subliminal Documentation

Release 0.7.5

Antoine Bertin

July 21, 2015

Contents

1 Providers **3**

2 Usage **5**
2.1 CLI . . . 5
2.2 Library . . . 5

3 How it works **7**

4 License **9**

5 Documentation **11**
5.1 Provider Guide . . . 11

6 API Documentation **13**
6.1 API . . . 13
6.2 Cache . . . 14
6.3 CLI . . . 14
6.4 Exceptions . . . 14
6.5 Providers . . . 15
6.6 Score . . . 16
6.7 Subtitle . . . 17
6.8 Video . . . 18

7 Changelog **21**
7.1 0.7.5 . . . 21
7.2 0.7.4 . . . 21
7.3 0.7.3 . . . 21
7.4 0.7.2 . . . 21
7.5 0.7.1 . . . 22
7.6 0.7.0 . . . 22
7.7 0.6.2 . . . 22
7.8 0.6.1 . . . 23
7.9 0.6.0 . . . 23
7.10 0.5.1 . . . 23
7.11 0.5 . . . 24
7.12 0.4 . . . 24
7.13 0.3 . . . 24
7.14 0.2 . . . 24
7.15 0.1 . . . 25

Python Module Index 27

Release v0.7.5

Subliminal is a python library to search and download subtitles. It comes with an easy to use CLI (command-line interface) suitable for direct use or cron jobs.

CHAPTER 1

Providers

Subliminal uses multiple providers to give users a vast choice and have a better chance to find the best matching subtitles. Providers are extensible through a dedicated entry point.

- Addic7ed
- OpenSubtitles
- Podnapisi
- TheSubDB
- TvSubtitles

CHAPTER 2

Usage

2.1 CLI

Download english subtitles:

```
$ subliminal -l en -- The.Big.Bang.Theory.S05E18.HDTV.x264-LOL.mp4
1 subtitle downloaded
```

See *subliminal.cli*

2.2 Library

Download best subtitles in French and English for videos less than one week old in a video folder, skipping videos that already have subtitles whether they are embedded or not:

```
from babelfish import Language
from datetime import timedelta
import subliminal

# configure the cache
subliminal.cache_region.configure('dogpile.cache.dbm', arguments={'filename': '/path/to/cachefile.dbm

# scan for videos in the folder and their subtitles
videos = subliminal.scan_videos(['/path/to/video/folder'], subtitles=True, embedded_subtitles=True, a

# download
subliminal.download_best_subtitles(videos, {Language('eng'), Language('fra')}, age=timedelta(week=1))
```

See *subliminal.api*, *scan_videos()* and *scan_video()*

CHAPTER 3

How it works

Subliminal makes use of various libraries to achieve its goal:

- enzyme to detect embedded subtitles in videos and retrieve metadata
- guessit to guess informations from filenames
- babelfish to work with languages
- requests to make human readable HTTP requests
- BeautifulSoup to parse HTML and XML
- dogpile.cache to cache intermediate search data
- charade to detect subtitles' encoding
- pysrt to validate downloaded subtitles

CHAPTER 4

License

MIT

CHAPTER 5

Documentation

5.1 Provider Guide

This guide is going to explain how to add a *Provider* to subliminal

5.1.1 Requirements

When starting a provider you should be able to answer to the following questions:

- What languages does my provider support?
- What are the language codes for the supported languages?
- Does my provider deliver subtitles for episodes? for movies?
- Does my provider require a video hash?

Each response of these questions will help you set the correct attributes for your *Provider*.

5.1.2 Video Validation

Not all providers deliver subtitles for *Episode*. Some may require a hash. The *check()* method does validation against a *Video* object and will return *False* if the given *Video* isn't suitable. If you're not happy with the default implementation, you can override it.

5.1.3 Configuration

API keys must not be configurable by the user and must remain linked to subliminal. Hence they must be written in the provider module.

Per-user authentication is allowed and must be configured at instantiation as keyword arguments. Configuration will be done by the user through the *provider_configs* argument of the *list_subtitles()* and *download_best_subtitles()* functions. No network operation must be done during instantiation, only configuration. Any error in the configuration must raise a *ProviderConfigurationError*.

