

---

**pkg***utilsdocumentation*

***Release 0.0.2***

**Karr Lab**

**Dec 04, 2017**



---

## Contents

---

<b>1</b>	<b>Contents</b>	<b>3</b>
1.1	Installation	3
1.1.1	Requirements	3
1.1.2	Optional requirements	3
1.1.3	Installing this package	3
1.2	Tutorial	4
1.2.1	Linking setuptools with package version numbers	4
1.2.2	Linking setuptools with GitHub README.md files	4
1.2.3	Linking setuptools with requirements	5
1.2.4	Restoring overridden console scripts during editable installations	6
1.2.5	Putting it all together	6
1.3	Testing	7
1.4	pkg_utils package	7
1.4.1	Submodules	7
1.4.2	pkg_utils.core module	7
1.4.3	Module contents	9
1.5	About	9
1.5.1	License	9
1.5.2	Development team	10
1.5.3	Questions and comments	10
	<b>Python Module Index</b>	<b>11</b>



Utilities for linking setuptools with package version metadata, GitHub README.md files, requirements.txt files, and restoring overridden entry points during for editable installations.



## 1.1 Installation

### 1.1.1 Requirements

First, install [Python](#) and [pip](#). The following command illustrates how to install Python and pip on Ubuntu Linux:

```
apt-get install python python-pip
```

### 1.1.2 Optional requirements

Second, optionally install pandoc to convert Markdown-formatted README files for GitHub into reStructuredText-formatted files for PyPI:

```
apt-get install pandoc
```

### 1.1.3 Installing this package

Use the following command to install this package from PyPI:

```
pip install pkg_utils
```

The latest version of this package can be installed from GitHub using this command:

```
pip install git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
```

Support for the pandoc can be installed using the following option:

```
pip install pkg_utils[pandoc]  
pip install git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_  
↪utils[pandoc]
```

## 1.2 Tutorial

### 1.2.1 Linking setuptools with package version numbers

The following example shows how to link a package number stored in package/VERSION with setuptools:

```
import os
import setuptools
try:
    import pkg_utils
except ImportError:
    import pip
    pip.main(['install', 'git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
↪'])
    import pkg_utils

# package name
name = 'my_package'
dirname = os.path.dirname(__file__)

# get package metadata
md = pkg_utils.get_package_metadata(dirname, name)

# install package
setuptools.setup(
    ...
    version=md.version,
)
```

### 1.2.2 Linking setuptools with GitHub README.md files

The following example shows how to link GitHub Markdown-formatted README.md files with setuptools which requires long descriptions in reStructuredText format. Note, this feature requires the pandoc option.

```
import os
import setuptools
try:
    import pkg_utils
except ImportError:
    import pip
    pip.main(['install', 'git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
↪'])
    import pkg_utils

# package name
name = 'my_package'
dirname = os.path.dirname(__file__)

# convert README.md to README.rst
pkg_utils.convert_readme_md_to_rst(dirname)

# get package metadata
md = pkg_utils.get_package_metadata(dirname, name)

# install package
```



```

setuptools.setup(
    ...
    long_description=md.long_description,
)

```

### 1.2.3 Linking setuptools with requirements

The following example illustrates how to link setuptools with requirements.txt files:

```

import os
import setuptools
try:
    import pkg_utils
except ImportError:
    import pip
    pip.main(['install', 'git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
↪'])
    import pkg_utils

# package name
name = 'my_package'
dirname = os.path.dirname(__file__)

# get package metadata
md = pkg_utils.get_package_metadata(dirname, name)

# install package
setuptools.setup(
    ...
    install_requires=md.install_requires,
    extras_require=md.extras_require,
    tests_require=md.tests_require,
    dependency_links=md.dependency_links,
)

```

This extracts dependencies from the following files:

- requirements.txt: dependencies
- requirements.optional.txt: optional dependencies
- tests/requirement.txt: dependencies to run the tests
- docs/requirement.txt: dependencies to build the documentation

