/",
            "id": 3265,
            "name": "E122627-lq405-L8095"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.65 Sampleannotation_Cv Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/schema/`

Perform read-only operations on `sampleannotation_cv` resources. This resource corresponds to the supported sample relationships (Self | Proband, Tumor, Normal, Mother, Father, etc) in Ion Reporter™ Software.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
annotationType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
isIRCompatible	Boolean data. Ex: True	false	false	false	true	false	boolean
sampleGroupType_CV	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
value	Unicode string data. Ex: "Hello World"		false	false	true	false	string
iRValue	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
iRAnnotationType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleannotation_cv/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleannotation_cv/", params={"fo
ts_api_response = ts_api_request.json()

sampleannotation_cvs = ts_api_response["objects"]

for sampleannotation_cv in sampleannotation_cvs:
    print sampleannotation_cv
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 34,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sampleannotation_cv/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "annotationType": "relationshipRole",
            "uid": "SAMPLEANNOTATE_CV_0001",
            "isIRCompatible": true,
```

```
            "sampleGroupType_CV": "/rundb/api/v1/samplegrouptype_cv/1/",
            "value": "Sample",
            "iRValue": "Sample",
            "iRAnnotationType": "Relation",
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/sampleannotation_cv/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.66 Sampleattribute Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
dataType_name	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
dataType	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
displayedName	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
isMandatory	Boolean data. Ex: True	false	false	false	true	false	boolean
sampleCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleattribute/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleattribute/", params={"format
ts_api_response = ts_api_request.json()

sampleattributes = ts_api_response["objects"]

for sampleattribute in sampleattributes:
    print sampleattribute
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.67 Sampleattributedatatype Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
dataType	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
id	Integer data. Ex: 2673		false	false	true	true	integer

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleattributedatatype/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleattributedatatype/", params=
ts_api_response = ts_api_request.json()

sampleattributedatatypes = ts_api_response["objects"]

for sampleattributedatatype in sampleattributedatatypes:
    print sampleattributedatatype
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 2,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sampleattributedatatype/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "dataType": "Text",
            "resource_uri": "/rundb/api/v1/sampleattributedatatype/1/",
            "description": "Up to 1024 characters",
            "isActive": true,
            "id": 1
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put

- delete
- patch

4.1.68 Samplegrouptype_Cv Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/schema/`

Perform read-only operations on `samplegrouptype_cv` resources. This resource corresponds to the supported relationship types (Single, Paired, Trio, etc) in Ion Reporter™ Software and to the sample set Grouping column in the Torrent Suite™ Software.

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isIRCompatible	Boolean data. Ex: True	false	false	false	true	false	boolean
description	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
sampleAnnotation_set	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
displayedName	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
iRValue	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
iRAnnotationType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
sampleSets	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
id	Integer data. Ex: 2673		false	false	true	true	integer
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplegrouptype_cv/", params={"fo
ts_api_response = ts_api_request.json()

samplegrouptype_cvs = ts_api_response["objects"]
```

```
for samplegrouptype_cv in samplegrouptype_cvs:
    print samplegrouptype_cv
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 5,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/samplegrouptype_cv/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isIRCompatible": true,
            "description": "",
            "sampleAnnotation_set": [
                "/rundb/api/v1/sampleannotation_cv/1/",
                "/rundb/api/v1/sampleannotation_cv/2/"
            ],
            "displayedName": "Sample_Control",
            "iRValue": "Paired_Sample|Sample_Control",
            "iRAnnotationType": "RelationshipType",
            "uid": "SAMPLEGROUP_CV_0001",
            "sampleSets": [],
            "id": 1,
            "isActive": true,
            "resource_uri": "/rundb/api/v1/samplegrouptype_cv/1/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.69 Sampleset Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sampleset/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sampleset/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
description	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
sampleCount	Integer data. Ex: 2673	n/a	false	true	false	false	integer
displayedName	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
SampleGroupType_CV	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
samples	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	true	false	true	false	related
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
sampleGroupTypeName	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sampleset/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sampleset/", params={"format": "j
ts_api_response = ts_api_request.json()

samplesets = ts_api_response["objects"]

for sampleset in samplesets:
    print sampleset
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 1,
        "offset": 0,
        "limit": 1,
        "next": null
```

```
    },
    "objects": [
        {
            "status": "created",
            "description": "",
            "sampleCount": 4,
            "displayedName": "Set",
            "SampleGroupType_CV": "/rundb/api/v1/samplegrouptype_cv/3/",
            "samples": [
                "/rundb/api/v1/samplesetitem/14/",
                "/rundb/api/v1/samplesetitem/15/",
                "/rundb/api/v1/samplesetitem/16/",
                "/rundb/api/v1/samplesetitem/17/"
            ],
            "lastModifiedDate": "2013-10-07T12:04:51.000432+00:00",
            "sampleGroupTypeName": "Tumor_Normal",
            "creationDate": "2013-10-07T12:04:51.000432+00:00",
            "id": 5,
            "resource_uri": "/rundb/api/v1/sampleset/5/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.70 Samplesetitem Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
sample	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
gender	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
relationshipGroup	Integer data. Ex: 2673	n/a	false	false	false	false	integer
cellularityPct	Integer data. Ex: 2673	n/a	true	false	false	false	integer
relationshipRole	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
cancerType	Unicode string data. Ex: "Hello World"	n/a	true	false	false	false	string
sampleSet	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
lastModifiedDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
dnabarcode	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
creationDate	A date & time as a string. Ex: "2010-11-10T03:07:43"	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplesetitem/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplesetitem/", params={"format"
ts_api_response = ts_api_request.json()

samplesetitems = ts_api_response["objects"]

for samplesetitem in samplesetitems:
    print samplesetitem
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
```

```
        "next": "/rundb/api/v1/samplesetitem/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "sample": "/rundb/api/v1/sample/4877/",
            "gender": "Female",
            "relationshipGroup": 1,
            "cellularityPct": null,
            "relationshipRole": "Self",
            "cancerType": null,
            "sampleSet": "/rundb/api/v1/sampleset/5/",
            "lastModifiedDate": "2013-10-07T12:04:51.000440+00:00",
            "dnabarcode": null,
            "creationDate": "2013-10-07T12:04:51.000440+00:00",
            "id": 14,
            "resource_uri": "/rundb/api/v1/samplesetitem/14/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.71 Samplesetiteminfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
sample	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
sampleSetPk	Integer data. Ex: 2673	n/a	true	true	true	false	integer
sampleExternalId	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
sampleDisplayedName	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
gender	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
relationshipGroup	Integer data. Ex: 2673	n/a	true	true	true	false	integer
cellularityPct	Integer data. Ex: 2673	n/a	true	false	false	false	integer
dnabarcodeKit	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
sampleDescription	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
relationshipRole	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
cancerType	Unicode string data. Ex: “Hello World”	n/a	true	false	false	false	string
samplePk	Integer data. Ex: 2673	n/a	true	true	true	false	integer
lastModifiedDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
dnabarcode	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	true	true	false	related
sampleSet	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	true	false	related
creationDate	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	datetime
id	Integer data. Ex: 2673		false	false	true	true	integer
sampleSetStatus	Unicode string data. Ex: “Hello World”	n/a	true	true	true	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/samplesetiteminfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/samplesetiteminfo/", params={"form
ts_api_response = ts_api_request.json()
```

```
samplesetiteminfos = ts_api_response["objects"]

for samplesetiteminfo in samplesetiteminfos:
    print samplesetiteminfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 4,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/samplesetiteminfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "sample": "/rundb/api/v1/sample/4877/",
            "samplePk": 4877,
            "sampleExternalId": "NA10859",
            "sampleDisplayedName": "1347-02",
            "gender": "Female",
            "relationshipGroup": 1,
            "cellularityPct": null,
            "dnabarcodeKit": "",
            "sampleDescription": "mother",
            "relationshipRole": "Self",
            "cancerType": null,
            "attribute_dict": {},
            "lastModifiedDate": "2013-10-07T12:04:51.000440+00:00",
            "dnabarcode": "",
            "sampleSetPk": 5,
            "sampleSet": "/rundb/api/v1/sampleset/5/",
            "creationDate": "2013-10-07T12:04:51.000440+00:00",
            "id": 14,
            "sampleSetStatus": "created",
            "resource_uri": "/rundb/api/v1/samplesetiteminfo/14/"
        }
    ]
}
```


Allowed HTTP methods

- get

4.1.72 Sequencingkitinfo Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
isActive	Boolean data. Ex: True	true	false	false	true	false	boolean
kitType	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
description	Unicode string data. Ex: "Hello World"		false	false	true	false	string
nucleotideType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
instrumentType	Unicode string data. Ex: "Hello World"		false	false	true	false	string
runMode	Unicode string data. Ex: "Hello World"		false	false	true	false	string
parts	Many related resources. Can be either a list of URIs or list of individually nested resource data.	n/a	false	false	false	false	related
flowCount	Integer data. Ex: 2673	n/a	false	false	false	false	integer
applicationType	Unicode string data. Ex: "Hello World"		true	false	false	false	string
uid	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
categories	Unicode string data. Ex: "Hello World"		true	false	false	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sequencingkitinfo/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sequencingkitinfo/", params={"form
ts_api_response = ts_api_request.json()

sequencingkitinfos = ts_api_response["objects"]

for sequencingkitinfo in sequencingkitinfos:
    print sequencingkitinfo
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 18,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sequencingkitinfo/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isActive": false,
            "kitType": "SequencingKit",
            "description": "(200bp) Ion Sequencing 200 Kit",
            "nucleotideType": "",
            "instrumentType": "pgm",
            "runMode": "",
            "parts": [
                {
                    "barcode": "4471258",
                    "id": 20005,
                    "resource_uri": "/rundb/api/v1/kitpart/20005/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471257",
                    "id": 20006,
                    "resource_uri": "/rundb/api/v1/kitpart/20006/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471259",
                    "id": 20007,
                    "resource_uri": "/rundb/api/v1/kitpart/20007/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                },
                {
                    "barcode": "4471260",
                    "id": 20008,
                    "resource_uri": "/rundb/api/v1/kitpart/20008/",
                    "kit": "/rundb/api/v1/kitinfo/20002/"
                }
            ],
            "flowCount": 520,
            "applicationType": "",
            "uid": "SEQ0002",
            "resource_uri": "/rundb/api/v1/sequencingkitinfo/20002/",
            "id": 20002,
            "categories": "",
            "name": "IonSeq200Kit"
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.73 Sequencingkitpart Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/`

Schema URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
barcode	Unicode string data. Ex: "Hello World"	n/a	false	false	false	true	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
kit	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/sequencingkitpart/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/sequencingkitpart/", params={"forr
ts_api_response = ts_api_request.json()

sequencingkitparts = ts_api_response["objects"]

for sequencingkitpart in sequencingkitparts:
    print sequencingkitpart
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 64,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/sequencingkitpart/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "barcode": "A25592",
            "id": 20132,
            "resource_uri": "/rundb/api/v1/sequencingkitpart/20132/",
            "kit": "/rundb/api/v1/kitinfo/20063/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.74 Supportupload Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/supportupload/`

Schema URL: `http://mytorrentserver/rundb/api/v1/supportupload/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
ticket_id	Unicode string data. Ex: “Hello World”		false	false	true	false	string
updated	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
local_message	Unicode string data. Ex: “Hello World”		false	false	true	false	string
description	Unicode string data. Ex: “Hello World”		false	false	false	false	string
created	A date & time as a string. Ex: “2010-11-10T03:07:43”	true	false	false	true	false	date-time
ticket_status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
contact_email	Unicode string data. Ex: “Hello World”		false	false	false	false	string
result	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
file	A single related resource. Can be either a URI or set of nested resource data.	n/a	true	false	false	false	related
celery_task_id	Unicode string data. Ex: “Hello World”		false	false	true	false	string
ticket_message	Unicode string data. Ex: “Hello World”		false	false	true	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
local_status	Unicode string data. Ex: “Hello World”		false	false	true	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/supportupload/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/supportupload/", params={"format"
ts_api_response = ts_api_request.json()

supportuploads = ts_api_response["objects"]

for supportupload in supportuploads:
    print supportupload
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
```

```
        "total_count": 0,
        "offset": 0,
        "limit": 1,
        "next": null
    },
    "objects": []
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.75 Template Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/template/`

Schema URL: `http://mytorrentserver/rundb/api/v1/template/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
isofficial	Boolean data. Ex: True	true	false	false	true	false	boolean
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
sequence	Unicode string data. Ex: “Hello World”		false	false	true	false	string
comments	Unicode string data. Ex: “Hello World”		false	false	true	false	string
key	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/template/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/template/", params={"format": "jso
ts_api_response = ts_api_request.json()

templates = ts_api_response["objects"]

for template in templates:
    print template
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 9,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/template/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "isofficial": true,
            "name": "TF1.2(tA-tB30') Hyb extend",
            "sequence": "GTTTTAGGGTCCCCGGGGTTAAAAGGTTTCGAACACGATGTCGGAGACACGCAGGGATGAGATGG",
            "comments": "",
            "key": "ATCGT",
            "id": 7,
            "resource_uri": "/rundb/api/v1/template/7/"
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.76 Tfmetrics Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/`

Schema URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/schema/`

Perform read-only operations on `tfmetrics` resources and data elements.

Fields table

field	help text	default	nullable	read-only	blank	unique	type
corrHPSNR	Unicode string data. Ex: "Hello World"		false	false	true	false	string
Q10Mean	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
SysSNR	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
HPAccuracy	Unicode string data. Ex: "Hello World"		false	false	true	false	string
Q17ReadCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
sequence	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
Q17Histo	Unicode string data. Ex: "Hello World"		false	false	true	false	string
name	Unicode string data. Ex: "Hello World"	n/a	false	false	false	false	string
aveKeyCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
number	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
id	Integer data. Ex: 2673		false	false	true	true	integer
keypass	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
Q10ReadCount	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
report	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
resource_uri	Unicode string data. Ex: "Hello World"	n/a	false	true	false	false	string
Q17Mean	Floating point numeric data. Ex: 26.73	n/a	false	false	false	false	float
Q10Histo	Unicode string data. Ex: "Hello World"		false	false	true	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/tfmetrics/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/tfmetrics/", params={"format": "js
ts_api_response = ts_api_request.json()

tfmetricss = ts_api_response["objects"]

for tfmetrics in tfmetricss:
    print tfmetrics
```

Torrent Server response

```
{
    "meta": {
        "previous": null,
        "total_count": 7320,
```

```
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/tfmetrics/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "corrHPSNR": "",
            "Q10Mean": 79.7,
            "SysSNR": 20.19,
            "HPAccuracy": "0 : 560244/582614, 1 : 356550/377171, 2 : 35115/45374, 3 : 0/0, 4 : 521/43
            "Q17ReadCount": 3992.0,
            "sequence": "TTGCGCGCGCTGTGAATGCGCTGCTGTCGAATCGCGCTGCGCTGAACGTCGCGTGCGCGAACGATCTGAGACTGCG
            "Q17Histo": "951 0 0 1 9 5 2 7 6 277 5 0 1 2 3 0 2 1 6 1 7 2 3 3 0 0 10 1 0 26 0 2 0 1 2
            "name": "TF_D",
            "aveKeyCount": 71.0,
            "number": 4119.0,
            "id": 1,
            "keypass": 5368.0,
            "Q10ReadCount": 4586.0,
            "report": "/rundb/api/v1/results/89/",
            "resource_uri": "/rundb/api/v1/tfmetrics/1/",
            "Q17Mean": 66.56,
            "Q10Histo": "40 0 0 1 8 3 0 4 2 1 587 5 3 5 1 8 0 2 6 1 5 1 3 3 2 1 5 9 0 0 2 0 2 1 5 1 0
        }
    ]
}
```


Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.77 Threeprimeadapter Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/`

Schema URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/schema/`

Fields table

field	help text	default	nullable	readonly	blank	unique	type
direction	Unicode string data. Ex: “Hello World”	Forward	false	false	false	false	string
name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
sequence	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
chemistryType	Unicode string data. Ex: “Hello World”		false	false	true	false	string
runMode	Unicode string data. Ex: “Hello World”	single	false	false	true	false	string
uid	Unicode string data. Ex: “Hello World”	n/a	false	false	false	true	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
isDefault	Boolean data. Ex: True	false	false	false	true	false	boolean
description	Unicode string data. Ex: “Hello World”		false	false	true	false	string

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/threeprimeadapter/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/threeprimeadapter/", params={"form
ts_api_response = ts_api_request.json()

threeprimeadapters = ts_api_response["objects"]

for threeprimeadapter in threeprimeadapters:
    print threeprimeadapter
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 17,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/threeprimeadapter/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "direction": "Forward",
            "name": "Ion P1B",
            "sequence": "ATCACCGACTGCCCATAGAGAGGCTGAGAC",
            "chemistryType": "",
            "runMode": "single",
            "uid": "FWD_0001",
            "resource_uri": "/rundb/api/v1/threeprimeadapter/1/",
            "id": 1,
            "isDefault": true,
            "description": "Default forward adapter"
```

```
        }
    ]
}
```

Allowed HTTP methods

- get
- post
- put
- delete
- patch

4.1.78 User Resource

Resource URL: `http://mytorrentserver/rundb/api/v1/user/`

Schema URL: `http://mytorrentserver/rundb/api/v1/user/schema/`

Fields table

field	help text	default	nullable	read-only	blank	unique	type
profile	A single related resource. Can be either a URI or set of nested resource data.	n/a	false	false	false	false	related
username	Required. 30 characters or fewer. Letters, numbers and @/./+/-/_ characters	n/a	false	false	false	true	string
first_name	Unicode string data. Ex: “Hello World”		false	false	true	false	string
last_name	Unicode string data. Ex: “Hello World”		false	false	true	false	string
is_active	Designates whether this user should be treated as active. Unselect this instead of deleting accounts.	true	false	false	true	false	boolean
email	Unicode string data. Ex: “Hello World”		false	false	true	false	string
last_login	A date & time as a string. Ex: “2010-11-10T03:07:43”	2014-06-28T14:04:05.000090+00:00	false	false	false	false	datetime
full_name	Unicode string data. Ex: “Hello World”	n/a	false	false	false	false	string
resource_uri	Unicode string data. Ex: “Hello World”	n/a	false	true	false	false	string
id	Integer data. Ex: 2673		false	false	true	true	integer
date_joined	A date & time as a string. Ex: “2010-11-10T03:07:43”	2014-06-28T14:04:05.000090+00:00	false	false	false	false	datetime

Example request

Request URL: `http://mytorrentserver/rundb/api/v1/user/?format=json&limit=1`

Python example

