Transmission Documentation

Release 0.7

Erik Svensson

October 11, 2011

CONTENTS

CHAPTER

ONE

INTRODUCTION

This is **transmissionrpc**. This module helps using Python to connect to a Transmission JSON-RPC service. transmissionrpc is compatible with Transmission 1.3 and later.

CHAPTER

TWO

GETTING STARTED

Transmission is available at Python Package Index. To install the transmissionrpc python module use easy_install or pip.

```
$ easy_install transmissionrpc
```

Note: You might need administrator privileges to install python modules.

You may also download the tarball from Python Package Index. Untar and install.

```
$ tar -xzf transmissionrpc-0.7.tar.gz
$ cd transmissionrpc-0.7
$ python setup.py install
```

2.1 Dependecies

transmissionrpc has the following dependencies.

- Python >= 2.5.
- simplejson >= 1.7.1 or Python >= 2.6.

If Python 2.6 or later is detected the standard library json implementation will be used.

2.2 Report a problem

Problems with transmissionrpc should be reported through the issue tracker at bitbucket. Please look through the existing issues before opening a new issue.

CHAPTER

THREE

INSTALLING FROM SOURCE

3.1 The source code

Transmission is hosted at bitbucket using mercurial. To get a working copy, run

```
$ hg clone http://www.bitbucket.org/blueluna/transmissionrpc/
```

The source code will be fetched and stored the directory transmissionrpc.

Then install the module using

```
$ python setup.py install
```

Or if you wish to further develop transmissionrpc itself use

```
$ python setup.py develop
```

This will link this directory to the library as transmissionrpc.

3.2 Poking around

Now that transmissionrpc has been installed, run python and start to poke around. Following will create a RPC client and list all torrents.

```
>>> import transmissionrpc
>>> tc = transmissionrpc.Client('localhost', port=9091)
>>> tc.list()
```

List will return a dictionary of Torrent object indexed by their id. You might not have any torrents yet. This can be remedied by adding an torrent.

```
>>> tc.add_url('http://releases.ubuntu.com/8.10/ubuntu-8.10-desktop-i386.iso.torrent')
{1: <Torrent 1 "ubuntu-8.10-desktop-i386.iso">}
>>> tc.info(1)
{1: <Torrent 1 "ubuntu-8.10-desktop-i386.iso">}
```

As you saw, the add_url and info calls also returns a dictionary with `{<id>: <Torrent>, ...}`. More information about a torrent transfer can be found in the Torrent object.

```
>>> torrent = tc.info(1)[1]
>>> torrent.name
'ubuntu-8.10-desktop-i386.iso'
>>> torrent.hashString
```

```
'33820db6dd5e5928d23bc811bbac2f4ae94cb882'
>>> torrent.status
'downloading'
>>> torrent.eta
datetime.timedelta(0, 750)
>>> for key, value in torrent.fields.iteritems():
...     print(key, value)
...
('comment', 'Ubuntu CD releases.ubuntu.com')
```

The last call will list all known data provided by Transmission.

Well, we weren't that interested in Ubuntu so lets stop the transfer and the remove it.

```
>>> tc.stop(1)
>>> tc.remove('33820db6dd5e5928d23bc811bbac2f4ae94cb882')
```

See what we did there? most methods in transmissionrpc can take both torrent id and torrent hash when referring to a torrent. lists and sequences are also supported.

```
>>> tc.info([2, 'caff87b88f50f46bc22da3a2712a6a4e9a98d91e'])
{2: <Torrent 2 "ubuntu-8.10-server-amd64.iso">, 3: <Torrent 3 "ubuntu-8.10-alternate-amd64.iso">}
>>> tc.info('1:3')
{2: <Torrent 2 "ubuntu-8.10-server-amd64.iso">, 3: <Torrent 3 "ubuntu-8.10-alternate-amd64.iso">}
```

Continue to explore and have fun! For more in depth information read the module reference.

3.3 A note about debugging information

If you ever need to see what's going on inside transmissionrpc, you can change the logging level of transmissionrpc. This is done with these easy steps

```
>>> import logging
>>> logging.getLogger('transmissionrpc').setLevel(logging.DEBUG)
```

Note that this will produce a whole lot of output! Other levels are (listed by severity)

- `logging.ERROR`
- `logging.WARNING`
- `logging.INFO`
- `logging.DEBUG`

The default logging level of transmissionrpc is `logging.ERROR`.

