目录

管理好Python的库：包管理器

- 最新版本： `v1.0`
- 更新时间： `20210425`

简介

介绍如何用包管理器尤其是pip去管理好Python的库。先进行基本的概述，包括常见的Python包管理器有easy_install、setuptools、pip；然后详细阐述pip：包括常用的命令，比如pip install、pip uninstall、pip show、pip list、pip search、pip freeze。其中包括pip install的各种语法和各种举例说明；以及如何给pip下载加速，包括更换源和加代理的方法和具体操作步骤；总结了pip常见的一些问题，包括command not found pip、Could not install packages due to an EnvironmentError Errno 13 Permission denied、Retrying after connection broken by NewConnectionError、使用pip时要注意是python2还是python3、以及部分可以情况下不用pip管理python的库；然后整理了pip相关的一些内容，包括虚拟环境工具pipenv、通过pip查看site-packages位置等；最后给出参考资料。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- crifan/python_summary_package_manager: 管理好Python的库：包管理器

如何使用此Gitbook源码去生成发布为电子书

详见：crifan/gitbook_template: demo how to use crifan gitbook template and demo

在线浏览

- 管理好Python的库：包管理器 book.crifan.com
- 管理好Python的库：包管理器 crifan.github.io

离线下载阅读

- 管理好Python的库：包管理器 PDF

- 管理好Python的库：包管理器 ePub
- 管理好Python的库：包管理器 Mobi

版权说明

鸣谢

感谢我的老婆**陈雪**的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

更多其他电子书

Python包管理器概述

Python开发期间，经常涉及到需要安装各种Python库，用于实现不同的功能。

Python中用**包管理器**来管理Python库。

目前现存有很多种Python的包管理器：

- Python包管理器
    - `easy_install`
        - `easy_install` 基于旧的包格式： `egg`
        - 现在很少用了
            - 只有当你只有 `egg` 的格式时，才需要用到 `easy_install`
    - `setuptools`
        - `setuptools` 包含了（内部用到了） `easy_install`
        - 现在用的很少
    - **pip**
        - 历史
            - 2008年发布
        - 内部实现
            - 基于新的包的格式： `wheel`
        - `pip` 往往是Python自带就有的，无需额外安装。且功能强大，很好用。
        - `pip` 是目前Python中**最常见**的包管理器

pip

`pip` 是Python中最常用的**包管理器**。

pip常用命令

pip常用的命令有：

- `pip install`
- `pip uninstall`
- `pip search`
- `pip show`
- `pip list`
- `pip freeze`

后续会详细介绍。

典型的pip使用逻辑

对于使用pip管理Python库的典型逻辑是：

用 `virtualevn` 或 `pipenv` 等创建某Python项目的Python虚拟环境后

进入虚拟环境，然后

- 去用 `pip install xxx` 安装需要用到的库

Python项目开发期间：

- 如果不需要某库，用 `pip uninstall xxx` 去删除掉
- 如果不知道某个库的包名是什么，但知道库的名字，可以用 `pip search xxx` 去找找源中是否有该库，以及具体包名是啥
- 想要知道具体某个已安装的库的详情（比如版本，具体安装位置等），可以用 `pip show xxx` 去查看
- 查看当前都安装了哪些库，用 `pip list`

项目开发完毕后， `冻结` 环境，即保存好当前Python环境，用：

```
pip freeze > requirements.txt
```

即生成了 `requirements.txt` ，记录了当前安装了哪些Python库，及其版本信息。

如果别人（比如你的同事接手你的项目）想要重新恢复你的Python项目的环境，则去：

（用和你一样的 `virtualenv` 或 `pipenv` ）创建对应的虚拟环境后，进入虚拟环境，再去：

```
pip install -r requirements.txt
```

安装全部的库，即恢复出了相同的Python环境。

即可，接着愉快的继续（协作）开发项目了。

pip语法

`pip` 的语法，可以通过help查看到：

```
pip --help
```

贴出来供参考：

```
 ⚙ limao@xxx □ ~ □ pip --help

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  download                    Download packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in
  list                        List installed packages.
  show                        Show information about insta
  check                       Verify installed packages ha
  config                      Manage local and global conf
  search                      Search PyPI for packages.
  cache                       Inspect and manage pip's whe
  wheel                       Build wheels from your requi
  hash                        Compute hashes of package ar
  completion                  A helper command used for co
  debug                       Show information useful for
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  --isolated                  Run pip in an isolated mode,
  -v, --verbose               Give more output. Option is
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output. Option is
                              ERROR, and CRITICAL logging
  --log <path>                Path to a verbose appending
  --no-input                  Disable prompting for input.
  --proxy <proxy>             Specify a proxy in the form
  --retries <retries>         Maximum number of retries ea
  --timeout <sec>             Set the socket timeout (defa
  --exists-action <action>    Default action when a path a
  --trusted-host <hostname>   Mark this host or host:port
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certifica
                              PEM format.
  --cache-dir <dir>           Store the cache data in <dir
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyP
                              download. Implied with --no-
  --no-color                  Suppress colored output.
  --no-python-version-warning
                              Silence deprecation warnings
  --use-feature <feature>     Enable new functionality, th
  --use-deprecated <feature>  Enable deprecated functional
```

