YACT Documentation

Release 0.5.1

Jesse Roberts

Oct 17, 2019

Contents

Release v0.5.1. (*Installation*)

YACT is a friendly configuration tool that does all the heavy lifting so you don't have to.

Here's how easy it is:

```
>>> config = yact.from_file('my-config.yaml')
>>> config.get('debug')
True
```

YACT allows you to write human readable configuration files using YAML, then load that configuration into your app without having to set up parsers, or search for config files, or any of that nonsense. Even better, YACT can automatically reload your configuration file when it detects the file has changed.:

```
>>> config = yact.from_file('my-config.yaml', auto_reload=True)
```

YACT is tested against Python 2.7, 3.3-3.6 and PyPy.

CHAPTER 1

Usage Guide

1.1 Quickstart

1.1.1 Installation

Via Pip

To install YACT, run this in your terminal:

```
$ pip install yact
```

Via Git

YACT is under active development on GitHub.

To install YACT via git, run this in your terminal:

```
$ git clone git@github.com/jesseops/yact.git $ cd yact && python setup.py install
```

1.1.2 Loading Config

YACT is designed to make loading your configuration as easy as possible. To that end, calling *yact.from_file* with the name of your config file will automatically search common locations and pull up your config. You may optionally give it a path to search in if it's not in a standard location.

Example

Standard loading:

```
>>> import yact
>>> config = yact.from_file('my-config.yaml')
>>> print(config.filename)
'/etc/my-config.yaml'
```

Explicit directory:

```
>>> config = yact.from_file('my-config.yaml', directory='/opt/my-app')
>>> print(config.filename)
'/opt/my-app/my-config.yaml'
```

1.1.3 Saving Config

There's no need! As long as you use the standard methods of updating a config entry, YACT will automatically save your changes to the original configuration file.

Example:

```
>>> config = yact.from_file('mynewconfig.yaml', create_if_missing=True)
>>> config.set('my.new.setting', True)
```

Done!

1.1.4 Auto Reloading

YACT will watch for changes to your config files and automatically reload your configuration without any extra code on your end. Just set *auto_reload* to *True* when loading your config:

```
>>> config = yact.from_file('myconfig.yaml', auto_reload=True)
```

1.2 Advanced

1.2.1 Here Be Dragons (AKA unsafe YAML)

TODO: This

CHAPTER 2

API Documentation

Here you will find documentation of the actual code. Wondering what exactly *yact.from_file* does? This is for you.

2.1 Main Interface

class yact.**Config**(*filename*, *unsafe=False*, *auto_reload=False*)

The *Config* object is a wrapper around YAML data. For most use cases, the basic functionality of reading a YAML file (extension does not matter) is sufficient.

While not currently tested, unsafe loading of YAML files is supported using the unsafe flag.

get(*key*, *default=None*)

Retrieve the value of a key (or consecutive keys joined by periods) or default, similar to dict.get

remove(*key*)

Remove an item from configuration file

Establishes lock on configuration data, deletes config entry matching the passed in key. Saves updated configuration back to file.

save()

Save current configuration back to file in YAML format

Acquires configuration lock, opens file in overwrite mode ('w') and writes the output of yaml.dump to the file object. default_flow_style is set to false to force proper YAML formatting

sections

Provided for users of the standard ConfigParser module.

set(*key*, *value*)

Set the value of a provided key (or nested keys joined by periods) to the provided value

CHAPTER 3

Contribution Guide

Index