```
import requests

ts_api_request = requests.get("http://mytorrentserver/rundb/api/v1/user/", params={"format": "json",
ts_api_response = ts_api_request.json()

users = ts_api_response["objects"]

for user in users:
    print user
```

Torrent Server response


```
{
    "meta": {
        "previous": null,
        "total_count": 20,
        "offset": 0,
        "limit": 1,
        "next": "/rundb/api/v1/user/?offset=1&limit=1&format=json"
    },
    "objects": [
        {
            "profile": {
                "phone_number": "",
                "name": "",
                "title": "user",
                "last_read_news_post": "1984-11-05T05:00:00+00:00",
                "note": "",
                "id": 6,
                "resource_uri": ""
            },
            "username": "dm_contact",
            "first_name": "",
            "last_name": "",
            "is_active": true,
            "email": "ts-admin@mailman.itw",
            "last_login": "2013-03-13T17:28:05.000596+00:00",
            "full_name": "",
            "resource_uri": "/rundb/api/v1/user/6/",
            "id": 6,
            "date_joined": "2013-03-13T17:28:05.000596+00:00"
        }
    ]
}
```


Allowed HTTP methods

- get

4.2 Torrent Server Database Tables

4.2.1 Database Table rundb_analysisargs

Postgres database: `iondb`

Postgres table: `rundb_analysisargs`

Analysis arguments data model.

Lifecycle

Updated when an analysis is launched or re-analyzed.

Schema

4.2.2 Database Table rundb_analysismetrics

Postgres database: `iondb`

Postgres table: `rundb_analysismetrics`

Analysis metrics data model.

Lifecycle

The `rundb_analysismetrics` table values are computed during the pipeline analysis phase.

Referenced by

- rundb_results

Schema

4.2.3 Database Table rundb_applicationgroup

Postgres database: `iondb`

Postgres table: `rundb_applicationgroup`

Application group data model.

Referenced by

- rundb_plannedexperiment

Schema

4.2.4 Database Table rundb_applproduct

Postgres database: `iondb`

Postgres table: `rundb_applproduct`

Application product data model.

Schema

4.2.5 Database Table rundb_backup

Postgres database: `iondb`

Postgres table: `rundb_backup`

Backup description data model.

Lifecycle

The `rundb_backup` table elements are updated when a particular experiment is archived.

Schema

4.2.6 Database Table rundb_backupconfig

Postgres database: `iondb`

Postgres table: `rundb_backupconfig`

Backup configuration specification data model.

Lifecycle

In previous releases, these database items are set to default values and are modified using the Torrent Browser Services tab Archive panel. In 3.6 and beyond, these are placed by the data management tables.

Schema

4.2.7 Database Table rundb_chip

Postgres database: `iondb`

Postgres table: `rundb_chip`

Chip description data model. These are the per-chip default analysis arguments.

Lifecycle

These database items are set to default values during Torrent Server installation.

Schema

4.2.8 Database Table rundb_content

Postgres database: `iondb`

Postgres table: `rundb_content`

Content description data model.

Schema

4.2.9 Database Table rundb_contentupload

Postgres database: `iondb`

Postgres table: `rundb_contentupload`

Content upload data model.

Referenced by

- rundb_content
- rundb_usereventlog

Schema

4.2.10 Database Table rundb_cruncher

Postgres database: `iondb`

Postgres table: `rundb_cruncher`

Cruncher data model.

Schema

4.2.11 Database Table rundb_dm_prune_field

Postgres database: `iondb`

Postgres table: `rundb_dm_prune_field`

Data management prune field data model.

Schema

4.2.12 Database Table rundb_dm_prune_group

Postgres database: `iondb`

Postgres table: `rundb_dm_prune_group`

Data management prune group data model.

Schema

4.2.13 Database Table rundb_dm_reports

Postgres database: `iondb`

Postgres table: `rundb_dm_reports`

Data management reports data model.

Schema

4.2.14 Database Table rundb_dmfileset

Postgres database: `iondb`

Postgres table: `rundb_dmfileset`

Data management file set data model.

Referenced by

- rundb_dmfilestat

Schema

4.2.15 Database Table rundb_dmfilestat

Postgres database: `iondb`

Postgres table: `rundb_dmfilestat`

Data management file statistics data model.

Schema

4.2.16 Database Table rundb_dnabarcode

Postgres database: `iondb`

Postgres table: `rundb_dnabarcode`

Barcode data model (for experiments that use a barcoding kit)

Lifecycle

These database elements are populated by user input on the sequencing instrument. The `rundb_experiment.barcodeId` element references this table to create the `/results/barcodeList.txt` file.

Schema

4.2.17 Database Table rundb_emailaddress

Postgres database: `iondb`

Postgres table: `rundb_emailaddress`

Report recipient email address description data model.

Lifecycle

These database items are updated in the Torrent Browser admin Configure tab Email panel.

Schema

4.2.18 Database Table rundb_eventlog

Postgres database: `iondb`

Postgres table: `rundb_eventlog`

Event log data model.

Schema

4.2.19 Database Table rundb_experiment

Postgres database: `iondb`

Postgres table: `rundb_experiment`

Experiment description data model.

Lifecycle

Each sequencing run has a corresponding `rundb_experiment` table record. These database items are set by the Crawler process, which monitors directories containing PGM or Proton experiment data and creates a record for each new experiment it finds.

Referenced by

- rundb_backup
- rundb_results

Related tables

The following tables hold data related to experiments:

- sample
- plannedexperiment

- plannedexperimentqc
- experimentanalysissettings

Schema

4.2.20 Database Table rundb_experimentanalysissettings

Postgres database: `iondb`

Postgres table: `rundb_experimentanalysissettings`

Experiment analysis settings data model.

Lifecycle

A versioned set of analysis parameter values is created when the user modifies settings for a reanalysis run.

Referenced by

- rundb_results

Related tables

The following tables hold data on experiments.

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings
- experiment
- project

Schema

4.2.21 Database Table rundb_filemonitor

Postgres database: `iondb`

Postgres table: `rundb_filemonitor`

File Monitor data model.

Referenced by

- rundb_referencegenome

Schema

4.2.22 Database Table rundb_fileserver

Postgres database: `iondb`

Postgres table: `rundb_fileserver`

Ion Torrent server file server description data model.

Schema

4.2.23 Database Table rundb_globalconfig

Postgres database: `iondb`

Postgres table: `rundb_globalconfig`

Global configuration parameters data model.

Lifecycle

These configuration items are set to default values at system installation.

Schema

4.2.24 Database Table rundb_kitinfo

Postgres database: `iondb`

Postgres table: `rundb_kitinfo`

Library kit information data model.

Referenced by

- rundb_applproduct
- rundb_kitpart

Schema

4.2.25 Database Table rundb_kitpart

Postgres database: `iondb`

Postgres table: `rundb_kitpart`

Library kit part number data model.

Schema

4.2.26 Database Table rundb_libmetrics

Postgres database: `iondb`

Postgres table: `rundb_libmetrics`

Library metrics data model.

Lifecycle

Library metrics are computed as part of the analysis pipeline.

Referenced by

- rundb_results

Schema

4.2.27 Database Table rundb_librarykey

Postgres database: `iondb`

Postgres table: `rundb_librarykey`

Library Key data model.

Schema

4.2.28 Database Table rundb_librarykit

Postgres database: `iondb`

Postgres table: `rundb_librarykit`

Library kit data model.

Schema

4.2.29 Database Table rundb_location

Postgres database: `iondb`

Postgres table: `rundb_location`

Ion Torrent server description data model.

Referenced by

- rundb_backupconfig
- rundb_cruncher
- rundb_fileserver
- rundb_rig

Schema

4.2.30 Database Table rundb_message

Postgres database: `iondb`

Postgres table: `rundb_message`

Message data model.

Schema

4.2.31 Database Table rundb_monitordata

Postgres database: `iondb`

Postgres table: `rundb_monitordata`

Monitor Data data model.

Schema

4.2.32 Database Table rundb_newspost

Postgres database: `iondb`

Postgres table: `rundb_newspost`

Newspost data model.

Lifecycle

Created when a news message is sent from Ion to Torrent Suite™ Software.

Schema

4.2.33 Database Table rundb_plannedexperiment

Postgres database: `iondb`

Postgres table: `rundb_plannedexperiment`

Planned experiment description data model.

Note: Use *this* table (not other tables in the related tables list) if you are doing LIMS integration.

Lifecycle

Created by user in the Torrent Browser Planning tab.

Referenced by

- rundb_experiment
- rundb_plannedexperimentqc

Related tables

The following other tables also hold data on experiments:

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings
- experiment
- project

Schema

4.2.34 Database Table rundb_plannedexperimentqc

Postgres database: `iondb`

Postgres table: `rundb_plannedexperimentqc`

Planned experiment QC description data model. Holds the QC threshold metrics from a planned experiment.

Related tables

The following tables also hold data on experiments:

- sample
- plannedexperiment
- plannedexperimentqc
- experimentanalysissettings
- experiment
- project

Schema

4.2.35 Database Table rundb_plugin

Postgres database: `iondb`

Postgres table: `rundb_plugin`

Plugin description data model.

Lifecycle

Created when a plugin is installed.

Referenced by

- rundb_pluginresult

Schema

4.2.36 Database Table rundb_pluginresult

Postgres database: `iondb`

Postgres table: `rundb_pluginresult`

Plugin result data model.

Schema

4.2.37 Database Table rundb_project

Postgres database: `iondb`

Postgres table: `rundb_project`

Project data model.

Schema

4.2.38 Database Table rundb_publisher

Postgres database: `iondb`

Postgres table: `rundb_publisher`

Publisher data model.

Referenced by

- rundb_content
- rundb_contentupload

Schema

4.2.39 Database Table rundb_qctype

Postgres database: `iondb`

Postgres table: `rundb_qctype`

QC type data model (the name of the QC metric, such as Bead Loading, Key Signal, or Usable Sequence).

Referenced by

- rundb_plannedexperimentqc

Schema

4.2.40 Database Table rundb_qualitymetrics

Postgres database: `iondb`

Postgres table: `rundb_qualitymetrics`

Quality metrics data model.

Lifecycle

Quality metrics are calculated during the alignment QC stage of the analysis pipeline.

Referenced by

- rundb_results

Schema

4.2.41 Database Table rundb_referencegenome

Postgres database: `iondb`

Postgres table: `rundb_referencegenome`

Reference genome description data model.

Lifecycle

This data is created when a reference genome is uploaded in the Torrent Browser. These data are used by the PGM and Proton to build a list of available genomes.

Schema

4.2.42 Database Table rundb_remoteaccount

Postgres database: `iondb`

Postgres table: `rundb_remoteaccount`

Schema

4.2.43 Database Table rundb_reportstorage

Postgres database: `iondb`

Postgres table: `rundb_reportstorage`

Report output location description data model.

Lifecycle

These data are used to generate the weekly report.

Referenced by

- rundb_results

Schema

4.2.44 Database Table rundb_results

Postgres database: `iondb`

Postgres table: `rundb_results`

Experiment results description data model.

Lifecycle

A `rundb_results` table record is created each time the analysis pipeline is executed.

Referenced by

- rundb_analysismetrics
- rundb_dmfilestat
- rundb_experimentanalysissettings
- rundb_libmetrics
- rundb_qualitymetrics
- rundb_pluginresult
- rundb_tfmetrics

Schema

4.2.45 Database Table rundb_rig

Postgres database: `iondb`

Postgres table: `rundb_rig`

PGM or Proton description data model.

Schema

4.2.46 Database Table rundb_runscript

Postgres database: `iondb`

Postgres table: `rundb_runscript`

Run script data model. The run script is the Python script that runs the analysis pipeline, adds metrics to the database, and generates reports.

Schema

4.2.47 Database Table rundb_runtype

Postgres database: `iondb`

Postgres table: `rundb_runtype`

Run type data model.

Referenced by

- rundb_applproduct

Schema

4.2.48 Database Table rundb_sample

Postgres database: `iondb`

Postgres table: `rundb_sample`

Sample data model.

Referenced by

- rundb_samplesetitem
- rundb_sampleattributevalue

Schema

4.2.49 Database Table rundb_sampleannotation_cv

Postgres database: `iondb`

Postgres table: `rundb_sampleannotation_cv`

Sample Annotation CV data model. This table corresponds to the supported sample relationships (Self | Proband, Tumor, Normal, Mother, Father, etc) in Ion Reporter™ Software.

Schema

4.2.50 Database Table rundb_sampleattribute

Postgres database: `iondb`

Postgres table: `rundb_sampleattribute`

Sample Attribute data model.

Lifecycle

Created when the user creates a sample attribute.

Referenced by

- rundb_sampleattributevalue

Schema

4.2.51 Database Table rundb_sampleattributedatatype

Postgres database: `iondb`

Postgres table: `rundb_sampleattributedatatype`

Sample Attribute Data Type data model.

Referenced by

- rundb_sampleattribute

Schema

4.2.52 Database Table rundb_sampleattributevalue

Postgres database: `iondb`

Postgres table: `rundb_sampleattributevalue`

Sample Attribute Value data model.

Lifecycle

Created when the user assigns a sample attribute value.

Schema

4.2.53 Database Table rundb_samplegrouptype_cv

Postgres database: `iondb`

Postgres table: `rundb_samplegrouptype_cv`

Sample Group Type CV data model. This table corresponds to the supported relationship types (Single, Paired, Trio, etc) in Ion Reporter™ Software and to the sample set Grouping column in the Torrent Suite™ Software.

Referenced by

- rundb_sampleannotation_cv

Schema

4.2.54 Database Table rundb_sampleset

Postgres database: `iondb`

Postgres table: `rundb_sampleset`

Sample Set data model.

Lifecycle

Created when the user creates a sample set in the Torrent Browser.

Referenced by

- rundb_samplesetitem
- rundb_plannedexperiment

Schema

4.2.55 Database Table rundb_samplesetitem

Postgres database: `iondb`

Postgres table: `rundb_samplesetitem`

Sample Set Item data model.

Lifecycle

Created when the user assigns a sample to a sample set.

Schema

4.2.56 Database Table rundb_sequencingkit

Postgres database: `iondb`

Postgres table: `rundb_sequencingkit`

Sequencing kit data model.

Schema

4.2.57 Database Table rundb_supportupload

Postgres database: `iondb`

Postgres table: `rundb_supportupload`

Schema

4.2.58 Database Table rundb_template

Postgres database: `iondb`

Postgres table: `rundb_template`

Test fragment template description data model.

Schema

4.2.59 Database Table rundb_tfmetrics

Postgres database: `iondb`

Postgres table: `rundb_tfmetrics`

Test Fragment (TF) metrics data model.

Lifecycle

TF metrics are calculated during the basecalling phase of the analysis pipeline.

Schema

4.2.60 Database Table rundb_threeprimeadapter

Postgres database: `iondb`

Postgres table: `rundb_threeprimeadapter`

Three prime adapter data model.

Schema

4.2.61 Database Table rundb_usereventlog

Postgres database: `iondb`

Postgres table: `rundb_usereventlog`

User event log data model.

Schema

4.2.62 Database Table rundb_userprofile

Postgres database: `iondb`

Postgres table: `rundb_userprofile`

User profile data model.

Schema

4.2.63 Database Table rundb_variantfrequencies

Postgres database: `iondb`

Postgres table: `rundb_variantfrequencies`

Variant frequencies data model.

Schema

CHAPTER 5

Other SDK documents and guides

5.1 Torrent Suite™ Software Database User Guide

5.1.1 Introduction

The Database User Guide describes how to work with the experiment metadata and analysis results stored in the Ion Torrent database. Refer to the Database Schema Reference for a detailed description of database content and structure.

The Torrent Server includes PostgreSQL, which is an open-source object-relational Database Management System (DBMS) that supports almost all SQL constructs. PostgreSQL APIs are available for the most popular programming languages to build applications using the database for backend data store. The main user interface to PostgreSQL is the `psql` command line program. The psql program permits you to enter database queries directly from a terminal or to execute a query sequence from a file. Database queries demonstrated in this guide use `psql`.

This document, as part of the Torrent Suite™ Software SDK, shows how to manipulate the database using the command line and complements the API, which provides a programmatic way of accessing the database.

Torrent SDK Getting Started with PostgreSQL (psql)

The `psql` program is a command line client that accesses the PostgreSQL database both programmatically and interactively.

References

- Practical PostgreSQL
- PostgreSQL 8.4.7 Documentation - Chapter 19. Client Authentication 19.1 - The pg_hba.conf file.
- On your PostgreSQL server, view the `psql manpage` with the following command: `man psql`

Connect to the database locally

If you are logged into the Torrent Server, you can interactively run `psql` and connect to the database using the following command:

```
ionadmin@myserver:~$ psql -U ion -d iondb
psql (8.4.7)
Type "help" for help.
```

```
iondb=>
```

After connecting, you can continue to interactively access the database using PostgreSQL queries. *iondb=>* is the command prompt.

Connect to the database remotely

To remotely connect to the database, you may need to do one or more of the following actions:

- Change security settings
- Install a PostgreSQL client
- Connect programmatically or using the `psql` command line client

Change remote access security settings

By default, the postgres database in the Torrent Server is configured to restrict remote access to the database according to the IP address of the local subnet. To change this security restriction, the PostgreSQL `pg-hba.conf` configuration file must be modified (see the *References* section for links to PostgreSQL documentation).

Install the PostgreSQL client

If you are on another Linux computer on the network, you can access the database remotely if a PostgreSQL client is installed. Install the client on Ubuntu using the following commands:

```
sudo apt-get install postgresql-client-common
sudo apt-get install postgresql-client-8.4
```

Connect to the database

When the client is installed, access the database using `psql` and provide your login username:

```
thisuser@mydesktop:~$ psql -h myserver -d iondb -U ion
psql (8.4.7)
SSL connection (cipher: DHE-RSA-AES256-SHA, bits: 256)
Type "help" for help.

iondb=>
```

Verify your database connection

Verify that you are connected to the Torrent Server database by checking the PostgreSQL version, using `psql`:

```
iondb=> select version();

--------------------------

PostgreSQL 8.4.7  ... elided ...
```

The installed PostgreSQL version is displayed with other information about the database.

After you verify that your system can access the database, you can run SQL queries on the database using `psql` or your programming language PostgreSQL API. Continue reading Database Access Examples for specific methods for working with the Ion Torrent database.

Database Access Examples

These examples give a brief demonstration of how to work with the Ion Torrent database. As a prerequisite, you should be familiar with SQL, the `psql` command, and a PostgreSQL API, and be able to easily see how these examples can be expanded to create more useful applications.

Connect to the database using the following command, for example:

```
ionadmin@myserver:~$ psql -U ion -d iondb
psql (8.4.7)
Type "help" for help.

iondb=>
```

End your psql session by entering the quit command:

```
iondb=> \q
```

Examples:

- *List database table elements*
- *Get the value of a table element*
- *Get the value of all elements in a table*
- *Get run and results data*
- *Query the database using a file*

List database table elements

To get a list and brief description of table contents, use the `\d <tableName>` command:

```
iondb-> \d rundb_template

                Table "public.rundb_template"
Column     |         Type          |        Modifiers
-----------+-----------------------+-------------------------
id         | integer               | not null default nextval
name       | character varying(64) | not null
sequence   | text                  | not null
key        | character varying(64) | not null
comments   | text                  | not null
isofficial | boolean               | not null

Indexes:
        "rundb_template_pkey" PRIMARY KEY, btree (id)

(END)
```

This example lists information about the `rundb_results` table elements, including name, datatype, attributes and relationship to other tables.

Get the value of a table element

Use the SQL `select` command to get the data associated with one or more table elements:

```
iondb=> select "experiment_id" from rundb_results;
experiment_id
---------------
                        4
                        3
(2 rows)
```

The example lists the IDs of all experiments stored in the rundb_results table. Here, results data are stored for two experiments. You can further qualify which experiments are listed using the SQL `where` operator, as shown in the *Get run and results data* example.

Get the value of all elements in a table

Use the wildcard (*) character to match all elements in a table:

```
iondb=> select * from rundb_template;
id | name |       sequence     | key  | comments | isofficial
----+------+--------------------+------+----------+-----------
 1 | TF_A | TGTTTCCGTGAGACTAGG | ATCG |          | t
 2 | TF_B | TGAAGCCGTGAGACTGG  | ATCG |          | t
 3 | TF_C | TACGAACGTGAGACTGG  | ATCG |          | t
 4 | TF_D | TTGCGGAAGAGACTAGG  | ATCG |          | t
(4 rows)
```

The example displays the four templates stored in the database, and their attributes.

Get run and results data

There are two interesting tables representing experiments (PGM™ or Proton™ sequencer runs) and results (runs analyses):

- A run has a single record on the rundb_experiment table.
- Each time the analysis pipeline is executed, a record is created in the rundb_results table.

In the rundb_results table, the experiment_id field provides an association between analysis results and a PGM experiment record in the rundb_experiment table. There may be zero, one or multiple results for each experiment. If an experiment has never been analyzed, it will have zero rundb_results records associated with it.

In the rundb_results table, the analysis pipeline updates the status field, indicating a Started, Terminated, ERROR, or Complete condition. Completed means that the analysis pipeline has completed normally and analysis results are available.

A `psql` program SQL query to return only completed analysis results records and associated experiment records has the following form:

```
iondb=> select "experiment_id", "resultsName", "reportLink"
iondb=> from rundb_results where "status" = 'Completed';
```

The `psql` program handles case sensitivity by wrapping text in quotes. If your queries fail unexpectedly, try using quotation marks around field names and tables. At the core of a relational database is the ability to maintain data dependencies. For tables that have links to related data, you can use the ID link to reference the associated data.

Building on the previous simple example, we know there are two experiments in the rundb_results table:

```
iondb=> select "experiment_id" from rundb_results;
experiment_id
---------------
     4
     3
(2 rows)
```

Suppose we want to know the experiment name and sample name associated with the results for experiment_id 3. Again, we use the SQL `select` command, specifying the table elements of interest, but qualifying the query with the experiment ID so only those data associated with that experiment are returned:

```
iondb=> select "expName", "sample"
iondb=> from rundb_experiment where "id"= '3';
                expName            |      sample
-----------------------------------+-------------------
R_2013_06_32_00_user_B6--237 | ms505_xm_indirect
(1 row)
```

The query returns the expName and sample fields for only the record that matched the specified experiment ID. SQL commands can be entered on one or more lines and are terminated with a semicolon.

Query the database using a file

Database queries can be specified in a file and executed by passing the filename to the `psql` program:

```
psql -d iondb -U ion -f test.sql
```

When accessing the database remotely, you must also specify the host:

```
psql -h myhost -d iondb -U ion -f test.sql
```

A `test.sql` file that contains the following SQL commands:

```
select * from rundb_rig;
select * from rundb_location where id = '1';
```

produces the following results:

```
ionadmin@myhost:~/example$ psql -d iondb -U ion -f test.sql
name     | location_id | comments
---------+-------------+----------
PGM_test |           1 |
B6       |           1 |
(2 rows)

id  | name | comments
----+------+----------
1   | Home |
(1 row)
```

The command sequence lists the rigs (PGM™ and Proton™ sequencers) stored in the database and uses the `location_id` element to display information about one of the rigs.

5.2 Torrent Suite™ Software API Cookbook

5.2.1 Introduction

This "cookbook" introduces you to the basic capabilities of the Torrent Suite™ Software API, using the learn-by-doing method.

About the Examples

- Cookbook examples are discussed in snippets to elaborate on important interface details. Refer to Torrent Suite™ Software SDK Source Code Samples for full example listings.
- Where the server name and authorization credentials are shown, the following convention is used:

Entity	**Placeholder**
Server name	myhost
Username	myusername
Password	mypassword

To run the examples, replace these strings with the host name and credentials required for your server.

- You can interactively explore the REST interface using either the cURL command line utility, a REST client, or a web browser. These tools require less infrastructure than program development and providing a more convenient way to learn the interface.
  - cURL
  - Firefox REST client
  - Chrome REST client
  - Generic REST client

  (!) Each example shows the equivalent URI used with these tools before describing the programming language implementation.
- If you run the examples in your browser using either the browser address window or a REST client, you must include the `?format=json` parameter. This is because the browser requests XML-formatted data before JSON-formatted data but the current implementation does not support XML. This requirement does not apply to your programs, although, the programming examples in this document include the format parameter.
- For examples that demonstrate the API using the Python programming language, one of the following REST libraries is used. You may need to modify the example code for your preferred REST library.
  - httplib2
  - restful_lib (deprecated)
  - requests

- Currently, only the JSON data format is supported. The examples use the simplejson library to encode and decode JSON data into Python objects.
- JavaScript examples use the jQuery framework.

Before Reading This Document

To facilitate your learning the API, we recommend that you run the examples as you work through the cookbook. You can install the necessary programming languages and libraries and run the examples as shown, or modify the examples for your particular language and programming environment. These simple examples are shown using an easy-to-read programming language, requiring minimal glue logic, so they should be easily portable to other languages.

To run the examples or to interactively work with the API using cURL or a REST client, you need a Torrent Server available. Remember to change the server name and credentials shown in the example to those required by your server.

Writing applications using the REST API involves basic web programming. You should have some previous experience developing service-oriented web applications.

You may find the documentation in the following list helpful to provide more insight into the topics presented in the cookbook. This content might prove most useful when you extend the examples and create your own applications:

- The cookbook is intended to provide only the essential information need to get started developing applications. To this end, the examples typically involve resources having a small number of data fields to reduce the amount of non-essential information in the presentation. Refer to the Torrent Suite™ Software Database Tables for a complete, detailed description of each resource. Notice that in the current API version, not all of the resources are exposed by the API.
- For a complete description of the API syntax and functionality for all resources, refer to the API references tables document Torrent Suite™ Software API Reference.
- Once you have gained a basic familiarity with API programming, use the Torrent Suite™ Software API Quick Reference to help recall details about the API that may take some time to remember.

API topics are presented in the form of examples that are "recipes" for mastering each of the various topics. The examples are arranged in order, from the simplest "hello, world" type of application to increasingly complex functionality, where each depends on the understanding gained in previous examples. Each example demonstrates a real-world application that can be easily extended. From the basic operations involved in retrieving data through selecting and sorting data and, finally, updating data and creating new resource objects, you gain the knowledge needed to begin your own application development.

See the API reference tables for the full syntax needed to extend the applications presented in this document:

- Torrent Suite™ Software REST API v1 Resources

Connect with the Server

To connect to a resource, you first authenticate with the server.

Topics on this page:

The connection and authentication is currently as simple as logging into the server and providing your username and password.

The following examples show:

- The general form of authentication using a browser or REST client. You are prompted for your username and password, if they are not provided in the request.
- The cURL command line form.
- Programmatic methods using various Python libraries, PHP, and JavaScript.

General form

```
http://myusername:mypassword@myhost/rundb/api/v1/experiment
```

cURL command

```
curl --user mysername:mypassword
     --header "Content-Type: application/json"
     --location 'http://myhost/rundb/api/v1/experiment'
```

Programatically

Python libraries *restful_lib*

NOTE: restful_lib has not been updated in over 5 years and is considered deprecated.