Beyond this point, if a network error occurs, a *ProviderNotAvailable* exception must be raised and an unexpected behavior must raise a *ProviderError* exception.

5.1.4 Initialization / Termination

Actual authentication operations must take place in the *initialize()* method. If you need anything to be executed when the provider isn't used anymore like logout, use *terminate()*.

5.1.5 Caching policy

To save bandwidth and improve querying time, intermediate data should be cached when possible. Typical use case is when a query to retrieve show ids is required prior to the query to actually search for subtitles. In that case the function that gets the show id from the show name must be cached.

5.1.6 Language

To be able to handle various language codes, subliminal makes use of babelfish Language and converters. You must set the attribute *languages* with a set of supported `babelfish.Language`.

If you cannot find a suitable converter for your provider, you can make one of your own.

5.1.7 Querying

The *query()* method parameters must include all aspects of provider's querying with simple types.

5.1.8 Subtitle

A custom *Subtitle* subclass must be created to represent a subtitle from the provider. It must have relevant attributes that can be used to compute the matches of the subtitle against a *Video* object.

5.1.9 Score computation

To be able to compare subtitles coming from different providers between them, the *compute_matches()* method must be implemented. If guessit is used to extract data from the *Subtitle* subclass, you can use *compute_guess_matches()* as a helper to compute matches between the *Video* and the `guessit.Guess`.

Refer to the *scores* attribute of *Episode* and *Movie* for a list of possible matches.

5.1.10 Unittesting

All possible uses of the *query()* method must be unittested including the uses that produce exceptions other than *ProviderNotAvailable*. The *compute_matches()* is used to validate the unittests.

As it is not possible to unittest all uses of the *list_subtitles()* and *download_subtitle()* methods, unitests are only required to cover most common use cases.

See existing unittests for more details on how to proceed.

CHAPTER 6

API Documentation

If you are looking for information on a specific function, class or method, this part of the documentation is for you.

6.1 API

subliminal.api.**PROVIDERS_ENTRY_POINT = u'subliminal.providers'**
Entry point for the providers

subliminal.api.**list_subtitles**(*videos*, *languages*, *providers=None*, *provider_configs=None*)
List subtitles for *videos* with the given *languages* using the specified *providers*

Parameters

- **videos** (set of *Video*) – videos to list subtitles for
- **languages** (set of babelfish.Language) – languages of subtitles to search for
- **providers** (*list of string or None*) – providers to use for the search, if not all
- **provider_configs** (*dict of provider name => provider constructor kwargs*) – configuration for providers

Returns found subtitles

Return type dict of *Video* => [*Subtitle*]

subliminal.api.**download_subtitles**(*subtitles*, *provider_configs=None*, *single=False*)
Download subtitles

Parameters

- **subtitles** (dict of *Video* => [*Subtitle*]) – subtitles to download
- **provider_configs** (*dict of provider name => provider constructor kwargs*) – configuration for providers
- **single** (*bool*) – download with .srt extension if *True*, add language identifier otherwise

subliminal.api.**download_best_subtitles**(*videos*, *languages*, *providers=None*, *provider_configs=None*, *single=False*, *min_score=0*, *hearing_impaired=False*, *hi_score_adjust=0*)
Download the best subtitles for *videos* with the given *languages* using the specified *providers*

Parameters

- **videos** (set of *Video*) – videos to download subtitles for

- **languages** (set of babelfish.Language) – languages of subtitles to download
- **providers** (*list of string or None*) – providers to use for the search, if not all
- **provider_configs** (*dict of provider name => provider constructor kwargs*) – configuration for providers
- **single** (*bool*) – download with .srt extension if *True*, add language identifier otherwise
- **min_score** (*int*) – minimum score for subtitles to download
- **hearing_impaired** (*bool*) – download hearing impaired subtitles
- **hi_score_adjust** (*int*) – Adjust hearing_impaired_scores if matched.

