



● 2.2



Search

● Introduction



# Introduction

# System requirements

# Installation

# Enable tab completion for Bash, Fish, or Zsh

# Bash

# Fish

# Zsh

Poetry is a tool for **dependency management** and **packaging** in Python. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you. Poetry offers a lockfile to ensure repeatable installs, and can build your project for distribution.

## System requirements

Poetry requires **Python 3.9+**. It is multi-platform and the goal is to make it work equally well on Linux, macOS and Windows.

# Installation


If you are viewing documentation for the development branch, you may wish to install a preview or development version of Poetry. See the **advanced** installation instructions to use a preview or alternate version of Poetry.

With `pipx`

With the official installer

Manually (advanced)

CI recommendations

`pipx`  is used to install Python CLI applications globally while still isolating them in virtual environments. `pipx` will manage upgrades and uninstalls when used to install Poetry.

## 1 Install `pipx`

If `pipx` is not already installed, you can follow any of the options in the [official `pipx` installation instructions](#). Any non-ancient version of `pipx` will do.

## 2 Install Poetry

```
pipx install poetry
```



## 3 Install Poetry (advanced)

You can skip this step, if you simply want the latest version and already installed Poetry as described in the previous step. This step details advanced usages of this installation method. For example, installing Poetry from source, having multiple versions installed at the same time etc.

`pipx` can install different versions of Poetry, using the same syntax as `pip`:

```
pipx install poetry==1.8.4
```

`pipx` can also install versions of Poetry in parallel, which allows for easy testing of alternate or prerelease versions. Each version is given a unique, user-specified suffix, which will be used to create a unique binary name:

```
pipx install --suffix=@1.8.4 poetry==1.8.4  
poetry@1.8.4 --version
```

```
pipx install --suffix=@preview --pip-args=--pre poetry  
poetry@preview --version
```

Finally, `pipx` can install any valid `pip` requirement spec, which allows for installations of the development version from `git`, or even for local testing of pull requests:

```
pipx install --suffix @main git+https://github.com/python-poetry/poetry
pipx install --suffix @pr1234 git+https://github.com/python-poetry/poetry
```

4

#### Update Poetry

```
pipx upgrade poetry
```

5

#### Uninstall Poetry

```
pipx uninstall poetry
```

Poetry should always be installed in a dedicated virtual environment to isolate it from the rest of your system. Each of the above described installation methods ensures that. It should in no case be installed in the environment of the project that is to be managed by Poetry. This ensures that Poetry's own dependencies will not be accidentally upgraded or uninstalled. In addition, the isolated virtual environment in which poetry is installed should not be activated for running poetry commands.

## Enable tab completion for Bash, Fish, or Zsh

`poetry` supports generating completion scripts for Bash, Fish, and Zsh.

You may need to restart your shell in order for these changes to take effect.

See `poetry help completions` for full details, but the gist is as simple as using one of the following:

## