

---

# py-deps Documentation

*Release 0.5.5*

**Kouhei Maeda <[mkouhei@palmtb.net](mailto:mkouhei@palmtb.net)>**

November 26, 2015



<b>1</b>	<b>About py-dep</b>	<b>3</b>
1.1	Status . . . . .	3
1.2	Requirements . . . . .	3
1.3	Features . . . . .	3
1.4	Known issue with the packages that depends on py-deps . . . . .	3
<b>2</b>	<b>Basic usage</b>	<b>5</b>
2.1	Search packages . . . . .	5
2.2	Show latest version . . . . .	6
2.3	Initialize . . . . .	6
2.4	Generate rendering data . . . . .	7
2.5	Check cache . . . . .	9
<b>3</b>	<b>References</b>	<b>11</b>
<b>4</b>	<b>Design notes</b>	<b>19</b>
4.1	Memo . . . . .	19
<b>5</b>	<b>History</b>	<b>21</b>
5.1	0.5.5 (2015-08-19) . . . . .	21
5.2	0.5.4 (2015-07-22) . . . . .	21
5.3	0.5.3 (2015-07-19) . . . . .	21
5.4	0.5.2 (2015-07-15) . . . . .	21
5.5	0.5.1 (2015-07-12) . . . . .	21
5.6	0.5.0 (2015-06-22) . . . . .	22
5.7	0.4.6 (2015-06-11) . . . . .	22
5.8	0.4.5 (2015-06-07) . . . . .	22
5.9	0.4.4 (2015-06-03) . . . . .	22
5.10	0.4.3 (2015-06-02) . . . . .	22
5.11	0.4.2 (2015-05-31) . . . . .	22
5.12	0.4.1 (2015-05-28) . . . . .	22
5.13	0.4.0 (2015-05-20) . . . . .	23
5.14	0.3.0 (2015-05-12) . . . . .	23
5.15	0.2.0 (2015-05-10) . . . . .	23
5.16	0.1.1 (2015-05-08) . . . . .	23
5.17	0.1.0 (2015-05-07) . . . . .	23
5.18	0.0.1 (2015-04-29) . . . . .	23
<b>6</b>	<b>Indices and tables</b>	<b>25</b>



Contents:



---

## About py-dep

---

The `py-dep` provides parsing the dependencies of Python packages and generating the metadata for graph.

The graph data is for [NetworkX](#), [Graphviz](#), [blockdiag](#), [Linkdraw](#), etc.

### 1.1 Status

### 1.2 Requirements

- Python 2.7 over or Python 3.3 over or PyPy 2.4.0 over
- pip 1.5.6 or 6.1.1 over
- wheel 0.24.0 over
- NetworkX 1.9 over
- pylibmc 1.4.3 over (optional)

### 1.3 Features

- Generating Linkdraw data (JSON and decoded JSON).
- Generating Networkx DiGraph object data.
- Cache the parsed dependencies.
- Searching packages from PyPI.

### 1.4 Known issue with the packages that depends on py-deps

The packages that depend on `py-deps`; after that called “X” package, there is a known issue that fails to install using the `pip`. This problem is caused by `py-deps` is a package that depends on the `pip` and `wheel`. When you install the “X” in the `pip` following exception occurs.:

The AssertionError: Multiple .dist-info directories occurs, because py-deps depends on

`pip, wheel.`

### 1.4.1 Workaround

The workaround for this problem is to use `setuptools` instead of `pip`.

- You should use the `easy_install` or `pip install --no-use-wheel` command when you are installing the X from [PyPI](#).
- Use the `python setup.py install` when you install from the source tree , such as repository.
- When you use the [Tox](#) in unit test , you should specify `install_command` in `[testenv]` section of `tox.ini`:

```
[testenv]
install_command = easy_install {opts} {packages}
```

See also [pgraph](#) is already corresponding to the above-mentioned problems.



---

## Basic usage

---

py-deps provides parsing the Python deps and generating graph data.

