

---

# PicoTorrent

*Release 0.15.0*

**Apr 26, 2018**



---

## Contents

---

<b>1</b>	<b>Overview</b>	<b>1</b>
1.1	Configuration . . . . .	1
1.2	How-to . . . . .	3
1.3	Keyboard shortcuts . . . . .	3
1.4	Plugins . . . . .	3
1.5	Privacy . . . . .	7



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!](#)

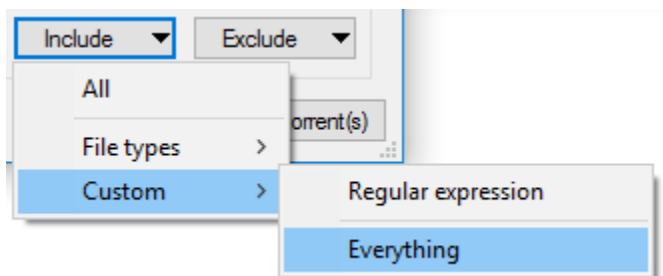
---

## 1.1 Configuration

PicoTorrent 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.

```
1 {
2   "file_filters": [
3     { "name": "Everything", "pattern": ".*" }
4   ]
5 }
```

### 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).

```
1 {
2   // If a user chooses to ignore an update, that version number is stored
3   // in this field.
4   "ignored_update": "",
5
6   // The API URL where PicoTorrent will check for the latest release.
7   "update_url": "https://api.github.com/repos/picotorrent/picotorrent/releases/latest
8   ↪",
9
10  "session": {
11    // The max number of simultaneous torrents to allow in 'checking'.
12    "active_checking": 1,
13
14    // The max number of torrents to announce to the DHT.
15    "active_dht_limit": 88,
16
17    // The max number of active downloads.
18    "active_downloads": 3,
19
20    // The max number of active torrents across states.
21    "active_limit": 15,
22
23    // The max number of torrents that are allowed to be *loaded* at
24    // any given time. PicoTorrent will at times load and unload inactive
25    // torrents from memory to conserve memory usage.
26    "active_loaded_limit": 100,
27
28    // The max number of torrents to announce to the local network over the
29    // local service discovery protocol.
30    "active_lsd_limit": 60,
31
32    // The max number of active seeds.
33    "active_seeds": 5,
34
35    // Set to `true` to enable anonymous mode
36    "anonymous_mode": false,
37
38    // The max number of torrents to announce to their trackers.
39    "active_tracker_limit": 1600,
40
41    // Set to `true` to enable DHT.
42    "enable_dht": true,
43
44    // Set to `true` to enable LSD (Local Service Discovery).
45    "enable_lsd": true,
```

```

45     // Set to `true` to require encryption for incoming connections.
46     "require_incoming_encryption": false,
47
48     // Set to `true` to require encryption for outgoing connections.
49     "require_outgoing_encryption": false
50 },
51
52 "ui": {
53     // Set to `false` to skip the Add Torrent dialog
54     "show_add_torrent_dialog": true
55 }
56 }
57 }

```

## 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.

```
1 {
2   "websocket": {
3     // Set to `true` to enable the WebSocket plugin.
4     "enabled": false,
5
6     // The port which the WebSocket server will listen on.
7     "listen_port": 7676
8   }
9 }
```

### 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

```
1 {
2   "type": "pico_state",
3
4   "version_info": {
5     // The WebSocket API version.
6     "api_version": 1
7   },
8
9   // An array of torrent objects managed by PicoTorrent.
10  "torrents": [ ... ]
11 }
```

#### 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

```

1 {
2   "type": "torrent_added",
3
4   // A torrent object representing the added torrent.
5   "torrent": { ... }
6 }

```

### The *torrent\_finished* message

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

### Example

```

1 {
2   "type": "torrent_finished",
3
4   // A torrent object representing the finished torrent.
5   "torrent": { ... }
6 }

```

### 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

```

1 {
2   "type": "torrent_removed",
3
4   // An info hash identifying the the removed torrent.
5   "info_hash": "...
6 }

```

### 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

```

1 {
2   "type": "torrent_updated",
3
4   // A list of torrents which has been updated.