pip install

可以用 `pip install` 安装Python的库。

语法介绍

可通过

```
pip install --help
```

或：

```
pip help install
```

查看细节。

贴出来供参考：

```
⚙ limao@xxx □ ~ □ pip install --help

Usage:
  pip install [options] <requirement specifier> [package-in
  pip install [options] -r <requirements file> [package-inc
  pip install [options] [-e] <vcs project url> ...
  pip install [options] [-e] <local project path> ...
  pip install [options] <archive url/path> ...

Description:
  Install packages from:

  - PyPI (and other indexes) using requirement specifiers.
  - VCS project urls.
  - Local project directories.
  - Local or remote source archives.

  pip also supports installing from "requirements files", w
  an easy way to specify a whole environment to be installe

Install Options:
  -r, --requirement <file>    Install from the given requir
  -c, --constraint <file>     Constrain versions using the
  --no-deps                   Don't install package depende
  --pre                       Include pre-release and deve
  -e, --editable <path/url>   Install a project in editable
                              or a VCS url.
  -t, --target <dir>          Install packages into <dir>.
                              Use --upgrade to replace exis
  --platform <platform>       Only use wheels compatible wi
                              this option multiple times to
  --python-version <python_version>
                              The Python interpreter versio
                              Defaults to a version derived
                              up to three dot-separated int
                              minor version can also be giv
  --implementation <implementation>
                              Only use wheels compatible wi
                              or 'ip'. If not specified, th
                              force implementation-agnostic
  --abi <abi>                 Only use wheels compatible wi
                              current interpreter abi tag i
                              supported by the target inter
                              --platform, and --python-vers
  --user                      Install to the Python user in
                              %APPDATA%\Python on Windows.
                              details.)
  --root <dir>                Install everything relative t
  --prefix <dir>              Installation prefix where lib
  --src <dir>                 Directory to check out editab
```

```
                              path>/src". The default for 
  -U, --upgrade               Upgrade all specified package
                              depends on the upgrade-strate
  --upgrade-strategy <upgrade_strategy>
                              Determines how dependency up
                              dependencies are upgraded re
                              requirements of the upgraded
                              satisfy the requirements of 
  --force-reinstall           Reinstall all packages even 
  -I, --ignore-installed      Ignore the installed package
                              package is of a different ve
  --ignore-requires-python    Ignore the Requires-Python i
  --no-build-isolation        Disable isolation when build
                              PEP 518 must be already inst
  --use-pep517                Use PEP 517 for building sou
  --install-option <options>  Extra arguments to be suppli
                              install-scripts=/usr/local/b
                              options to setup.py install.
                              absolute path.
  --global-option <options>   Extra global options to be s
  --compile                   Compile Python source files 
  --no-compile                Do not compile Python source
  --no-warn-script-location   Do not warn when installing 
  --no-warn-conflicts         Do not warn about broken dep
  --no-binary <format_control>
                              Do not use binary packages. 
                              value. Accepts either ":all:
                              the colons), or one or more 
                              packages are tricky to compi
  --only-binary <format_control>
                              Do not use source packages. 
                              value. Accepts either ":all:
                              or more package names with c
                              to install when this option 
  --prefer-binary             Prefer older binary packages
  --require-hashes            Require a hash to check each
                              implied when any package in 
  --progress-bar <progress_bar>
                              Specify type of progress to 
  --no-clean                  Don't clean up build directo

Package Index Options:
  -i, --index-url <url>       Base URL of the Python Packa
                              repository compliant with PE
                              the same format.
  --extra-index-url <url>     Extra URLs of package indexe
                              as --index-url.
  --no-index                  Ignore package index (only l
  -f, --find-links <url>      If a URL or path to an html 
                              wheel (.whl) files. If a loc
                              in the directory listing. Li
```

```
General Options:
  -h, --help                  Show help.
  --isolated                  Run pip in an isolated mode,
  -v, --verbose               Give more output. Option is a
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output. Option is a
                              ERROR, and CRITICAL logging l
  --log <path>                Path to a verbose appending l
  --no-input                  Disable prompting for input.
  --proxy <proxy>             Specify a proxy in the form
  --retries <retries>         Maximum number of retries eac
  --timeout <sec>             Set the socket timeout (defau
  --exists-action <action>    Default action when a path a
  --trusted-host <hostname>   Mark this host or host:port p
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certificat
                              PEM format.
  --cache-dir <dir>           Store the cache data in <dir>
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI
                              download. Implied with --no-i
  --no-color                  Suppress colored output.
  --no-python-version-warning
                              Silence deprecation warnings
  --use-feature <feature>     Enable new functionality, tha
  --use-deprecated <feature>  Enable deprecated functionali
```

用pip安装库

- 安装
  - 普通安装

    ```
    pip install xxx
    ```

    - 把pip作为python的模块去安装

      ```
      /your/specific/python -m pip install xxx
      ```

  - 安装特定版本

    ```
    pip install 'xxx==version_number'
    ```

  - 安装 `A` 时，同时安装A所依赖的库： `B`

    ```
    pip install A[B]
    ```

- 升级
  - 升级已安装的库

    ```
    pip install --upgrade xxx
    ```

    - 等价于

      ```
      pip install -U xxx
      ```

安装来源和方式

- 从文件安装
  - `whl文件`
    - `在线` 文件

      ```
      pip install https://storage.googleapis.com/tens
      ```

    - `本地` 文件

      ```
      pip install /Users/crifan/dev/dev_tool/tensorfl
      ```

- 从源安装
  - `源码包`
    - 语法

```
pip install PackageName --no-index --find-links
```

- 举例

```
pip install -U SQLAlchemy --no-index --find-lin
```

`pip install` 举例

- pipenv
  - 安装pinenv

    ```
    pip3 install pipenv --user
    ```

  - 用pip升级pipenv

    ```
    pip install --user --upgrade pipenv
    ```

- 安装Django的sendfile插件

  ```
  pip3 install django-sendfile
  ```

- 安装Django的CORS插件

  ```
  pip3 install django-cors-headers
  ```

- 安装查看文件类型的MIME

  ```
  pip install mime
  ```

- 安装Amazon的boto3

  ```
  pip install boto3
  ```

- 安装Web自动化工具 `selenium`

  ```
  pip install -U selenium
  ```

- 安装虚拟环境工具 `virtualenv`

  ```
  pip install virtualenv
  ```

  - 升级virtualenv

    ```
    pip install --upgrade virtualenv
    ```

- 安装MongoDB数据库的Python的driver

  ```
  pip install pymongo
  ```

  - 升级MongoDB

    ```
    pip install --upgrade pymongo
    ```

- 安装数据库ORM：sqlalchemy

  ```
  pip install -U sqlalchemy
  ```

  - 安装指定版本

    ```
    pip install 'sqlalchemy==1.1.0b3'
    ```

- 安装爬虫工具scrapy

  ```
  pip install scrapy
  ```

通过python的模块方式调用pip去安装

- 通过python的模块方式调用pip去安装pylint

  ```
  /usr/local/bin/python3 -m pip install -U pylint --user
  ```

  - 类似的写法

    ```
    python -m pip install -U pylint
    ```

安装之前加额外参数

- 在重新安装pycurl之前，加上额外的参数

  ```
  pip uninstall pycurl
  export PYCURL_SSL_LIBRARY=openssl
  export LDFLAGS=-L/usr/local/opt/openssl/lib;export CPPF
  ```