The requirements.txt files should follow the [pip](#) format:

```

package_1
package_2[package_2_option_2] >= 1.0.0; python_version >= "2.7.14"

```

The requirements.optional.txt should follow the same format, but with section headings to indicate the options:

```

[my_option_1]
package_1
package_2[package_2_option_2] >= 1.0.0; python_version >= "2.7.14"

[my_option_2]

```

```
package_3
package_4
```

In addition to the installation options described in `requirements.optional.txt`, `pkg_utils` will create `tests`, `docs` and `all` options to install the test, documentation, and all dependencies.

## 1.2.4 Restoring overridden console scripts during editable installations

The following example illustrates how to restore overridden console scripts during editable installations. This useful for generating console scripts for specific versions of Python.

```
import os
import setuptools
try:
    import pkg_utils
except ImportError:
    import pip
    pip.main(['install', 'git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
↪'])
    import pkg_utils

# package name
name = 'my_package'
dirname = os.path.dirname(__file__)

# read old console scripts
console_scripts = pkg_utils.get_console_scripts(dirname, name)

# install package
setuptools.setup(...)

# restore old console scripts
pkg_utils.add_console_scripts(dirname, name, console_scripts)
```

## 1.2.5 Putting it all together

The following example shows how to use all of the features of this package:

```
import os
import setuptools
try:
    import pkg_utils
except ImportError:
    import pip
    pip.main(['install', 'git+https://github.com/KarrLab/pkg_utils.git#egg=pkg_utils
↪'])
    import pkg_utils

# package name
name = 'my_package'
dirname = os.path.dirname(__file__)

# get package metadata
md = pkg_utils.get_package_metadata(dirname, name)
```

```

# read old console scripts
console_scripts = pkg_utils.get_console_scripts(dirname, name)

# install package
setuptools.setup(
    ...
    version=md.version,
    long_description=md.long_description,
    install_requires=md.install_requires,
    extras_require=md.extras_require,
    tests_require=md.tests_require,
    dependency_links=md.dependency_links,
)

# restore old console scripts
pkg_utils.add_console_scripts(dirname, name, console_scripts)

```

## 1.3 Testing

The package can be tested by running these commands:

```

pip install pytest
python -m pytest tests

```

## 1.4 pkg\_utils package

### 1.4.1 Submodules

### 1.4.2 pkg\_utils.core module

Utilities for linking setuptools with package version metadata, GitHub README.md files, requirements.txt files, and restoring overridden entry points during for editable installations.

**Author** Jonathan Karr <[jonrkarr@gmail.com](mailto:jonrkarr@gmail.com)>

**Date** 2017-12-03

**Copyright** 2017, Karr Lab

**License** MIT

**class** pkg\_utils.core.PackageMetadata

Bases: object

Metadata about a package

**name**  
str

**description**  
str – short description

**long\_description**  
str – long description, e.g. from README.rst

**version**

str – version, e.g. from package/VERSION

**install\_requires**

list of str – dependencies, e.g. from requirements.txt

**extras\_require**

dict of list of str – optional dependencies, e.g. from requirements.optional.txt

**tests\_require**

list of str – test dependencies, e.g. from tests/requirements.txt

**dependency\_links**

list of str – documentation dependencies, e.g. from docs/requirements.txt

`pkg_utils.core.add_console_scripts (dirname, package_name, console_scripts)`

Add console scripts for a package

**Parameters**

- **dirname** (str) – path to the package
- **package\_name** (str) – package name
- **console\_scripts** (dict) – console script names and locations

`pkg_utils.core.convert_readme_md_to_rst (dirname)`

Convert the README.md to README.rst

**Parameters** **dirname** (str) – path to the package

`pkg_utils.core.get_console_scripts (dirname, package_name)`

Get the console scripts for a package

**Parameters**

- **dirname** (str) – path to the package
- **package\_name** (str) – package name