```
from restful_lib import Connection
base_url = 'http://myhost/rundb/api/v1'
conn = Connection(base_url, username="myusername", password="mypassword")
```

httplib2

```
import httplib2
h = httplib2.Http()
h.add_credentials('myusername', 'mypassword')
```

requests (recommended)

```
import requests
resp = requests.get('http://myhost/rundb/api/v1?format=json', auth=('myusername', 'mypassword'))
```

PHP

```
<?php
$context = stream_context_create(array(
'http' => array(
'header'  =>
    "Authorization: Basic " . base64_encode("myusername:mypassword")
)
));

$url = "http://myhost/rundb/api/v1?format=json";
$feed = file_get_contents($url, false, $context);
?>
```

JavaScript jQuery AJAX call

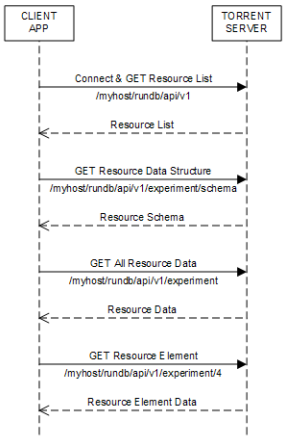
```
$.ajax({
    url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment",
    dataType: 'jsonp',
    success: handleResponse(json_results)
});
```

Cookbook - Your First Request and More

Read this topic for a hands-on introduction to using the REST API. Through a logical progression, using Python examples, you learn the fundamentals of how to make a REST call and how to use the API to access and manipulate a particular data element.

Before you access a resource, you need to get the list of resources exposed by the API, and the mechanism for accessing your particular resource of interest. At each step, you use the API to traverse the relational database or functional component (file or plugin), to obtain your specific data item of interest.

The following diagram shows the request message sequence for to discover a particular data element:



1. Begin with the highest-level request, /myhost/rundb/api/v1, without specifying a resource so all supported resources are returned. The first request sent to the server always requires authentication, which is your username and password.
2. To find the data fields contained by a resource, request the resource schema: /myhost/rundb/api/v1/experiment/schema. You can use these fields to qualify your searches for specific resource elements or data sets.
3. To find all of the resource elements, or objects, send a request that includes only the resource name, or list_endpoint: /myhost/rundb/api/v1/experiment.
4. Once you locate the particular resource element of interest, pass the primary key for the resource, usually the id field, to retrieve only the data for that element: /myhost/rundb/api/v1/experiment/4.

The rest of this document shows how to build on this basic message sequence to refine your searches.

> (!) The interface mechanism demonstrated here is the same for all REST operations, and subsequent more involved examples differ only in particular resource and access request parameters.

Get the list of resources

URIs The REST interface uses a Uniform Resource Identifier (URI) to name and locate a resource. This is the same as an address you commonly type in your browser to retrieve a Web page.

For example:

```
http://myhost/rundb/api/v1/experiment
```

You can see that REST uses the same HTTP protocol as the Web and the rest of the address specifies the location of your resource. This example requests the experiment resource, which is located on the host named myhost in the resource directory /rundb/api/v1, where v1 is the API version.

List available resources If you do not already know what the API name for the resource containing your data item is, you need to get a list of resources:

```
http://myhost/rundb/api/v1?format=json
```

This is the basic URI without specifying a resource after the API version.:

```
(!) The format=json parameter name:value pair is appended to the request to specify the format, JavaS
```

Enter the URI in your browser window or REST client, replacing myhost with your host name. This step also verifies connectivity before continuing with the tutorial. If you are using a REST client and the request is successful, an HTTP status code of 200 is returned. Otherwise, the request failed.

The examples in this tutorial use the Python programming language, because the API is intended to be used programmatically and because Python has both a low barrier to entry and also syntax similar to numerous other commonly used languages.

Also, the examples depend on the json and requests libraries (note that restful_lib is depreciated), so each example assumes the following statements are included:

```
import json
import requests
```

You can use any equivalent libraries and modify the code snippets as needed.

Now, you can programmatically make the same request for a list of resources using the following code snippet:

```
resp = requests.get('http://myhost/rundb/api/v1, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

(!) The examples in this tutorial do not include error handling and assume an HTTP status code of 2xx is always returned.

- On the first API call, it is necessary to connect to the server, providing basepath and authentication parameters. (To run the example, replace myhost with your host name and replace myusername and mypassword with your username and password.)

The connection request returns a connection handle, which is used for subsequent requests.

- The second statement performs a simple GET request, without specifying a resource. The response from the server is returned in the result variable:


```
{
        "account": {
                "list_endpoint": "/rundb/api/v1/account/",
                "schema": "/rundb/api/v1/account/schema/"
        },
        "activeionchefprepkitinfo": {
                "list_endpoint": "/rundb/api/v1/activeionchefprepkitinfo/",
                "schema": "/rundb/api/v1/activeionchefprepkitinfo/schema/"
```

```
},
"activelibrarykitinfo": {
        "list_endpoint": "/rundb/api/v1/activelibrarykitinfo/",
        "schema": "/rundb/api/v1/activelibrarykitinfo/schema/"
},
"activepgmlibrarykitinfo": {
        "list_endpoint": "/rundb/api/v1/activepgmlibrarykitinfo/",
        "schema": "/rundb/api/v1/activepgmlibrarykitinfo/schema/"
},
"activepgmsequencingkitinfo": {
        "list_endpoint": "/rundb/api/v1/activepgmsequencingkitinfo/",
        "schema": "/rundb/api/v1/activepgmsequencingkitinfo/schema/"
},
"activeprotonlibrarykitinfo": {
        "list_endpoint": "/rundb/api/v1/activeprotonlibrarykitinfo/",
        "schema": "/rundb/api/v1/activeprotonlibrarykitinfo/schema/"
},
"activeprotonsequencingkitinfo": {
        "list_endpoint": "/rundb/api/v1/activeprotonsequencingkitinfo/",
        "schema": "/rundb/api/v1/activeprotonsequencingkitinfo/schema/"
},
"activesequencingkitinfo": {
        "list_endpoint": "/rundb/api/v1/activesequencingkitinfo/",
        "schema": "/rundb/api/v1/activesequencingkitinfo/schema/"
},
"analysismetrics": {
        "list_endpoint": "/rundb/api/v1/analysismetrics/",
        "schema": "/rundb/api/v1/analysismetrics/schema/"
},
"applproduct": {
        "list_endpoint": "/rundb/api/v1/applproduct/",
        "schema": "/rundb/api/v1/applproduct/schema/"
},
"availableionchefplannedexperiment": {
        "list_endpoint": "/rundb/api/v1/availableionchefplannedexperiment/",
        "schema": "/rundb/api/v1/availableionchefplannedexperiment/schema/"
},
"availableionchefplannedexperimentsummary": {
        "list_endpoint": "/rundb/api/v1/availableionchefplannedexperimentsummary/",
        "schema": "/rundb/api/v1/availableionchefplannedexperimentsummary/schema/"
},
"availableonetouchplannedexperiment": {
        "list_endpoint": "/rundb/api/v1/availableonetouchplannedexperiment/",
        "schema": "/rundb/api/v1/availableonetouchplannedexperiment/schema/"
},
"availableonetouchplannedexperimentsummary": {
        "list_endpoint": "/rundb/api/v1/availableonetouchplannedexperimentsummary/",
        "schema": "/rundb/api/v1/availableonetouchplannedexperimentsummary/schema/"
},
"availableplannedexperimentsummary": {
        "list_endpoint": "/rundb/api/v1/availableplannedexperimentsummary/",
        "schema": "/rundb/api/v1/availableplannedexperimentsummary/schema/"
},
"chip": {
        "list_endpoint": "/rundb/api/v1/chip/",
        "schema": "/rundb/api/v1/chip/schema/"
},
"compositedatamanagement": {
```

```
        "list_endpoint": "/rundb/api/v1/compositedatamanagement/",
        "schema": "/rundb/api/v1/compositedatamanagement/schema/"
},
"compositeexperiment": {
        "list_endpoint": "/rundb/api/v1/compositeexperiment/",
        "schema": "/rundb/api/v1/compositeexperiment/schema/"
},
"compositeresult": {
        "list_endpoint": "/rundb/api/v1/compositeresult/",
        "schema": "/rundb/api/v1/compositeresult/schema/"
},
"content": {
        "list_endpoint": "/rundb/api/v1/content/",
        "schema": "/rundb/api/v1/content/schema/"
},
"contentupload": {
        "list_endpoint": "/rundb/api/v1/contentupload/",
        "schema": "/rundb/api/v1/contentupload/schema/"
},
"datamanagementhistory": {
        "list_endpoint": "/rundb/api/v1/datamanagementhistory/",
        "schema": "/rundb/api/v1/datamanagementhistory/schema/"
},
"dnabarcode": {
        "list_endpoint": "/rundb/api/v1/dnabarcode/",
        "schema": "/rundb/api/v1/dnabarcode/schema/"
},
"emailaddress": {
        "list_endpoint": "/rundb/api/v1/emailaddress/",
        "schema": "/rundb/api/v1/emailaddress/schema/"
},
"eventlog": {
        "list_endpoint": "/rundb/api/v1/eventlog/",
        "schema": "/rundb/api/v1/eventlog/schema/"
},
"experiment": {
        "list_endpoint": "/rundb/api/v1/experiment/",
        "schema": "/rundb/api/v1/experiment/schema/"
},
"experimentanalysissettings": {
        "list_endpoint": "/rundb/api/v1/experimentanalysissettings/",
        "schema": "/rundb/api/v1/experimentanalysissettings/schema/"
},
"fileserver": {
        "list_endpoint": "/rundb/api/v1/fileserver/",
        "schema": "/rundb/api/v1/fileserver/schema/"
},
"globalconfig": {
        "list_endpoint": "/rundb/api/v1/globalconfig/",
        "schema": "/rundb/api/v1/globalconfig/schema/"
},
"ionchefplantemplate": {
        "list_endpoint": "/rundb/api/v1/ionchefplantemplate/",
        "schema": "/rundb/api/v1/ionchefplantemplate/schema/"
},
"ionchefplantemplatesummary":{
        "list_endpoint": "/rundb/api/v1/ionchefplantemplatesummary/",
        "schema": "/rundb/api/v1/ionchefplantemplatesummary/schema/"
```

```
},
"ionchefprepkitinfo": {
        "list_endpoint": "/rundb/api/v1/ionchefprepkitinfo/",
        "schema": "/rundb/api/v1/ionchefprepkitinfo/schema/"
},
"ionreporter": {
        "list_endpoint": "/rundb/api/v1/ionreporter/",
        "schema": "/rundb/api/v1/ionreporter/schema/"
}, "kitinfo": {
        "list_endpoint": "/rundb/api/v1/kitinfo/",
        "schema": "/rundb/api/v1/kitinfo/schema/"
},
"kitpart": {
        "list_endpoint": "/rundb/api/v1/kitpart/",
        "schema": "/rundb/api/v1/kitpart/schema/"
},
"libmetrics": {
        "list_endpoint": "/rundb/api/v1/libmetrics/",
        "schema": "/rundb/api/v1/libmetrics/schema/"
},
"librarykey": {
        "list_endpoint": "/rundb/api/v1/librarykey/",
        "schema": "/rundb/api/v1/librarykey/schema/"
},
"librarykitinfo": {
        "list_endpoint": "/rundb/api/v1/librarykitinfo/",
        "schema": "/rundb/api/v1/librarykitinfo/schema/"
},
"librarykitpart": {
        "list_endpoint": "/rundb/api/v1/librarykitpart/",
        "schema": "/rundb/api/v1/librarykitpart/schema/"
},
"location": {
        "list_endpoint": "/rundb/api/v1/location/",
        "schema": "/rundb/api/v1/location/schema/"
},
"log": {
        "list_endpoint": "/rundb/api/v1/log/",
        "schema": "/rundb/api/v1/log/schema/"
},
"message": {
        "list_endpoint": "/rundb/api/v1/message/",
        "schema": "/rundb/api/v1/message/schema/"
},
"monitorexperiment": {
        "list_endpoint": "/rundb/api/v1/monitorexperiment/",
        "schema": "/rundb/api/v1/monitorexperiment/schema/"
},
"obsoletereferencegenome": {
        "list_endpoint": "/rundb/api/v1/obsoletereferencegenome/",
        "schema": "/rundb/api/v1/obsoletereferencegenome/schema/"
},
"onetouchplantemplate": {
        "list_endpoint": "/rundb/api/v1/onetouchplantemplate/",
        "schema": "/rundb/api/v1/onetouchplantemplate/schema/"
},
"onetouchplantemplatesummary": {
        "list_endpoint": "/rundb/api/v1/onetouchplantemplatesummary/",
```

```
        "schema": "/rundb/api/v1/onetouchplantemplatesummary/schema/"
    },
    "plannedexperiment": {
        "list_endpoint": "/rundb/api/v1/plannedexperiment/",
        "schema": "/rundb/api/v1/plannedexperiment/schema/"
    },
    "plannedexperimentdb": {
        "list_endpoint": "/rundb/api/v1/plannedexperimentdb/",
        "schema": "/rundb/api/v1/plannedexperimentdb/schema/"
    },
    "plannedexperimentqc": {
        "list_endpoint": "/rundb/api/v1/plannedexperimentqc/",
        "schema": "/rundb/api/v1/plannedexperimentqc/schema/"
    },
    "plantemplatesummary": {
        "list_endpoint": "/rundb/api/v1/plantemplatesummary/",
        "schema": "/rundb/api/v1/plantemplatesummary/schema/"
    },
    "plugin": {
        "list_endpoint": "/rundb/api/v1/plugin/",
        "schema": "/rundb/api/v1/plugin/schema/"
    },
    "pluginresult": {
        "list_endpoint": "/rundb/api/v1/pluginresult/",
        "schema": "/rundb/api/v1/pluginresult/schema/"
    },
    "project": {
        "list_endpoint": "/rundb/api/v1/project/",
        "schema": "/rundb/api/v1/project/schema/"
    },
    "publisher": {
        "list_endpoint": "/rundb/api/v1/publisher/",
        "schema": "/rundb/api/v1/publisher/schema/"
    },
    "qctype": {
        "list_endpoint": "/rundb/api/v1/qctype/",
        "schema": "/rundb/api/v1/qctype/schema/"
    },
    "qualitymetrics": {
        "list_endpoint": "/rundb/api/v1/qualitymetrics/",
        "schema": "/rundb/api/v1/qualitymetrics/schema/"
    },
    "referencegenome": {
        "list_endpoint": "/rundb/api/v1/referencegenome/",
        "schema": "/rundb/api/v1/referencegenome/schema/"
    },
    "results": {
        "list_endpoint": "/rundb/api/v1/results/",
        "schema": "/rundb/api/v1/results/schema/"
    },
    "rig": {
        "list_endpoint": "/rundb/api/v1/rig/",
        "schema": "/rundb/api/v1/rig/schema/"
    },
    "runtype": {
        "list_endpoint": "/rundb/api/v1/runtype/",
        "schema": "/rundb/api/v1/runtype/schema/"
    },
```

```
        "sample": {
                "list_endpoint": "/rundb/api/v1/sample/",
                "schema": "/rundb/api/v1/sample/schema/"
        },
        "sequencingkitinfo": {
                "list_endpoint": "/rundb/api/v1/sequencingkitinfo/",
                "schema": "/rundb/api/v1/sequencingkitinfo/schema/"
        },
        "sequencingkitpart": {
                "list_endpoint": "/rundb/api/v1/sequencingkitpart/",
                "schema": "/rundb/api/v1/sequencingkitpart/schema/"
        },
        "template": {
                "list_endpoint": "/rundb/api/v1/template/",
                "schema": "/rundb/api/v1/template/schema/"
        },
        "tfmetrics": {
                "list_endpoint": "/rundb/api/v1/tfmetrics/",
                "schema": "/rundb/api/v1/tfmetrics/schema/"
        },
        "threeprimeadapter": {
                "list_endpoint": "/rundb/api/v1/threeprimeadapter/",
                "schema": "/rundb/api/v1/threeprimeadapter/schema/"
        },
        "torrentsuite": {
                "list_endpoint": "/rundb/api/v1/torrentsuite/",
                "schema": "/rundb/api/v1/torrentsuite/schema/"
        },
        "user": {
                "list_endpoint": "/rundb/api/v1/user/",
                "schema": "/rundb/api/v1/user/schema/"
        }
}
```


You now have a listing of all resources available through the REST API. Notice that each resource is described by a list_endpoint and a schema, which are partial URIs. Use these URIs in subsequent calls to read and write resource data.

Get the resource data structure

Use the schema URI to get the resource data structure, which limits the names and fields of all resource data elements.

Example:

```
http://myhost/rundb/api/v1/experiment/schema?format=json
```

Get a list of experiments

This section shows how to get the experiment resource data. This example uses the experiment resource, but the experiment field in the URI could be replaced by any resource name.

> (!) By default, a maximum of 20 resource objects are returned. Add the limit parameter, as shown, to return all objects for a resource (for some resources, this may result in a large amount of data):

This request uses the experiment `list_endpoint` URI and has the following general form:

```
http://myhost/rundb/api/v1/experiment?format=json&limit=0
```

These steps show how to get experiment resource data programmatically:

1. Connect to the resource.
2. Post a request for data using the GET method.

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

In the code snippet, a base_url variable is used to define the part of the URI that is common for all requests. For the experiment resource, the following example data are returned:


```
{
    "meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
    },
    "objects": [
        {
                "autoAnalyze": true,
                "baselineRun": false,
                "chipBarcode": "",
                "chipType": "318",
                "cycles": 0,
                "date": "2013-02-08T21:00:52.000449+00:00",
                "diskusage": 0,
                "displayName": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
                "eas_set": [
                        {
                                "barcodeKitName": "",
                                "barcodedSamples": {},
                                "date": "2012-12-15T00:53:29.000428+00:00",
                                "experiment": "/rundb/api/v1/experiment/7/",
                                "hotSpotRegionBedFile": "",
                                "id": 3,
                                "isDuplicateReads": false,
                                "isEditable": true,
                                "isOneTimeOverride": false,
                                "libraryKey": "TCAG",
                                "libraryKitBarcode": null,
                                "libraryKitName": "Ion Xpress Plus Fragment Library Kit",
                                "reference": "",
                                "resource_uri": "/rundb/api/v1/experimentanalysissettings/3/",
                                "results": [
                                        "/rundb/api/v1/results/55/",
                                        "/rundb/api/v1/results/26/"
                                ],
                                "selectedPlugins": {
                                        "IonReporterUploader": {"features": ["export"], "id": 167, "
                                        "SFFCreator": {"features": [], "id": 157, "name": "SFFCreato
                                        "variantCaller": {"features": [], "id": 165, "name": "variant
                                "status": "planned",
                                "targetRegionBedFile": "",
```

```
                        "threePrimeAdapter": "ATCACCGACTGCCCATAGAGAGGCTGAGAC"
                }
        ],
        "expCompInfo": "",
        "expDir": "",
        "expName": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
        "flows": 500,
        "flowsInOrder": "",
        "ftpStatus": "Complete",
        "id": 7,
        "isReverseRun": false,
        "log": { ... },
        "metaData": {},
        "notes": "",
        "pgmName": "",
        "plan": "/rundb/api/v1/plannedexperiment/41/",
        "rawdatastyle": "single",
        "reagentBarcode": "",
        "resource_uri": "/rundb/api/v1/experiment/7/",
        "resultDate": "2013-02-08T21:00:52.000450+00:00",
        "results": [],
        "reverse_primer": null,
        "runMode": "single",
        "runtype": "GENS",
        "sample": "Example_2",
        "samples": [
                {"date": "2012-12-15T00:53:29.000428+00:00",
                "description": null,
                "displayedName": "Example_2",
                "experiments": ["/rundb/api/v1/experiment/7/"],
                "externalId": null,
                "id": 2,
                "name": "Example_2",
                "resource_uri": "/rundb/api/v1/sample/2/",
                "status": "planned"}
        ],
        "seqKitBarcode": "",
        "sequencekitbarcode": "",
        "sequencekitname": "IonPGM200Kit",
        "star": false,
        "status": "planned",
        "storageHost": null,
        "storage_options": "A",
        "unique": "5c2f8551-ac03-4c99-a9a7-83a5c0b41842",
        "usePreBeadfind": false,
        "user_ack": "U"
    } ]
}
```


`meta` field The `meta` field contains data about the object data. The metadata of interest for the experiment resource is that the resource currently contains 1 experiment.