CHAPTER

FOUR

MODULE REFERENCE

4.1 `transmissionrpc` — Module reference

This documentation will not describe all RPC fields in detail. Please refer to the RPC specification for more information on RPC data.

Contents

4.1.1 Exceptions

class `transmissionrpc.`**`TransmissionError`**(*message=''*, *original=None*)

This exception is raised when there has occured an error related to communication with Transmission. It is a subclass of Exception.

`original`

The original exception.

class `transmissionrpc.`**`HTTPHandlerError`**(*httpurl=None*, *httpcode=None*, *httpmsg=None*, *httpheaders=None*, *httpdata=None*)

This exception is raised when there has occured an error related to the HTTP handler. It is a subclass of Exception.

`url`

The requested url.

`code`

HTTP error code.

`message`

HTTP error message.

`headers`

HTTP headers.

data
HTTP data.

4.1.2 Torrent object

Torrent is a class holding the information received from Transmission regarding a bittorrent transfer. All fetched torrent fields are accessible through this class using attributes. The attributes use underscore instead of hyphen in the names though. This class has a few convenience attributes using the torrent information.

Example:

```
>>> import transmissionrpc
>>> t = transmissionrpc.Torrent({'id': 1, 'comment': 'My torrent', 'addedDate': 1232281019})
>>> t.comment
'My torrent'
>>> t.date_added
datetime.datetime(2009, 1, 18, 13, 16, 59)
>>>
```

class transmissionrpc.**Torrent**(*client*, *fields*)
Torrent is a class holding the data raceived from Transmission regarding a bittorrent transfer. All fetched torrent fields are accessable through this class using attributes. This class has a few convenience properties using the torrent data.

date_active
Get the attribute "activityDate" as datetime.datetime.

date_added
Get the attribute "addedDate" as datetime.datetime.

date_done
Get the attribute "doneDate" as datetime.datetime.

date_started
Get the attribute "startDate" as datetime.datetime.

eta
Get the "eta" as datetime.timedelta.

files()
Get list of files for this torrent.

This function returns a dictionary with file information for each file. The file information is has following fields:

```
{
        <file id>: {
                'name': <file name>,
                'size': <file size in bytes>,
                'completed': <bytes completed>,
                'priority': <priority ('high'|'normal'|'low')>,
                'selected': <selected for download>
        }

        ...
}
```

format_eta()
Returns the attribute *eta* formatted as a string.

- If eta is -1 the result is 'not available'
- If eta is -2 the result is 'unknown'
- Otherwise eta is formatted as <days> <hours>:<minutes>:<seconds>.

priority
Get the priority as string. Can be one of 'low', 'normal', 'high'.

progress
Get the download progress in percent.

ratio
Get the upload/download ratio.

status
Returns the torrent status. Is either one of 'check pending', 'checking', 'downloading', 'seeding' or 'stopped'. The first two is related to verification.

update(*other*)
Update the torrent data from a Transmission JSON-RPC arguments dictinary

4.1.3 Session object

Session is a class holding the session data for a Transmission session.

Access the session field can be done through attributes. The attributes available are the same as the session arguments in the Transmission RPC specification, but with underscore instead of hyphen. `download-dir` -> `download_dir`.

class `transmissionrpc.`**Session**(*fields = {}*)
fields should be an dictionary build from session information from an Transmission JSON-RPC result.

`Session.`**update**(*other*)
Updates the Session object with data from *other*.

other should be a Session object or session information from an Transmission JSON-RPC result.

4.1.4 Client object

This is it. This class implements the JSON-RPC protocol to communicate with Transmission.

Torrent ids

Many functions in Client takes torrent id. A torrent id can either be id or hashString. When suppling multiple id's it is possible to use a list mixed with both id and hashString.

