PicoTorrent

Release 0.15.0

Apr 26, 2018

Contents

1 Overview **1**
- 1.1 Configuration . . . 1
- 1.2 How-to . . . 3
- 1.3 Keyboard shortcuts . . . 3
- 1.4 Plugins . . . 3
- 1.5 Privacy . . . 7

CHAPTER 1

Overview

PicoTorrent is a tiny BitTorrent client for Windows. It is written in C++ using Rasterbar-libtorrent and the raw Win32 API.

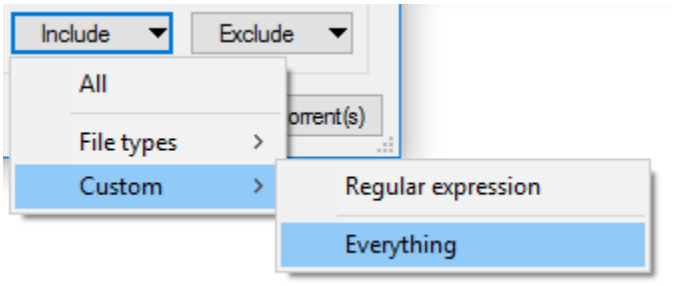
Note: In case you find errors in this documentation you can help by sending pull requests!

1.1 Configuration

PicoTorrent is stores its configuration in a JSON file and depending on if PicoTorrent was installed or not the file will reside at different locations.

- If installed, the file can be found at `%APPDATA%/PicoTorrent/PicoTorrent.json`.
- If not installed, the file will be placed next to `PicoTorrent.exe`.

1.1.1 File filters



File filters are used in the *Add torrent* dialog and allows a user to store custom include/exclude filters.

Each filter contains a *name* and a *pattern*. The name is visible in the include/exclude context menus, and the pattern is a regular expression which is matched on each file name.

```
{
  "file_filters": [
    { "name": "Everything", "pattern": ".*" }
  ]
}
```

1.1.2 Advanced settings

The following is an annotated example file with all settings that are not available for configuration from the GUI (ie. you need to edit the JSON file directly).

```
{
  // If a user chooses to ignore an update, that version number is stored
  // in this field.
  "ignored_update": "",

  // The API URL where PicoTorrent will check for the latest release.
  "update_url": "https://api.github.com/repos/picotorrent/picotorrent/releases/latest
↪",

  "session": {
     // The max number of simultaneous torrents to allow in 'checking'.
     "active_checking": 1,

     // The max number of torrents to announce to the DHT.
     "active_dht_limit": 88,

     // The max number of active downloads.
     "active_downloads": 3,

     // The max number of active torrents across states.
     "active_limit": 15,

     // The max number of torrents that are allowed to be *loaded* at
     // any given time. PicoTorrent will at times load and unload inactive
     // torrents from memory to conserve memory usage.
     "active_loaded_limit": 100,

     // The max number of torrents to announce to the local network over the
     // local service discovery protocol.
     "active_lsd_limit": 60,

     // The max number of active seeds.
     "active_seeds": 5,

     // Set to `true` to enable anonymous mode
     "anonymous_mode": false,

     // The max number of torrents to announce to their trackers.
     "active_tracker_limit": 1600,

     // Set to `true` to enable DHT.
     "enable_dht": true,

     // Set to `true` to enable LSD (Local Service Discovery).
     "enable_lsd": true,
```

```
    // Set to `true` to require encryption for incoming connections.
    "require_incoming_encryption": false,

    // Set to `true` to require encryption for outgoing connections.
    "require_outgoing_encryption": false
  },

  "ui": {
    // Set to `false` to skip the Add Torrent dialog
    "show_add_torrent_dialog": true
  }
}
```

1.2 How-to

1.2.1 Use regular expressions to filter files

The *Add torrent dialog* has the option to include or exclude files within a torrent based on a regular expression.

The regular expression should be ECMAScript compatible, and will match against the whole torrent file path.