### 2.1 Search packages

Search packages from PyPI:

```
$ python
>>> from py_deps.deps import search
>>> search('deps')
[{'_pypi_ordering': False,
  'name': 'anybox.recipe.sysdeps',
  'summary': 'A buildout recipe to check system dependencies',
  'version': '0.5'},
 {'_pypi_ordering': False,
  'name': 'appdynamics-bindeps-linux-x64',
  'summary': 'Dependencies for AppDynamics Python agent',
  'version': '4.0.5.0'},
 {'_pypi_ordering': False,
  'name': 'appdynamics-bindeps-linux-x86',
  'summary': 'Dependencies for AppDynamics Python agent',
  'version': '4.0.5.0'},
 {'_pypi_ordering': False,
  'name': 'appdynamics-bindeps-osx-x64',
  'summary': 'Dependencies for AppDynamics Python agent',
  'version': '4.0.5.0'},
 {'_pypi_ordering': False,
  'name': 'deps',
  'summary': 'deps discovers your Python dependencies',
  'version': '0.1.0'},
 {'_pypi_ordering': False,
  'name': 'gtkeggdeps',
  'summary': 'Interactive egg dependency browser',
  'version': '0.0.7'},
 {'_pypi_ordering': False,
  'name': 'htmldeps',
  'summary': 'Expand CSS and javascript dependency links in HTML',
  'version': '1.2.1'},
 {'_pypi_ordering': False,
  'name': 'py-deps',
  'summary': 'parsing the Python deps and generating graph data',
  'version': '0.3.0'},
```

```
{'_pypi_ordering': False,
 'name': 'pydeps',
 'summary': 'Display module dependencies',
 'version': '1.2.1'},
{'_pypi_ordering': False,
 'name': 'runestone-test-deps',
 'summary': 'This is dependencies for RSI',
 'version': '0.1'},
{'_pypi_ordering': False,
 'name': 'tl.eggdeps',
 'summary': 'Compute a dependency graph between active Python eggs.',
 'version': '0.4'},
{'_pypi_ordering': False,
 'name': 'tt.eggdeps',
 'summary': 'Compute a dependency graph between active Python eggs.',
 'version': '0.5'}]
```

## 2.2 Show latest version

Retrieve latest package from PyPI.:

```
$ python
>>> from py_deps.deps import latest_version
>>> latest_version('deps')
'0.1.0'
```

## 2.3 Initialize

Cache the parsed dependencies into the `py-deps.pickle` on current working directory. This file format is [pickle](#).:

```
$ python
>>> from py_deps import Package
>>> pkg = Package('py-deps')
```

`py-deps` retrieve latest version from PyPI without `version` argument as above. Specify version use `version` argument.:

```
>>> pkg = Package('py-deps', version='0.3.0')
>>> pkg.version
0.3.0
```

### 2.3.1 Change cache file

Use `cache_name` argument.:

```
>>> pkg = Package('py-deps', cache_name='some-cache.name')
```

### 2.3.2 Override cache forcibly

Use `update_force` argument. (default: False):

```
>>> pkg = Package('py-deps', update_force=True)
```

## 2.3.3 Changes the cache backend to Memcached

Installing libmemcached-dev package and pylibmc.:

```
$ sudo apt-get install libmemcached-dev
$ . /path/to/venv/bin/activate
(venv)$ cd /path/to/py-deps
(venv)$ python setup.py memcache
```

Use servers argument. The argument syntax follows pylibmc.Client.:

```
>>> pkg = Package('py-deps', servers=['127.0.0.1:11211'])
```

## 2.4 Generate rendering data

Supports follows currently.

