webvtt-py Documentation

Release 0.1.0

Alejandro Mendez

Jun 08, 2018

Contents

Contents:

CHAPTER 1

Quickstart

1.1 Installation

You can install `webvtt-py` with pip:

```
$ pip install webvtt-py
```

To install with easy_install:

```
$ easy_install webvtt-py
```

1.2 Requirements

This module requires Python 3.3+.

1.3 Source code

This project is hosted on GitHub.

1.4 License

Licensed under the MIT License.

CHAPTER 2

Usage

2.1 Reading WebVTT caption files

```
import webvtt

# we can iterate over the captions
for caption in webvtt.read('captions.vtt'):
    print(caption.start)  # start timestamp in text format
    print(caption.end)  # end timestamp in text format
    print(caption.text)  # caption text

# you can also iterate over the lines of a particular caption
for line in vtt[0].lines:
    print(line)

# caption text is returned clean without class tags
# we can access the raw text of a caption with raw_text
>>> vtt[0].text
'This is a caption text'
>>> vtt[0].raw_text
'This is a <c.colorE5E5E5>caption</c> text'

# caption identifiers
>>> vtt[0].identifier
'crédit de transcription'
```

2.2 Creating captions

```
from webvtt import WebVTT, Caption

vtt = WebVTT()
```

(continues on next page)

(continued from previous page)

```
# creating a caption with a list of lines
caption = Caption(
    '00:00:00.500',
    '00:00:07.000',
    ['Caption line 1', 'Caption line 2']
)

# adding a caption
vtt.captions.append(caption)

# creating another caption with a text
caption = Caption(
    '00:00:07.000',
    '00:00:11.890',
    'Caption line 1\nCaption line 2']
)

vtt.captions.append(caption)
```

2.3 Manipulating captions

```
import webvtt

vtt = webvtt.read('captions.vtt')

# update start timestamp
vtt[0].start = '00:00:01.250'

# update end timestamp
vtt[0].end = '00:00:03.890'

# update caption text
vtt[0].text = 'My caption text'

# delete a caption
del vtt.captions[2]
```

2.4 Saving captions

```
import webvtt

vtt = webvtt.read('captions.vtt')

# save to original file
vtt.save()

# save to a different file
vtt.save('my_captions.vtt')

# write to opened file
```

(continues on next page)

(continued from previous page)

```
with open('my_captions.vtt', 'w') as fd:
    vtt.write(fd)
```

2.5 Converting captions

You can read captions from the following formats:

- SubRip (.srt)
- YouTube SBV (.sbv)

```
import webvtt

# to read from a different format use the method from_ followed by
# the extension.
vtt = webvtt.from_sbv('captions.sbv')
vtt.save()

# if we just want to convert the file we can do this in one line
webvtt.from_sbv('captions.sbv').save()
```

Also we can convert WebVTT to other formats:

- SubRip (.srt)

```
import webvtt

# save in SRT format
vtt = webvtt.read('captions.vtt')
vtt.save_as_srt()

# write to opened file in SRT format
with open('my_captions.srt', 'w') as fd:
    webvtt.write(fd, format='srt)
```

CHAPTER 3

webvtt-py package reference

3.1 webvtt.webvtt

class webvtt.webvtt.**WebVTT**(*file=''*, *captions=None*, *styles=None*)

Parse captions in WebVTT format and also from other formats like SRT.

To read WebVTT:

> WebVTT().read('captions.vtt')

For other formats like SRT, use from_[format in lower case]:

> WebVTT().from_srt('captions.srt')

A list of all supported formats is available calling list_formats().

captions

Returns the list of captions.

classmethod from_sbv(*file*)

Reads captions from a file in YouTube SBV format.

classmethod from_srt(*file*)

Reads captions from a file in SubRip format.

static list_formats()

Provides a list of supported formats that this class can read from.

classmethod read(*file*)

Reads a WebVTT captions file.

save(*output=''*)

Save the document. If no output is provided the file will be saved in the same location. Otherwise output can determine a target directory or file.

total_length

Returns the total length of the captions.

