pip Python package installer

CONCEPTS

> Use python -m pip so pip always belongs to the interpreter you mean.

Index PyPI or mirror
↓ resolve
Wheel / sdist cached download
↓ build + unpack
site-packages active environment
↓ freeze
requirements.txt pinned list

> Install into a virtual environment; on system Pythons marked EXTERNALLY-MANAGED pip refuses.

> Set PIP_REQUIRE_VIRTUALENV=1 (or --require-virtualenv) to block accidental global installs.

INSTALL

pip install *<package>*
Install the latest version

pip install *<package>*==*<version>*
Install an exact version

pip install "*<package>*≥1.2,<2"
Install within a range

pip install -r *<file>*
Install from a requirements file

pip install -U *<package>*
Upgrade to the newest version

pip install -e *<path>*
Editable install of a local project

pip install "*<package>*[*<extra>*]"
Install with optional extras

pip install *<path or url>*
Install a wheel, archive or directory

pip install git+*<url>*@*<ref>*
Install from a Git repository

pip install --no-deps *<package>*
Skip dependencies

pip install --dry-run *<package>*
Show what would be installed

pip install --pre *<package>*
Allow pre-release versions

INSPECT

pip list
List installed packages

pip list --outdated
Show packages with newer versions

pip show *<package>*
Show version, location, dependents

pip show -f *<package>*
Also list installed files

pip freeze
Print installed packages as pins

pip list --not-required
Only top-level packages

pip list --format json
Machine-readable list

pip check
Verify dependencies are consistent

pip index versions *<package>*
List versions on the index

pip debug
Show tags and platform info

REMOVE

pip uninstall *<package>*
Remove a package

pip uninstall -y *<package>*
Remove without confirmation

pip uninstall -r *<file>*
Remove everything in a file

REQUIREMENTS

pip freeze > requirements.txt
Snapshot the environment

pip install -r requirements.txt
Restore a snapshot

pip install -c constraints.txt -r *<file>*
Cap versions without adding packages

pip install --require-hashes -r *<file>*
Verify each hash

> In a file: -r other.txt includes another, -e . adds the project, -c sets constraints, # comments, ; marker after a pin.

INDEXES AND DOWNLOAD

pip install -i *<url>* *<package>*
Use another index

pip install --extra-index-url *<url>* *<package>*
Add an index to PyPI

pip install --no-index -f *<dir>* *<package>*
Install offline from a directory

pip download -d *<dir>* **-r** *<file>*
Fetch packages without installing

pip wheel -w *<dir>* *<package>*
Build wheels into a directory

CONFIG AND CACHE

pip config list
Show active configuration

pip config set global.index-url *<url>*
Set a default index

pip cache dir
Print the cache location

pip cache purge
Empty the cache

pip --python *<interpreter>* *<command>*
Manage another interpreter's packages

pip --version
Show pip and Python versions

python -m pip install -U pip
Upgrade pip itself

RECIPES

New project environment

```
python3 -m venv .venv
source .venv/bin/activate
python -m pip install -U pip
pip install -r requirements.txt
```

Review and update outdated packages

```
pip list --outdated
pip install -U <package>
pip check
pip freeze > requirements.txt
```

Develop a local package

```
pip install -e ".[dev]"
pip show <package>
pip uninstall <package>
```