PHYS 4332 – Computational Physics

Python Tutorial

2.3 – Installing packages using pip

The Package Installer for Python (`pip`) provides a simple way of installing third-party packages from the Python Package Index (and other indexes). These are repositories of Python software developed and shared by the Python community.

- To get the version of pip you currently have installed, from a terminal/command prompt type:

```
>> pip --version

pip 22.3.1 from C:\Python310\lib\site-packages\pip (python 3.10)
```

- To upgrade pip to the latest version, type:

```
>> python -m pip install --upgrade pip

Requirement already satisfied: pip in c:\python310\lib\site-packages (22.3.1)
```

- To get a list of all available commands and options for pip, use `help`:

```
>> pip help

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  download                    Download packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  check                       Verify installed packages have compatible dependencies.
  config                      Manage local and global configuration.
  search                      Search PyPI for packages.
  cache                       Inspect and manage pip's wheel cache.
  index                       Inspect information available from package indexes.
  wheel                       Build wheels from your requirements.
  hash                        Compute hashes of package archives.
  completion                  A helper command used for command completion.
  debug                       Show information useful for debugging.
  help                        Show help for commands.
```

- You can also get detailed help for specific commands (e.g. `install`):

```
>> pip help install

Usage:
  pip install [options] <requirement specifier> [package-index-options] ...
  pip install [options] -r <requirements file> [package-index-options] ...
  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", which provide
  an easy way to specify a whole environment to be installed.
...
```

- To list the packages currently installed, use `list`:

```
>> pip list
```

- This provides both the package and version installed. To see if any of these are outdated, use the `--outdated` option:

```
>> pip list --outdated
```

- To install a new package, use `install`. For example, to install the `numpy` package:

```
>> pip install numpy

Collecting numpy
Downloading numpy-1.24.1-cp310-cp310-win_amd64.whl (14.8 MB)
---------------------------------------- 14.8/14.8 MB 14.9 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-1.24.1
```

- To upgrade a package to the latest version, type:

```
>> pip install -U numpy
```

- To uninstall a package, use `uninstall` and type `y` to confirm:

```
>> pip uninstall numpy
```

- To get a list of installed packages in a requirements format, use `freeze`:

```
>> pip freeze
```

- To export this list to a file called `Requirements.txt`:

```
>> pip freeze > Requirements.txt
```

- To automatically install all the packages listed in `Requirements.txt` (with their corresponding versions):

```
>> pip install -r Requirements.txt
```