iscompatible Documentation

Release 0.1.1

Marcus Ottosson

October 09, 2014

Contents

iscompatible v0.1.1. gives you the power of the pip requirements.txt syntax for everyday python packages, modules, classes or arbitrary functions.

The requirements.txt syntax allows you to specify inexact matches between a set of requirements and a version. For example, let’s assume that the single package foo-5.6.1 exists on disk. The following requirements are all compatible with foo-5.6.1.

Requirement	Description
foo	any version of foo
foo>=5	any version of foo, above or equal to 5
foo>=5.6	any version of foo, above or equal to 5.6
foo==5.6.1	exact match
foo>5	foo-5 or greater, including minor and patch
foo>5, <5.7	foo-5 or greater, but less than foo-5.7
foo>0, <5.7	any foo version less than foo-5.7

Example

```
>>> iscompatible("foo>=5", (5, 6, 1))
True
>>> iscompatible("foo>=5.6.1, <5.7", (5, 0, 0))
False
>>> MyPlugin = type("MyPlugin", (), {’version’: (5, 6, 1)})
>>> iscompatible("foo==5.6.1", MyPlugin.version)
True
```

References

- The requirements file-format

iscompatible.**iscompatible**(*requirements*, *version*)

Return whether or not *requirements* is compatible with *version*

Parameters

- **requirements** (*str*) – Requirement to compare, e.g. foo==1.0.1
- **version** (*tuple*) – Version to compare against, e.g. (1, 0, 1)

Example

```
>>> iscompatible("foo", (1, 0, 0))
True
>>> iscompatible("foo<=1", (0, 9, 0))
True
>>> iscompatible("foo>=1, <1.3", (1, 2, 0))
True
>>> iscompatible("foo>=0.9.9", (1, 0, 0))
True
>>> iscompatible("foo>=1.1, <2.1", (2, 0, 0))
True
>>> iscompatible("foo==1.0.0", (1, 0, 0))
True
>>> iscompatible("foo==1.0.0", (1, 0, 1))
False
```

iscompatible.**parse_requirements**(*line*)

Return list of tuples with (operator, version) from *line*

Note: This is a minimal re-implementation of pkg_utils.parse_requirements and doesn't include support for VCS or extras.

Example

```
>>> parse_requirements("foo==1.0.0")
[('==', '1.0.0')]
>>> parse_requirements("foo>=1.1.0")
[('>=', '1.1.0')]
>>> parse_requirements("foo>=1.1.0, <1.2")
[('>=', '1.1.0'), ('<', '1.2')]
```

iscompatible.**string_to_tuple**(*version*)

Convert version as string to tuple

Example

```
>>> string_to_tuple("1.0.0")
(1, 0, 0)
>>> string_to_tuple("2.5")
(2, 5)
```

Python Module Index