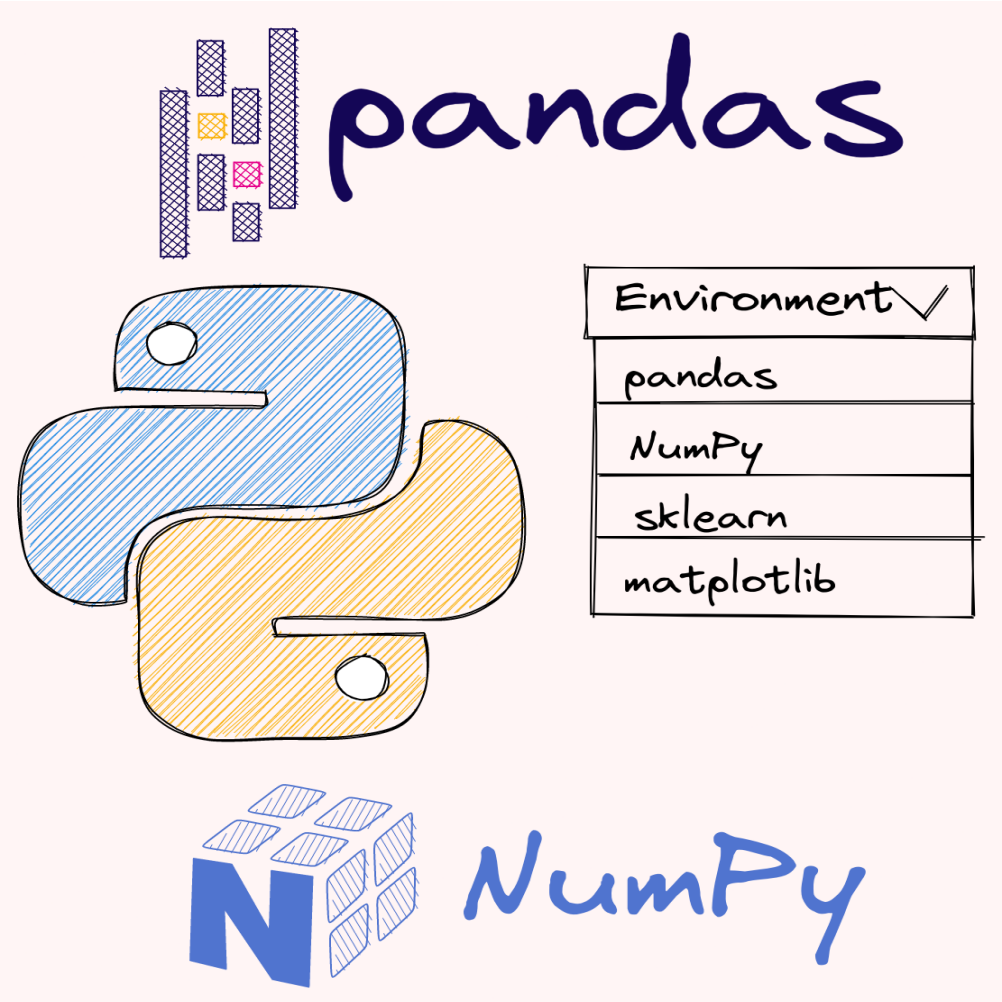
Efficient Python Tricks and Tools for Data Scientists

Environment Management - By Khuyen Tran



virtualenv-clone: Create a Copy of a Virtual Environment

Sometimes you might want to use the same virtual environment for 2 different directories. If you want to create a copy of a virtual environment, use virtualenv-clone.

The code below shows how to use virtualenv-clone.

```
$ pip install virtualenv-clone
$ virtualenv-clone old_venv/ new_venv/

$ source new_venv/bin/activate
```

Link to virtualenv-clone.

pip-autoremove: Remove a Package and Its Unused Dependencies

When using `pip uninstall`, you only remove a specific package.

```
$ pip uninstall pandas-profiling[notebook] -y
```

```
Found existing installation: pandas-profiling
3.1.0
Uninstalling pandas-profiling-3.1.0:
  Successfully uninstalled pandas-profiling-
3.1.0
```

Wouldn't it be nice if you can uninstall that package and its unused dependencies? That is when `pip-autoremove` comes in handy.

```
!pip install -U pandas-profiling[notebook]
```

```
$ pip-autoremove pandas-profiling[notebook] -y
```

```
Found existing installation: pandas-profiling
3.1.0
Uninstalling pandas-profiling-3.1.0:
  Successfully uninstalled pandas-profiling-
3.1.0
Found existing installation: seaborn 0.11.2
Uninstalling seaborn-0.11.2:
  Successfully uninstalled seaborn-0.11.2
Found existing installation: tangled-up-in-
unicode 0.1.0
Uninstalling tangled-up-in-unicode-0.1.0:
  Successfully uninstalled tangled-up-in-
unicode-0.1.0
```

By using pip-autoremove, pandas-profiling and its unused dependencies are removed!

