
emoji-unicode Documentation

Release 0.1

Esteban Castro Borsani

November 19, 2015

1	API Reference	1
1.1	API	1
2	Additional Notes	3
2.1	Python 2 Support	3
2.2	Changelog	3
2.3	License	3
	Python Module Index	5

API Reference

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

1.1 API

1.1.1 Helpers

`emoji_unicode.replace(txt, callback)`
Replace all unicode emojis in a given text

Parameters

- **txt** (*str*) – Text source to be parsed
- **callback** (*callable*) – A callable that should accept a instance of *Emoji* and return a str to replace the match

Returns Parsed text with all the emojis replaced by the callback result

Return type str

`emoji_unicode.normalize(code_points, separator=u'-')`
Normalize a code point removing joiner characters such as emoji variations, Zero Width Joiner and leading zeros

Parameters

- **code_points** (*str*) – code points separated by an hyphen or the separator param value
- **separator** (*str*) – The separator used to split the code points, process them and merge them back

Returns Code points with no joiner chars, leading zeros, and lower cased.

Return type str

1.1.2 Emoji Object

`class emoji_unicode.Emoji(unicode)`
Emoji is used in the process of *replace()* unicode emojis in a text

Parameters **unicode** (*str*) – Unicode emoji

as_map()

A map containing the individual unicode chars and code points. The code points are normalized as by `normalize()`

Returns Sequence of tuples of the form [(unicode, code_point)]

Return type list

code_points

Code points representing the unicode emoji, the result is normalized as by `normalize()`

Getter Code points representing the emoji, with no joiner chars and lower cased, ie: 1f3c3-1f3fc

Type str

Additional Notes

Design notes, legal information and changelog are here for the interested.

2.1 Python 2 Support

Python 2 is by default build with *narrow* unicode characters support, this means it does not support unicode code points above 0xffff which most emojis are. Python +3.3 does not suffer from this.

2.1.1 Requirements

To support *wide unicode characters*, python 2.7 must be build from source with `--enable-unicode=ucs4` flag.

To find if python 2 has support for *wide* unicode characters, run:

```
$ python
>>> import sys
>>> 'Is this a wide-build? {}'.format(sys.maxunicode > 65536)
```

2.2 Changelog

2.2.1 0.2

- Adds `Emoji.as_map()` function.

2.2.2 0.1

- Initial release

2.3 License

The MIT License (MIT)

Copyright (c) 2015 Esteban Castro Borsani <ecastroborsani@gmail.com>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the ‘Software’), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ‘AS IS’, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

e

emoji_unicode, 1

A

`as_map()` (`emoji_unicode.Emoji` method), 1

C

`code_points` (`emoji_unicode.Emoji` attribute), 2

E

`Emoji` (class in `emoji_unicode`), 1

`emoji_unicode` (module), 1

N

`normalize()` (in module `emoji_unicode`), 1

R

`replace()` (in module `emoji_unicode`), 1