`object` field The `object` field is a list containing actual experiment data, or properties. Two elements are listed, which is also indicated by the metadata `total_count` field.

Refer to the database schema for a description of each data item.

Notice that the results data item is another URI list, containing the locations of results data for the experiment.

Get data for a specific experiment

You can get the data for a specific experiment by specifying the experiment resource primary key value in the URI, for the desired experiment. For most resources, the primary key is the id field. The exception is the rig resource, which has the name field as the primary key.

A request for the experiment whose id field is 4 has the following form:

General form

```
http://myhost/rundb/api/v1/experiment/4?format=json
```

Python snippet

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/4?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

The response includes the data for the single experiment whose id is 4. The output is similar to the output shown in *Get a list of experiments*.

Alternate method Using the primary key, you can request data simply by encoding the key value in the main part of the URI (as seen in the previous examples). If instead only one of the field properties of a resource is known, you can use an alternative method of requesting a particular resource element by passing a filtering parameter in the request.

In the following example, instead of specifying a primary key value, the experiment name field, expName, is passed as a parameter to get the same experiment resource element:

General form

```
http://myhost/rundb/api/v1/experiment?format=json&expName=5c2f8551-ac03-4c99-a9a7-83a5c0b41842
```

Python snippet

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName=5c2f8551-ac03-4c99-a9a7-83a5c0b4184'%base_url,
resp_json = resp.json()
```

Try both of these requests to verify that the same experiment data are returned.

Next

This section assumed that no errors occurred in the request-response transaction between your application and the server. In the next section, you learn about the kinds of errors that can occur as you develop more complex applications.

This section finished with a brief mention of filtering as a way of selecting a particular resource element. Following sections show the flexibility provided by filtering in selecting single or groups of resource elements.

Building on selecting the desired resource elements returned by the server, you also learn how to order, or sort, results.

Handle Errors Reported by the Server

Now that you have successfully sent API requests and processed the responses, the topic of error handling is introduced to help overcome problems that might be introduced with more complex applications.

API errors can be classified as either communication or internal server processing errors:

- Communication errors can be caused by connectivity issues, failure to authenticate or message protocol errors, which are listed in section 6 of the RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 standard.
- Internal processing errors are those detected by the Django framework and include software bugs, database anomalies and invalid request formats.

The *HTTP Response Codes* section of the *Torrent Server REST API v1 Resources* document lists errors that might typically occur when using the Torrent Suite™ Software API, and possible causes.

Topics on this page:

- *HTTP status codes*
- *Debug internal errors*

Errors detected by the server are reported in the status code returned with any create, read, update or delete request. To help determine the cause of the error, the returned message body contains additional information about the error. For internal, 500-series server errors, in particular, Django returns very detailed information about the error cause and location. Debugging must be enable to receive the more extensive Django error reports.

Using a REST client with your browser to interactively exercise the API provides an easy way to investigate error conditions, by examining the status code and message body returned with the request. When writing an application program, it is helpful to include exception handling around API calls to catch possible errors, and to always check the returned status code before continuing to process a response.

HTTP status codes

Successful API requests return a status code of 200 or 201.

All other status codes indicate some kind of error condition, and after some experience using HTTP the cause of the error can often readily be determined. To demonstrate an error condition, the following example omits the question mark (?) symbol preceding request parameters, effectively making a request on an undefined resource:

```
http://myhost/rundb/api/v1/rigformat=json
```

If you try sending this request, you will see that the server returns a 404 status code, indicating the resource was Not Found. Additionally, the response message body contains a server-specific HTML page for the 404-type error.

Debug internal errors

Turn on debugging to receive additional detailed information in the response message body for internal, 500-series errors, which are detected by the Django framework:

1. On your server, open the `settings.py` file for editing, found at the following location:

```
/opt/ion/iondb/settings.py
```

2. Set the DEBUG environment variable to True.

```
DEBUG = True
```

3. Restart Apache:

```
sudo /etc/init.d/apache2 restart
```

Thereafter, whenever a 500-series error occurs, a message similar to the following example is provided in the response message with detailed information about the type of error and the source code location where the error was detected:

```
<Response [500]>
{"error_message": "The format indicated 'application/x-www-form-urlencoded' had no available deserial
```

Filter and Sort

You can select a particular element or a group of resource elements by specifying filtering criteria. A filter may specify an exact match or a partial match using a filter qualifier.

Query results can be sorted in either ascending or descending order, using the order_by parameter and specifying the field on which to sort.

Select a Subset of Resources

All resource elements If you specify only the resource in the URI, all of the resource elements are returned. For example:

```
http://myhost/rundb/api/v1/dnabarcode/?format=json
```

A single resource element Similarly, you can select a specific resource element by providing the primary key value of the element, usually the id field:

```
http://myhost/rundb/api/v1/dnabarcode/34?format=json
```

Multiple resource elements To request multiple elements, use the set keyword following the resource name in the URI, then separate each desired element using a semicolon:

```
http://myhost/rundb/api/v1/dnabarcode/set/34;35?format=json
```

This example returns only elements with id 34 and 35.

Basic Filters

Topics on this page:

Get the resource schema and filter list When you request the resource schema, the response includes a filtering field, which is a dictionary of fields you can filter on.

Filters are used in subsequent requests by adding the filter as a request parameter and assigning the filter a value, and possibly a value qualifier. All elements that match the filter criteria are returned for the request.

General form of a schema request

```
http://myhost/rundb/api/v1/location/schema?format=json
```

Python implementation

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location/schema?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Schema request response

```
{
        "default_format": "application/json",
        "fields": {
        "comments": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "id": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "name": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": false,
                                "type": "string"
                        },
        "resource_uri": {
                                "help_text": "Unicode string data. Ex: \"Hello World\"",
                                "nullable": false,
                                "readonly": true,
                                "type": "string"
                        }
        },
        "filtering": {
                        "backupconfig": 2,
                        "comments": 2,
                        "cruncher": 2,
                        "fileserver": 2,
                        "id": 2,
                        "name": 2,
                        "rig": 2
                },
        "ordering": \[
        "backupconfig",
        "comments",
        "cruncher",
        "fileserver",
        "id",
```

```
        "name",
        "rig"
        \]
}
```

Select by filter value The location resource is used here as an example, where location contains two elements:

```
"objects": [
{
        "comments": "",
        "id": "1",
        "name": "Home",
"resource_uri": "/rundb/api/v1/location/1/"
},
{
        "comments": "Test comment.",
        "id": "2",
        "name": "testDir",
        "resource_uri": "/rundb/api/v1/location/2/"
}
]
```

Using the name field, a valid filter according to the schema, a request is made to get all elements matching the value (Home) assigned to the name parameter. Only one element is expected to match.

General form of a URI with a filter parameter

```
http://myhost/rundb/api/v1/location?format=json&name=Home
```

Python implementation of a request with a filter parameter

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location?formatjson&name=Home'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Only one location element is returned, having a name field with a value of Home:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 1
        },
        "objects": [
                {
                        "comments": "",
                        "id": "1",
                        "name": "Home",
                        "resource_uri": "/rundb/api/v1/location/1/"
                }
        ]
}
```

Specify multiple filters You can use more than one filter to select resource elements by using multiple request parameters.

General form to specify multiple filters

```
http://myhost/rundb/api/v1/experiment?format=json&cycles=0&rawdatastyle=single
```

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&cycles=0&rawdatastyle=single'%base_url,
        auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

The request to return only those elements whose cycles are 0 and whose rawdatastyle is single returns a single element:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 1
        },
        ...
}
```

Non-matching filter response Where no resource elements match your filter criteria, an empty object list is returned.

For multiple filters, all filters must match.

The following example is similar to the previous one, except that the comments fiilter is assigned a value of Test.

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/location?format=json&name=Nothing could possibly have this name'%base_url,
        auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

No elements match both filter values so no elements are returned for the request, confirmed by "total_count": 0.

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 0
        },
        "objects": [ ]
}
```

Qualifying Filters

Topics on this page:

Select all experiments with a common `expName` field Filter values can be qualified so the value does not need to be an exact match to select an element.

> (!) The full set of filter qualifiers is listed in the *Filter Qualifiers* section of the *Torrent Suite™ Software API User Guide*.

Use the following syntax to specify a filter qualifier, where two underscore characters (__) separate the filter name from the filter qualifier name:

```
<filterName>__<filterQualifierName>=<value>
```

For some qualifiers, the behavior is similar to using a wildcard. The names of most qualifiers is self-explanatory, describing how it matches on a value.

In the following example, the `startswith` qualifier is used so any element whose field value "starts with" the specified value is returned, for the specified field.

General form of a URI request with a filter qualifier

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013
```

Python implementation of a filter qualifier

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName__startswith=R_2013'%base_url, auth=('myusernar
resp_json = resp.json()
```

The example returns two elements whose experiment name, expName, starts with R_2013.


```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
```

```
                .
                .
            "unique": "/results/PGM_test/sample",
            "usePreBeadfind": true
        },
        {
            "autoAnalyze": true,
            "barcodeId": "",
            "baselineRun": false,
            "chipBarcode": "AA0011641",
            "chipType": "\"314R\"",
            "cycles": 55,
            "date": "2013-11-05T18:32:00",
            "expCompInfo": "",
            "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
            "expName": "R_2013_11_05_18_32_00_user_B6--237",
                .
                .
                .
            "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
            "usePreBeadfind": true
        }
    ]
}
```

A more restrictive example This example is the same as the previous example, except that the stricter criteria are applied by specifying that the experiment name must start with R_2013_11_05. From the results of the previous example, you can see that only one element is expected to meet this qualification.

General form of a more restrictive filter qualifier

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013_11_05
```

Python implementation of a stricter filter qualifier

```
import json
import requests
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&expName__startswith=R_2013_11_05'%base_url, auth=('myu
resp_json = resp.json()
```

The response shows that only one element matches the `expName` filter:

```
{
    "meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
    },
    "objects": [
        {
            "autoAnalyze": true,
            "barcodeId": "",
```

```
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                }
        ]
}
```


Select experiments with a given date field The filter qualifier shown in this example works, as most qualifiers do, similar to the previous examples. Here, instead of searching for an element that "starts with" a particular value, you are searching for elements that "contain" a particular value. This example looks for elements whose date field contains the string value 2013-03.

General form of a URI with a filter qualifier on the date field

```
http://myhost/rundb/api/v1/experiment?format=json&date__icontains=2013-03
```

Python implementation of applying a filter qualifier on the date field

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&date__icontains=2013-03'%base_url,
        auth=('myusername', 'mypassword'))
```

One experiment is returned whose date field contains the string `2013-03`. Notice that the `startswith` filter qualifier could also have been used. Considerable flexibility is available to you in choosing a qualifier and the best choice depends on the application and the data set.


```
{
"meta": {
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
},
"objects": [
        {
                "autoAnalyze": true,
                "barcodeId": "",
                "baselineRun": false,
                "chipBarcode": "AA0000000",
                "chipType": "\"314R\"",
                "cycles": 27,
                "date": "2011-03-07T17:48:53",
```

```
                "expCompInfo": "",
                "expDir": "/results/PGM_test/sample",
                "expName": "R_2010_11_08_22_30_04_user_B15-45",
                        .
                        .
                        .
                "unique": "/results/PGM_test/sample",
                "usePreBeadfind": true
        }]
}
```

Sort Response Output demonstrates how to sort the returned experiment data by date.

Sort Response Output

Topics on this page:

- *Sort by date*
- *Sort in reverse order*

Sort by date To sort multiple elements, add a sort parameter to your request. Otherwise, elements are returned in the order they occur in the database.

Sorting is specified by using the keyword order_by, which works the same way as the SQL ordering statement. You must also specify the field you want to sort on. The sort parameter has the following syntax:

```
order_by=<field>
```

You can request sorting in ascending or descending alphanumeric order, as these example will demonstrate.

> (!) Elements with, for example, field values of 1, 2, 10 are returned in 1, 10, 2 order.

The first example requests elements to be sorted by the date field. This is the default form of the order_by parameter and returns elements in ascending order. (You should already be familiar with the startswith filter qualifier used in previous examples.)

General form of a sort request

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013&order_by=date
```

Python implementation of a sort request

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&order_by=date'%base_url, auth=('myusername', 'mypassw
resp_json = resp.json()
```

Two matching elements are returned, sorted in ascending order, by date:

```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
```

```
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                },
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
                                .
                                .
                        "unique": "/results/PGM_test/sample",
                        "usePreBeadfind": true
                }
        ]
}
```


Sort in reverse order To reverse the order of the elements returned in the previous example, add a minus symbol before the name of the field you are sorting on. This returns elements in descending order, for the specified field.

General form of a descending-order request

```
http://myhost/rundb/api/v1/experiment?format=json&expName__startswith=R_2013&order_by=-date
```

Python implementation of a descending-order request

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment?format=json&order_by=-date'%base_url, auth=('myusername', 'mypassw
resp_json = resp.json()
```

You can see that the elements are returned in inverse order of the previous example:


```
{
        "meta": {
                "limit": 20,
                "next": null,
                "offset": 0,
                "previous": null,
                "total_count": 2
        },
        "objects": [
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0000000",
                        "chipType": "\"314R\"",
                        "cycles": 27,
                        "date": "2013-03-07T17:48:53",
                        "expCompInfo": "",
                        "expDir": "/results/PGM_test/sample",
                        "expName": "R_2013_11_08_22_30_04_user_B15-45",
                                .
                                .
                                .
                        "unique": "/results/PGM_test/sample",
                        "usePreBeadfind": true
                },
                {
                        "autoAnalyze": true,
                        "barcodeId": "",
                        "baselineRun": false,
                        "chipBarcode": "AA0011641",
                        "chipType": "\"314R\"",
                        "cycles": 55,
                        "date": "2013-11-05T18:32:00",
                        "expCompInfo": "",
                        "expDir": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "expName": "R_2013_11_05_18_32_00_user_B6--237",
                                .
                                .
                                .
                        "unique": "/results/B6/R_2013_11_05_18_32_00_user_B6--237",
                        "usePreBeadfind": true
                }
        ]
}
```


The selection semantics are the same as those of the SQL SELECT statement, where a selection filter equates to the SELECT column name parameter. The API provides that any resource field, with the exception of the resource_uri, can be used as a filter. This gives considerable flexibility in selecting only the desired resource elements. Filters are specified as URI parameters, a filter=value pair, and any resource elements matching the filter criteria are returned in the response. Filter values can be further qualified using certain keywords that act as wildcards or logical operators.

Any of the fields in the resource schema ordering list can be used to sort responses. Ordering is alphanumeric, so elements with, for example, name field values of 1, 2, 10 are returned in the order 1, 10, 2.

You can use filters to retrieve metadata and analysis metrics for runs, for instance, with a given project name or genome name, or within a specific date range.

Work with the Database

Most REST operations involve reading data from the database or updating the database with data from your own applications. The following example applications use more advanced programming procedures than those already presented:

Get Run Metadata and Metrics

This section describes a more involved programming example that begins to approach an actual application. From the previous sections, you gained the basic knowledge needed to begin to write simple applications.

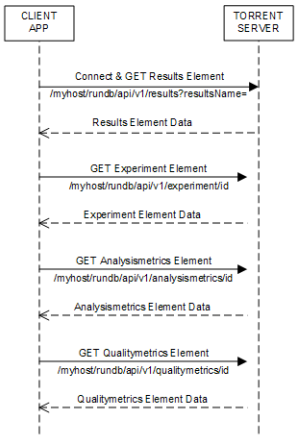
This example gets basic information about a particular run by requesting the results resource element for the run. To demonstrate getting data using links, the run gets experiment data and metrics by traversing results links to the experiment, analysismetrics and qualitymetrics resources. The program outputs experiment metadata and run metrics.

Use the following command to run the program in your Python environment:

```
getruninfo.py <runName>
```

Example: `getruninfo.py Auto_B6--237_3`

The following sequence diagram shows the request-response flow:



The program

(!) You can view the full source code at Torrent Suite API Source Code Samples.

The example uses the Python libraries requests to make REST requests to the server and simplejson to parse JSON data into Python objects. You can modify the program as needed for your libraries.

```
import requests
import simplejson as json
import sys
```

Handle the command line and save the run name input parameter. The run name is used in formatting the request to the server to return results for the particular run.

```
if len(sys.argv) == 2:
        [runName] = sys.argv[1:2]
else:
        print '\n\tUsage:   getruninfo.py <runName>'
```

```
        print '\n\tExample: getruninfo.py Auto_user_f4--134-br_21'
        sys.exit(1)
```

Connect to the server on sending the first request and GET the results element associated with the desired run name.

The requests KeyError and IndexError exceptions are also handled.

```
base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/results?format=json&resultsName=%s'%(base_url, runName), auth=('myusername',
resp_json = resp.json()

try:
        runData = resp_json[u'objects'][0]
        experLoc = runData[u'experiment']
except (KeyError, IndexError):
        print 'ERROR: Invalid name given.'
        sys.exit(1)
```

Connect to the server on sending the first request and GET the results element associated with the desired run name.

Decode the JSON data received from the server into Python objects.

The objects field contains the run data. Of these data, the following fields are of interest for output display or for linking to additional data:

- resultsName
- timeStamp
- experiment
- analysismetrics
- qualitymetrics

Use the experiment field to get the URI of the experiment data associated with this run, and parse the JSON response data.

Display the experiment metadata from the following experiment element fields:

- project
- expName
- pgmName
- library
- notes

The dictionary KeyError exception is also handled.

```
expResult = requests.get('http://%s%s'%(myhost, experLoc))
expData = expResult.json()

try:
        print '\nProject:\t\t%s'%expData[u'log'][u'project']
        print 'Experiment Name:\t%s'%expData[u'expName']
        print 'PGM Name:\t\t%s'%expData[u'pgmName']
        print 'Library:\t\t%s'%expData[u'log'][u'library']
        print 'Notes:\t\t\t%s'%expData[u'notes']
except KeyError:
        print 'ERROR: Invalid key in expData.'
```

Display the results element data previously parsed.

```
try:
        print 'Results:\t\t%s'%runData['resultsName']
        print 'Timestamp:\t\t%s'%runData['timeStamp']
except KeyError:
        print 'ERROR: Invalid key in runData.'
```

Similar to the way you retrieved experiment data, use the analysismetrics and qualitymetrics fields to get the respective URIs for retrieving metrics data from the server. Parse the JSON response data for both elements and display the results. All returned metrics data are displayed.

```
ametricsLoc = runData[u'analysismetrics'][0]
aResult = requests.get('http://%s%s'%(myhost,ametricsLoc))
aData = aResult.json()

print '\n\nAnalysis Metrics:\n==================\n'
for propType, propVal in aData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t= %s'%(propType, propVal)

qmetricsLoc = runData[u'qualitymetrics'][0]
qResult = requests.get('http://%s%s'%(myhost,qmetricsLoc))
qData = qResult.json()

print '\n\nQuality Metrics:\n==================\n'
for propType, propVal in qData.iteritems():
        if propType != 'resource_uri':
                print '%s\t\t=%s'%(propType, propVal)
```

The output Run on a sample database, the program described above produces the following output. You should get similar results running the program against your database.