Timeouts

In Python 2.6 it is possible to supply a timeout to a HTTP request. This is accessible through transmissionrpc by either changing the timeout property of a Client object or supply the named argument `timeout` in most methods of Client. The default timeout is 30 seconds.

class `transmissionrpc.`**Client**(*address='localhost', port=9091, user=None, password=None, http_handler=None, timeout=None*)
Client is the class handling the Transmission JSON-RPC client protocol.

add(*data*, *timeout=None*, ***kwargs*)

Add torrent to transfers list. Takes a base64 encoded .torrent file in data. Additional arguments are:

Argument	RPC	Description
bandwidthPriority	8 -	Priority for this transfer.
download_dir	1 -	The directory where the downloaded contents will be saved in.
filename	1 -	A filepath or URL to a torrent file or a magnet link.
files_unwanted	1 -	A list of file id's that shouldn't be downloaded.
files_wanted	1 -	A list of file id's that should be downloaded.
metainfo	1 -	The content of a torrent file, base64 encoded.
paused	1 -	If True, does not start the transfer when added.
peer_limit	1 -	Maximum number of peers allowed.
priority_high	1 -	A list of file id's that should have high priority.
priority_low	1 -	A list of file id's that should have low priority.
priority_normal	1 -	A list of file id's that should have normal priority.

add_uri(*uri*, ***kwargs*)

Add torrent to transfers list. Takes a uri to a torrent, supporting all uri's supported by Transmissions torrent-add 'filename' argument. Additional arguments are:

Argument	RPC	Description
bandwidthPriority	8 -	Priority for this transfer.
download_dir	1 -	The directory where the downloaded contents will be saved in.
files_unwanted	1 -	A list of file id's that shouldn't be downloaded.
files_wanted	1 -	A list of file id's that should be downloaded.
paused	1 -	If True, does not start the transfer when added.
peer_limit	1 -	Maximum number of peers allowed.
priority_high	1 -	A list of file id's that should have high priority.
priority_low	1 -	A list of file id's that should have low priority.
priority_normal	1 -	A list of file id's that should have normal priority.

blocklist_update(*timeout=None*)

Update block list. Returns the size of the block list.

change(*ids*, *timeout=None*, ***kwargs*)

Change torrent parameters for the torrent(s) with the supplied id's. The parameters are:

Argument	RPC	Replaced by	Description
bandwidthPriority	5 -		Priority for this transfer.
downloadLimit	5 -		Set the speed limit for download in Kib/s.
downloadLimited	5 -		Enable download speed limiter.
files_unwanted	1 -		A list of file id's that shouldn't be downloaded.
files_wanted	1 -		A list of file id's that should be downloaded.
honorsSessionLimits	5 -		Enables or disables the transfer to honour the upload limit set in the session.
ids	1 -		Local download location.
peer_limit	1 -		The peer limit for the torrents.
priority_high	1 -		A list of file id's that should have high priority.
priority_low	1 -		A list of file id's that should have normal priority.
priority_normal	1 -		A list of file id's that should have low priority.
seedIdleLimit	10 -		Seed inactivity limit in minutes.
seedIdleMode	10 -		Seed inactivity mode. 0 = Use session limit, 1 = Use transfer limit, 2 = Disable limit.
seedRatioLimit	5 -		Seeding ratio.
seedRatioMode	5 -		Which ratio to use. 0 = Use session limit, 1 = Use transfer limit, 2 = Disable limit.
speed_limit_down	1 - 5	downloadLimit	Set the speed limit for download in Kib/s.
speed_limit_down_enabled	1 - 5	downloadLimited	Enable download speed limiter.
speed_limit_up	1 - 5	uploadLimit	Set the speed limit for upload in Kib/s.
speed_limit_up_enabled	1 - 5	uploadLimited	Enable upload speed limiter.
trackerAdd	10 -		Array of string with announce URLs to add.
trackerRemove	10 -		Array of ids of trackers to remove.
trackerReplace	10 -		Array of (id, url) tuples where the announce URL should be replaced.
uploadLimit	5 -		Set the speed limit for upload in Kib/s.
uploadLimited	5 -		Enable upload speed limiter.

Note: transmissionrpc will try to automatically fix argument errors.

get_files(*ids=None*, *timeout=None*)

Get list of files for provided torrent id(s). If ids is empty, information for all torrents are fetched. This function returns a dictonary for each requested torrent id holding the information about the files.

```
{
        <torrent id>: {
                <file id>: {
                        'name': <file name>,
                        'size': <file size in bytes>,
                        'completed': <bytes completed>,
                        'priority': <priority ('high'|'normal'|'low')>,
                        'selected': <selected for download (True|False)>
                }

                ...
        }
```

```
        ...
}
```

get_session(*timeout=None*)
Get session parameters

info(*ids=None*, *arguments=None*, *timeout=None*)
Get detailed information for torrent(s) with provided id(s).