Link to pip-autoremove.

pipreqs: Generate requirements.txt File for Any Project Based on Imports

```
!pip install pipreqs
```

`pip freeze` saves all packages in the environment, including ones that you don't use in your current project. To generate a `requirements.txt` based on imports, use pipreqs.

For example, to save all packages in your current project to a `requirements.txt` file, run:

```
$ pipreqs .
```

```
INFO: Successfully saved requirements file in
./requirements.txt
```

Your `requirements.txt` should look like below:

```
numpy==1.21.4
pandas==1.3.4
pyinstrument==4.0.3
typer==0.4.0
```

Usage of pipreqs:

```
Usage:
    pipreqs [options] [<path>]

Arguments:
    <path>                The path to the
directory containing the application files for
which a requirements file
                          should be generated
(defaults to the current working directory)

Options:
    --use-local           Use ONLY local
package info instead of querying PyPI
    --pypi-server <url>   Use custom PyPi
server
    --proxy <url>         Use Proxy, parameter
will be passed to requests library. You can
also just set the
                          environments
parameter in your terminal:
                          $ export
HTTP_PROXY="http://10.10.1.10:3128"
                          $ export
HTTPS_PROXY="https://10.10.1.10:1080"
    --debug               Print debug
information
    --ignore <dirs>...    Ignore extra
directories, each separated by a comma
    --no-follow-links     Do not follow
symbolic links in the project
```

```
    --encoding <charset>  Use encoding
parameter for file open
    --savepath <file>     Save the list of
requirements in the given file
    --print               Output the list of
requirements in the standard output
    --force               Overwrite existing
requirements.txt
    --diff <file>         Compare modules in
requirements.txt to project imports
    --clean <file>        Clean up
requirements.txt by removing modules that are
not imported in project
    --mode <scheme>       Enables dynamic
versioning with <compat>, <gt> or <non-pin>
schemes
                          <compat> | e.g.
Flask~=1.1.2
                          <gt>     | e.g.
Flask>=1.1.2
                          <no-pin> | e.g.
Flask
```

Link to pipreqs.

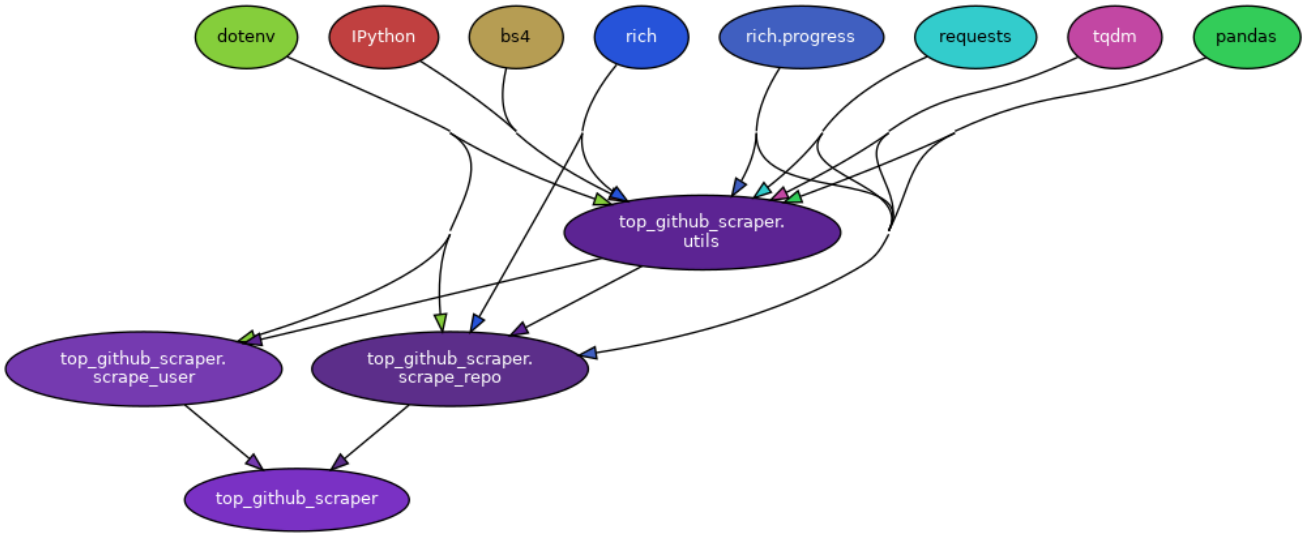
pydeps: Python Module Dependency Visualization

If you want to generate the graph showing the dependencies of your Python modules, try pydeps.

For example, to generate the dependency graph for files in the folder top_github_scraper, I type:

```
$ pydeps top_github_scraper
```

The image below is the output of the command:



The folder structure of top_github_scraper looks like the below:

```
top_github_scraper
├── __init__.py
├── scrape_repo.py
├── scrape_user.py
└── utils.py
```

Link to pydeps.