```
Project:                test
Experiment Name:        R_2012_12_05_19_34_18_user_F4--134-br
PGM Name:               f4
Library:                hg19
Notes:
Results:                barcode_test_large
Timestamp:              2013-06-06T15:28:15.000486+00:00


Analysis Metrics:
==================

libLive         = 0
ignored         = 30065
washout_ambiguous               = 0
sysIE           = 0.600278610364
bead            = 736200
tfKp            = 0
washout_live            = 0
id              = 15
libFinal                = 452234
lib             = 720367
keypass_all_beads               = 0
dud             = 15616
sysCF           = 0.877433363348
```

```
pinned          = 56051
live            = 720584
excluded                = 0
tf              = 217
empty           = 137684
tfFinal         = 200
amb             = 0
lib_pass_basecaller             = 0
lib_pass_cafie          = 0
washout_dud             = 0
libMix          = 0
report          = /rundb/api/v1/results/17/
libKp           = 0
tfLive          = 0
sysDR           = 0.0382701400667
washout_test_fragment           = 0
washout_library         = 0
washout         = 0
tfMix           = 0


Quality Metrics:
==================

q0_reads                =451883
q17_max_read_length             =173
q20_reads               =451883
report          =/rundb/api/v1/results/17/
q17_mean_read_length            =87.0
q17_100bp_reads         =263410
q0_max_read_length              =181
q20_100bp_reads         =105246
id              =15
q20_mean_read_length            =49
q17_bases               =39133239
q0_bases                =47709033
q20_150bp_reads         =6
q17_reads               =451883
q17_50bp_reads          =346855
q20_50bp_reads          =198227
q0_50bp_reads           =414922
q17_150bp_reads         =89
q0_150bp_reads          =298
q0_mean_read_length             =105.0
q20_max_read_length             =156.0
q0_100bp_reads          =333009
q20_bases               =35345630
```

Update Experiment Notes

So far, all of the examples have involved getting data from the server. This example shows you how to modify resource data by sending a PUT request to add a note to an experiment.

Get the current notes First, see what is currently stored for the experiment with id=3:

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/experiment/3/'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Among other data, the response data shows there are no notes in the notes field of experiment 3:

```
        .
        .
        .
"notes": "",
        .
        .
        .
```

Add a Note Construct a JSON notes string, using the json library dumps method.

```
metaData = json.dumps({ "notes" : "This is a sample note." })
```

For PUT and POST reqeusts, data are passed in the message body instead of as a parameter. Set the message body to the notes JSON string created, above.

Also, the JSON data format must be specified in the message header, using the form: 'content-type':'application/json'.

```
putResp = requests.put('%s/experiment/3/'%base_url,
                                                    data=metaData,
                                                    headers={'content-type':'application/json'},
                                                    auth=('myusername', 'mypassword'))
```

Now send a GET request for the same experiment to verify that the text was added to the notes field:

```
resp = requests.get('%s/experiment/3/'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

Typically, you would also test the response status code to verify the action was performed successfully:

```
resp.status_code
```

The notes field now contains the string sent with the PUT request:

```
        .
        .
        .
"notes": "This is a sample note.",
        .
        .
        .
```

Add a PGM™ or Proton™ Sequencer

Topics on this page:

- *View the rig resource before adding an element*
- *Analyze the rig schema*
- *Add a rig element*

- *View the result*

In the previous example, you modified the notes field of an experiment element. In this example, you add an element to the rig resource, which is another name for the PGM™ or Proton™ Sequencer. This example also includes the added complexity of updating a resource that includes a link to another resource.

View the rig resource before adding an element First, use the `cURL` command line program or your REST client to view the rigs defined for your system. Using these tools is a convenient way to view the database while developing and debugging your program. For example:

```
http://myhost/rundb/api/v1/rig?format=json
```

This rig resource contains three PGM™ Sequencers:


```
{
"meta": {
                                .
                                .
                                .
                "total_count": 3
        },
"objects": [
{
                                        .
                                        .
                                        .
                        "name": "B6",
                        "resource_uri": "/rundb/api/v1/rig/B6/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "default",
                        "resource_uri": "/rundb/api/v1/rig/default/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "PGM_test",
                        "resource_uri": "/rundb/api/v1/rig/PGM_test/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
]
}
```


Analyze the rig schema The following example shows the JSON structure of a rig:


```
{
        "comments": "",
        "ftppassword": "ionguest",
```

```
        "ftpserver": "ts",
        "ftpusername": "ionguest",
        "location": {"comments": "", "id": "1", "name": "Home"},
        "name": "PGM_test",
        "updateflag": false,
        "updatehome": "ts"
}
```

What makes this more interesting is that the structure includes a nested dictionary for the location field, with the location schema.

When creating or modifying the rig structure, you also need to provide the location structure, either an existing location or by adding a location resource to the database before adding a rig.

In the programming example, a copy of one of the existing rigs is used but the example shows how to reference a nested dictionary.

Add a `rig` element Because the intention is to copy an existing rig data structure, modifying the desired fields, a GET request is sent to get the `rig` element `PGM_test`, to be copied.

```
import json
import requests

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/rig/PGM_test?format=json'%base_url, auth=('myusername', 'mypassword'))
```

The JSON data structure of the existing rig is returned in the resp variable. Use the .json() method to get a Python json object that can be manipulated as needed.

```
resp_json = resp.json()
```

Only the program name field is changed in the copied rig data. It is changed from `PGM_test` to `myNewPgm`.

Remember that, for almost all resources, all fields must be included in the JSON string when making a PUT or POST request, not only the field you modified. An exception is the `resource_uri` field contained in all resources. The `resource_uri` field is removed in the example using the simplejson pop method.

This example demonstrates the added complication of also removing the `resource_uri` field from the nested location data structure, showing how to access nested data in the process.

```
resp_json.update(name='myNewPgm')
resp_json.pop('resource_uri')
resp_json['location'].pop('resource_uri')
```

Use the json dumps method to encode the Python object into a json string.

```
pdata = json.dumps(resp_json)
```

Use the PUT request to add the new PGM™ or Proton™ Sequencer to the database, passing the URI and message body, pdata, of the new element as parameters. You must also provide the message header and specify the content data type: {'content-type':'application/json'}.

```
status = requests.put('%s/rig/myNewPgm/'%base_url, data=pdata, headers={'content-type':'application/
```

View the result If you again use cURL or a REST client to view the `rig` resource, you can see that a PGM™ or Proton™ Sequencer named `myNewPgm` is added:

```
{
"meta": {
                                .
                                .
                                .
                "total_count": 4
        },
"objects": [
{
                                        .
                                        .
                                        .
                        "name": "B6",
                        "resource_uri": "/rundb/api/v1/rig/B6/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "default",
                        "resource_uri": "/rundb/api/v1/rig/default/",
                        "updateflag": false,
                        "updatehome": "ts"
                },
{
                                        .
                                        .
                                        .
                        "name": "PGM_test",
                        "resource_uri": "/rundb/api/v1/rig/PGM_test/",
                        "updateflag": false,
                        "updatehome": "ts"
                },{
                                        .
                                        .
                                        .
                        "name": "myNewPgm",
                        "resource_uri": "/rundb/api/v1/rig/myNewPgm/",
                        "updateflag": false,
                        "updatehome": "ts"
                }
]
}
```


Work with PGM™ or Proton™ Status

The `rig` resource API has a unique syntax that permits you to update individual fields. All other resources require that you provide all fields when updating the resource.

By using the status keyword, following the key or sequencer name in the URI, you can update the following `rig` resource fields, individually:

- state
- last_init_date

- last_clean_date
- last_experiment
- version
- alarms

General form of the rig status request

```
http://myhost/rundb/api/v1/PGM_test/status?format=json
```

When you update rig status, you can provide either one or all of the status items as data in the request body.

A rig status update example This example formats a status request, updating all of the status fields, and displays the results.

In the example, a simple cURL or REST client request to GET the PGM_test resource element returns the following results:

```
http://myhost/rundb/api/v1/rig/PGM_test?format=json

{"alarms": {}, "comments": "", "ftppassword": "ionguest",
"ftprootdir": "results", "ftpserver": "192.168.201.1",
"ftpusername": "anonymous", "last_clean_date": "", "last_experiment": "",
"last_init_date": "", "location": {"comments": "", "id": "1",
"name": "Home", "resource_uri": "/rundb/api/v1/location/1/"},
"name": "PGM_test", "resource_uri": "/rundb/api/v1/rig/PGM_test/",
"state": "", "updateflag": false, "updatehome": "192.168.201.1",
"version": {}}
```

You can refine the GET request to only retrieve the status fields, returning the following result:

```
http://myhost/rundb/api/v1/rig/PGM_test/status?format=json

{"alarms": {}, "last_clean_date": "", "last_experiment": "",
"last_init_date": "", "state": "", "version": {}}
```

You can see in this example that all of the field values are empty.

The programming example to update these fields uses the requests and simplejson Python libraries:

```
import requests
import simplejson as json
```

A local status variable is initialized to assign a value to each of the status fields:

```
status = {}

status["last_init_date"] = "rig.last_init_date"
status["state"] = "rig.state"
status["last_clean_date"] = "rig.last_clean_date"
status["last_experiment"] = "rig.last_experiment"
status["version"] = {"version":"test"}
status["alarms"] = {"rig.alarms":"test"}
```

And the Python status object is encoded into a JSON string:

```
pdata = json.dumps(status)
print pdata
```

The program displays the JSON string to be sent to the server in the request body:

```
{"last_clean_date": "rig.last_clean_date",
"last_experiment": "rig.last_experiment",
"state": "rig.state", "version": {"version": "test"},
"last_init_date": "just this", "alarms": {"rig.alarms": "test"}}
```

Now, send the PUT request to the server to update the status fields, providing the JSON string as data:

```
status = requests.put('http://myhost/rundb/api/v1/rig/PGM_test/status/',
                      data=pdata,
                      headers={'content-type':'application/json'},
                      auth=('myusername', 'mypassword'))
print status
```

The server returns an HTTP status code of 204, indicating a successful PUT request.

To verify that the status fields have been updated, a GET request is sent, and the response is displayed:

```
resp1 = requests.get('http://myhost/rundb/api/v1/rig/PGM_test/status/',
                      auth=('myusername', 'mypassword'))
print resp1.content
```

You can see the status fields now contain the data sent with the PUT request:

```
{"alarms": {"rig.alarms": "test"}, "last_clean_date": "rig.last_clean_date",
"last_experiment": "rig.last_experiment",
"last_init_date": "rig.last_init_date", "state": rig.state",
"version": {"version": "test"}}
```

These examples show more complex and involved database query sequences than the basic operations used to introduce REST API programming. They get run metadata then use linked fields to navigate to analysis and quality metrics associated with a run.

Some examples demonstrate how to use the PUT and POST methods to update data resource fields and to create new resource elements.

Although simple resources are shown, having a limited number of fields, the procedures demonstrated in this section apply for any of the resources exposed by the REST API.

Work with the File System

Using the API, you can find and download analysis results files.

Download a FASTQ File

This example shows the REST API facilities for working with the file system.

Making the following request on the results resource,

```
http://myhost/rundb/api/v1/results/13?format=json
```

shows the path of the associated FASTQ file. The database schema includes a number of file path entries, which can all be accessed in the same way.

```
{
                    .
                    .
                    .
```

```
        "fastqLink": "/output/Home/Auto_B15-45_4_013/R_2010_11 ... B15-45_Au...",
                .
                .
                .
}
```

You can get the file contents by copying the path to the URI, following the host name.

```
http://myhost/output/Home/Auto_B15-45_4_013/R_2010_11 ... B15-45_4.fastq
```

The entire sequence is shown in the following programming example.

The GET request on the results resource returns the FASTQ file path in the fastqLink field.

```
import requests
import simplejson as json

base_url = 'http://myhost/rundb/api/v1'
resp = requests.get('%s/results/13?format=json'%base_url, auth=('myusername', 'mypassword'))
resp_json = resp.json()
```

To GET the file contents, append the fastqLink value to the URI, following the host name.

```
resp = requests.get('http://myhost/%s'%resp_json['fastqLink'], auth=('myusername', 'mypassword'))
```

Display the FASTQ file path and the contents of the file.

```
print resp_json['fastqlink']
print resp.content
```

List File Servers

This example accesses the fileserver resource to find all file server directories.

This example demonstrates the use of the `httplib2` Python REST library:

```
import httplib2
import json
```

On the first request, perform authentication:

```
h = httplib2.Http(".cache")
h.add_credentials('myusername', 'mypassword')
```

Request all fileserver elements using the GET method:

```
resp, content = h.request("http://localhost/rundb/api/v1/fileserver?format=json", "GET")
```

Parse the JSON string response into Python objects:

```
contentdict = json.loads(content)
```

Loop through each object in the list and display the directory name:

```
objects = contentdict['objects']

for obj in objects:
        print obj['filesPrefix']
```

In addition to the FASTQ example, other files included in the results resource include the BAM, test fragment BAM, and default report files. The reference genome file can be also located and downloaded using the link in the `referencegenome` resource.

This example demonstrates the unique way files are referenced using the REST API.

Run a Plugin

You can use the API to run plugins programmatically, including passing parameters to plugins.

Topics on this page:

- *Get a list of plugins*
- *Start a plugin without parameters*
- *Start a plugin with parameters*

For more information about the plugin execution environment and plugins development, see the following documents on the Ion Community:

- Plugin SDK Documentation
- Introduction to Python Plugins

Get a list of plugins

Enter the plugin resource name in the URI to get a list of all plugins. Use the parameter `active=True` to restrict the list to the currently installed plugins.

```
http://myhost/rundb/api/v1/plugin/?format=json&active=True
```

The response includes plugin metadata and the data for each plugin in the system. Notice that, by default, the response only included 20 elements but the total_count meta field indicates there are 28 plugins. (Append `limit=0` to show all the results in a single response, or use `offset=20` to get the next 20 entries.)


```
{
        "meta": {
                "limit": 20,
                "next": "/rundb/api/v1/plugin/?offset=20&limit=20&format=json",
                "offset": 0,
                "previous": null,
                "total_count": 28
        },
        "objects": [
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.438365",
                        "id": "23",
                        "libraryName": "",
                        "name": "top100Ionogram",
                        "path": "/results/plugins/top100Ionogram",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/23/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                },
```

```
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.477418",
                        "id": "24",
                        "libraryName": "",
                        "name": "AmpliconRep",
                        "path": "/results/plugins/AmpliconRep",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/24/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                },
                        .
                        .
                        .
                {
                        "autorun": true,
                        "chipType": "",
                        "date": "2011-05-06T19:09:45.760567",
                        "id": "42",
                        "libraryName": "",
                        "name": "igv",
                        "path": "/results/plugins/igv",
                        "project": "",
                        "resource_uri": "/rundb/api/v1/plugin/42/",
                        "sample": "",
                        "selected": false,
                        "version": "0"
                }
        ]
}
```


Specify the filtering criteria or the plugin id to retrieve the data for a single plugin.

Start a plugin without parameters

The following code snippet shows how to start a plugin that requires no parameters. (The requests and simplejson Python libraries are used, as in previous examples.)

Use a dictionary that has the plugin keyword and the plugin name as the value:

```
myPlugin = json.dumps( {"plugin": ["AmpliconRep"]} )
```

Send a POST request to run the plugin with the plugin name in the request body:

```
status = requests.post('http://myhost/rundb/api/v1/plugin/84/',
                data=myPlugin,
                headers={'content-type':'application/json'},
                auth=('myusername', 'mypassword'))
```

Start a plugin with parameters

To run a plugin requiring runtime parameters, simply add the parameters to the dictionary, as in the following code snippet, and include the plugin name and parameters in the request body:

```
myPlugin = json.dumps(
{
        "plugin": ["AmpliconRep"],
        "pluginconfig" : { "user_variables" : "foo" }
})
```

Again, send a POST request to run the plugin.

Here is a complete example using httplib2. (The shebang #! just allows for easy execution.)

```
#!/usr/bin/python
import httplib2
import json
#the primary key for the report
reportPrimaryKey = "1234"
#the name of the plugin to run
pluginName = "YOUR_PLUGIN"
h = httplib2.Http()
h.add_credentials('ionadmin', 'ionadmin')
headers = {"Content-type": "application/json","Accept": "application/json"}
url = 'http://ionwest' + '/rundb/api/v1/results/' + reportPrimaryKey + "/plugin/"
pluginUpdate  = {"plugin": [pluginName]}
resp, content = h.request(url, "POST", body=json.dumps(pluginUpdate),headers=headers )
print resp
print content
```

Write a Web Service Client

You can use the REST API to write a web service client.

Topics on this page:

- *Get a list of experiments*
- *Display results data*
- *CSS used in these examples*

Some application environments have timing complexities that make a simple request-response communication paradigm undesirable. A non-deterministic, blocking protocol, like HTTP, may take a long time to complete, causing connections to time out or degraded application performance. Using AJAX, you can achieve the asynchronous behavior needed for such applications.

These examples uses the jQuery library to show how to set up and make a REST API call and handle the pending response. The first example simply lists all experiments on the server, sorted by date. The second example uses the experiment resource link to the results data for the experiment to also display all analysis results for the experiment. (The CSS code is only provided to show the UI presentation mechanism used in the examples.)

Because JavaScript prevents data requests to servers in a different domain, the JSONP data format is used handle this limitation.

Building on the fundamental procedures shown in these examples, you might easily modify the application to also monitor the status of a run and report when analysis processing has completed. Such an application could be implemented on a mobile device to allow remote monitoring and real-time notification.

(!) The sample JavaScript code has been tested with the Chrome 11 and Firefox 3.6 browsers.

Get a list of experiments

The first example is to display the list of experiments, including the run date and the PGM™ or Proton™ Sequencer name.

Experiment	Date	PGM
R_2010_11_05_18_32_00_user_B6--237	2010-11-05T18:32:00	B6
R_2010_11_08_22_30_04_user_B15-45	2011-03-07T17:48:53	PGM_test

The HTML is simply a tag encapsulating the display. The JavaScript functions write data to the element whose id is mainPage.

```
<div id="mainPage"></div>
```

The JavaScript uses the jQuery library.

```
<script type="text/javascript" src="http://code.jquery.com/jquery-1.6.min.js"></script>
```

On loading the HTML page, a jQuery AJAX function sets up the request for experiment data:

- The URL is set to the URI for the experiment resource.
- The data type is set to jsonp to handle JavaScript cross-domain restrictions.
- A function is defined to handle the response from the server.

```
$(function(){
        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment \
                        ?format=jsonp&order_by=date",
                dataType: 'jsonp',
                success: function(json_results){
                        $('#mainPage').append('<table class="expTable">
                                                                <tr><th>Experiment</th>
                                                                    <th>Date</th>
                                                                    <th>PGM</th></table>');
                        listItems = $('#mainPage').find('table');
                        $.each(json_results.objects, function(key) {
                                html =  '<td>' + json_results.objects[key].expName + '</td>';
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].date + '</td>' ;
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].pgmName + '</td>' ;
                                listItems.append('<tr class="expTableRow">'+html+'</tr>');
                        });
                }
        });
});
```

Within the `id=mainPage` element, the response handling function constructs an HTML table and populates the cells with the following experiment resource fields, for each element returned in the response:

- `expName`, the experiment name.
- `date`, the date of the experiment.
- `pgmName`, the name of the PGM sequencer where the experiment was run.

Display results data

The previous example is now extended to include the location of results data associated with the experiment. This might be useful for monitoring run status.

Experiment	Date	PGM	Results
R_2010_11_05_18_32_00_user_B6--237	2010-11-05T18:32:00	B6	/rundb/api/v1/results/20/ /rundb/api/v1/results/12/
R_2010_11_08_22_30_04_user_B15-45	2011-03-07T17:48:53	PGM_test	/rundb/api/v1/results/13/

Further, by clicking on the results link in the table, results metadata are displayed:

Results Name	B6--237-TEST2
Date	2011-05-13T14:51:58.518491
Frames Processed	0
Cycles Processed	55
Status	Completed
Time to Complete	0

The JavaScript code snippet is the same as the previous example with the addition of a loop to get all of the results for an experiment and display them in the *results* column.

Notice that the results table entry includes an onClick event handler, which call the showResult function to display results metadata, passing the results location from the results field as a parameter.

```
$(function(){
        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/experiment \
                        ?format=jsonp&order_by=date",
                dataType: 'jsonp',
                success: function(json_results){
                        $('#mainPage').append('<table class="expTable">
                                                        <tr><th>Experiment</th>
                                                                <th>Date</th><th>PGM</th>
                                                                <th>Results</th></table>')
                        listItems = $('#mainPage').find('table');
                        $.each(json_results.objects, function(key) {
                                html =  '<td>' + json_results.objects[key].expName + '</td>';
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].date + '</td>' ;
                                html += '<td class="expTableCol">' + \
                                                json_results.objects[key].pgmName + '</td>' ;
                                html += '<td class="expTableCol">' ;
                for (var result in json_results.objects[key].results)
                {
                        resultPath = json_results.objects[key].results[result];
                        resultId = resultPath.split('/');
                        html += '<a href="#" onClick= \
                                                        "showResult(' + resultId[5] + '); \
                                                        return false;">' + resultPath + '</a
                }
                                html += '</td>' ;
                                listItems.append('<tr class="expTableRow">'+html+'</tr>');
                        });
                }
        });
});
```

The call to `showResult` constructs an AJAX GET request, including the `results` field parameter in the URI and,

again, specifying a jsonp data type.

```
function showResult(resultLocation) {
/\*     alert("resultLocation: " + resultLocation)  \*/

        $.ajax({
                url: "http://myusername:mypassword@myhost/rundb/api/v1/results/" + \
                          resultLocation + "?format=jsonp",
                dataType: 'jsonp',
```

When a successful response is received from the server, the response handler constructs a table of metadata in the mainPage element.