安装库A同时安装依赖库B

- 安装gunicorn同时安装所依赖的其他库
  - eventlet

    ```
    pip install gunicorn[eventlet]
    ```

  - gevent

```
pip install gunicorn[gevent]
```

- tornado

  ```
  pip install gunicorn[tornado]
  ```

- gthread

  ```
  pip install gunicorn[gthread]
  ```

给出部分库的详细安装日志

- 安装Python项目部署工具supervisor

  ```
  pip install supervisor
  ```

  - 如果遇到权限问题，则加上 `--user`

    ```
    pip install supervisor --user
    ```

    - 详细日志

      ```
      ➜  robotDemo pip install supervisor --user
      Collecting supervisor
      Collecting meld3&gt;=0.6.5 (from supervisor)
        Using cached https://files.pythonhosted.org/p
      pipenv 11.10.0 requires certifi, which is not i
      pipenv 11.10.0 requires requests[security], whi
      pipenv 11.10.0 requires virtualenv, which is no
      matplotlib 1.3.1 requires nose, which is not in
      matplotlib 1.3.1 requires tornado, which is not
      Installing collected packages: meld3, superviso
        The scripts echo_supervisord_conf, pidproxy,
        Consider adding this directory to PATH or, if
      Successfully installed meld3-1.0.2 supervisor-3
      ```

- 安装Excel处理工具 `openpyxl`

  ```
  pip install openpyxl
  ```

  - 详细日志

```
➜  英语资源 pip install openpyxl
Collecting openpyxl
  Downloading openpyxl-2.5.1.tar.gz (169kB)
    100% |████████████████████████████████| 174kB 1
Collecting jdcal (from openpyxl)
  Downloading jdcal-1.3.tar.gz
Collecting et_xmlfile (from openpyxl)
  Downloading et_xmlfile-1.0.1.tar.gz
Building wheels for collected packages: openpyxl, j
  Running setup.py bdist_wheel for openpyxl … done
  Stored in directory: /Users/crifan/Library/Caches
  Running setup.py bdist_wheel for jdcal … done
  Stored in directory: /Users/crifan/Library/Caches
  Running setup.py bdist_wheel for et-xmlfile … don
  Stored in directory: /Users/crifan/Library/Caches
Successfully built openpyxl jdcal et-xmlfile
Installing collected packages: jdcal, et-xmlfile, o
Successfully installed et-xmlfile-1.0.1 jdcal-1.3 o
```

给pip下载加速

用 `pip` 去安装库时，往往速度**很慢**，甚至连不上

因为通过pip去安装库，需要先去下载库的安装包，而安装包的来源，默认的是：

https://pypi.org

其服务器在国外，国内连接的速度往往很慢，导致：

pip安装库的速度很慢

解决办法：

- 方式1：**更换pip的源**
  - 为某个国内的pypi.org的镜像服务器
    - 国内用户访问速度更快
- 方式2：**给网络加代理**
  - 通过代理访问 pypi.org，实现下载加速的效果

更换pip的源

思路

- 新建（全局的、系统的） `pip` 的配置文件
    - `Windows` ： `%HOMEPATH%\pip\pip.ini`
        - 举例
            - `C:\Users\Administrator\pip\pip.ini`
    - `macOS` ： `~/.pip/pip.conf`
        - 举例
            - `/Users/crifanli/.pip/pip.conf`
- 给配置文件加上国内的某个pip的源

步骤

（如果不存在的话）新建pip的配置文件

```
mkdir ~/.pip
vim ~/.pip/pip.conf
```

去更新配置内容，格式是：

```
[global]
index-url = pypi_mirror_url

[install]
trusted-host=pypi_mirror_url_host
```

参数解释：

- pypi_mirror_url
    - 常见可选的pip的源
        - 阿里云：http://mirrors.aliyun.com/pypi/simple/
        - 清华大学：https://pypi.tuna.tsinghua.edu.cn/simple
        - 豆瓣：http://pypi.douban.com/simple
- pypi_mirror_url_host
    - 对应 `host`
        - `mirrors.aliyun.com`
        - `pypi.tuna.tsinghua.edu.cn`
        - `pypi.douban.com`

下面给出实例：

推荐：清华大学的pip的源

```
[global]
index-url = https://pypi.tuna.tsinghua.edu.cn/simple

[install]
trusted-host=pypi.tuna.tsinghua.edu.cn
```

实测：速度暴快，> `1MB/s`

推荐：阿里云的pip的源

```
[global]
index-url = http://mirrors.aliyun.com/pypi/simple

[install]
trusted-host=mirrors.aliyun.com
```

速度也不错，也很稳定。

可选：豆瓣的pip的源

```
[global]
index-url = http://pypi.douban.com/simple

[install]
trusted-host=pypi.douban.com
```

待确认：淘宝taobao的源

后来看到淘宝的npm的镜像

taobao Mirrors

其中也能找到python的源：

Python Mirror

抽空试试是否可用

特殊：现已不推荐：ustc的源

之前推荐的

```
[global]
index-url = http://pypi.mirrors.ustc.edu.cn/simple

[install]
trusted-host=pypi.mirrors.ustc.edu.cn
```