**Returns** console script names and locations

**Return type** dict

`pkg_utils.core.get_long_description (dirname)`

Get the long description of a package from its README.rst file

**Parameters** **dirname** (str) – path to the package

**Returns** long description

**Return type** str

`pkg_utils.core.get_package_metadata (dirname, package_name)`

Get meta data about a package

**Parameters**

- **dirname** (str) – path to the package
- **package\_name** (str) – package name

**Returns** meta data

**Return type** *PackageMetadata*

**Raises** **Exception:** if test or documentation dependencies are defined in *requirements.optional.txt*

`pkg_utils.core.get_version (dirname, package_name)`

Get the version a package from its VERSION file (package/VERSION)

**Parameters**

- **dirname** (str) – path to the package
- **package\_name** (str) – package name

**Returns** version

**Return type** str

`pkg_utils.core.install_dependencies (dependencies, upgrade=False)`

Install dependencies

**Parameters**

- **dependencies** (list) – list of dependencies
- **upgrade** (bool, optional) – if True, upgrade package

`pkg_utils.core.parse_optional_requirements_file (filename)`

Parse a requirements.optional.txt file into list of requirements and dependency links

**Parameters** **filename** (str) – path to requirements.txt file

**Returns** requirements list of str: dependency links

**Return type** dict of list of str

**Raises** Exception – if a line cannot be parsed

`pkg_utils.core.parse_requirement_lines (lines)`

Parse lines from a requirements.txt file into list of requirements and dependency links

**Parameters** **lines** (list of str) – lines from a requirements.txt file

**Returns** requirements list of str of str: dependency links

**Return type** list of str

**Raises** Exception – if a line cannot be parse

`pkg_utils.core.parse_requirements_file (filename)`

Parse a requirements.txt file into list of requirements and dependency links

**Parameters** **filename** (str) – path to requirements.txt file

**Returns** requirements list of str: dependency links

**Return type** list of str

## 1.4.3 Module contents

## 1.5 About

### 1.5.1 License

The software is released under the MIT license

The MIT License (MIT)

Copyright (c) 2017 Karr Lab

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.

## 1.5.2 Development team

This package was developed by the [Karr Lab](#) at the Icahn School of Medicine at Mount Sinai in New York, USA.

## 1.5.3 Questions and comments

Please contact the [Karr Lab](#) with any questions or comments.

### p

`pkg_utils`, 9  
`pkg_utils.core`, 7





## A

`add_console_scripts()` (in module `pkg_utils.core`), 8

## C

`convert_readme_md_to_rst()` (in module `pkg_utils.core`), 8

## D

`dependency_links` (`pkg_utils.core.PackageMetadata` attribute), 8

`description` (`pkg_utils.core.PackageMetadata` attribute), 7

## E

`extras_require` (`pkg_utils.core.PackageMetadata` attribute), 8

## G

`get_console_scripts()` (in module `pkg_utils.core`), 8

`get_long_description()` (in module `pkg_utils.core`), 8

`get_package_metadata()` (in module `pkg_utils.core`), 8

`get_version()` (in module `pkg_utils.core`), 8

## I

`install_dependencies()` (in module `pkg_utils.core`), 9

`install_requires` (`pkg_utils.core.PackageMetadata` attribute), 8

## L

`long_description` (`pkg_utils.core.PackageMetadata` attribute), 7

## N

`name` (`pkg_utils.core.PackageMetadata` attribute), 7

## P

`PackageMetadata` (class in `pkg_utils.core`), 7

`parse_optional_requirements_file()` (in module `pkg_utils.core`), 9

`parse_requirement_lines()` (in module `pkg_utils.core`), 9

`parse_requirements_file()` (in module `pkg_utils.core`), 9

`pkg_utils` (module), 9

`pkg_utils.core` (module), 7

## T

`tests_require` (`pkg_utils.core.PackageMetadata` attribute), 8

## V

`version` (`pkg_utils.core.PackageMetadata` attribute), 7