The following results resource fields are appended to the table:

- `resultsName`, the name assigned to the analysis.
- `timeStamp`, the time of the analysis.
- `framesProcessed`, the number of frames processed.
- `processedCycles`, the number of cycles processed.
- `status`, the analysis status.
- `timeToComplete`, the time remaining to complete the analysis.

```
success: function(json_results){
                $('#mainPage').replaceWith('<div id="mainPage"> \
                                                        <table class="expTable"></table></div>');
                listItems = $('#mainPage').find('table');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Results Name</td><td>' + \
                                                                json_results.resultsName + '</td></
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Date</td> \
                                                                <td>' + json_results.timeStamp + '<
                                                                </tr>');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Frames Processed</td><td>' + \
                                                                json_results.framesProcessed + '</t
                                                                </tr>');
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Cycles Processed</td><td>' + \
                                                                json_results.processedCycles + '</t
                                                                </tr>');
                listItems.append('<tr class="expTableRow"><td>Status</td> \
                                                                <td>' + json_results.status + '</td
                listItems.append('<tr class="expTableRow"> \
                                                                <td>Time to Complete</td><td>' + \
                                                                json_results.timeToComplete + '</td
                }
        });

}
```

CSS used in these examples

```
#mainPage
{
```

```
        background-color: #f9f1cd;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}

ul
{
        list-style-type: none;
}

table.expTable
{
        border:none;
        border-spacing:0px;
        margin-left:25px;
        vertical-align:top;
        padding:0px;
}

table.expTable th
{
        border-color: #dddddd;
        border-width: 1px 1px 1px 1px;
        border-style: solid;
        background-color: #333333;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
        color:#f9f1cd;
        font-size:8pt;
        margin-left:25px;
        padding-right:10px;
        vertical-align:top;
}

table.expTable td
{
        border-color: #dddddd;
        border-width: 1px 1px 1px 1px;
        border-style: solid;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
        color:#666666;
        font-size:8pt;
        margin-left:25px;
        padding-right:10px;
        vertical-align:top;
}
```

The API Quick Reference provides a summary of select APIs:

Ion Torrent™ Server API Quick Reference

REST request format

Syntax

```
http://[<username>:<password>@]
        <host>/rundb/api/<version>/<resource>?format=json
        [[&<filter>{=<value> | __<qualifier>=<value>}]...][&order_by=[-]<filter>]
```

Examples

```
http://ionuser:ionuser@ionwest.itw/rundb/api/v1/experiment
        ?format=json&expName__startswith=R_2010_11&order_by=date

curl --user ionuser:ionuser --header "Content-Type: application/json"
        --location 'http://ionwest.itw/rundb/api/v1/results'
```

REST methods

Method	Function	Element URI	List URI
POST	Create	Create new entry in element	Create new entry in list
GET	Read	Get URI content	List URI members
PUT	Update	Replace URI content	Replace URI members
DELETE	Delete	Delete URI element	Delete list members

HTTP status codes

Method	Code	Meaning
	200 301	Resource exists Permanently moved
GET	401	Authorization err or
	404	Not found
	410	No longer exists
	200 201	Resource replaced Resource created
	204	No response
	301	Redirect
PUT/POST	400	Invalid data
	401	Authorization error
	409	Resource state conflict
	500	Internal error
	501	Method not implemented
DELETE	200 400	Resource deleted Resource not deleted
	401	Authorization error

Top-level requests

URI	Scope
/rundb/api/v1/	Resource list
/rundb/api/v1/ <resource>/	Resource element list (default limit: 20)
/rundb/api/v1/ <resource>?limit =0	Resource element list; all elements
/rundb/api/v1/ <resource>/<key>/	Resrouce element
/rundb/api/v1/ <resource>/set/ <key>;<key>;.../	Multiple resource elements
/rundb/api/v1/ <resource>/schema	Resource schema

Resources

- ++ *indicates PUT/POST/DELETE permitted*

- *Bold type indicates KEY field*

analysismetrics amb, bead, dud, empty, excluded, **id**, ignored, keypass_all_beads, lib, libFinal, libKp, libLive, libMix, lib_pass_basecaller, lib_pass_cafie, live, pinned, report, sysCF, sysDR, sysIE, tf, tfFinal, tfKp, tfLive, tfMix, washout, washout_ambiguous, washout_dud, washout_library, washout_live, washout_test_fragment

dnabarcode ++ adapter, annotation, floworder, **id**, index, length, name, sequence, type

experiment ++ autoAnalyze, backup, barcodeId, baselineRun, chipBarcode, chipType, cycles, date, expCompInfo, expDir, expName, flows, flowsInOrder, ftpStatus, **id**, library, libraryKey, log, metaData, notes, pgmName, project, reagentBarcode, results, sample, seqKitBarcode, star, storageHost, storage_options, unique, usePreBeadfind

fileserver comments, filesPrefix, **id**, location, name

globalconfig default_command_line, default_flow_order, default_library_key, default_plugin_script, default_storage_options, default_test_fragment_key, fasta_path, **id**, name, plugin_folder, plugin_output_folder, records_to_display, reference_path, selected, sfftrim, sfftrim_args, site_name, web_root

libmetrics See Filters (continued)

location comments, **id**, name

plugin autorun, chipType, date, **id**, libraryName, name, path, project, sample, selected, version

qualitymetrics **id**, q0_100bp_reads, q0_15bp_reads, q0_50bp_reads, q0_bases, q0_max_read_length, q0_mean_read_length, q0_reads, q17_100bp_reads, q17_150bp_reads, q17_50bp_reads, q17_bases, q17_max_read_length, q17_mean_read_length, q17_reads, q20_100bp_reads, q20_150bp_reads, q20_50bp_reads, q20_bases, q20_max_read_length, q20_mean_read_length, q20_reads, report

referencegenome bled, **id**, index_version, name, notes, reference_path, short_name, source, species, status, verbose_error, version

results ++ analysisVersion, analysismetrics, experiment, fastqLink, framesProcessed, **id**, libmetrics, log, metaData, pluginState, pluginStore, processedCycles, qualitymetrics, reportLink, reportstorage, resultsName, sffLink, status, tfFastq, tfSffLink, tfmetrics, timeStamp, timeToComplete

rig ++ alarms, comments, ftppassword, ftprootdir, ftpserver, ftpusername, last_clean_date, last_experiment, last_init_date, location, **name**, <nameValue>/status, state, updateflag, updatehome, version

runtype ++ barcode, description, **id**, runType

tfmetrics See Filters (continued)

Extended resources

Plugins

POST: *Request Header:*

```
Content-Type: application/json
```

Request Body:

```
{"plugin":["<pluginName>"]}
```

or

```
{"plugin":["<pluginName>"], pluginconfig : { json params } }
```

```
http://myhost/rundb/api/v1/results/<key>/plugin?format=json
```

Files Example:

1. From *results* resource response:

```
{"log": "/output/Home/Auto_B15-45_4_013/log.html"}
```

2. Get file:

```
http://myhost/output/Home/Auto_B15-45_4_013/log.html
```

Filter qualifiers

Usage: `<field>__<qualifier>=<value>`

Example: `library__contains=coli`

contains	icontains	istartswith	search
day	iendswith	lt	startswith
endswith	iexact	lte	week_day
exact	in	month	year
gt	iregex	range	
gte	isnull	regex	

Sort parameter

Usage: `order_by=[-]<filter>`

Examples:

(ascending) `order_by=date`
(descending) `order_by=-date`

Data format parameter

Format	Parameter	Note
JSON	?format=json	
XML	?format=xml	Not supported

Supported run types (>= 3.x)

Run Type	Description
RunType.FULLCHIP	Whole chip PGM run.
RunType.THUMB	Thumbnail run.
RunType.COMPOSITE	Proton run.

Supported run levels (>= 3.x)

Run Level	Description
RunLevel.PRE	Runs after all analysis jobs have been submitted, but before any finish.
RunLevel.BLOCK	Runs when an individual block finishes analysis. Occurs once for each block.
RunLevel.POST	Runs after all blocks are done processing.
RunLevel.LAST	Runs after everything, including other plugins. Multiple plugins w/ LAST will run at the same time.
RunLevel.DEFAULT	Default run level, generally PGM runs.

Ion Torrent™ Server API Quick Reference - 2

Return to API Quick Reference main page

Filters (continued)

libmetrics

Genome_Version, Index_Version, align_sample, aveKeyCounts, cf, dr, extrapolated_100q10_reads, extrapolated_100q17_reads, extrapolated_100q20_reads, extrapolated_100q47_reads, extrapolated_100q7_reads, extrapolated_200q10_reads, extrapolated_200q17_reads, extrapolated_200q20_reads, extrapolated_200q47_reads, extrapolated_200q7_reads, extrapolated_50q10_reads, extrapolated_50q17_reads, extrapolated_50q20_reads, extrapolated_50q47_reads, extrapolated_50q7_reads, extrapolated_from_number_of_sampled_reads, extrapolated_mapped_bases_in_q10_alignments, extrapolated_mapped_bases_in_q17_alignments, extrapolated_mapped_bases_in_q20_alignments, extrapolated_mapped_bases_in_q47_alignments, extrapolated_mapped_bases_in_q7_alignments, extrapolated_q10_alignments, extrapolated_q10_coverage_percentage, extrapolated_q10_longest_alignment, extrapolated_q10_mean_alignment_length, extrapolated_q10_mean_coverage_depth, extrapolated_q17_alignments, extrapolated_q17_coverage_percentage, extrapolated_q17_longest_alignment, extrapolated_q17_mean_alignment_length, extrapolated_q17_mean_coverage_depth, extrapolated_q20_alignments, extrapolated_q20_coverage_percentage, extrapolated_q20_longest_alignment, extrapolated_q20_mean_alignment_length, extrapolated_q20_mean_coverage_depth, extrapolated_q47_alignments, extrapolated_q47_coverage_percentage, extrapolated_q47_longest_alignment, extrapolated_q47_mean_alignment_length, extrapolated_q47_mean_coverage_depth, extrapolated_q7_alignments, extrapolated_q7_coverage_percentage, extrapolated_q7_longest_alignment, extrapolated_q7_mean_alignment_length, extrapolated_q7_mean_coverage_depth, genome, genomelength, genomesize, i100Q10_reads, i100Q17_reads, i100Q20_reads, i100Q47_reads, i100Q7_reads, i200Q10_reads, i200Q17_reads, i200Q20_reads, i200Q47_reads,

i200Q7_reads, i50Q10_reads, i50Q17_reads, i50Q20_reads, i50Q47_reads, i50Q7_reads, **id**, ie, q10_alignments, q10_coverage_percentage, q10_longest_alignment, q10_mapped_bases, q10_mean_alignment_length, q10_qscore_bases, q17_alignments, q17_coverage_percentage, q17_longest_alignment, q17_mapped_bases, q17_mean_alignment_length, q17_qscore_bases, q20_alignments, q20_coverage_percentage, q20_longest_alignment, q20_mapped_bases, q20_mean_alignment_length, q20_qscore_bases, q47_alignments, q47_coverage_percentage, q47_longest_alignment, q47_mapped_bases, q47_mean_alignment_length, q47_qscore_bases, q7_alignments, q7_coverage_percentage, q7_longest_alignment, q7_mapped_bases, q7_mean_alignment_length, q7_qscore_bases, r100Q10, r100Q17, r100Q20, r200Q10, r200Q17, r200Q20, r50Q10, r50Q17, r50Q20, rCoverage, rLongestAlign, rMeanAlignLen, rNumAlignments, report, s100Q10, s100Q17, s100Q20, s200Q10, s200Q17, s200Q20, s50Q10, s50Q17, s50Q20, sCoverage, sLongestAlign, sMeanAlignLen, sNumAlignments, sampled_100q10_reads, sampled_100q17_reads, sampled_100q20_reads, sampled_100q47_reads, sampled_100q7_reads, sampled_200q10_reads, sampled_200q17_reads, sampled_200q20_reads, sampled_200q47_reads, sampled_200q7_reads, sampled_50q10_reads, sampled_50q17_reads, sampled_50q20_reads, sampled_50q47_reads, sampled_50q7_reads, sampled_mapped_bases_in_q10_alignments, sampled_mapped_bases_in_q17_alignments, sampled_mapped_bases_in_q20_alignments, sampled_mapped_bases_in_q47_alignments, sampled_mapped_bases_in_q7_alignments, sampled_q10_alignments, sampled_q10_coverage_percentage, sampled_q10_longest_alignment, sampled_q10_mean_alignment_length, sampled_q10_mean_coverage_depth, sampled_q17_alignments, sampled_q17_coverage_percentage, sampled_q17_longest_alignment, sampled_q17_mean_alignment_length, sampled_q17_mean_coverage_depth, sampled_q20_alignments, sampled_q20_coverage_percentage, sampled_q20_longest_alignment, sampled_q20_mean_alignment_length, sampled_q20_mean_coverage_depth, sampled_q47_alignments, sampled_q47_coverage_percentage, sampled_q47_longest_alignment, sampled_q47_mean_alignment_length, sampled_q47_mean_coverage_depth, sampled_q7_alignments, sampled_q7_coverage_percentage, sampled_q7_longest_alignment, sampled_q7_mean_alignment_length, sampled_q7_mean_coverage_depth, sysSNR, totalNumReads, total_number_of_sampled_reads

tfmetrics

CF, DR, HPAccuracy, HPSNR, IE, Q10Histo, Q10Mean, Q10Mode, Q10ReadCount, Q17Histo, Q17Mean, Q17Mode, Q17ReadCount, SysSNR, aveHqReadCount, aveKeyCount, aveQ10ReadCount, aveQ17ReadCount, corOverlap, corrHPSNR, corrIonogram, error, hqReadCount, **id**, keypass, matchMismatchHisto, matchMismatchMean, matchMismatchMode, name, number, postCorrSNR, preCorrSNR, rawIonogram, rawOverlap, report, sequence

Return to API Quick Reference main page

5.3 Torrent Suite™ Software API User Guide

The Torrent Suite™ Software API User Guide describes the technology behind API and how to leverage the technology using the REST interface.

5.3.1 Purpose

The API User Guide presents background and conceptual information to provide context for using the Torrent Suite™ Software API.

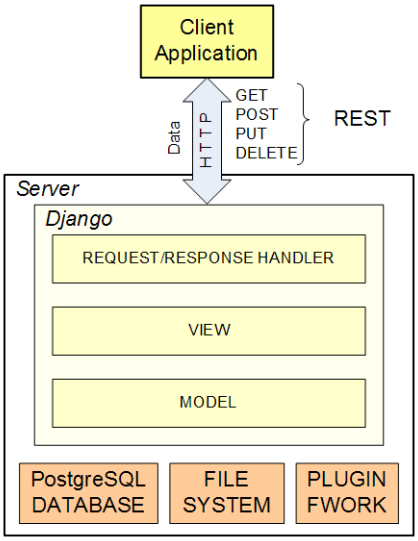
You should already have a basic understanding of software development related to web services, and familiarity with the Torrent Browser user interface. The purpose of this document is to focus that basic knowledge in the area of the Torrent Suite™ Software web service, showing how the API is implemented and the mechanisms available for developing applications that take full advantage of Torrent Suite functionality exposed by the API.

Once you know the basic architecture and concepts behind the API, you are encouraged to work through the *Torrent Suite™ Software API Cookbook* as a hands-on way to discover API functionality in a step-by-step way.

5.3.2 General Architecture

The Torrent Suite™ Software API is implemented as conventional web service, where your client application sends and receive messages to access server functionality.

The following figure describes the client-server relationship and the layered implementation of the web service implemented by the server.



Programming model

Client application

Your application is the client application in the client-server architecture, which can be further described as a service-oriented architecture. The client makes requests for information, or data or some service and the server responds to the request. The server makes system facilities available to your application as a service. Your application behaves very much like the web browser with which you are familiar, except that as a specialized program your application can do more than simply view the data.

REST

Representational State Transfer (REST) is another architectural concept applied to data interchange between clients and servers. Using the communication methods inherent in the underlying HTTP protocol, clients make requests for resources resident on the server. Resources are usually database items but can be any service or functionality implemented by the server. These communication methods are the following:

- GET to retrieve information from the server.
- PUT to update information on the server.
- POST to create a new information record on the server.
- DELETE to delete information on the server.

Web developers should already be familiar with these methods.

Other characteristics that make REST the architecture of choice for the Torrent Suite API include:

- Server resources are addressable using a Universal Resource Identifier (URI), which is similar to the URL in your browser address window.
- The JavaScript Object Notation (JSON) data format is supported, providing a simple data encoding, transfer and decoding, while meeting the needs of most applications.
- REST is stateless, as the name implies, meaning the server does not have or provide the additional complexity of maintaining the state of the application.
- REST is easy to implement and easy to use.

Server

The server makes resources and functionality available to client applications upon request.

Web application servers typically implement the Model-View-Controller architecture pattern. Simply, the server software implementation abstracts the backend database, or *model*, and presents a view of the data in a way that the application can access and use. The *controller* part of the pattern implements the business logic around the data.

This decomposition of web application server software is in widespread use and Django is used to implement an MVC-like pattern in Torrent Server.

Django

Django is a web application framework that implements a RESTful API using the Tastypie framework.

Modern web-like services are typically implemented using the Model-View-Controller architectural pattern, which is similarly implemented by Django using the Python programming language.

Resources

The API operates on resources, which are addressable using a URI. The conventional notion of a web application resource is a database item but the API extends the notion to include the file system and plugins.

Resources that are not database entities, such as plugins, are not included in the REST API schema listing.

PostgreSQL database

Torrent Server uses a PostgreSQL database for persistent backend data storage. PostgreSQL supports access to the data using standardized SQL.

Django integrates PostgreSQL as the model of an MVC framework and exports an SQL-like interface, using Tastypie, through the REST API.

File system

The API can be used to download files, typically, analysis data and results files.

A number of database resources, such as results, have fields that link to results files. An application can use these links to reference and download files.

Plugin framework

Plugins provide a mechanism for you to write code that can be run, automatically or manually, using a shell script after analysis pipeline processing completes. The plugin feature extends Torrent Server capabilities in an open-ended, flexible way.

The API provides methods to submit plugins to the job scheduler.

5.3.3 Technology

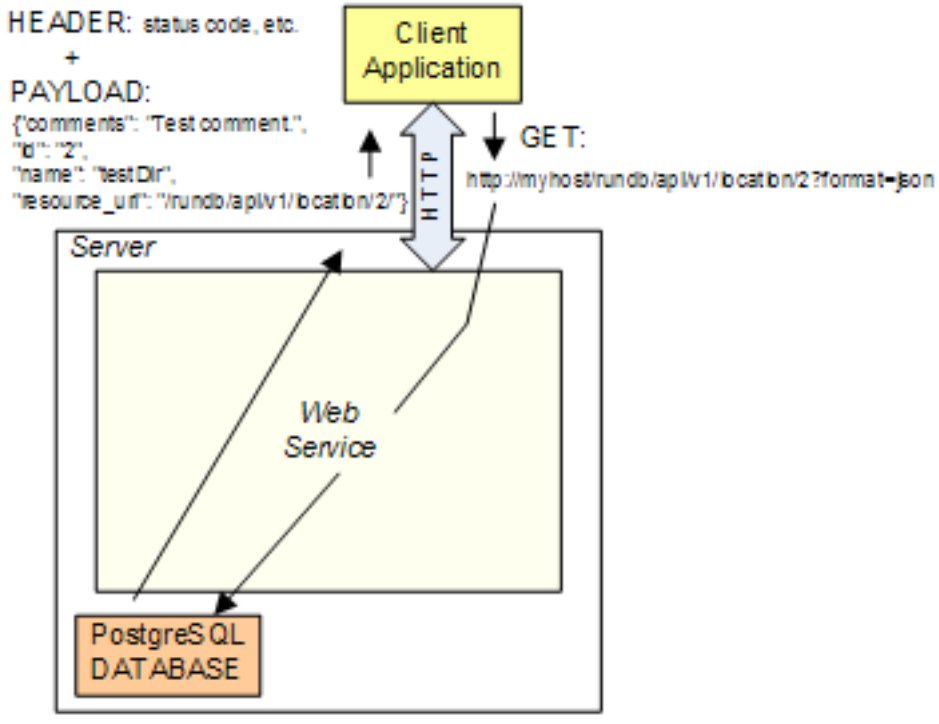
The following software elements are the core technologies used to implement the Torrent Suite™ Software web application service:

REST

Representational State Transfer (REST) is an architectural model for implementing loosly-coupled, distributed applications, making REST similar to the RPC communication models. While not a standard, REST embraces the communication methods defined by the HTTP standard, adding the notion of resources as communication endpoints.

The resources REST operates on are addressable using a Uniform Resource Identifier (URI). Resources are located on the server. A core notion of the REST model is that resources are stateless. Communication by the client application with the server are in the form of requests of resource and any needed state information is carried in the request and not maintained on the server.

On Torrent Server, a resource is most often a database item located in the backend PostgreSQL database. (Files and plugins are also defined as resources in the API.) The following figure shows the components involved in making a REST request.



The client application sends an HTTP GET request to retrieve an item from the database; in this example, the URI identifies a location resource whose ID is 2, also specifying that the data format is JSON:

```
http://myhost/rundb/api/v1/location/2?format=json
```

The server decodes the request, finding the location resource and returning the data for location 2 to the client application in a response message. The response has a header and payload. The header includes the request status code, indicating if the request was successful or not, and the payload contains the requested resource data.

HTTP provides the underlying communication mechanism in the REST model.

HTTP methods

The following HTTP communication methods are considered sufficient for a RESTful application to perform all necessary operations on resources. These are commonly abbreviated using the CRUD acronym: C - create, R - read, U - update and D - Delete.

GET Use the GET method to retrieve resource information, or information about resources.

PUT Use the PUT method to update resource data.

POST Use the POST method to create a resource.

DELETE Use the DELETE resource to delete a resource.

Universal Resource Indicator (URI)

The Uniform Resource Identifier (URI) is a address string used to locate a named resource or group of resources over a network. The REST communication model uses a URI to identify resources with each resource having a unique URI.

Th URI structure is hierarchical so a group of resources can be identified by specifying the location at a higher level in the address hierarchy, without specifying a particular resource.