现已不推荐

原因：

- 其内部已跳转到tsinghua的源
    - 202104 后记：现已跳转到 BFSU（北京外国语大学）的镜像了：mirrors.bfsu.edu.cn

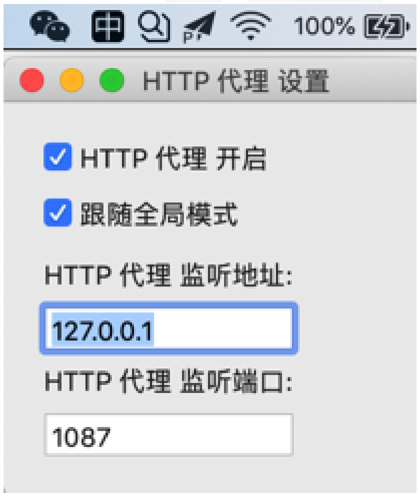
给pip加代理

此处也可以通过给pip加代理的方式实现加速pip下载。

实现方式：用 `pip` 安装时，加上代理：

```
pip --proxy http://127.0.0.1:1087 install pandas
```

其中：

- 代理地址： `127.0.0.1:1087`
  - 对应着此处本地的 `ss` 的 `http代理` ：
    - 

      HTTP 代理 设置

      ✓ HTTP 代理 开启

      ✓ 跟随全局模式

      HTTP 代理 监听地址: 127.0.0.1

      HTTP 代理 监听端口: 1087

pip uninstall

- 卸载已安装的Python库

  ```
  pip uninstall xxx
  ```

- 有时候，pip会提示你，确认是否删除，需要输入 `y` 才能继续删除
  - 如果不希望被提示，直接删除，可以加上 `y`

    ```
    pip uninstall -y xxx
    ```

- 想要卸载某个特定版本，可以指定版本号

  ```
  pip uninstall xxx==verion_number
  ```

`pip uninstall` 举例

- 卸载（已安装，但有问题的）pycurl

  ```
  pip uninstall pycurl
  ```

  - 不询问(是否删除)而直接删除

    ```
    pip uninstall -y pycurl
    ```

- 卸载uiautomator2

  ```
  pip uninstall uiautomator2
  ```

- 卸载facebook-wda

  ```
  pip uninstall facebook-wda
  ```

- 卸载 `1.7.0rc1` 版本的 `tensorflow`

  ```
  pip uninstall tensorflow==1.7.0rc1
  ```

- 卸载pyinstaller

  ```
  pip uninstall pyinstaller
  ```

- 卸载pipenv

  ```
  pip3 uninstall pipenv
  ```

- 卸载pip自己

  ```
  python -m pip uninstall pip
  ```

- 卸载mime

  ```
  pip uninstall mime
  ```

- 卸载selenium

  ```
  pip uninstall selenium
  ```

给出部分库的详细卸载日志

- 卸载virtualenv

  ```
  pip uninstall virtualenv
  ```

  - 完整log举例

    ```
    ➜  virtualenv pip uninstall virtualenv
    Uninstalling virtualenv-15.1.0:
    /usr/local/bin/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    /usr/local/lib/python2.7/site-packages/virtualenv
    Proceed (y/n)? y
    Successfully uninstalled virtualenv-15.1.0
    ```

    - 期间需要输入 `y` 才可继续卸载
      - 想要实现，不需要输入 `y` 就可以直接自动卸载，可以加上 `-y` 参数

        ```
        pip uninstall -y virtualenv
        ```

- 卸载supervisor

  ```
  pip2 uninstall supervisor
  ```

  - 详细日志

```
[root@xxx-general-01 robotDemo]# pip2 uninstall s
Uninstalling supervisor-3.3.4:
Would remove:
    /usr/bin/echo_supervisord_conf
    /usr/bin/pidproxy
    /usr/bin/supervisorctl
    /usr/bin/supervisord
    /usr/lib/python2.7/site-packages/supervisor-3
    /usr/lib/python2.7/site-packages/supervisor-3
    /usr/lib/python2.7/site-packages/supervisor/c
    /usr/lib/python2.7/site-packages/supervisor/c
    /usr/lib/python2.7/site-packages/supervisor/d
    /usr/lib/python2.7/site-packages/supervisor/d
    /usr/lib/python2.7/site-packages/supervisor/e
    /usr/lib/python2.7/site-packages/supervisor/h
    /usr/lib/python2.7/site-packages/supervisor/h
    /usr/lib/python2.7/site-packages/supervisor/l
    /usr/lib/python2.7/site-packages/supervisor/m
    /usr/lib/python2.7/site-packages/supervisor/o
    /usr/lib/python2.7/site-packages/supervisor/p
    /usr/lib/python2.7/site-packages/supervisor/p
    /usr/lib/python2.7/site-packages/supervisor/p
    /usr/lib/python2.7/site-packages/supervisor/r
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/s
    /usr/lib/python2.7/site-packages/supervisor/t
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/u
    /usr/lib/python2.7/site-packages/supervisor/v
    /usr/lib/python2.7/site-packages/supervisor/w
    /usr/lib/python2.7/site-packages/supervisor/x
```

```
Proceed (y/n)? y
Successfully uninstalled supervisor-3.3.4
```

pip show

可以用

```
pip show
```

去查看已安装的库/包的详细信息。

语法：

```
pip show xxx
```

`pip show` 举例

Django

```
[root@xx-general-01 xxx]# pip3 show Django
Name: Django
Version: 2.0.6
Summary: A high-level Python Web framework that encourages
Home-page: https://www.djangoproject.com/
Author: Django Software Foundation
Author-email: foundation@djangoproject.com
License: BSD
Location: /usr/lib/python3.4/site-packages
Requires: pytz
Required-by: django-redis
```

supervisor

```
[root@xx-general-01 robotDemo]# pip show supervisor
Name: supervisor
Version: 3.3.4
Summary: A system for controlling process state under UNIX
Home-page: http://supervisord.org/
Author: Chris McDonough
Author-email: chrism@plope.com
License: BSD-derived (http://www.repoze.org/LICENSE.txt)
Location: /usr/lib/python2.7/site-packages
Requires: meld3
Required-by:
```

kafka

```
# pip3 show kafka
Name: kafka
Version: 1.3.5
Summary: Pure Python client for Apache Kafka
Home-page: https://github.com/dpkp/kafka-python
Author: Dana Powers
Author-email: dana.powers@gmail.com
License: Apache License 2.0
Location: /usr/local/lib/python3.6/site-packages
Requires:
Required-by:
```

flask

```
(xxx) ➜  xxx pip show flask
Metadata-Version: 2.0
Name: Flask
Version: 0.11.1
Summary: A microframework based on Werkzeug, Jinja2 and goo
Home-page: http://github.com/pallets/flask/
Author: Armin Ronacher
Author-email: armin.ronacher@active-4.com
Installer: pip
License: BSD
Location: /root/Envs/xxx/lib/python2.7/site-packages
Requires: itsdangerous, Jinja2, Werkzeug, click
Classifiers:
  Development Status :: 4 - Beta
  Environment :: Web Environment
  Intended Audience :: Developers
  License :: OSI Approved :: BSD License
  Operating System :: OS Independent
  Programming Language :: Python
  Programming Language :: Python :: 2
  Programming Language :: Python :: 2.6
  Programming Language :: Python :: 2.7
  Programming Language :: Python :: 3
  Programming Language :: Python :: 3.3
  Programming Language :: Python :: 3.4
  Programming Language :: Python :: 3.5
  Topic :: Internet :: WWW/HTTP :: Dynamic Content
  Topic :: Software Development :: Libraries :: Python Modu
Entry-points:
  [console_scripts]
  flask=flask.cli:main
```