6.2 Cache

subliminal.cache.**CACHE_VERSION = 2**
Subliminal's cache version

subliminal.cache.**subliminal_key_generator**(*namespace*, *fn*, *to_str=<type 'str'>*)
Add a *CACHE_VERSION* to dogpile.cache's default function_key_generator

subliminal.cache.**region**
The dogpile.cache region

Refer to dogpile.cache's documentation to see how to configure the region

6.3 CLI

6.3.1 subliminal

Command u'subliminal –help' failed: [Errno 2] No such file or directory

6.4 Exceptions

class subliminal.exceptions.**Error**
Base class for exceptions in subliminal

class subliminal.exceptions.**ProviderError**
Exception raised by providers

class subliminal.exceptions.**ProviderConfigurationError**
Exception raised by providers when badly configured

class subliminal.exceptions.**ProviderNotAvailable**
Exception raised by providers when unavailable

class subliminal.exceptions.**InvalidSubtitle**
Exception raised by providers when the downloaded subtitle is invalid

6.5 Providers

class subliminal.providers.**Provider**(***kwargs*)
Base class for providers

If any configuration is possible for the provider, like credentials, it must take place during instantiation

Parameters ****kwargs** – configuration

Raise *ProviderConfigurationError* if there is a configuration error

languages = set([])
Supported BabelFish languages

video_types = (<class 'subliminal.video.Episode'>, <class 'subliminal.video.Movie'>)
Supported video types

required_hash = None
Required hash, if any

initialize()
Initialize the provider

Must be called when starting to work with the provider. This is the place for network initialization or login operations.

Raise *ProviderNotAvailable* if the provider is unavailable

terminate()
Terminate the provider

Must be called when done with the provider. This is the place for network shutdown or logout operations.

Raise *ProviderNotAvailable* if the provider is unavailable

classmethod check(*video*)
Check if the *video* can be processed

The video is considered invalid if not an instance of *video_types* or if the *required_hash* is not present in *Video*'s *hashes* attribute.

Parameters **video** (*Video*) – the video to check

Returns *True* if the *video* and *languages* are valid, *False* otherwise

Return type bool

query(*languages*, **args*, ***kwargs*)
Query the provider for subtitles

This method arguments match as much as possible the actual parameters for querying the provider

Parameters

- **languages** (set of babelfish.Language) – languages to search for
- ***args** – other required arguments
- ****kwargs** – other optional arguments

Returns the subtitles

Return type list of *Subtitle*

Raise *ProviderNotAvailable* if the provider is unavailable

Raise *ProviderError* if something unexpected occured

list_subtitles(*video*, *languages*)
List subtitles for the *video* with the given *languages*

This is a proxy for the *query()* method. The parameters passed to the *query()* method may vary depending on the amount of information available in the *video*

Parameters

- **video** (*Video*) – video to list subtitles for
- **languages** (set of babelfish.Language) – languages to search for

Returns the subtitles

Return type list of *Subtitle*

Raise *ProviderNotAvailable* if the provider is unavailable

Raise *ProviderError* if something unexpected occured

download_subtitle(*subtitle*)
Download the *subtitle*

Parameters **subtitle** (*Subtitle*) – subtitle to download

Returns the subtitle text

Return type string

Raise *ProviderNotAvailable* if the provider is unavailable

Raise *InvalidSubtitle* if the downloaded subtitle is invalid

Raise *ProviderError* if something unexpected occured

6.6 Score

subliminal.score.**get_episode_equations**()
Get the score equations for a *Episode*

The equations are the following:

1. hash = resolution + video_codec + audio_codec + series + season + episode + release_group
2. series = resolution + video_codec + audio_codec + season + episode + 1
3. tvdb_id = series
4. season = resolution + video_codec + audio_codec + 1
5. imdb_id = series + season + episode
6. resolution = video_codec
7. video_codec = 2 * audio_codec
8. title = season + episode
9. season = episode
10. release_group = season
11. audio_codec = 1

Returns the score equations for an episode

Return type list of `sympy.Eq`

`subliminal.score.`**`get_movie_equations`**`()`
Get the score equations for a *Movie*

The equations are the following:

1.hash = resolution + video_codec + audio_codec + title + year + release_group

2.imdb_id = hash

3.resolution = video_codec

4.video_codec = 2 * audio_codec

5.title = resolution + video_codec + audio_codec + year + 1

6.release_group = resolution + video_codec + audio_codec + 1

7.year = release_group + 1

8.audio_codec = 1

Returns the score equations for a movie

Return type list of `sympy.Eq`

6.7 Subtitle

class `subliminal.subtitle.`**`Subtitle`**(*language*, *hearing_impaired=False*)
Base class for subtitle