- pretty print
- Linkdraw

### 2.4.1 Pretty print

```
>>> print(pkg.draw())
py-deps -> [Sphinx, setuptools, pip, wheel, tox]
setuptools -> [certifi, wincertstore, setuptools[ssl], pytest]
pip -> [pytest, virtualenv, scripttest, mock, pytest, virtualenv,
scripttest, mock]
wheel -> [ed25519ll, keyring, argparse, pyxdg, jsonschema, pytest,
coverage, pytest-cov]
>>>
```

### 2.4.2 Linkdraw

```
>>> import json
>>> json.loads(pkg.draw('linkdraw'))
{'descr': u'py-deps dependencies',
 'lines': [{u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'py-deps', u'target': u'Sphinx', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'py-deps', u'target': u'setuptools', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'py-deps', u'target': u'pip', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'py-deps', u'target': u'wheel', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'py-deps', u'target': u'tox', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
  u'source': u'setuptools', u'target': u'certifi', u'width': u'1'},
 {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
```

```
    u'source': u'setuptools', u'target': u'wincertstore', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'setuptools', u'target': u'setuptools____ssl',
     u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'setuptools', u'target': u'pytest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'pytest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'virtualenv', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'scriptttest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'mock', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'pytest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'virtualenv', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'scriptttest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'pip', u'target': u'mock', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'ed2551911', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'keyring', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'argparse', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'pyxdg', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'jsonschema', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'pytest', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'coverage', u'width': u'1'},
    {u'color': u'#5F9EA0', u'descr': u'->', u'link': u'',
     u'source': u'wheel', u'target': u'pytest-cov', u'width': u'1'}],
u'nodes': [{u'color': u'', u'link': u'https://github.com/mkouhei/py-deps',
            u'name': u'py-deps', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'Sphinx', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'https://bitbucket.org/pypa/setuptools',
            u'name': u'setuptools', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'https://pip.pypa.io/',
            u'name': u'pip', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'http://bitbucket.org/pypa/wheel/',
            u'name': u'wheel', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'tox', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'certifi', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'wincertstore',
            u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'setuptools____ssl',
            u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'pytest', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'virtualenv', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'scriptttest', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'mock', u'r': u'6'},
            {u'color': u'#5F9EA0', u'link': u'', u'name': u'ed2551911', u'r': u'6'},
```

```
{u'color': u'#5F9EA0', u'link': u'', u'name': u'keyring', u'r': u'6'},
{u'color': u'#5F9EA0', u'link': u'', u'name': u'argparse', u'r': u'6'},
{u'color': u'#5F9EA0', u'link': u'', u'name': u'pyxdg', u'r': u'6'},
{u'color': u'#5F9EA0', u'link': u'', u'name': u'jsonschema', u'r': u'6'},
{u'color': u'#5F9EA0', u'link': u'', u'name': u'coverage', u'r': u'6'},
{u'color': u'#5F9EA0', u'link': u'', u'name': u'pytest-cov',
 u'r': u'6'}]],
u'time': u'2015-05-08T03:52:59.542732'}
```

See also [How to use linkdraw](#).

## 2.4.3 NetworkX

```
>>> pkg.draw('networkx')
>>> <networkx.classes.digraph.DiGraph at 0x7fbe2311dbd0>
```

## 2.5 Check cache

Stores parsed dependency metadata to pickles data file. The file name is `py-deps.pickle` in default.

Listing cached data with the `list_data` method of `Container`..

```
>>> from py_deps import Container
>>> Container().list_data()
{'py-deps': [py-deps, setuptools, pip, wheel, networkx, decorator],
 (snip)}
```

Read the cached package with `read_data` method of `Container`. This method returns `Package.traced_chain`.

```
>>> Container().read_data(('py-deps', '0.3.0'))
[py-deps, setuptools, pip, wheel, networkx, decorator]
```



---

## References

---

py\_deps.deps module.

**class** `py_deps.deps.Depth`

Bases: `object`

Cache of the dependency level.

**get** (*pkg\_name*)

Get depth of package.

**Return type** `int`

**Returns** dependency level.

**Parameters** **pkg\_name** (*str*) – package name.

**parse** (*pkgs*, *pkg\_name*)

Parse dependency depth.

**Parameters**

- **pkgs** (*list*) – package nodes list.
- **pkg\_name** (*str*) – package name.

**set** (*pkg\_name*, *depth*)

Set depth of package.