pip list

可以用

```
pip list
```

来查看当前（Python）环境中的已安装的Python库（及其版本信息）

`pip list` 举例

```
 ⚙ limao@xxx □ ~ □ pip list
Package           Version    Location
----------------- ---------- ------------------------------
adbutils          0.7.1
apkutils2         1.0.0
asn1crypto        1.3.0
attrdict          2.0.1
attrs             19.3.0
Automat           20.2.0
autopep8          1.4.4
beautifulsoup4    4.8.2
blinker           1.4
Brotli            1.0.7
bs4               0.0.1
cached-property   1.5.1
certifi           2019.11.28
cffi              1.14.0
chardet           3.0.4
cigam             0.0.3
Click             7.0
constantly        15.1.0
cryptography      2.4.2
cssselect         1.1.0
decorator         4.4.1
defusedxml        0.6.0
Deprecated        1.2.7
deprecation       2.0.7
facebook-wda      1.3.4
Flask             1.1.1
Flask-Login       0.5.0
future            0.18.2
greenlet          1.0.0
h11               0.9.0
h2                3.1.1
hpack             3.0.0
humanize          0.5.1
hyperframe        5.2.0
hyperlink         19.0.0
idna              2.8
incremental       17.5.0
itemadapter       0.1.0
itsdangerous      1.1.0
Jinja2            2.11.1
json5             0.9.5
kaitaistruct      0.8
ldap3             2.6.1
logzero           1.5.0
lxml              4.5.0
MarkupSafe        1.1.1
mitmproxy         5.0.1
```

```
numpy              1.18.1
packaging          20.1
pandas             0.24.2
parsel             1.6.0
passlib            1.7.2
PGet               0.5.0
Pillow             7.0.0
pip                21.0.1
pipenv             2018.11.26
playwright         1.10.0
progress           1.5
Protego            0.1.16
protobuf           3.10.0
publicsuffix2      2.20191221
py                 1.8.1
pyasn1             0.4.8
pyasn1-modules     0.2.8
pycodestyle        2.5.0
pycparser          2.19
pycurl             7.43.0.5
PyDispatcher       2.0.5
pyee               8.1.0
pyelftools         0.26
PyHamcrest         2.0.2
pyOpenSSL          19.0.0
pyparsing          2.4.6
pyperclip          1.7.0
pyquery            1.4.1
pyspider           0.3.10
python-dateutil    2.8.1
pytz               2019.3
PyYAML             5.3
queuelib           1.5.0
requests           2.22.0
retry              0.9.2
ruamel.yaml        0.16.7
Scrapy             2.2.1
service-identity   18.1.0
setuptools         41.2.0
six                1.14.0
sortedcontainers   2.1.0
soupsieve          1.9.5
spark-parser       1.8.9
tblib              1.7.0
tornado            4.5.3
tushare            1.2.48
Twisted            20.3.0
typing-extensions  3.7.4.3
u-msgpack-python   2.6.0
uiautomator2       2.5.3
uncompyle6         3.6.2      /Users/limao/dev/tools/python_
```

```
urllib3            1.25.8
urwid              2.0.1
virtualenv         16.7.9
virtualenv-clone   0.5.3
w3lib              1.22.0
weditor            0.4.2
Werkzeug           1.0.0
wheel              0.35.1
whichcraft         0.6.1
wrapt              1.11.2
WsgiDAV            3.0.3
wsproto            0.14.1
xdis               4.2.2
xmltodict          0.12.0
zope.interface     5.1.0
zstandard          0.12.0
```

另外一个 `pipenv` 的虚拟环境中的例子：

```
➜  tensorflow pip list --format=columns
Package                        Version
------------------------------ ---------
asn1crypto                     0.24.0
astroid                        1.5.3
attrs                          17.3.0
Automat                        0.6.0
backports.functools-lru-cache 1.4
browsermob-proxy               0.8.0
certifi                        2017.11.5
cffi                           1.11.2
chardet                        3.0.4
configparser                   3.5.0
constantly                     15.1.0
cryptography                   2.1.4
cssselect                      1.0.1
enum34                         1.1.6
hyperlink                      17.3.1
idna                           2.6
incremental                    17.5.0
ipaddress                      1.0.19
isort                          4.2.15
lazy-object-proxy              1.3.1
lxml                           4.1.1
mccabe                         0.6.1
parsel                         1.2.0
pip                            9.0.1
pyasn1                         0.4.2
pyasn1-modules                 0.2.1
pycparser                      2.18
PyDispatcher                   2.0.5
pylint                         1.7.2
pyOpenSSL                      17.5.0
queuelib                       1.4.2
requests                       2.18.4
Scrapy                         1.4.0
selenium                       3.7.0
service-identity               17.0.0
setuptools                     38.4.0
singledispatch                 3.4.0.3
six                            1.10.0
speedtest-cli                  1.0.7
Twisted                        17.9.0
urllib3                        1.22
virtualenv                     15.1.0
w3lib                          1.18.0
wheel                          0.30.0
wrapt                          1.10.11
zope.interface                 4.4.3
```