Parameters

- **`language`** (`babelfish.Language`) – language of the subtitle
- **`hearing_impaired`** (*bool*) – *True* if the subtitle is hearing impaired, *False* otherwise

`compute_matches`(*video*)
Compute the matches of the subtitle against the *video*

Parameters **`video`** (*Video*) – the video to compute the matches against

Returns matches of the subtitle

Return type set

`compute_score`(*video*, *hi_score_adjust=0*)
Compute the score of the subtitle against the *video*

There are equivalent matches so that a provider can match one element or its equivalent. This is to give all provider a chance to have a score in the same range without hurting quality.

•Matching *Video*'s *hashes* is equivalent to matching everything else

•Matching *Episode*'s *season* and *episode* is equivalent to matching *Episode*'s *title*

•Matching *Episode*'s *tvdb_id* is equivalent to matching *Episode*'s *series*

Parameters

- **`video`** (*Video*) – the video to compute the score against
- **`hi_score_adjust`** – adjust hearing impaired matched videos by this value

Returns score of the subtitle

Return type int

subliminal.subtitle.**get_subtitle_path**(*video_path*, *language=None*)
Create the subtitle path from the given *video_path* and *language*

Parameters

- **video_path** (*string*) – path to the video
- **language** (babelfish.Language or None) – language of the subtitle to put in the path

Returns path of the subtitle

Return type string

subliminal.subtitle.**is_valid_subtitle**(*subtitle_text*)
Check if a subtitle text is a valid SubRip format

Returns *True* if the subtitle is valid, *False* otherwise

Return type bool

subliminal.subtitle.**compute_guess_matches**(*video*, *guess*)
Compute matches between a *video* and a *guess*

Parameters

- **video** (*Video*) – the video to compute the matches on
- **guess** (guessit.Guess) – the guess to compute the matches on

Returns matches of the *guess*

Return type set

6.8 Video

subliminal.video.**VIDEO_EXTENSIONS = (u'.3g2', u'.3gp', u'.3gp2', u'.3gpp', u'.60d', u'.ajp', u'.asf', u'.asx', u'.avchd',**
Video extensions

subliminal.video.**SUBTITLE_EXTENSIONS = (u'.srt', u'.sub', u'.smi', u'.txt', u'.ssa', u'.ass', u'.mpl')**
Subtitle extensions

class subliminal.video.**Video**(*name*, *release_group=None*, *resolution=None*, *video_codec=None*, *audio_codec=None*, *imdb_id=None*, *hashes=None*, *size=None*, *subtitle_languages=None*)
Base class for videos

Represent a video, existing or not, with various properties that defines it. Each property has an associated score based on equations that are described in subclasses.

Parameters

- **name** (*string*) – name or path of the video
- **release_group** (*string*) – release group of the video
- **resolution** (*string*) – screen size of the video stream (480p, 720p, 1080p or 1080i)
- **video_codec** (*string*) – codec of the video stream

- **audio_codec** (*string*) – codec of the main audio stream
- **imdb_id** (*int*) – IMDb id of the video
- **hashes** (*dict*) – hashes of the video file by provider names
- **size** (*int*) – byte size of the video file
- **subtitle_languages** (*set*) – existing subtitle languages

class subliminal.video.**Episode**(*name*, *series*, *season*, *episode*, *release_group=None*, *resolution=None*, *video_codec=None*, *audio_codec=None*, *imdb_id=None*, *hashes=None*, *size=None*, *subtitle_languages=None*, *title=None*, *tvdb_id=None*)

Episode *Video*

Scores are defined by a set of equations, see *get_episode_equations()*

Parameters

- **series** (*string*) – series of the episode
- **season** (*int*) – season number of the episode
- **episode** (*int*) – episode number of the episode
- **title** (*string*) – title of the episode
- **tvdb_id** (*int*) – TheTVDB id of the episode

class subliminal.video.**Movie**(*name*, *title*, *release_group=None*, *resolution=None*, *video_codec=None*, *audio_codec=None*, *imdb_id=None*, *hashes=None*, *size=None*, *subtitle_languages=None*, *year=None*)