**Parameters**

- **pkg\_name** (*str*) – package name
- **depth** (*int*) – dependency level

**class** `py_deps.deps.Node` (*name*, *version=None*, *url=None*)

Bases: `object`

Node object class.

**add\_targets** (*nodes*)

Add targets.

**Parameters** **nodes** (*list*) – nodes list

**add\_test\_targets** (*nodes*)

Add test targets.

**Parameters** **nodes** (*list*) – nodes list for testing.

**remove\_targets** (\*nodes)

Remove targets.

**Parameters** \*nodes (\*list) – nodes list

**update\_depth** (depth)

set dependency level.

**Parameters** depth (int) – dependency level.

**class** py\_deps.deps.**Package** (name, version=None, update\_force=False, \*\*kwargs)

Bases: object

Package class.

**cleanup** (alldir=False)

Cleanup temporary build directory.

**Parameters** alldir (bool) – Remove all temporary directories. (default: False)

**Return type** None

**draw** (draw\_type=None, decode\_type='', disable\_time=False, disable\_descr=False, link\_prefix=None)

Generate drawing data.

**Parameters**

- **draw\_type** (str) – [dot|blockdiag|linkdraw]
- **decode\_type** (str) – ['|json(linkdraw)]

**trace\_chain** (pkg\_name=None)

Trace dependency chain.

**Parameters** pkg\_name (str) – package name

**Return type** None

**class** py\_deps.deps.**Target** (nodename, specs, extras=False)

Bases: [py\\_deps.deps.Node](#)

Target objects.

**add\_targets** (nodes)

Add targets.

**Parameters** nodes (list) – nodes list

**add\_test\_targets** (nodes)

Add test targets.

**Parameters** nodes (list) – nodes list for testing.

**remove\_targets** (\*nodes)

Remove targets.

**Parameters** \*nodes (\*list) – nodes list

**update\_depth** (depth)

set dependency level.

**Parameters** depth (int) – dependency level.

**py\_deps.deps.latest\_version** (pkg\_name)

retrieve latest version.

**Return type** str



**Returns** latest version

**Parameters** **pkg\_name** (*str*) – package name.

`py_deps.deps.search(pkg_name, exactly=False)`  
Search package.

**Return type** list

**Returns** search packages

**Parameters**

- **pkg\_name** (*str*) – package name.
- **exactly** (*bool*) – exactly match only.

`py_deps.deps.u2h(name)`  
Change underscore to hyphen of package name.

**Return type** str

**Returns** string replaced underscore with hyphen

**Parameters** **name** (*str*) – package name

`py_deps.graph` module.

**class** `py_deps.graph.Graph(chain_data, link_prefix=None)`  
Bases: `object`

Graph data generate abstract class.

**count\_depth()**  
Count each depth.

**generate\_nodes()**  
Generate nodes data.

**class** `py_deps.graph.Linkdraw(chain_data, link_prefix=None)`  
Bases: `py_deps.graph.Graph`

Linkdraw object class.

**count\_depth()**  
Count each depth.

**disable\_descr()**  
Disable description.

**disable\_time()**  
Disable time.

**generate\_data()**  
Generate Linkdraw data.

**generate\_edges()**  
Generate edges data.

**generate\_nodes()**  
Generate nodes data.

**generate\_position()**  
Generate position.

level 0 is the base depth, that is one only.

Nodes depth images is follows.:

```
0---1---2---3
|   \--2---3---4---5
|   \--2---3
|   \--3---4
1---2---3
    \--3
```

**base position** 0

**level distance** the value that the smaller of the SVG width and height divided by the depth level.

**node distance** the circumference drawn by the same level nodes divided by the num same level.

**depth level** `len(self.count_depth().key())`

**deepest level** `self.count_depth().keys()[-1]`

**num same level** `self.count_depth().get(int:'some level')`

**class** `py_deps.graph.Metadata`

Bases: `object`

Metadata object class.

**class** `py_deps.graph.Networkx(chain_data, link_prefix=None)`

Bases: `py_deps.graph.Graph`

Networkx object class.