Syntax

```
http://<username>:<password>@]<host>/rundb/api/<version>
          [/<resource>[/<key>]?format=json
          [[&<filter>{=<value> | __<qualifier>=<value>}]...]
          [&order_by=[-]<filter>\]

         username ::= User login name.

         password ::= User login password.

         host ::= Host server name.

         version ::= API version ID; e.g., 'V1'.

         resource ::= "analysismetrics" | "experiment" |
                        "fileserver" | "globalconfig" | "libmetrics" |
                        "location" | "plugin" |"qualitymetrics" |
                        "referencegenome" | "results" |"rig" | "tfmetrics"

         key ::= Specific resource instance name or identifier;
```

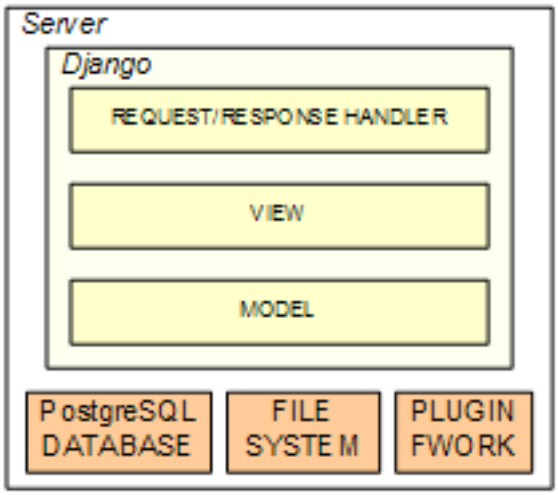
```
                                    Example: '12' for experiment 'id' = 12.

            filter ::= (resource-dependent)

            value ::= (filter-dependent)

            qualifier ::= contains | icontains | iexact | month | day |
                          in | range | endswith | iregex | regex | exact |
                          isnull | search | gt | istartswith | startswith |
                          gte | lt | week_day | iendswith | lte | year
```

Example:

```
http://ionuser:ionuser@ionwest.itw/rundb/api/v1/experiment
        ?format=json&expName__startswith=R_2013_06&order_by=date
```

Django

Django is a web application framework that implements a RESTful API using the Tastypie framework. Modern web-like services are typically implemented using the Model-View-Controller architectural pattern, which is similarly implemented by Django using the Python programming language. The following figure shows a simplified Django architecture, resident on the server with a backend database and other resources.



- The view component handles UI presentation tasks. In addition to supporting the REST API, Django also implements the Torrent Browser UI. In Django, the concept of view is implemented using templates.
- Django also includes a *request/response handler* to handle the HTTP request and transmit protocol, mapping JSON- or XML-formatted data to Python objects. HTTP requests and responses are another mechanism into the model view.
- The *model* component provides for database item representation, and access to persistent data. Persistent Torrent Server data are maintained in a PostgreSQL database.
- The *controller* component interacts with the view and model components to implement the Ion Torrent business logic of application processing, including view functions that populate the view.

With the addition of the Tastypie framework to Django, the Torrent Server supports a REST API in addition to the user interface.

PostgreSQL

Torrent Server uses a PostgreSQL database for persistent backend data storage. PostgreSQL supports access to the data using standardized SQL.

Django integrates PostgreSQL as the model of an MVC framework and exports an SQL-like interface, using Tastypie, through the REST API.

5.3.4 Links to References

This page provides links to background information about the underlying web service technology.

Note: While we attempt to keep these links current, your mileage may vary.

Technology

HTTP

RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 (IETF)

Hypertext Transfer Protocol (Wikipedia)

REST

Representational State Transfer (Wikipedia)

Architectural Styles and the Design of Network-based Software Architectures (Fielding - dissertation)

RESTful Web Services (O'Reilly Media)

Learn REST: A Tutorial

Django

Django Homepage

Django 1.3 Documentation - QuerySet API reference - Field lookups

Interacting With The API

Django and The MVC pattern in web development

PostgreSQL

PostgreSQL Documentation

Tools

cURL

cURL Documentation and Installation

REST clients

Firefox REST Client

Chrome REST Client

RESTClient

Python REST libraries used in the examples

requests.py API

restful_lib.py

httplib2.py

Python JSON parser used in the examples

simplejson.py

5.3.5 Get started

This section introduces you to the Torrent Suite REST API by giving a simple example of the basic request-response interaction between client and server. But even more involved interactions, such as select and sorting data sets, or writing data to the server use the same fundamental operations described here. (The *Torrent Suite™ Software API Cookbook* builds on this simple example, introducing you to all of the functionality made available by the API.)

Your application needs data. You can programmatically get the data, stored in the PostgreSQL database, by sending requests to the serve following the prescribed format and protocol conventions.

A convenient way to become familiar with sending requests to the server and receiving responses is to use a tool that permits you to work interactively. This section uses the Client URL Library (cURL) command line tool, which you can download and run in most environments. A convenient alternative is a browser-based REST client, but all tools and REST programming libraries work on the same principles, primarily differing in syntax and display modes.

Download and install `cURL` to try the API yourself as you follow along in this tutorial. (Remember to replace the hostname, myhost, with your actual server name, and to replace myusername:mypassword with your username and password.)

Request resource data

A request for data is made on an resource endpoint using an Uniform Resource Identifier (URI). The URI is, effectively, the endpoint address. Depending on how you format the URI, you can get a single resource or a set of resources.

Get all resources

First, we get a set of resources. From the response, we can find out which specific resources are available. The `cURL` command for getting a rig resource has the following format. (Requests and responses are shown on multiple lines to make the individual parts easier to see. You should enter commands on a single line.)

```
curl --user myusername:mypassword
         --header "Content-Type: application/json"
         --location 'http://myhost/rundb/api/v1/rig'
```

The command requires an username and password for authentication, which is the only form of authentication required by the API. The command also specifies the data format in which the server should send the response data: JSON.

The URI `http://myhost/rundb/api/v1/rig` tells the server the location of the desired data. The URI format is similar to the familiar browser URL. Specifying the location without indicating a particular rig, requests the data for all rigs:


```
{
meta: {
        limit: 20,
        next: null,
        offset: 0,
        previous: null,
        total_count: 4
},
objects: [
{
        alarms: { },
        comments: "This is a model PGM. Do not delete.",
        ftppassword: "ionguest",
        ftprootdir: "results",
        ftpserver: "192.168.201.1",
        ftpusername: "ionguest",
        last_clean_date: "",
        last_experiment: "",
        last_init_date: "",
        location: {
                comments: "",
                defaultlocation: true,
                id: 1,
                name: "Home",
                resource_uri: "/rundb/api/v1/location/1/"
        },
        name: "default",
        resource_uri: "/rundb/api/v1/rig/default/",
        serial: null,
        state: "",
        updateflag: false,
        updatehome: "192.168.201.1",
        version: { }
        },
        {
        alarms: { },
        comments: "",
        ftppassword: "ionguest",
        ftprootdir: "results",
        ftpserver: "192.168.201.1",
        ftpusername: "ionguest",
        last_clean_date: "",
        last_experiment: "",
        last_init_date: "",
        location: {
                comments: "",
                defaultlocation: true,
                id: 1,
                name: "Home",
                resource_uri: "/rundb/api/v1/location/1/"
        },
        name: "PGM_test",
```

```
        resource_uri: "/rundb/api/v1/rig/PGM_test/",
        serial: "",
        state: "",
        updateflag: false,
        updatehome: "192.168.201.1",
        version: { }
}
]
```

A list of resource elements, or objects, are returned, and the meta field indicates the total object count is two. Looking at the object name elements, you can see one rig is named PGM_test and the other is named default.

Get a single resource

Knowing the available rigs in the database, you can then request only the data for a single rig, PGM_test, by appending the rig name to the URI in the following way:

```
curl --user myusername:mypassword
         --header "Content-Type: application/json"
         --location 'http://myhost/rundb/api/v1/rig/PGM_test'
```

Only fields defined as primary key fields in the database can be used in this way. Other fields can be used to select resource elements but they are passed as URI parameters. These details are left for more extensive presentation in other API documents.

You can see that the server returns only the data for the PGM_test rig, and no metadata are included in the response:

```
{
alarms: { },
comments: "",
ftppassword: "ionguest",
ftprootdir: "results",
ftpserver: "192.168.201.1",
ftpusername: "ionguest",
last_clean_date: "",
last_experiment: "",
last_init_date: "",
location: {
        comments: "",
        defaultlocation: true,
        id: 1,
        name: "Home",
        resource_uri: "/rundb/api/v1/location/1/"
},
name: "PGM_test",
resource_uri: "/rundb/api/v1/rig/PGM_test/",
serial: "",
state: "",
updateflag: false,
updatehome: "192.168.201.1",
version: { }
}
```

Check for errors

So far, all of the requests have been successful, returning resource data. Requests and responses include a message body and a message header component. You may have noticed that the request specified the JSON data format in the

request header argument. For responses, the data are returned in the message body and the request status, success or some kind of failure, are returned in the message header.

A successful response

Adding the –dump-header headers.txt argument to your cURL command permits you to see the response header:

```
$ curl --user myusername:mypassword
           --dump-header headers.txt
           --header "Content-Type: application/json"
           --location 'http://myhost/rundb/api/v1/rig/PGM_test'
```

Viewing the file `headers.txt` confirms the successful response received for the earlier commands:

```
HTTP/1.1 200 OK
Date: Thu, 30 Jun 2011 20:09:17 GMT
Server: Apache/2.2.14 (Ubuntu)
Content-Type: application/json; charset=utf-8
Transfer-Encoding: chunked
```

The key header item is the HTTP status code in the first line, which has a value of 200. All 200-series status codes indicate a successful operation.

A request failure

To demonstrate an unsuccessful operation, enter a cURL command with a URI for a rig that does not exist, for instance `PGM_xyz`. (Review the response when you request data for all rigs, and you can see there is no rig named `PGM_xyz`).

```
$ curl --user myusername:mypassword
           --dump-header headers.txt
           --header "Content-Type: application/json"
           --location 'http://myhost/rundb/api/v1/rig/PGM_xyz'
```

The returned status code is 410, which means the resource element does not exist:

```
HTTP/1.1 410 Gone
Date: Thu, 30 Jun 2011 20:13:38 GMT
Server: Apache/2.2.14 (Ubuntu)
Content-Type: text/html; charset=utf-8
Vary: Accept-Encoding
Content-Length: 0
```

Next steps

Continue using `cURL` or a REST client to experiment with the API, consulting the *Ion Torrent™ Server API Quick Reference* for the various ways of requesting data.

When you are sufficiently familiar with basic API usage, read the *Torrent Suite™ Software API Cookbook* for more in-depth tutorials.

Topics in this list provide background and conceptual information on parameters, requests, responses, and endpoints:

5.3.6 Parameters

These required and optional parameters specify the format of the data sent and received and throttle the transferred data set.

For the GET method, options are appended as parameters to the URI with a question mark (?) and multiple parameters are separated by an ampersand (&), as shown in the following example:

```
http://myhost/rundb/api/v1/experiment?format=json&order_by=date
```

For the PUT, POST and DELETE methods, options are specified in the request header, as shown in the following cURL example:

```
--header "Content-Type: application/json"
```

The following tables indicate to which REST method(s) the option applies:

- C = Create (POST)
- R = Read (GET)
- U = Update (PUT)
- D = Delete (DELETE)

Topics on this page:

Required parameter

Parameter	C	R	U	D	Usage	Description
format	X	X	X	X	format=json	The `format` parameter specifies the format of sent data and the expected format of received data: • `json` = JSON-formatted data • `xml` = XML-formatted data (not supported currently)

Example syntax:

```
http://myhost.itw//rundb/api/v1/experiment?format=json
```

Optional parameters

Parameter	C	R	U	D	Example	Description
limit		X			limit=100	The `limit` parameter specifies the maximum number of elements to be returned in the response. Use with the `offset` parameter to indicate a range.
offset		X			offset=25	The zero-based `offset` parameter specifies the first data element to return in the response, for `limit` or the default number of elements.
order-by		X			order_by=date	The `order_by` parameter alphanumerically orders the received data set by the specified field name. The default is to sort in ascending order. A minus (-) symbol in front of the sort field reverses the sort order.
<field>		X	X	X	name=testRun1	Any field, except `resource_uri` can be included in the parameter list. The request matches on any and all resources whose field value equals the parameter value (or qualified value). Field parameters can be qualified with partially matching values using Filter Qualifiers (see *Filter Qualifiers*).

Example syntax:

```
http://myhost.itw//rundb/api/v1/experiment?format=json&limit=100

http://myhost.itw//rundb/api/v1/experiment?format=json&offset=50

http://myhost.itw//rundb/api/v1/experiment?format=json&offset=10&limit=25

http://myhost.itw//rundb/api/v1/experiment?format=json&library__startswith=e_coli

http://myhost.itw//rundb/api/v1/experiment?format=json&library__startswith=e_coli&order_by=date
```

5.3.7 Request and Response Headers

HTTP messages are composed of a header and message body containing the data.

Header lines provide information about the request or response, or about the object sent in the message body. A header line is ASCII text in the form "header-name:value". The "header-name" is not case-sensitive although the "value" may be. A header may have as many header lines as needed.

Section 14 of RFC 2616 Fielding, et al., Hypertext Transfer Protocol – HTTP/1.1 defines header lines.

For requests, the Torrent Suite™ Software API usually only requires headers to be specified for PUT, POST, and DELETE methods. For GET requests, the necessary information is provided as a parameter. Response messages from the server always include a header.

Topics on this page:

- *Request header*
- *Response header*

Request header

A request header might be as simple as the following example:

```
Content-Type:application/json
```

This specifies the format of the transmitted data, if it is not specified as a parameter.

Response header

A response header is usually similar to the following example:

```
Status Code:200 OK
Date: Wed, 01 Jun 2011 22:38:49 GMT
Server: Apache/2.2.2.14 (Ubuntu)
Content-Type: application/json; charset=utf-8
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
```

The `Status Code` is of particular interest, and the `Content-Type` header line describes the data format.

5.3.8 HTTP Response Codes

The HTTP response status codes are specified in section 6 of the RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 standard.

The following table lists the more common status codes you may receive:

Method	Code	Meaning	Resolution
	200	Resource exists	–
	301	Permanently moved	–
GET	401	Authorization error	–
	404	Not found	–
	410	No longer exists	–
	200	Resource replaced	–
	201	Resource created	–
	204	No response	–
	301	Redirect	Usually, missing the trailing slash '/' in the URI
PUT/POST	400	Invalid data	–
	401	Authorization error	–
	409	Resource state conflict	–
	500	Internal error	Django error; enable debugging to evaluate (see *Debug API Errors*)
	501	Method not implemented	–
	200	Resource deleted	–
DELETE	400	Resource not deleted	–
	401	Authorization error	–

5.3.9 Endpoint Metadata

Each endpoint response includes metadata similar to the following example:

```
"meta":
{
        "limit": 20,
        "next": null,
        "offset": 0,
        "previous": null,
        "total_count": 1
}
```

Meta-data field	Description
limit	The maximum number of elements to return in the response. A response may include fewer elements if the resource has less than `limit` elements. Default = 20.
offset	The number of the first element to return in the response. The total number returned is determined by the `limit` field. Default = 0.
to-tal_count	The total number of resource elements. This is not the number of elements actually returned in the response unless the response included all resource elements.
next	The URI of the next element, following the last element returned in the response. If all resource elements are included in the response, this value is null.
previous	The URI of the previous resource element, if the first element returned in the response is not the first resource element. If all resource elements are included in the response, this value is null.

Tip: Recall that an endpoint request does not specify a particular resource element. Example:

`http://myhost/rundb/api/v1/results`

The following topics provide more detail about implementing and maintaining REST applications:

5.3.10 API Implementation

This section contains the following API implementation topics:

URI Structure

The Uniform Resource Identifier (URI) is a address string used to locate a named resource or group of resources over a network. The REST communication model uses a URI to identify resources with each resource having a unique URI.

The URI structure is hierarchical so a group of resources can be identified by specifying the location at a higher level in the address hierarchy, without specifying a particular resource.

Simplified syntax

The URI has a similar appearance to the Uniform Resource Locator (URL) you use in your browser. In fact, URL is a subset of the URI specification. REST URI components include:

- The protocol (HTTP)
- domain
- path
- resource ID

Examples

REST URI	Description
`http://myhost/rundb/api/v1/rig/testPGM`	Reference the specific rig resource named `testPGM` located at domain `myhost` and path `rundb/api/v1/rig/`.
`http://myhost/rundb/api/v1/rig/`	Reference all resources located at domain `myhost` and path `rundb/api/v1/rig/`.
`rundb/api/v1/rig/testPGM`	Reference the specific rig resource named `testPGM` on `localhost`.

The URI may include your login username and password, and port number, as part of the host name.

Parameters

Parameters can be appended to the URI to qualify requests.

The following symbols are used to specify URI parameters:

Symbol	Description
?	Introduces the parameter list.
&	Parameter separator character.

Parameters are usually specified as name-value pairs.

Example

```
http://myhost/rundb/api/v1/rig/?format=json&limit=0&order_by=name
```

Syntax specification

```
http://<username>:<password>@]<host>/rundb/api/<version>
        [/<resource>[/<key>]?format=json
        [[&<filter>{=<value> | __<qualifier>=<value>}]...]
        [&order_by=[-]<filter>\]

        username ::= User login name.

        password ::= User login password.

        host ::= Host server name.

        version ::= API version ID; e.g., 'V1'.

        resource ::= "analysismetrics" | "experiment" |
                     "fileserver" | "globalconfig" | "libmetrics" |
                     "location" | "plugin" |"qualitymetrics" |
                     "referencegenome" | "results" |"rig" | "tfmetrics"

        key ::= Specific resource instance name or identifier;
                Example: '12' for experiment 'id' = 12.

        filter ::= (resource-dependent)

        value ::= (filter-dependent)

        qualifier ::= contains | icontains | iexact | month | day |
                      in | range | endswith | iregex | regex | exact |
                      isnull | search | gt | istartswith | startswith |
                      gte | lt | week_day | iendswith | lte | year
```

Authentication

Currently, logging in to the server using HTML/cookie authentication (username/password) is the only required authentication.

On making your first request to the server, you are prompted for your username and password credentials, if they are not included in the URI.

Responses

Responses are HTTP messages from the server in response to application requests. The response includes header and body parts to the message.

> **Tip:** Using a REST client in your browser is a convenient way to visualize the response header and body.

Response header

A response header contains information similar to the following example:

```
Status Code:200 OK
Date: Wed, 01 Jun 2011 22:38:49 GMT
Server: Apache/2.2.2.14 (Ubuntu)
Content-Type: application/json; charset=utf-8
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
```

`Status Code` is of particular interest, indicating if the requested succeeded or not. Responses with 200-series status codes indicate the request was handled successfully. Another informative data item in the header is `Content-Type`, which describes the data format. Currently, only the JSON data format is supported.

Response body

For a GET request, the response body contains the data representing the resource or resource set requested. These data are formatted as specified by the `Content-Type` property in the header.

The following code snippet shows an example response body returned by the server, in response to a location request (`http://myhost/rundb/api/v1/location/?format=json`):

```
{
"meta": {"limit": 20, "next": null, "offset": 0, "previous": null, "total_count": 1},
 "objects":
   [{"comments": "",
         "id": "2",
         "name": "IonWest",
         "resource_uri": "/rundb/api/v1/location/2/"
   }]
}
```

A response body is not returned by the server for PUT, POST and DELETE requests.

Error Messages

API errors can be classified as either communication or internal server processing errors:

- Communication errors can be caused by connectivity issues, failure to authenticate or message protocol errors, which are listed in section 6 of the RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 standard.
- Internal processing errors are those detected by the Django framework and include software bugs, database anomalies and invalid request formats.

The *HTTP Response Codes* section of the *Torrent Server REST API v1 Resources* document lists errors that might typically occur when using the Torrent Suite™ Software API, and their possible causes.

Topics on this page:

Note: Errors detected by the server are reported in the status code returned with any create, read, update or delete request. To help determine the cause of the error, the returned message body contains additional information about the error. For internal, 500-series server errors, in particular, Django returns very detailed information about the error cause and location. Debugging must be enable to receive the more extensive Django error reports.

Using a REST client with your browser to interactively exercise the API provides an easy way to investigate error conditions, by examining the status code and message body returned with the request. When writing an application program, it is helpful to include exception handling around API calls to catch possible errors, and to always check the returned status code before continuing to process a response.

HTTP status codes

Successful API requests have a return status code of 200 or 201.

All other status codes indicate some kind of error condition, and after some experience using HTTP the cause of the error can often readily be determined. To demonstrate an error condition, the following example omits the question mark (?) symbol preceding request parameters, effectively making a request on an undefined resource:

```
http://myhost/rundb/api/v1/rigformat=json
```

If you try sending this request, you will see that the server returns a 404 status code, indicating the resource was `Not Found`. Additionally, the response message body contains a server-specific HTML page for the 404-type error.

Debug internal errors

You can receive additional, detailed information in the response message body for internal, 500-series errors, which are detected by the Django framework, by turning on debugging.