Movie *Video*

Scores are defined by a set of equations, see *get_movie_equations()*

Parameters

- **title** (*string*) – title of the movie
- **year** (*int*) – year of the movie

subliminal.video.**hash_opensubtitles**(*video_path*)

Compute a hash using OpenSubtitles' algorithm

Parameters **video_path** (*string*) – path of the video

Returns the hash

Return type string

subliminal.video.**hash_thesubdb**(*video_path*)

Compute a hash using TheSubDB's algorithm

Parameters **video_path** (*string*) – path of the video

Returns the hash

Return type string

subliminal.video.**scan_subtitle_languages**(*path*)

Search for subtitles with alpha2 extension from a video *path* and return their language

Parameters **path** (*string*) – path to the video

Returns found subtitle languages

Return type set

subliminal.video.**scan_video**(*path*, *subtitles=True*, *embedded_subtitles=True*, *video=None*)
Scan a video and its subtitle languages from a video *path*

Parameters

- **path** (*string*) – absolute path to the video
- **subtitles** (*bool*) – scan for subtitles with the same name
- **embedded_subtitles** (*bool*) – scan for embedded subtitles

:parm *Video*: optionally specify a video if you've already detected on by other means.

Returns the scanned video

Return type *Video*

Raise ValueError if cannot guess enough information from the path

subliminal.video.**scan_videos**(*paths*, *subtitles=True*, *embedded_subtitles=True*, *age=None*)
Scan *paths* for videos and their subtitle languages

Params paths absolute paths to scan for videos

Parameters

- **subtitles** (*bool*) – scan for subtitles with the same name
- **embedded_subtitles** (*bool*) – scan for embedded subtitles
- **age** (*datetime.timedelta or None*) – age of the video, if any

Returns the scanned videos

Return type list of *Video*

CHAPTER 7

Changelog

7.1 0.7.5

release date: 2015-03-04

- Update requirements
- Remove BierDopje provider
- Add pre-guessed video optional argument in scan_video
- Improve hearing impaired support
- Fix TVSubtitles and Podnapisi providers

7.2 0.7.4

release date: 2014-01-27

- Fix requirements for guessit and babelfish

7.3 0.7.3

release date: 2013-11-22

- Fix windows compatibility
- Improve subtitle validation
- Improve embedded subtitle languages detection
- Improve unittests

7.4 0.7.2

release date: 2013-11-10

- Fix TVSubtitles for ambiguous series
- Add a CACHE_VERSION to force cache reloading on version change

- Set CLI default cache expiration time to 30 days
- Add podnapisi provider
- Support script for languages e.g. Latn, Cyrl
- Improve logging levels
- Fix subtitle validation in some rare cases

7.5 0.7.1

release date: 2013-11-06

- Improve CLI
- Add login support for Addic7ed
- Remove lxml dependency
- Many fixes

7.6 0.7.0

release date: 2013-10-29

WARNING: Complete rewrite of subliminal with backward incompatible changes

- Use enzyme to parse metadata of videos
- Use babelfish to handle languages
- Use dogpile.cache for caching
- Use charade to detect subtitle encoding
- Use pysrt for subtitle validation
- Use entry points for subtitle providers
- New subtitle score computation
- Hearing impaired subtitles support
- Drop async support
- Drop a few providers
- And much more...

7.7 0.6.2

release date: 2012-09-15

- Fix BierDopje
- Fix Addic7ed
- Fix SubsWiki
- Fix missing enzyme import

- Add Catalan and Galician languages to Addic7ed
- Add possible services in help message of the CLI
- Allow existing filenames to be passed without the ./ prefix

7.8 0.6.1

release date: 2012-06-24

- Fix subtitle release name in BierDopje
- Fix subtitles being downloaded multiple times
- Add Chinese support to TvSubtitles
- Fix encoding issues
- Fix single download subtitles without the force option
- Add Spanish (Latin America) exception to Addic7ed
- Fix group_by_video when a list entry has None as subtitles
- Add support for Galician language in Subtitulos
- Add an integrity check after subtitles download for Addic7ed
- Add error handling for if not strict in Language
- Fix TheSubDB hash method to return None if the file is too small
- Fix guessit.Language in Video.scan
- Fix language detection of subtitles