3.2 webvtt.segmenter

class webvtt.segmenter.**WebVTTSegmenter**
Provides segmentation of WebVTT captions for HTTP Live Streaming (HLS).

seconds
Returns the number of seconds used for segmenting captions.

segment(*webvtt*, *output=''*, *seconds=10*, *mpegts=900000*)
Segments the captions based on a number of seconds.

segments
Return the list of segments.

total_segments
Returns the total of segments.

3.3 webvtt.cli

Usage: webvtt segment <file> [–target-duration=SECONDS] [–mpegts=OFFSET] [–output=<dir>] webvtt -h | –help webvtt –version

Options: -h –help Show this screen. –version Show version. –target-duration=SECONDS Target duration of each segment in seconds [default: 10]. –mpegts=OFFSET Presentation timestamp value [default: 900000]. –output=<dir> Output to directory [default: ./].

Examples: webvtt segment captions.vtt –output destination/directory

webvtt.cli.**main**()
Main entry point for CLI commands.

webvtt.cli.**segment**(*f*, *output*, *target_duration*, *mpegts*)
Segment command.

3.4 webvtt.generic

3.5 webvtt.parsers

class webvtt.parsers.**SBVParser**(*parse_options=None*)
Bases: *webvtt.parsers.TextBasedParser*

YouTube SBV parser.

class webvtt.parsers.**SRTParser**(*parse_options=None*)
Bases: *webvtt.parsers.TextBasedParser*

SRT parser.

class webvtt.parsers.**TextBasedParser**(*parse_options=None*)
Bases: object

Parser for plain text caption files. This is a generic class, do not use directly.

read(*file*)
Reads the captions file.

class webvtt.parsers.**WebVTTParser**
Bases: *webvtt.parsers.TextBasedParser*

WebVTT parser.

3.6 webvtt.writers

3.7 webvtt.exceptions

CHAPTER 4

History

4.1 0.4.2 (08-06-2018) Rename of modules and usability improvements

- Renamed and reorganized few of the modules
- Parsing methods are now class methods: read, from_srt and from_sbv
- Improved usability with the addition of shortcuts to avoid instantiating the classes so we can do:

import webvtt

webvtt.read('captions.vtt') # this will return a WebVTT instance

4.2 0.4.1 (24-12-2017) Hot fix on cue identifiers

- Support for saving cue identifiers

4.3 0.4.0 (18-09-2017) Refactor and parse compatibility

The main goal of this release is a refactor of the WebVTT parser to be able to parse easier and give support to new features of the format.

New features:

- Support for cue identifiers
- Support for parsing WebVTT captions with comments
- Support for parsing WebVTT captions with Style blocks
- Support for BOM in caption files

- Added method to write the captions to an opened file
- Convert WebVTT to SRT format
- Ignore empty captions in SRT format

Other:

- Refactored WebVTT parser

4.4 0.3.3 (23-08-2017) Hot fix on cue tags

The text for the caption is now returned clean (tags removed). The cue text could contain tags like: * timestamp tags: *<00:19.000>* * class tags: *<c.classname>text</c>* * and others... **Important**: It currently removes any tag present in the cue text. For example <b> would be removed.

Also a new attribute is available on captions to retrieve the text without cleaning tags: **raw_text**

4.5 0.3.2 (11-08-2017) Hot fix for compatibility

The goal of this release if to allow the WebVTT parser to be able to read caption files that contain metadata headers that extend to more than one line.

4.6 0.3.1 (08-08-2017) Compatibility updates

- Made hours in WebVTT parser optional as per specs.
- Added support to parse WebVTT files that contain metadata headers.

4.7 0.3.0 (02-06-2016) YouTube SBV

New features:

- Added support for YouTube SBV captions.
- Added easy iteration to WebVTT class.
- New CLI command for segmenting captions for HLS.

Other:

- Improved parsers to reuse functionality.
- Added an exception for invalid timestamps in captions.
- Added an exception when saving without a filename.

4.8 0.2.0 (23-05-2016) Module refactor

- Refactor of the main module and parsers.

4.9 0.1.0 (20-05-2016) First release

This module is released with the following initial features:

- Read/Edit/Write WebVTT captions.
- Read SRT captions and convert to WebVTT.
- Segment WebVTT files for captioning HLS video.

CHAPTER 5

Indices and tables

- genindex
- modindex

Python Module Index

W

Index