pip search

当想要安装某个库，但却又不知道该库的包名时，可以用：

```
pip search xxx
```

去数据源中查询有哪些相关的包。

`pip search` 举例

kafka

```
[root@xxx-dev-01 exercise]# pip3 search kafka
kafka (1.3.5)                                     - Pure Py
  INSTALLED: 1.3.5 (latest)
heroku-kafka (2.1.2)                              - Python
pytest-kafka (0.3.1)                              - Zookeep
kafka-utils (2.3.0)                               - Kafka r
chance-kafka (0.0.6)                              - The ka
dask-kafka (0.1.0.dev0)                           - Dask-Ka
kafka-bundle (1.0)                                - Kafka s
kafka-helper (0.2)                                - Makes i
scrapy-kafka (0.1.1)                              - Kafka-b
kafka-connector (0.0.7)                           - A pytho
kafka-logger (0.3.0)                              - A simpl
eventcore-kafka (0.3.3)                           - Produce
kafka-shell (0.1.1)                               - A super
kafka-quixey (0.9.0-q3)                           - Pure Py
confluent-kafka (1.1.0)                           - Conflue
kafka-python (1.4.6)                              - Pure Py
timi-kafka (0.0.2)                                - Pure Py
kafka-scanner (0.3.4)                             - High Le
log-to-kafka (1.1.4)                              - log to
sentry-kafka (1.1)                                - A Sentr
kafka-rest (0.0.12)                               - Async c
kafka-tools (0.1.9)                               - A colle
kafka-connect-python (0.0.3)                      - Kafka C
kafka-dev-tools (0.0.1)                           - Tools f
kafka-python-with-confluent-kafka (0.2)           - This py
aio-kafka-daemon (0.5.1)                          - Asynchr
scrapy-kafka-export (0.1.1)                       - Export
kafka-rest-client (0.8.0)                         - A pytho
chatora.confluent-kafka-ext (0.3)                 - Apache
kafka-utils-netease (1.4.9)                       - Easy wa
stack-kafka-python (1.4.11)                       - Pure Py
scrapy-kafka-redis (0.0.7)                        - Kafka a
kafka-splunk-connector (1.0.6)                    - Library
confluent-kafka-helpers (0.7.3)                   - Helpers
netbox-kafka-producer (1.0.10)                    - Easily
gc-kafka-python (0.9.8)                           - Pure Py
robinhood-kafka-python (1.4.3)                    - Pure Py
kafka-connect-healthcheck (0.1.0)                 - A simpl
kafka-logging-handler (0.2.2)                     - A Pytho
netbox-kafka-consumer (1.0.6)                     - Easily
heroku-kafka-eze (0.0.2)                          - Python
logwatcher-kafka-plugin (1.0)                     - LogWatc
python-elk-kafka (0.0.1)                          - A pytho
confluent-kafka-smyte (0.9.3)                     - Smyte f
swissbib-kafka-event-hub (2.2.0)                  - Swissbi
kafka-log-handler (1.2)                           - Simple
python-kafka-logger (0.5)                         - Simple
batching-kafka-consumer (0.0.4)                   - Kafka C
```

```
python-kafka-logging (0.6)                        - Simple
reverse-kafka-logger (0.1.2)                      - grep ka
pipelinewise-tap-kafka (1.0.0)                    - Singer.
ctodd-python-lib-kafka (1.0.2)                    - Python
kafka-avro-binary-consumer (0.0.0.2)              - Kafka a
kafka-avro-producer-topkrabbensteam (1.3.1)       - Kafka (
prometheus-kafka-consumer-group-exporter (0.5.1)  - Kafka c
kafka-python-log-handler (0.0.1.dev12)            - A log h
kafka-metrics-producer-topkrabbensteam (1.0)      - Library
gevent-kafka (0.2)                                - ApacheK
kafka-cffi (0.11.4a5)                             - A CFFI
kafka-tfrx (0.14.0)                               - Add a s
dox-kafka (0.0.1)                                 -
kafka-transport (0.6.7)                           -
ansible-runner-kafka (0.1)                        -
kser-transport-kafka (0.1.0)                      - Kser li
samsa (0.3.11)                                    - Kafka (
azure-functions-kafka-binding (1.0.1)             -
ickafka (0.1.4)                                   - improve
esque (0.1.4a0)                                   - A usabl
pysubman (1.20.1.4)                               - kafka s
zpython-tools (0.1.1)                             - kafka t
kafkaloghandler (0.9.0)                           - Kafka L
kafka_influxdb (0.9.3)                            - A Kafka
dhcpkit_kafka (1.0.0)                             - Kafka e
robinhood-aiokafka (1.0.3)                        - Kafka i
aiokafka (0.5.2)                                  - Kafka i
kafkaBridgeClient (0.1.5)                         - A packa
kafka_store (0.1.4)                               - Kafka S
streamsx.kafka (1.2.4)                            - IBM Str
kq (2.0.0)                                        - Kafka J
fjord_kafka_migration (0.2.5)                     - fjord k
kafkatos3 (0.2.0)                                 - Archive
bubuku (0.10.47)                                  - AWS sup
xoskafka (3.3.1)                                  - Wrapper
robotframework-kafkalibrary (0.0.2)               - Kafka T
kser (0.8.19)                                     - Kafka s
nico (0.1)                                        - job sch
kafkaesque (1.0)                                  - an easy
kiel (0.9.3)                                      - Kafka c
testing.kafka (0.0.1)                             - automat
kser-crypto (0.1.4)                               - Crypto
yelp_kafka (5.2.1)                                - A libra
kafkacrypto (0.9.4)                               - Message
asynckafka (0.1.2)                                - Fast py
kafka_replayer (1.0.1)                            - Timesta
robotframework-confluentkafkalibrary (0.3.0)      - Conflue
kafkacli (0.0.7)                                  - Apache
klag (1.0.2)                                      - A Kafka
Kafthon (0.0.1)                                   - High le
```

```
afkak (3.0.0)                                    - Twistec
flasfka (1.1.6)
```