```

```
5  "torrents": [ ... ]
6  }
```

### 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

```
1  {
2  // The torrent info hash.
3  "info_hash": "...",
4
5  // The name of the torrent.
6  "name": "...",
7
8  // The position of this torrent in the queue. Will be -1 for torrents
9  // not in the queue.
10 "queue_position": 1,
11
12 // The size of the torrent in bytes.
13 "size": 1024,
14
15 // The progress of the current operation.
16 "progress": 0.43,
17
18 // The current status of the torrent.
19 // -1: unknown
20 // 0: checking resume data
21 // 1: complete
22 // 2: downloading
23 // 3: downloading (checking)
24 // 4: downloading (forced)
25 // 5: downloading metadata
26 // 6: downloading paused
27 // 7: downloading (queued)
28 // 8: downloading (stalled)
29 // 9: error
30 // 10: uploading
31 // 11: uploading (checking)
32 // 12: uploading (forced)
33 // 13: uploading (paused)
34 // 14: uploading (queued)
35 // 15: uploading (stalled)
36 "status": 2,
37
38 // The number of seconds until the torrent finishes its current operation.
39 "eta": 100,
40 }
```

```
41 // The current transfer rates (up/down) for the torrent.
42 "dl_rate": 1024,
43 "ul_rate": 1024,
44
45 // The number of seeds connected.
46 "seeds_connected": 1,
47
48 // The total number of seeds.
49 "seeds_total": 12,
50
51 // The number of non-seeds connected.
52 "nonseeds_connected": 4,
53
54 // The total number of non-seeds.
55 "nonseeds_total": 54
56 }
```

## 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.

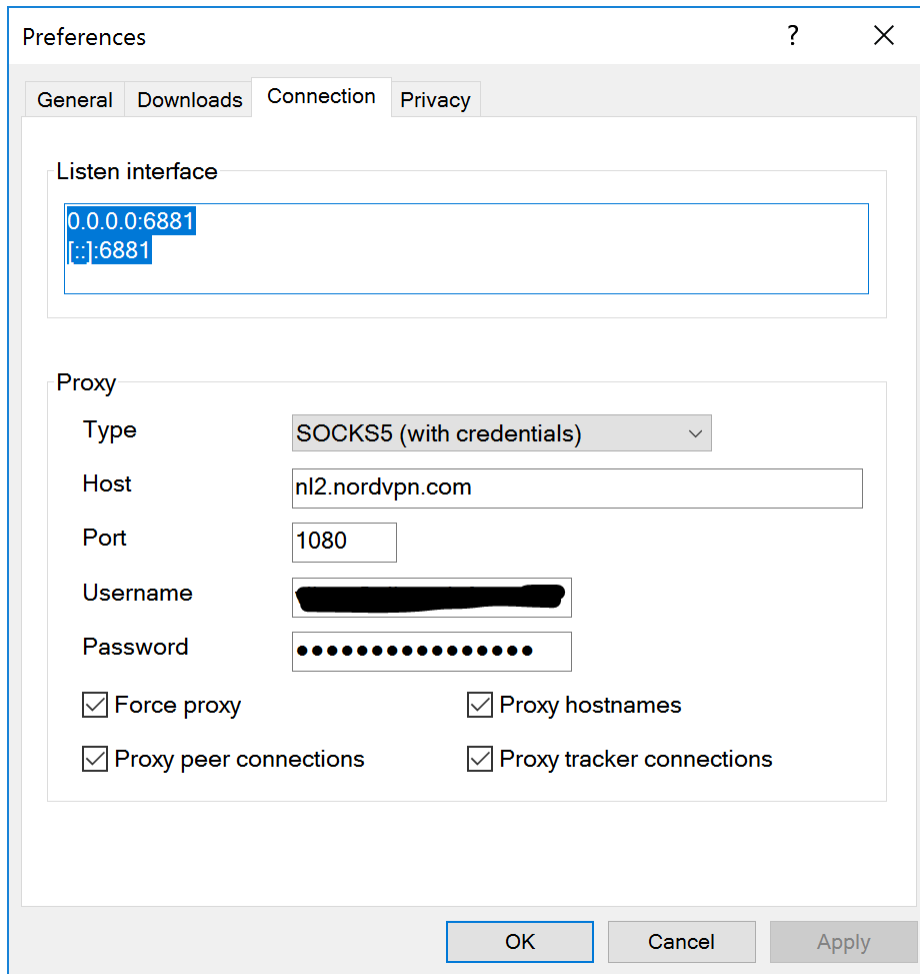
### 1.5.1 Picking a VPN provider

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.