PHP BitTorrent Library API

Introduction

This documentation provides summary information on the functionality of m4dm4n's PHP BitTorrent Library (http://m4dm4n.homelinux.net:8086/download/detail/3). The library aims to provide a simple interface to programatically interact with and process data from BitTorrent Tracker Servers and bencoded Torrent files.

For ease of navigation, the document has been divided into the individual files provided as part of the library. These files are logically seperated into their seperate functions and some are even partially functionally independent from one another.

An example (very basic) implementation of the library is provided in example.php, which demonstrates the use of the library to process and query information on a torrent file.

Much of the information in this document draws on the unofficial BitTorrent specification found here: http://wiki.theory.org/BitTorrentSpecification. This document does not deal specifically with the BitTorrent protocol, except as it pertains to the library itself. If you require further information on the protocol, it is best to consult the unofficial specification at the address provided.

The library is open source and so may be freely distributed and modified, on the condition that the credit information is left intact.

Errors

The potential for errors to occur in a networked environment is quite high, so it is imporant to understand the error handling policy of the library. In all situations where an error occurs, the function will report an E_USER_WARNING level error and return false. By these means it is possible to discretely handle errors in the library and suppress the warnings by use of the “@” prefix (See: http://www.php.net/@).

bt.config.php

This file defines settings used primarily (at this stage, version 0.6) by the caching functions of *tracker.php*. **In order for caching to function properly you must configure this file to work on your server.** Caching of these functions improves the load time of the library by a very significant amount and reduces network traffic at the tracker and the local server. A misconfigured implementation of this library may result in abuse of tracker servers, causing a ban which prevents queries for a period of time, as well as creating a significant page load delay.

The configuration constants are as follows.

Constant	Description
BT_IMPLICIT_CACHE	Turn caching on or off. If set to *true*, the library will automatically apply an internal caching policy to all functions which require network activity.
BT_CACHE_PATH	The location to store cached data. **It is very important to set this or caching will not work.** This directory must be writeable by PHP and it is recommended you provide a seperate directory specifically for these files, as the number of subdirectories and files is quite large.
BT_CACHE_DURATION	The default amount of time before cached data is considered stale. Note that this does not apply to the *tracker_get_announce*, which uses the interval information provided by the tracker itself.

bencode.reader.php

Class: BEncodeReader		
Method	**Parameters**	**Description**
BEncodeReader	([String : $filename])	Main constructor with optional *$filename* parameter indicating a file to open and parse.
setData	(String : $data)	Sets the data to be parsed by the reader.
readNext	(*None*)	Recursively parses the data string set through *setData*, returning the next element read. In most cases, this will fully parse a bencoded message in the form of a .torrent file or from a tracker, returning it as a series of nested arrays. Returns *false* if an error occurs in parsing.
readNextDictionary	(*None*)	**Internal.**
readNextList	(*None*)	**Internal.**
readNextString	(*None*)	**Internal.**
readNextInteger	(*None*)	**Internal.**

torrent.php

Class: Torrent		
Method	**Parameters**	**Description**
Torrent	(String : $filename)	Main constructor. *$filename* indicates a .torrent file to parse. A failure to parse the file will be flagged in the $error field and produce an E_USER_WARNING error.
Field	**Type**	**Description**
$announce	String	The primary tracker address.
$announceList	Array(Array(String) | String)	*Optional.* A multi-dimensional array of the backup tracker list. The structure of this list is defined according to the tier extension to the protocol. Full documentation here: http://home.elp.rr.com/tur/multitracker-spec.txt
$createdBy	String	*Optional.* The program or author of the original .torrent file.
$creationDate	Integer	*Optional.* The creation time of the .torrent, given as a Unix timestamp.
$encoding	String	*Optional, undocumented*. The filename encoding format (ASCII, UTF-8, etc).
$name	String	In *single file mode*, this indicates the name of the file. In *multiple file mode* (this is not part of the standard), it indicates the name of the directory in which to store the files.
$length	Integer	*Single-file mode only.* This indicates the length of the file in bytes.
$md5sum	String	*Single-file mode only.* This indicates the md5 hash of the

		file.
$files	Array(TorrentFile)	*Multi-file mode only.* An array of TorrentFile objects containing the details of the individual files in the torrent.
$pieceLength	Integer	The number of bytes in each piece.
$pieces	String	A concatenation of 20 byte SHA1 values for each piece in the torrent.
$comment	String	*Optional*. A comment about the torrent by the author.
$private	Boolean	*Optional (defaults to false).* If the torrent is private, the client must not use external sources of peers, using only the trackers specified in the torrent. Otherwise, other peer sources such as DHT are acceptable.
$infoHash	String	Urlencoded SHA1 hash of the "info" section of the torrent file. This is necessary for identification to most trackers and used by the *tracker* functions.
$totalSize	Integer	The total size of the torrent files (in bytes).
$modifiedBy	String	*Optional*. Some torrent index sites (such as isoHunt) include this tag to indicate that they have added trackers or otherwise modified the original torrent file.
$error	Boolean	Flag indicating whether there was an error in parsing the torrent file.