Match against a file extension

```
.*\.iso$
```

File name should contain X

This will match against any file that has *md5sum* in its file name.

```
.*md5sum.*
```

1.3 Keyboard shortcuts

PicoTorrent has a few keyboard shortcuts which simplifies various tasks.

- `Control+A`. Select all torrents.
- `Delete`. Removes the selected torrents while keeping the downloaded data.
- `Shift+Delete`. Removes the selected torrents and the downloaded data.

1.4 Plugins

PicoTorrent has an advanced C++ API making it possible to extend the application in various ways by listening to events and reacting.

1.4.1 WebSocket

The WebSocket plugin adds a WebSocket server to PicoTorrent where torrent updates are broadcasted to connected clients. It is currently unprotected and therefore not suitable for internet traffic.

Configuring the WebSocket

Configuration is stored in the `PicoTorrent.json` file.

```
{
  "websocket": {
    // Set to `true` to enable the WebSocket plugin.
    "enabled": false,

    // The port which the WebSocket server will listen on.
    "listen_port": 7676
  }
}
```

Messages

The message API is JSON based and each message contains the *type* hash which tells the consumer what type of message it is.

The first message sent is the *pico_state* message, which contains the full state for PicoTorrent.

The *pico_state* message

The *pico_state* message is sent when a client connection is opened and contains the full PicoTorrent state (all managed torrents) as well as some version information.

Example

```
{
  "type": "pico_state",

  "version_info": {
    // The WebSocket API version.
    "api_version": 1
  },

  // An array of torrent objects managed by PicoTorrent.
  "torrents": [ ... ]
}
```

The *torrent_added* message

The *torrent_added* message is sent when a torrent is successfully added to the session and contains a *torrent* object representing the newly added torrent.

Example

```
{
  "type": "torrent_added",

  // A torrent object representing the added torrent.
  "torrent": { ... }
}
```

The *torrent_finished* message

The *torrent_finished* message is sent when all non-skipped files of a torrent has finished downloading.

Example

```
{
  "type": "torrent_finished",

  // A torrent object representing the finished torrent.
  "torrent": { ... }
}
```

The *torrent_removed* message

The *torrent_removed* message is sent when a torrent is removed from the session. It contains an info hash identifying the removed torrent.

Example

```
{
  "type": "torrent_removed",

  // An info hash identifying the the removed torrent.
  "info_hash": "..."
}
```

The *torrent_updated* message

The *torrent_updated* message is sent approximately once per second and contains a list of torrents which have been updated (i.e transfer rates, progress, etc).

Example

```
{
  "type": "torrent_updated",

  // A list of torrents which has been updated.
```

```
  "torrents": [ ... ]
}
```

Objects

Each message may contain one or more well-defined objects.

The *torrent* object

The torrent object represents a torrent in the session and contains various fields with statistics and data.

Example

```
{
  // The torrent info hash.
  "info_hash": "...",

  // The name of the torrent.
  "name": "...",

  // The position of this torrent in the queue. Will be -1 for torrents
  // not in the queue.
  "queue_position": 1,

  // The size of the torrent in bytes.
  "size": 1024,

  // The progress of the current operation.
  "progress": 0.43,

  // The current status of the torrent.
  // -1: unknown
  //  0: checking resume data
  //  1: complete
  //  2: downloading
  //  3: downloading (checking)
  //  4: downloading (forced)
  //  5: downloading metadata
  //  6: downloading paused
  //  7: downloading (queued)
  //  8: downloading (stalled)
  //  9: error
  // 10: uploading
  // 11: uploading (checking)
  // 12: uploading (forced)
  // 13: uploading (paused)
  // 14: uploading (queued)
  // 15: uploading (stalled)
  "status": 2,

  // The number of seconds until the torrent finishes its current operation.
  "eta": 100,

```

```
  // The current transfer rates (up/down) for the torrrent.
  "dl_rate": 1024,
  "ul_rate": 1024,

  // The number of seeds connected.
  "seeds_connected": 1,

  // The total number of seeds.
  "seeds_total": 12,

  // The number of non-seeds connected.
  "nonseeds_connected": 4,

  // The total number of non-seeds.
  "nonseeds_total": 54
}
```

1.5 Privacy

To support users with an increased need for privacy, we've made it easy to use a proxy for BitTorrent traffic. You can also enable *anonymous mode* to further obfuscate your identity.