此处，找到我们希望安装的库是：

```
pip3 install kafka-python
```

pip freeze

用 `pip freeze` + `pip install` 备份和恢复Python环境

pip的常见用法之一包括：

在别处，用pip给某项目安装了相应的库之后，去 `freeze` = `冻结` ：

```
pip freeze > requirements.txt
```

即，把当前Python中安装的所有的库的信息，写入到一个文本文件中。

该文件文件常写成 `requirements.txt` ，表示当前Python环境**所需**的库的信息。

而换到另外一个地方，想要恢复之前项目的环境，则可以用：

```
pip install -r requirements.txt
```

即可自动安装所依赖的，各种Python的库，恢复了之前的项目的Python环境。

pip常见问题

command not found pip

现象：用pip去安装 `virtualenv` 期间出错：

```
~ pip
zsh: command not found: pip
```

原因：当前CentOS服务器中没有安装pip

解决办法：去CentOS中安装pip

操作步骤：

先用 `yum` 安装 `python-pip` ：

```
yum -y install python-pip
```

再去升级到最新版：

```
pip install --upgrade pip
```

即可

Errno 13 Permission denied

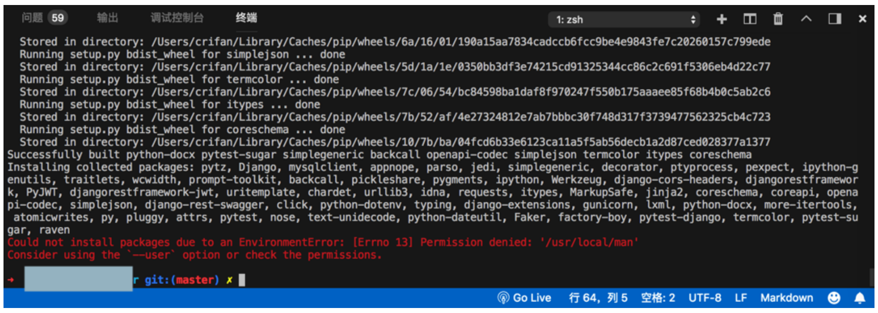
Could not install packages due to an EnvironmentError: [Errno 13] Permission denied: /usr/local/man

现象：

```
pip3 install -r requirements.txt
```

报错：

```
  xxx git:(master) pip3 install -r requirements.txt
Collecting Django==2.0.6 (from -r requirements.txt (line 1
  Downloading https://files.pythonhosted.org/packages/56/0
    100% |████████████████████████████████| 7.1MB 12kB/s
Collecting mysqlclient==1.3.12 (from -r requirements.txt (
Requirement already satisfied: pymongo==3.6.1 in /usr/loca
Collecting ipython==6.4.0 (from -r requirements.txt (line 
  。。。
  Running setup.py bdist_wheel for itypes … done
  Stored in directory: /Users/crifan/Library/Caches/pip/whe
  Running setup.py bdist_wheel for coreschema … done
  Stored in directory: /Users/crifan/Library/Caches/pip/whe
Successfully built python-docx pytest-sugar simplegeneric b
Installing collected packages: pytz, Django, mysqlclient, a
Could not install packages due to an EnvironmentError: [Er
Consider using the `--user` option or check the permission
```



```
  Stored in directory: /Users/crifan/Library/Caches/pip/wheels/6a/16/01/190a15aa7834cadccb6fcc9be4e9843fe7c20260157c799ede
  Running setup.py bdist_wheel for simplejson ... done
  Stored in directory: /Users/crifan/Library/Caches/pip/wheels/5d/1a/1e/0350bb3df3e74215cd91325344cc86c2c691f5306eb4d22c77
  Running setup.py bdist_wheel for termcolor ... done
  Stored in directory: /Users/crifan/Library/Caches/pip/wheels/7c/06/54/bc84598ba1daf8f970247f550b175aaaee85f68b4b0c5ab2c6
  Running setup.py bdist_wheel for itypes ... done
  Stored in directory: /Users/crifan/Library/Caches/pip/wheels/7b/52/af/4e27324812e7ab7bbbc30f748d317f3739477562325cb4c723
  Running setup.py bdist_wheel for coreschema ... done
  Stored in directory: /Users/crifan/Library/Caches/pip/wheels/10/7b/ba/04fcd6b33e6123ca11a5f5ab56decb1a2d87ced028377a1377
Successfully built python-docx pytest-sugar simplegeneric backcall openapi-codec simplejson termcolor itypes coreschema
Installing collected packages: pytz, Django, mysqlclient, appnope, parso, jedi, simplegeneric, decorator, ptyprocess, pexpect, ipython-g
enutils, traitlets, wcwidth, prompt-toolkit, backcall, pickleshare, pygments, ipython, Werkzeug, django-cors-headers, djangorestframewor
k, PyJWT, djangorestframework-jwt, uritemplate, chardet, urllib3, idna, requests, itypes, MarkupSafe, jinja2, coreschema, coreapi, opena
pi-codec, simplejson, django-rest-swagger, click, python-dotenv, typing, django-extensions, gunicorn, lxml, python-docx, more-itertools,
 atomicwrites, py, pluggy, attrs, pytest, nose, text-unidecode, python-dateutil, Faker, factory-boy, pytest-django, termcolor, pytest-su
gar, raven
Could not install packages due to an EnvironmentError: [Errno 13] Permission denied: '/usr/local/man'
Consider using the `--user` option or check the permissions.

→ [redacted]r git:(master) ✗ 
```

原因：没有权限

解决办法：给（当前用户）加上权限，或者（通过 `--user` ）改用当前用户权限

操作步骤：

- 给（当前用户）加上权限

  - 给/usr/local加上权限，或者把拥有者ower换成自己

    ```
    sudo chown -R crifan /usr/local
    ```

    - 或：

      ```
      sudo chown -R $USER /usr/local
      ```

- 通过 `--user` 改用当前用户权限

  ```
  pip3 install --user -r requirements.txt
  ```

  - 其他类似的问题的解决办法，也是加 `--user`

    ```
    python -m pip install -U pylint --user
    ```

Retrying after connection broken by NewConnectionError

现象：

```
pip install boto3
```

报错：

```
(RunningFast) ➜  stable pip install boto3
Collecting boto3
  Retrying (Retry(total=4, connect=None, read=None, redirec
...
  Retrying (Retry(total=0, connect=None, read=None, redirec
  Could not find a version that satisfies the requirement b
No matching distribution found for boto3

Failed to establish a new connection Errno -2  Name or serv

Retrying (Retry(total=4, connect=None, read=None, redirect=
```

原因：当时网络不稳定，连不上pip的源，导致报网络相关的错误

解决办法：

- 换个时间（网络就正常了）
  - 此处当时就是这么做的：换了个时间，第二天早上，网络正常了，再去 `pip install boto3` ，就可以了
- 给pip加上代理
  - 确保可以正常访问pip的源
    - 如何操作，详见前面章节：给pip加代理

使用pip时注意是Pyhton2还是Python3

在使用pip时，要注意 `pip` 的版本，是 `Python 2` 还是 `Python 3`

可以通过：

```
pip --version
```

去确认当前pip到底是什么版本。

举例：

- Python 2

  ```
  ➜ pip --version
  pip 10.0.1 from /Users/crifan/Library/Python/2.7/lib/
  ```

- Python 3

  ```
  ➜ pip --version
  pip 10.0.1 from /usr/local/lib/python3.6/site-package
  ```