Class: TorrentFile

Field	Type	Description
$md5sum	String	This indicates the md5 hash of the file.
$name	String	The full path of the file relative to the download directory.
$length	Integer	The length of the file in bytes.

tracker.php

These functions deal with the querying of trackers for information on a particular *Torrent*. Note that as of version 0.6 all functions which engage in network activity (*tracker_scrape_all*, *tracker_scrape* and *tracker_get_announce*) employ a caching procedure, configured by *bt.config.php*.

Function	Parameters	Description
tracker_scrape_summarise	(Array : $scrape_results)	Takes the return value of *tracker_scrape_all* and returns a single *ScrapeResult* object summarising all of the data.
tracker_scrape_all	(Torrent : $torrent, [Integer : $timeout = 5, Boolean : $force_refresh = false])	"Scrapes" all of the trackers in the specified *$torrent* and returns the result as an associative array of *ScrapeResult* objects, with the keys being the tracker addresses. *$timeout* indicates how long to try connecting to each tracker before giving up. *$force_refresh* indicates whether to ignore any cached data and reload from the trackers.
tracker_scrape	(Torrent : $torrent,	"Scrapes" the specified *$tracker* or the

	[String : $tracker = $torrent->announce, Integer : $timeout = 5, Boolean : $force_refresh = false])	default tracker for the *$torrent*. *$timeout* indicates how long to try connecting to the tracker before giving up. *$force_refresh* indicates whether to ignore any cached data and reload from the tracker. Returns a *ScrapeResult* object.
tracker_get_scrape_address	(String : $announce)	Converts an "announce" address for a tracker into a "scrape" address (See: http://wiki.theory.org/BitTorrentSpecification#Tracker_.27scrape.27_Convention). This is used by the tracker_scrape_* functions. Returns a string.
tracker_get_announce	(Torrent : $torrent, [String : $tracker = $torrent->announce, Integer : $timeout = 5, Integer : $max_peers = 200, Boolean : $force_refresh = false])	Impersonates a BitTorrent client (currently Azureus 3.1.1.0) in order to retrieve announce information from *$tracker* or the primary tracker from *$torrent*, primarily containing a peer list (maximum length specified by *$max_peers*). *$timeout* indicates how long to try connecting to the tracker before giving up. *$force_refresh* indicates whether to ignore any cached data and reload from the tracker. Returns an *AnnounceResult* object. This function's caching policy takes into account the information provided by the tracker on the minimum query interval (See: *AnnounceResult*).
tracker_cache_exists	(Torrent : $torrent, String : $tracker, String : $type)	**Internal.**
tracker_cache_get	(Torrent : $torrent, String : $tracker, String : $type)	**Internal.**
tracker_cache_store	(Torrent : $torrent, String : $tracker, Mixed : $return_value, String : $type)	**Internal.**
tracker_cache_filename	(Torrent : $torrent, String : $tracker, String : $type)	**Internal.**

Class: ScrapeResult

Field	Type	Description
$seeds	Integer	The number of "seeders"; peers with complete copies of the torrent.
$leeches	Integer	The number of "leechers"; peers who are downloading the torrent.
$downloads	Integer	The number of times the tracker has received a notice of completion of the torrent download from clients.

Class: AnnounceResult

Field	Type	Description
$warning	String	*Optional.* A warning message from the tracker which does not indicate a fatal error.
$interval	Integer	The interval at which clients are advised to query the tracker for new announces.
$minInterval	Integer	*Optional*. The very minimum interval at which clients should query the tracker for new announces.
$trackerId	String	An identifier which the tracker has given to the client for use with later queries. **This is not used by any of the library functions currently.**
$seeds	Integer	The number of “seeders”; peers with complete copies of the torrent.
$leeches	Integer	The number of “leechers”; peers who are downloading the torrent.
$peers	Array(String)	An array of peer IP addresses, in the format: “xxx.xxx.xxx.xxx:xxxx”. The number of peers returned depends on the parameters provided to *tracker_get_announce* and the actual number of peers (whichever is smaller). If the number of peers in the swarm is greater than the amount requested, an arbitrary selection is returned.