7.9 0.6.0

release date: 2012-06-16 **WARNING:** Backward incompatible changes

- Fix –workers option in CLI
- Use a dedicated module for languages
- Use beautifulsoup4
- Improve return types
- Add scan_filter option
- Add –age option in CLI
- Add TvSubtitles service
- Add Addic7ed service

7.10 0.5.1

release date: 2012-03-25

- Improve error handling of enzyme parsing

7.11 0.5

release date: 2012-03-25 **WARNING:** Backward incompatible changes

- Use more unicode
- New list_subtitles and download_subtitles methods
- New Pool object for asynchronous work
- Improve sort algorithm
- Better error handling
- Make sorting customizable
- Remove class Subliminal
- Remove permissions handling

7.12 0.4

release date: 2011-11-11

- Many fixes
- Better error handling

7.13 0.3

release date: 2011-08-18

- Fix a bug when series is not guessed by guessit
- Fix dependencies failure when installing package
- Fix encoding issues with logging
- Add a script to ease subtitles download
- Add possibility to choose mode of created files
- Add more checks before adjusting permissions

7.14 0.2

release date: 2011-07-11

- Fix plugin configuration
- Fix some encoding issues
- Remove extra logging

7.15 0.1

release date: not released yet

- Initial release

S

subliminal.api, 13
subliminal.cache, 14
subliminal.cli, 14
subliminal.exceptions, 14
subliminal.providers, 15
subliminal.score, 16
subliminal.subtitle, 17
subliminal.video, 18

C

CACHE_VERSION (in module subliminal.cache), 14
check() (subliminal.providers.Provider class method), 15
compute_guess_matches() (in module subliminal.subtitle), 18
compute_matches() (subliminal.subtitle.Subtitle method), 17
compute_score() (subliminal.subtitle.Subtitle method), 17

D

download_best_subtitles() (in module subliminal.api), 13
download_subtitle() (subliminal.providers.Provider method), 16
download_subtitles() (in module subliminal.api), 13

E

Episode (class in subliminal.video), 19
Error (class in subliminal.exceptions), 14

G

get_episode_equations() (in module subliminal.score), 16
get_movie_equations() (in module subliminal.score), 17
get_subtitle_path() (in module subliminal.subtitle), 18

H

hash_opensubtitles() (in module subliminal.video), 19
hash_thesubdb() (in module subliminal.video), 19

I

initialize() (subliminal.providers.Provider method), 15
InvalidSubtitle (class in subliminal.exceptions), 14
is_valid_subtitle() (in module subliminal.subtitle), 18

L

languages (subliminal.providers.Provider attribute), 15
list_subtitles() (in module subliminal.api), 13
list_subtitles() (subliminal.providers.Provider method), 15

M

Movie (class in subliminal.video), 19

P

Provider (class in subliminal.providers), 15
ProviderConfigurationError (class in subliminal.exceptions), 14
ProviderError (class in subliminal.exceptions), 14
ProviderNotAvailable (class in subliminal.exceptions), 14
PROVIDERS_ENTRY_POINT (in module subliminal.api), 13

Q

query() (subliminal.providers.Provider method), 15

R

region (in module subliminal.cache), 14
required_hash (subliminal.providers.Provider attribute), 15

S

scan_subtitle_languages() (in module subliminal.video), 19
scan_video() (in module subliminal.video), 19
scan_videos() (in module subliminal.video), 20
subliminal.api (module), 13
subliminal.cache (module), 14
subliminal.cli (module), 14
subliminal.exceptions (module), 14
subliminal.providers (module), 15
subliminal.score (module), 16
subliminal.subtitle (module), 17
subliminal.video (module), 18
subliminal_key_generator() (in module subliminal.cache), 14
Subtitle (class in subliminal.subtitle), 17
SUBTITLE_EXTENSIONS (in module subliminal.video), 18

T

terminate() (subliminal.providers.Provider method), 15

V

Video (class in subliminal.video), 18
VIDEO_EXTENSIONS (in module subliminal.video), 18
video_types (subliminal.providers.Provider attribute), 15