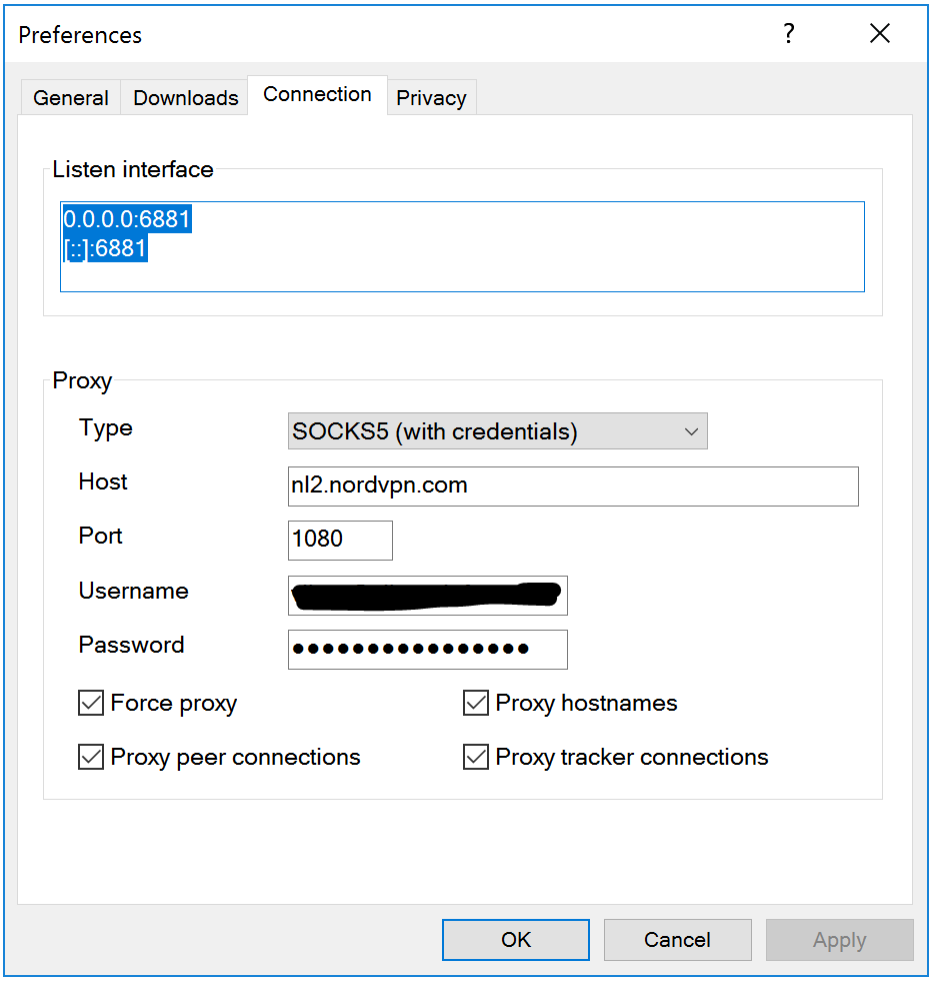
1.5.1 Picking a VPN provider

There are many VPN providers, and you can even host your own somewhere. However, for those looking for a quick and easy way of getting started, we've compiled a small list of VPN providers we like.

- NordVPN
- Mullvad

1.5.2 Configuring a proxy

Configuring a proxy is easy. Just open the *Preferences*, go to the *Connection* tab and enter your details.



1.5.3 Proxy settings

- *Force proxy* disables any connections not going through the proxy.
- *Proxy hostnames* will perform hostname lookups through the proxy.
- *Proxy peer connections* will route all peer connections through the proxy.
- *Proxy tracker connections* will route all tracker connections through the proxy.