在使用时，不要搞混了。

如果搞混了，容易出现问题：

用pip安装了库（到某个Python环境=python版本）中

但是去代码中（用到了另外一个环境=Python版本）去导入对应的库，却找不到该库，import会报错。

不用pip安装python库

有些python的包（第三方库），也可以不用pip，而用系统相关包管理工具去管理：

举例：

- 安装supervisord
  - CentOS
    - 用yum

      ```
      yum install supervisord
      ```

  - Ubuntu
    - 用 `apt-get`

      ```
      apt-get install supervisord
      ```

pip相关

pipenv

`pipenv` 是个Python虚拟环境管理工具。

`pipenv` 中主要是利用 `pip` 去安装包的。

`pipenv` 安装库有2种方式：

- 先进入 `pipenv` 环境，再用pip去安装库

  ```
  pipenv shell
  pip install xxx
  ```

- 直接用 `pipenv` 去安装

  ```
  pipenv install xxx
  ```

通过pip查看site-packages位置

有时候，希望知道当前Python的 `site-packages` 的位置。

此时可以借用 `pip show xxx` 去实现。

举例：

```
# pip3 show Django
Name: Django
Version: 2.0.6
Summary: A high-level Python Web framework that encourages
Home-page: https://www.djangoproject.com/
Author: Django Software Foundation
Author-email: foundation@djangoproject.com
License: BSD
Location: /usr/lib/python3.4/site-packages
Requires: pytz
Required-by: django-redis
```

从其中的：

`/usr/lib/python3.4/site-packages`

可以得知：

- 当前Python是： `/usr/lib/python3.4`
- 对应 `site-packages` 的位置是： `/usr/lib/python3.4/site-packages`

附录

下面列出相关参考资料。

参考资料

- 【已解决】macOS中全局设置pip源是ustc但是当前项目却从tsinghua的源去安装
- 【已解决】Mac中给pip更换源以加速下载
- 【已解决】macOS中pip安装powerline-status报错：WARNING Retrying Retry total 4 after connection broken by ConnectTimeoutError
- 【已解决】Mac中安装Python的1.7.0rc1的tensorflow
- 【已解决】Mac中启动PySpider
- 【已解决】自动抓包工具库facebook-wda优化：合并之前已优化的wda代码
- 【已解决】Mac中安装guesslang报错：ERROR No matching distribution found for tensorflow 1.7.0rc1 from guesslang
- 【已解决】windows中PyInstaller打包exe运行报错：Failed to execute script pyi_rth_pkgres ModuleNotFoundError pkg_resources.py2_warn
- 【已解决】树莓派中卸载重新安装Python3.7的pipenv
- 【已解决】树莓派升级Python3.7.3后pip出错：ImportError cannot import name main from pip
- 【已解决】PyCharm调试python代码import mime出错：UnicodeDecodeError ascii codec can't decode byte 0xc3 in position 7213 ordinal not in range 128
- 【已解决】CentOS中用pip3查看不到库kafka-python的信息
-
- 【已解决】CentOS中用python2的pip去安装supervisor后找不到/etc/supervisor中的默认配置文件supervisord.conf
- 【已解决】python解析excel文件并读取其中的sheet和row和column的值
- 【已解决】Mac本地用supervisor去管理gunicorn的Python3的Flask
- 【记录】用Python的Scrapy去爬取cbeebies.com
- ［已解决］CentOS中zsh: command not found: pip
- ［已解决］通过pip升级SQLAlchemy为最新的1.1版本
- 【已解决】python中从文件名后缀推断出MIME类型
- 【已解决】VSCode提示安装lint出错：Could not install packages due to an EnvironmentError Errno 13 Permission denied
- 【已解决】pipenv安装后警告：The script virtualenv is installed in which is not on PATH
- 【已解决】mac中卸载virtualenv出错：Not uninstalling virtualenv at outside environment
- 【已解决】Mac中把Python3的pip重新换成Python2的pip
- 【已解决】CentOS中如何查看Python的site-packages位置

- 【已解决】如何使用pip去本地安装Python的包文件
- 【已解决】pip3 install出错：Could not install packages due to an EnvironmentError Errno 13 Permission denied /usr/local/man
- 【已解决】Mac中给pip3添加代理以提升下载python包的速度 – 在路上
- 【已解决】pipenv虚拟环境中用pip安装pyspider出错：**main**.ConfigurationError: Curl is configured to use SSL, but we have not been able to determine which SSL backend it is using
- 【已解决】pipenv install PySpider卡死在：Locking [packages] dependencies
- 【已解决】Mac中升级Python 3的pipenv的版本
- 【已解决】Ant Design的Reactjs页面中点击下载word文件不弹框而直接下载
- 【已解决】Chrom中js去POST本地Django的API出错：The value of the Access-Control-Allow-Origin header in the response must not be the wildcard * when the request's credentials mode is include
- 【已解决】CentOS中用python2的pip去安装supervisor后找不到/etc/supervisor中的默认配置文件supervisord.conf
- 【已解决】CentOS中用pip或easy_install安装supervisor
- 【已解决】Python3中选择合适的虚拟环境工具
- 【记录】用VSCode开发和调试Python
- 【记录】Mac中安装和运行pyspider
- 【记录】Amazon AWS中尝试使用S3
- 【已解决】pip install失败：Failed to establish a new connection Errno -2 Name or service not known – 在路上
- 【已解决】Mac中安装selenium出错：OSError Errno 13 Permission denied /usr/local/selenium
- 【貌似无需解决】Mac中active后virtualenv没有生效
- 【已解决】pyspider运行出错：ImportError pycurl libcurl link-time ssl backend (openssl) is different from compile-time ssl backend (none/other)
- 【已解决】升级CentOS中的supervisor到最新版本
- 【已解决】用Python去连接本地mongoDB去用GridFS保存文件
- 【记录】安装Tensor Flow
-
- pip vs easy_install — Python Packaging User Guide
- MacOS 下利用 pyenv 管理Python 版本和虚拟环境 - 掘金
- 关于python Django与Flask学习的一些疑惑？ - 知乎
- pypi
- Which Installer is Better? Pip or Easy Install | Liquid Web
- Project Summaries — Python Packaging User Guide
- User Guide - pip documentation v21.2.dev0
- Settings — Gunicorn 19.9.0 documentation
-