**count\_depth()**

Count each depth.

**generate\_data()**

Generate networkx graph data.

**generate\_edges()**

Generate edges data.

**generate\_nodes()**

Generate nodes data.

`py_deps.graph.color(depth)`

color by depth level.

**Return type** `str`

**Returns** hex color code based blue.

**Parameters** `depth(int)` – dependency level

`py_deps.graph.pretty_print(chain_data)`

Pretty print on terminal.

**Parameters** `chain_data(list)` – List of `deps.Node`

`py_deps.graph.router(chain_data, draw_type=None, decode_type='', disable_time=False, disable_descr=False, link_prefix=None)`

Routing drawing tool.

`py_deps.cache` module.

```
class py_deps.cache.Container (cache_name=None)
```

Bases: object

Package container class.

```
list_data ()
```

Return dictionary stored package metadata.

**Return type** dict

**Returns** packages metadata

```
read_data (key)
```

Read traced\_chain data.

**Return type** list

**Returns** dependency chain list

**Parameters** **key** (*tuple*) – package name, version

```
store_data (key, data)
```

store traced\_chain data.

```
class py_deps.cache.Memcached (servers=None, username=None, password=None, behaviors=None)
```

Bases: [py\\_deps.cache.Container](#)

Cache backend is Memcached.

```
list_data ()
```

Return dictionary stored package metadata.

**Return type** dict

**Returns** packages metadata

```
read_data (key)
```

Read traced\_chain data.

**Return type** list

**Returns** dependency chain list

**Parameters** **key** (*tuple*) – package name, version

```
store_data (key, data)
```

Store traced\_chain data.

**Parameters**

- **key** (*tuple*) – package name, version
- **data** (*list*) – traced dependency chain data

```
class py_deps.cache.Pickle (cache_name=None)
```

Bases: [py\\_deps.cache.Container](#)

Cache backend is Pickle.

```
list_data ()
```

Return dictionary stored package metadata.

**Return type** dict

**Returns** packages metadata

```
load_cache ()
```

Load cache file.

**read\_data** (*key*)

Read traced\_chain data.

**Return type** list

**Returns** dependency chain list

**Parameters** **key** (*tuple*) – package name, version

**save\_cache** ()

Save cache file.

**store\_data** (*key*, *data*)

Store traced\_chain data.

**Parameters**

- **key** (*tuple*) – package name, version
- **data** (*list*) – traced dependency chain data

`py_deps.cache.backend` (*\*\*kwargs*)

Specify cache backend.

**Return type** `py_deps.cache.Container`

**Returns** Pickle object or Memcached object.

**Parameters** **\*\*kwargs** – parameters

**servers** Memcached servers (required in Memcached)

Required when Memcached. Using Pickle in default without `servers`.

**username** Memcached SASL username (optional)

**password** Memcached SASL password (optional)

**cache\_name** Pickle filename (default, optional)

`py_deps.exceptions` module.

**exception** `py_deps.exceptions.BackendFailure` (*message=None*)

Bases: `py_deps.exceptions.Error`

PyPI service down.

**with\_traceback** ()

Exception.with\_traceback(tb) – set self.\_\_traceback\_\_ to tb and return self.

**exception** `py_deps.exceptions.BrokenPackage` (*message=None*)

Bases: `py_deps.exceptions.Error`

BrokenPackage.

**with\_traceback** ()

Exception.with\_traceback(tb) – set self.\_\_traceback\_\_ to tb and return self.

**exception** `py_deps.exceptions.Error` (*message=None*)

Bases: `Exception`

Base error class.

**with\_traceback** ()

Exception.with\_traceback(tb) – set self.\_\_traceback\_\_ to tb and return self.

**exception** `py_deps.exceptions.InvalidMetadata` (*message=None*)

Bases: `py_deps.exceptions.Error`

Invalid package metadata.

**with\_traceback** ()

Exception.with\_traceback(tb) – set self.\_\_traceback\_\_ to tb and return self.