Follow these steps to turn on debugging:

1. On your server, open the settings.py file for editing, found at the following location:

   ```
   /opt/ion/iondb/settings.py
   ```

2. Set the DEBUG environment variable to true:

   ```
   DEBUG = True
   ```

3. Restart Apache:

   ```
   sudo /etc/init.d/apache2 restart
   ```

Thereafter, whenever a 500-series error occurs, a message similar to the following example is provided in the response message with detailed information about the type of error and the source code location where the error was detected:

```
<Response [500]>

{"error_message": "The format indicated 'application/x-www-form-urlencoded' had
no available deserialization method. Please check your ``formats`` and ``content
_types`` on your Serializer.", "traceback": "Traceback (most recent call last):\
n\n  File \"/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line
 175, in wrapper\n    response = callback(request, *args, **kwargs)\n\n  File \"
/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line 343, in dis
patch_detail\n    return self.dispatch('detail', request, **kwargs)\n\n  File \"
/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line 364, in dis
patch\n    response = method(request, **kwargs)\n\n  File \"/usr/local/lib/pytho
n2.6/dist-packages/tastypie/resources.py\", line 1007, in put_detail\n    deseri
alized = self.deserialize(request, request.raw_post_data, format=request.META.ge
t('CONTENT_TYPE', 'application/json'))\n\n  File \"/usr/local/lib/python2.6/dist
-packages/tastypie/resources.py\", line 325, in deserialize\n    return self._me
ta.serializer.deserialize(data, format=request.META.get('CONTENT_TYPE', 'applica
tion/json'))\n\n  File \"/usr/local/lib/python2.6/dist-packages/tastypie/seriali
zers.py\", line 159, in deserialize\n    raise UnsupportedFormat(\"The format in
dicated '%s' had no available deserialization method. Please check your ``format
s`` and ``content_types`` on your Serializer.\" % format)\n\nUnsupportedFormat:
The format indicated 'application/x-www-form-urlencoded' had no available deseri
alization method. Please check your ``formats`` and ``content_types`` on your Se
rializer.\n"}
```

Data Formats

The JavaScript Object Notation (JSON) is currently the only supported data format for exchanging data with Torrent Server.

JSON is a lightweight data interchange format organized as a list, or list of lists, of name-value pairs.

You must specify JSON as the desired data format for sending and receiving data as either a URI parameter or in the request header:

URI example

```
http://myhost/rundb/api/v1/results/122/?format=json
```

Request header example

```
Content-Type:application/json
```

5.3.11 Work with Django

The following topics apply to the Django API framework:

Key

Primary key

A key is the resource field defined as the primary key in the database schema. Most often, the key is the id field. Refer to the resource tables in the Database Schema Reference for the primary key definition of each resource.

Any particular resource element can be retrieved by specifying the key value following the resource name in the URI. For example to request an experiment resource element whose id field value is 3, use the following syntax:

```
http://myhost/rundb/api/v1/experiment/3
```

This returns the single experiment.

Use keys to get a resource

You can get a set of resources by adding the set keyword to the URI and specifying the primary key value of each resource, separated by a semicolon. For example:

```
http://myhost/rundb/api/v1/experiment/set/3;5
```

This returns two resource elements whose id field values are 3 and 5.

Fields

Fields, as used in this documentation, are the resource schema columns.

A field is a name-value pair, where each field has a name identifier and may or may not have an associated value assigned. For example:

```
"chipBarcode": "AA0000000",
"chipType": "\"314R\"",
"date": "2013-05-07T17:48:53",
```

Field names are useful in the API for selecting resource elements and sorting results.

Select resources using field names

You can identify one or more resources by specifying the field name and the value to match on:

```
http://myhost/rundb/api/v1/location?format=json&name=myLab
```

In this example, all location resource elements whose name field is `myLab` are returned by the server.

See the Basic Filtering and Qualifying Filters sections of the *Torrent Suite™ Software API Cookbook* for a more detailed description of selecting resources using field names.

Sort resources by field names

You sort resources returned by the server using the order_by key word and assigning it a field name. All resource elements returned are sorted by the value of the specified field. The following example sorts experiment elements by date:

```
http://myhost/rundb/api/v1/experiment?format=json&order_by=date
```

See the Sort Response Output section of the *Torrent Suite™ Software API Cookbook* for more detailed information about using field names to sort results.

Filter Qualifiers

When specifying selection criteria using field-value pair parameters, these qualifiers can be applied to field values to filter results.

See QuerySet API reference - Field lookups for a detailed description of each qualifier in the Django documentation.

Syntax

```
<field>__<qualifier>=<value>
```

Note: There are *two* underscore characters between the field and qualifier names.

Qualifiers

Important: The datatype of the specified value(s) must match the expected field datatype.

exact	*iexact*	*contains*	*icontains*
in	*gt*	*gte*	*lt*
lte	*startswith*	*istartswith*	*endswith*
iendswith	*range*	*year*	*month*
day	*week_day*	*isnull*	*search*
regex	*iregex*	–	–

exact Return element(s) whose value for the specified field is an exact match (case-sensitive). Example:

```
library__exact=E_coli_dh10b
```

iexact Return element(s) whose value for the specified field is an exact match (case-insensitive). Example:

```
library__iexact=E_coli_dh10b
```

contains Return element(s) whose value for the specified field contains the specified value (case-sensitive). (See also *search*.) Example:

```
library__contains=E_coli
```

icontains Return element(s) whose value for the specified field contains the specified value (case-insensitive). Example:

```
library__icontains=E_coli
```

in Return element(s) whose value for the specified field is in the specified list. Example:

```
id__in=[1, 3, 4]
```

gt Return element(s) whose value for the specified field is greater than the specified value. Example:

```
name__gt=B4
```

gte Return element(s) whose value for the specified field is greater than or equal to the specified value. Example:

```
name__gte=B4
```

lt Return element(s) whose value for the specified field is less than the specified value. Example:

```
name__lt=B24
```

lte Return element(s) whose value for the specified field is less than or equal to the specified value. Example:

```
name__lte=B24
```

startswith Return element(s) whose value for the specified field starts with the specified value (case-sensitive). Example:

```
library__startswith=E_coli
```

istartswith Return element(s) whose value for the specified field starts with the specified value (case-insensitive). Example:

```
library__istartswith=E_coli
```

endswith Return element(s) whose value for the specified field ends with the specified value (case-sensitive). Example:

```
library__endswith=dh10b
```

iendswith Return element(s) whose value for the specified field ends with the specified value (case-insensitive). Example:

```
library__iendswith=dh10b
```

range Return element(s) whose value for the specified field is in the range of the specified values (inclusive). Example:

```
date__range=(start_date, end_date))
```

year Return element(s) whose value for the specified date/datetime field matches the specified year. Example:

```
date__year=2013
```

month Return element(s) whose value for the specified date/datetime field matches the specified integer month. Example:

```
date__month=6
```

day Return element(s) whose value for the specified date/datetime field matches the specified integer day-of-month. Example:

```
date__day=17
```

week_day Return element(s) whose value for the specified date/datetime field matches the specified integer day-of-week, where Sunday = 1 and Saturday = 7. Example:

```
date__week_day=2
```

isnull Return element(s) whose value for the specified field is NULL, where True = NULL and False = NOT NULL. Example:

```
date__isnull=True
```

search Return element(s) whose value for the specified field contains the specified value. (See also *contains*.) Example:

```
comment__search="my test results"
```

regex Return element(s) whose value for the specified field matches the regular expression (case-sensitive). Example:

```
name__regex=r'^(An?|The) +'
```

iregex Return element(s) whose value for the specified field matches the regular expression (case-insensitive). Example:

```
name__iregex=r'^(an?|the) +'
```

5.3.12 Debug API Errors

API faults can be classified as either communication faults, backend server processing errors or client application program errors:

- Communication errors can be caused by connectivity issues, failure to authenticate or message protocol errors, which are listed in section 6 of the RFC 2616, Hypertext Transfer Protocol – HTTP/1.1 standard.
- Internal processing errors are those detected by the Django framework and include software bugs, database anomalies and invalid request formats.

- Client application program errors can be of many types, which can be detected and reported using common program debugging methods and tools. Some classes of errors deserving particular mention, because of the web application environment are errors related to HTML and JavaScript coding.

The *HTTP Response Codes* section of the *Torrent Server REST API v1 Resources* document lists errors that might typically occur when using the Torrent Suite™ Software API, and their possible causes.

HTTP status codes

Successful API requests return a status code of 200 or 201.

All other status codes indicate some kind of error condition, and after some experience using HTTP the cause of the error can often readily be determined. To demonstrate an error condition, the following example omits the question mark (?) symbol preceding request parameters, effectively making a request on an undefined resource:

```
http://myhost/rundb/api/v1/rigformat=json
```

If you try sending this request, you will see that the server returns a 404 status code, indicating the resource was `Not Found`. Additionally, the response message body contains a server-specific HTML page for the 404-type error.

Additional debugging facilities are available for backend 500-series errors, described below.

Debug backend errors

You can receive additional, detailed information in the response message body for internal, 500-series errors, which are detected by the Django framework, by turning on debugging:

1. On your server, open the settings.py file for editing, found at the following location:

   ```
   /opt/ion/iondb/settings.py
   ```

2. Set the DEBUG environment variable to true:

   ```
   DEBUG = True
   ```

3. Restart Apache:

   ```
   sudo /etc/init.d/apache2 restart
   ```

Thereafter, whenever a 500-series error occurs, a message similar to the following example is provided in the response message with detailed information about the type of error and the source code location where the error was detected:

```
<Response [500]>

{"error_message": "The format indicated 'application/x-www-form-urlencoded' had
no available deserialization method. Please check your ``formats`` and ``content
_types`` on your Serializer.", "traceback": "Traceback (most recent call last):\
n\n  File \"/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line
 175, in wrapper\n    response = callback(request, *args, **kwargs)\n\n  File \"
/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line 343, in dis
patch_detail\n    return self.dispatch('detail', request, **kwargs)\n\n  File \"
/usr/local/lib/python2.6/dist-packages/tastypie/resources.py\", line 364, in dis
patch\n    response = method(request, **kwargs)\n\n  File \"/usr/local/lib/pytho
n2.6/dist-packages/tastypie/resources.py\", line 1007, in put_detail\n    deseri
alized = self.deserialize(request, request.raw_post_data, format=request.META.ge
t('CONTENT_TYPE', 'application/json'))\n\n  File \"/usr/local/lib/python2.6/dist
-packages/tastypie/resources.py\", line 325, in deserialize\n    return self._me
ta.serializer.deserialize(data, format=request.META.get('CONTENT_TYPE', 'applica
tion/json'))\n\n  File \"/usr/local/lib/python2.6/dist-packages/tastypie/seriali
```

```
zers.py\", line 159, in deserialize\n    raise UnsupportedFormat(\"The format in
dicated '%s' had no available deserialization method. Please check your ``format
s`` and ``content_types`` on your Serializer.\" % format)\n\nUnsupportedFormat:
The format indicated 'application/x-www-form-urlencoded' had no available deseri
alization method. Please check your ``formats`` and ``content_types`` on your Se
rializer.\n"}
```

Debug HTML or JavaScript

Two useful browser tools available for debugging suspected HTML and JavaScript problems are Firefox Firebug and Chrome Inspector.

Firefox Firebug

If you are using the Firefox browser, download and install the Firebug plugin.

Once installed, click on the Firebug icon in the status bar to begin debugging your HTML and JavaScript code.

Chrome Inspector

The Chrome Inspector debugger comes pre-installed with the Chrome browser.

To begin debugging, right-click on your HTML page and select Inspect Element from the drop-down menu.

Event logging

The system logs events, which can be a useful debugging tool. Logs are located in the following directory:

```
/var/log/ion/
```

For plugins, the event log has the following name:

```
/var/log/ion/ionPlugin.log
```

5.3.13 Tools

The following tools and programming libraries are useful for developing web applications:

cURL

The Client URL Library (cURL) is a command line utility to transfer files using URL syntax.

The `cURL` utility is useful for interactively making REST requests. Using `cURL` gives you more control and flexibility than using a program while you learn the API and debug the communication part of your application.

You can download and install cURL from http://curl.haxx.se/docs/.

REST Clients

Browser-based REST clients are available for Firefox and Chrome browsers. You can download and install these clients at the following locations:

- Firefox REST Client
- Chrome REST Client

Like the cURL utility, a REST client is useful for interactively making REST requests, giving you more control and flexibility than using a program when learning the API and debugging the communication part of your application.

A Python-based REST client is also available for download and installation at the following location:

- RESTClient

REST Programming Libraries

Most of the examples in these documents demonstrate API functionality using the Python programming language. The following popular Python libraries are available to make authentication and sending requests easier. these libraries generally have similar functionality but differ in call syntax:

- requests.py
- restful_lib.py (deprecated)
- httplib2.py

You can use the library with which you are most familiar.

The API is built on the Django Framework, which implements a Model View Controller (MVC) architecture and supports the REST communication model. Using the Torrent Suite™ Software REST API, client applications operate on resources modelled in the back-end PostgreSQL database. Database, file, and system resources are addressable using a Universal Resource Identifier (URI), and can be created, read, modified, searched, and sorted using parameterized REST methods.

This document provides a semantic description of the API, which complements the API syntax presented in the *Torrent Server REST API v1 Resources*.

5.4 Torrent Suite™ Software API FAQ

Contents

5.4.1 General

What is an API and how do I use it?

An Application Programming Interface (API) provides a well-defined interface to a computing system for accessing system resources and functionality. Torrent Suite provides a REST-ful API, which uses the REST communication model to expose system functionality to client applications. Applications use the API by sending create, read, update and delete requests to the server and processing the responses. API request syntax is described in the *Torrent Server REST API v1 Resources* document and the semantics of the interface are described in the *Torrent Suite™ Software API User Guide*.

What is REST?

Representation State Transfer (REST) is a client-server communication model for transferring representations of resources, which can be database or functional components. REST uses HTTP as the communication mechanism without assuming that resource state is maintained between request messages. The Torrent Suite API uses REST as the underlying communication model between client application programs and server resources. Using a noun-verb abstraction, the REST create, read, update and delete methods are the verbs that operate on resources as nouns. Resources are addressable using a Universal Resource Identifier (URI), which is formatted like a typical Web address.

Which programming languages are supported?

As a REST specification, the API is programming language independent. Any language or tools that provides facilities for HTTP communication using the GET, PUT, POST and DELETE methods can interact with Torrent Server using the API. However, for any particular language used, standard Web programming constraints and conventions apply for the language

What is the difference between POST and PUT?

A common acronym used to describe the primary data storage functions is CRUD (Create-Read-Update-Delete). POST and PUT are used to describe similar HTTP functions: Generally, POST corresponds to the create function, creating a new data item, and PUT corresponds to the update function, for modifying an existing data item. You can, however, use also PUT to create a data item. The fundamental difference between the two functions is that PUT is idempotent but POST is not.

What is JSON?

JavaScript Object Notation (JSON) is a language independent data interchange format using JavaScript conventions. JSON is currently the only data transport format supported by the Torrent Suite™ Software API.

5.4.2 Tips

I noticed that the API query returns 20 results maximum. How do I access the rest of them?

by default, the maximum number of resource items returned is 20. To get more items with a single request, set the limit parameter to a higher value, such as 100:

```
http://myhost/rundb/api/v1/results/?format=json&limit=100
```

To get all items, set limit to zero:

```
http://myhost/rundb/api/v1/results/?format=json&limit=0
```

Be careful because some of the resources on your server may have a very large number of items.

How do I sort the returned results in inverse order?

You can change the way the results are ordered using the order_id query string, inserting a minus (-) symbol before filter name:

```
http://myhost/rundb/api/v1/results/?format=json&limit=0&order_by=-id
```

Why is a 301 HTTP status code returned when I try to write to the server?

The most common cause of a 301, redirect, error is failing to terminate the URI with a trailing slash character. Another common error when writing or deleting a resource is to include the `resource_uri` field in the request body.

What are the previous and next fields in the returned meta data?

If available, these are URI links to the previous and next pages of the resource data, when all of the data was not returned with the current page.

What is the `resource_uri` field?

The `resource_uri` field is the URI for a detailed view of the data item. The `resource_uri` field must not be included in your data structure when writing or deleting a resource, including nested `resource_uri` fields.

How can I retrieve multiple data items with a single request?

The simplest way to retrieve multiple data items with a single request is to specify all of the data items by id, separated by a colon:

```
http://myhost/rundb/api/v1/results/1;3?format=json
```

You can also get multiple results using filters and filter qualifiers to get data items matching specific criteria.

Is there a way to query two fields at once? For example, I wanted to find only those experiments that used a particular library and chip type.

Try something like:

```
http://myhost/rundb/api/v1/experiment?
  format=json&library=e_coli_dh10b&chipType__contains=316
```

This returns a list of experiments with the e_coli_dh10b reference library whose chip type contains 316. The ampersand (&) notation separates multiple request parameters.

When I PUT or POST to a resource, do I always need to provide all of the fields in the JSON string in the message body?

In general, yes. The only exception is the `rig` resource, where the following fields can up read and updated individually:

- state
- last_init_date
- last_clean_date
- last_experiment
- version
- alarms

5.5 API Acronyms and Abbreviations

Acronym	Expanded	Description
API	Application Programming Interface	A well-defined programmatic interface to a software component. The programmatic interface to the Torrent Server database and software functionality is through the API described in this document.
AJAX	Asynchronous JavaScript and XML	JavaScript examples in this documentation use AJAX to make requests to the server using the REST API. (wikipedia AJAX)
CRUD	Create, Read, Update. Delete	This common acronym lists the REST API methods. (wikipedia CRUD)
CSS	Cascading Style Sheets	Style sheets are used in some examples in this documentation. (http://www.w3.org/Style/CSS)
HTTP	Hypertext Transfer Protocol	A request-response communication model for client-server computing architectures, commonly known as the underlying communication mechanism of the Web. The protocol defines message for create, read, update and delete (CRUD), among others, used by the REST communication model. (http://tools.ietf.org/html/rfc2616)
JSON		JSON is a language-independent data interchange format using JavaScript conventions. JSON is currently the only data transport format supported by the Torrent Suite™ Software API. See XML. (http://www.json.org/)
JSONP	JSON with Padding	JSONP is an extension of the JSON data format to support cross-domain data transfer. JavaScript restricts such cross-domain operations using simple JSON, for security reasons. Implementing applications using the REST API with JavaScript necessitates using JSONP.
MVC	Model View Controller	MVC describes a common architectural pattern used to implement web-based application servers, where core functionality is decomposed into three components: model, view, and controller. The model part is the back-end database component, the view part is the UI or presentation component, and the controller part implements the application logic. While Django may not strictly implemented the MVC pattern, it is conceptually similar enough to discuss the Torrent Suite architectural framework in MVC terms. (wikipedia)
REST	Representation State Transfer	A client-server communication model for transferring representations of resources, which can be database or functional components. REST uses HTTP as the communication mechanism without assuming that resource state is maintained between request messages. The Torrent Suite™ Software API uses REST as the underlying communication model between client application programs and server resources. Using a noun-verb abstraction, the REST methods are the verbs that operate on resources as nouns. Resources are addressable by their URI. (wikipedia REST)
RPC	Remote Procedure Call	A Remote Procedure Call is similar to a programming language subroutine call, except that the “call” is to a remote system and the parameters and data associated with the call are contained in a message. A synchronous RPC is also like a subroutine call in that the calling function pends on the response, or completion. REST is a form of RPC.
SGE	Sun Grid Engine	An opensource distributed computing solution from Sun Microsystems that enables multiple computers or servers to be linked. The SGE provides a mechanism for creating and managing a job queue to distribute computing tasks over a cluster of machines, reducing CPU utilization on any single machine. The SGE is used to schedule plugins among compute resources, including running in parallel on a single server as resources permit.
SQL	Structured Query Language	Torrent Server uses a PostgreSQL database for persistent backend data storage. PostgreSQL supports access to the data using standardized SQL. Django integrates PostgreSQL as the model of an MVC framework and exports an SQL-like interface, using Tastypie, through the REST API.
URI	Uniform Resource Identifier	REST resources are addressable by their HTTP URI global identifier. (http://tools.ietf.org/html/rfc3986)
XML	Extensible Markup	A set of rules for encoding documents. XML is one the REST API data transfer formats but is not currently supported by the Torrent Suite™ Software API.

Plugins:

Example: Fastq creator plugin (3.x)

Example: Proton runlevel demonstration (3.x)

Example: Convert 2.x plugin to 3.x plugin

CHAPTER 6

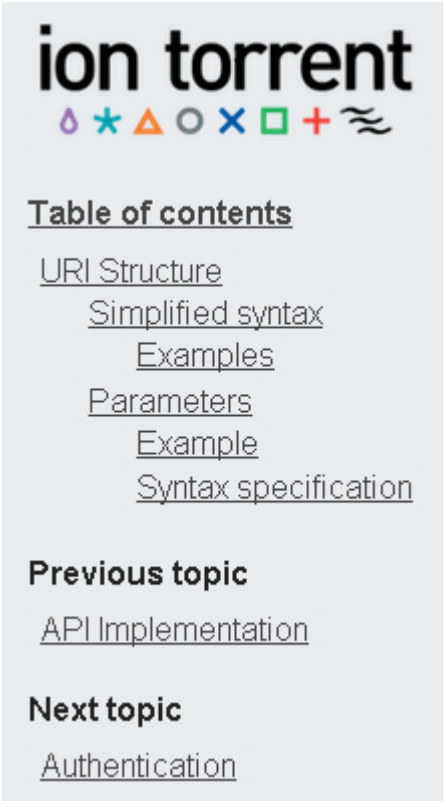
About these documents

See this page for descriptions of the left panel table of contents, the breadcrumbs at the top of each page, and the previous and next navigation links:

6.1 About These Documents

6.1.1 Table of contents

The left panel contains a table of contents for the current page. Click on any entry to jump to that section on the current page:



You can also jump to the previous or next topic using this panel.

6.1.2 Navigation links

Other navigation links to the previous page and next page are available in the right corner of the top and bottom banners:

6.1.3 Breadcrumbs

The banners at both the top and the bottom of each page shows the location of the current page:

You can click on any step of the breadcrumb trail to jump to that section.