shellp Documentation

Release 0.1.0

Dull Bananas

Jul 02, 2019

Contents

ShellP is an easy to use shell implemented completely in Python.

CHAPTER 1

Contents

1.1 Installing ShellP

1.1.1 Requirements and Compatibility

ShellP is supposed to work on any platform, but it has only been tested on Linux. If you are using a different platform and ShellP isn't working on it, report it by opening an issue on ShellP's GitHub repository.

ShellP is compatible with Python 3.6 and newer.

1.1.2 Installation

To install ShellP, use this command (without `sudo`):

```
pip3 install --user shellp
```

1.2 Basic Usage

1.2.1 Starting ShellP

To start ShellP, use this command:

```
python3 -m shellp
```

If `~/.local/bin` is in your `$PATH`, you can also use this command:

```
shellp
```

1.2.2 Exiting ShellP

To exit ShellP, type the `exit` command.

1.3 Configuration

You can create a configuration file at `~/.shellp/config.py` to customize ShellP. In this file, you simple define them as variables, just like in Sphinx's `conf.py` file. Here is an example of a configuration file:

```
aliases = {
    'cd': 'cd --color',
    'ga': 'git add',
}
debug = True
timeout = 3600
```

1.3.1 List of Options

These are the options that you can use in your config file:

`aliases` A dictionary mapping aliases to commands. Example:

```
aliases = {
    'ga': 'git add .',
    'l': 'ls --color -l',
}
```

`bash_alias_files` This option allows you to make ShellP parse one or more Bash files and extract aliases from it. Example:

```
bash_alias_files = ['/home/user/.bashrc']
```

`debug` This option enables debug mode when set to `True`. These are the changes that take effect when you enable this option:

- Before a command is run, the array of arguments that will be passed to the command will be shown (e.g. `['git', 'commit', '-m', 'i hate you']`)

`env_lists` This option allows you to set colon-separated environment variables such as `$PATH` with arrays instead of messy colon-separated strings. The items you add in the array are prepended to the environment variable's existing value. Example:

```
env_lists = {
    'PATH': [
        '/home/user/bin',
        '/other/path',
    ],
}
```

`env_vars` This is a dictionary of environment variables to set. Example:

```
env_vars = {
    'EDITOR': 'vim',
}
```

ps1 This is the prompt that is shown before the command you type. See *Prompt Format* for details on the format of this option.

timeout This sets the timeout for command input in seconds. You can use either an integer or a float.

1.3.2 Prompt Format

`ps1` uses a clean format that is much more readable than Bash's escape codes. It is parsed using `str.format()`. Example:

```
ps1 = '{style.green}{cwd} {symbol} '
```

Here are the values you can use:

{bell} ASCII BEL character; same as `chr(7)`

{cwd} The current working directory

{git_branch} The current Git branch, or an empty string if you're not in a Git directory.

{hostname} Your device's hostname

{platform["*"]} Shows the result of the specified function in the `platform` module; for example, `platform["processor"]`

{shellp_version} The version of ShellP that you are using

{style.*} The `beautiful_ansi` module

{symbol} A `#` if you are root, otherwise `$`

{time["*"]} The current time formatted with `time.strftime()`

{uid} Your user ID

{user} Your username

1.4 Special Commands

ShellP has several built-in commands:

cd Changes the current working directory. If no directory is specified, it defaults to your home directory

exit Exits ShellP

eval Evaluates a Python expression (must be in quotes)

reload Reloads your user configuration file