**exception** `py_deps.exceptions.NotFound` (*message=None*)

Bases: `py_deps.exceptions.Error`

Not Found.

**with\_traceback** ()

Exception.with\_traceback(tb) – set self.\_\_traceback\_\_ to tb and return self.



---

**Design notes**

---

**4.1 Memo**

purpose	setup.py	Egg	wheel (metadata.json)
requires	install_requires	requires.txt	run_requires.requires
no PyPI	dependency_links	dependency_links.txt	
optional	extras_require		extras
extra resource			run_requires.extra
test			test_requires.requires





---

**History**

---

**5.1 0.5.5 (2015-08-19)**

- Adds TimeoutError, ConnectionRefusedError / socket.error exceptions.
- Adds error handling the PyPI service down.
- Changes Sphinx theme to sphinx\_rtd\_theme.
- Adds counter of the each depth.

**5.2 0.5.4 (2015-07-22)**

- Adds latest\_version function.
- Adds link\_prefix for overriding node link.

**5.3 0.5.3 (2015-07-19)**

- Adds exception InvalidMetadata type.

**5.4 0.5.2 (2015-07-15)**

- Fixes duplicated line of linkdraw.
- Changes linkdraw colors by dependencies depth.
- Adds depth property to graph nodes.
- Adds parsing the package dependency depth.

**5.5 0.5.1 (2015-07-12)**

- Changes Package.search method to a function.
- Fixes infinity loop trace\_chain.

- Fixes *None* redundant second argument of *dict.get()*.
- Fixes *len()* used to check if collection has items.
- Fixes old-style string formatting.

## 5.6 0.5.0 (2015-06-22)

- Supports memcached as the backend of cache.

## 5.7 0.4.6 (2015-06-11)

- Fixes not control the version of package correctly.

## 5.8 0.4.5 (2015-06-07)

- Adds disable time, descr for Linkdraw.

## 5.9 0.4.4 (2015-06-03)

- Removes debug print.

## 5.10 0.4.3 (2015-06-02)

- Adds JSON decoder for Linkdraw.

## 5.11 0.4.2 (2015-05-31)

- Fixes #7 not handling the failure of python setup egg\_info.
- Adds py\_deps.exceptions module.
- Adds py\_deps.logger module.
- Fixes issues of DistributionNotFound, InstallationError.

## 5.12 0.4.1 (2015-05-28)

- Adds Container.list\_data method.
- Unsupports wheel format for distribution.

### 5.13 0.4.0 (2015-05-20)

- Searching packages from PyPI.

### 5.14 0.3.0 (2015-05-12)

- Supports NetworkX DiGraph objects.
- Changes to use mock instead of `pip.req.RequirementSet.prepare_files`.
- Coverage 98% over.

### 5.15 0.2.0 (2015-05-10)

- Cache the parsed dependencies.
- Fixes setting the url of node and targets.

### 5.16 0.1.1 (2015-05-08)

- Fixes test data of `pretty_print`, `linkdraw`.

### 5.17 0.1.0 (2015-05-07)

- Supports generating `linkdraw` data.
- Supports pip 6.1.1 over.
- Supports wheel format for distribution.
- Adds unit tests.

### 5.18 0.0.1 (2015-04-29)

- First release



---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`



## p

- `py_deps`, [5](#)
- `py_deps.cache`, [14](#)
- `py_deps.deps`, [11](#)
- `py_deps.exceptions`, [16](#)
- `py_deps.graph`, [13](#)





## A

`add_targets()` (`py_deps.deps.Node` method), 11  
`add_targets()` (`py_deps.deps.Target` method), 12  
`add_test_targets()` (`py_deps.deps.Node` method), 11  
`add_test_targets()` (`py_deps.deps.Target` method), 12

## B

`backend()` (in module `py_deps.cache`), 16  
`BackendFailure`, 16  
`BrokenPackage`, 16