list(*timeout=None*)
list all torrents

locate(*ids*, *location*, *timeout=None*)
Locate torrent data at the location.

move(*ids*, *location*, *timeout=None*)
Move torrent data to the new location.

port_test(*timeout=None*)
Tests to see if your incoming peer port is accessible from the outside world.

reannounce(*ids*, *timeout=None*)
Reannounce torrent(s) with provided id(s)

remove(*ids*, *delete_data=False*, *timeout=None*)
remove torrent(s) with provided id(s). Local data is removed if delete_data is True, otherwise not.

rpc_version
Get the Transmission RPC version. Trying to deduct if the server don't have a version value.

session_stats(*timeout=None*)
Get session statistics

set_files(*items*, *timeout=None*)
Set file properties. Takes a dictonary with similar contents as the result of *get_files*.

```
{
        <torrent id>: {
                <file id>: {
                        'priority': <priority ('high'|'normal'|'low')>,
                        'selected': <selected for download (True|False)>
                }

                ...
        }

        ...
}
```

set_session(*timeout=None*, ***kwargs*)
Set session parameters. The parameters are:

Argument	RPC	Replaced by	Description
alt_speed_down	5 -		Alternate session download speed limit (in Kib/s).
alt_speed_enabled	5 -		Enables alternate global download speed limiter.
alt_speed_time_begin	5 -		Time when alternate speeds should be enabled. Minutes after m
alt_speed_time_day	5 -		Enables alternate speeds scheduling these days.
alt_speed_time_enabled	5 -		Enables alternate speeds scheduling.
alt_speed_time_end	5 -		Time when alternate speeds should be disabled. Minutes after

Continued on next page

Table 4.1 – continued from previous page

Argument	RPC	Replaced by	Description
alt_speed_up	5 -		Alternate session upload speed limit (in Kib/s).
blocklist_enabled	5 -		Enables the block list
cache_size_mb	10 -		The maximum size of the disk cache in MB
dht_enabled	6 -		Enables DHT.
download_dir	1 -		Set the session download directory.
encryption	1 -		Set the session encryption mode, one of required, prefer
idle_seeding_limit	10 -		The default seed inactivity limit in minutes.
idle_seeding_limit_enabled	10 -		Enables the default seed inactivity limit
incomplete_dir	7 -		The path to the directory of incomplete transfer data.
incomplete_dir_enabled	7 -		Enables the incomplete transfer data directory. Otherwise data
lpd_enabled	9 -		Enables local peer discovery for public torrents.
peer_limit	1 - 5	peer-limit-global	Maximum number of peers
peer_limit_global	5 -		Maximum number of peers
peer_limit_per_torrent	5 -		Maximum number of peers per transfer
peer_port	5 -		Peer port.
peer_port_random_on_start	5 -		Enables randomized peer port on start of Transmission.
pex_allowed	1 - 5	pex-enabled	Allowing PEX in public torrents.
pex_enabled	5 -		Allowing PEX in public torrents.
port	1 - 5	peer-port	Peer port.
port_forwarding_enabled	1 -		Enables port forwarding.
rename_partial_files	8 -		Appends ".part" to incomplete files
script_torrent_done_enabled	9 -		Whether or not to call the "done" script.
script_torrent_done_filename	9 -		Filename of the script to run when the transfer is done.
seedRatioLimit	5 -		Seed ratio limit. 1.0 means 1:1 download and upload ratio.
seedRatioLimited	5 -		Enables seed ration limit.
speed_limit_down	1 -		Download speed limit (in Kib/s).
speed_limit_down_enabled	1 -		Enables download speed limiting.
speed_limit_up	1 -		Upload speed limit (in Kib/s).
speed_limit_up_enabled	1 -		Enables upload speed limiting.
start_added_torrents	9 -		Added torrents will be started right away.
trash_original_torrent_files	9 -		The .torrent file of added torrents will be deleted.

Note: transmissionrpc will try to automatically fix argument errors.

---

start (*ids*, *timeout=None*)
start torrent(s) with provided id(s)

stop (*ids*, *timeout=None*)
stop torrent(s) with provided id(s)

timeout
HTTP query timeout.

verify (*ids*, *timeout=None*)
verify torrent(s) with provided id(s)

CHAPTER

FIVE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

t