## C

`cleanup()` (`py_deps.deps.Package` method), 12  
`color()` (in module `py_deps.graph`), 14  
`Container` (class in `py_deps.cache`), 14  
`count_depth()` (`py_deps.graph.Graph` method), 13  
`count_depth()` (`py_deps.graph.Linkdraw` method), 13  
`count_depth()` (`py_deps.graph.Networkx` method), 14

## D

`Depth` (class in `py_deps.deps`), 11  
`disable_descr()` (`py_deps.graph.Linkdraw` method), 13  
`disable_time()` (`py_deps.graph.Linkdraw` method), 13  
`draw()` (`py_deps.deps.Package` method), 12

## E

`Error`, 16

## G

`generate_data()` (`py_deps.graph.Linkdraw` method), 13  
`generate_data()` (`py_deps.graph.Networkx` method), 14  
`generate_edges()` (`py_deps.graph.Linkdraw` method), 13  
`generate_edges()` (`py_deps.graph.Networkx` method), 14  
`generate_nodes()` (`py_deps.graph.Graph` method), 13  
`generate_nodes()` (`py_deps.graph.Linkdraw` method), 13  
`generate_nodes()` (`py_deps.graph.Networkx` method), 14  
`generate_position()` (`py_deps.graph.Linkdraw` method), 13  
`get()` (`py_deps.deps.Depth` method), 11  
`Graph` (class in `py_deps.graph`), 13

## I

`InvalidMetadata`, 16

## L

`latest_version()` (in module `py_deps.deps`), 12  
`Linkdraw` (class in `py_deps.graph`), 13  
`list_data()` (`py_deps.cache.Container` method), 15  
`list_data()` (`py_deps.cache.Memcached` method), 15  
`list_data()` (`py_deps.cache.Pickle` method), 15  
`load_cache()` (`py_deps.cache.Pickle` method), 15

## M

`Memcached` (class in `py_deps.cache`), 15  
`Metadata` (class in `py_deps.graph`), 14

## N

`Networkx` (class in `py_deps.graph`), 14  
`Node` (class in `py_deps.deps`), 11  
`NotFound`, 17

## P

`Package` (class in `py_deps.deps`), 12  
`parse()` (`py_deps.deps.Depth` method), 11  
`Pickle` (class in `py_deps.cache`), 15  
`pretty_print()` (in module `py_deps.graph`), 14  
`py_deps` (module), 5  
`py_deps.cache` (module), 14  
`py_deps.deps` (module), 11  
`py_deps.exceptions` (module), 16  
`py_deps.graph` (module), 13

## R

`read_data()` (`py_deps.cache.Container` method), 15  
`read_data()` (`py_deps.cache.Memcached` method), 15  
`read_data()` (`py_deps.cache.Pickle` method), 15  
`remove_targets()` (`py_deps.deps.Node` method), 11  
`remove_targets()` (`py_deps.deps.Target` method), 12  
`router()` (in module `py_deps.graph`), 14

## S

`save_cache()` (`py_deps.cache.Pickle` method), [16](#)  
`search()` (in module `py_deps.deps`), [13](#)  
`set()` (`py_deps.deps.Depth` method), [11](#)  
`store_data()` (`py_deps.cache.Container` method), [15](#)  
`store_data()` (`py_deps.cache.Memcached` method), [15](#)  
`store_data()` (`py_deps.cache.Pickle` method), [16](#)

## T

`Target` (class in `py_deps.deps`), [12](#)  
`trace_chain()` (`py_deps.deps.Package` method), [12](#)

## U

`u2h()` (in module `py_deps.deps`), [13](#)  
`update_depth()` (`py_deps.deps.Node` method), [12](#)  
`update_depth()` (`py_deps.deps.Target` method), [12](#)

## W

`with_traceback()` (`py_deps.exceptions.BackendFailure` method), [16](#)  
`with_traceback()` (`py_deps.exceptions.BrokenPackage` method), [16](#)  
`with_traceback()` (`py_deps.exceptions.Error` method), [16](#)  
`with_traceback()` (`py_deps.exceptions.InvalidMetadata` method), [17](#)  
`with_traceback()` (`py_deps.exceptions.NotFound` method), [17](#)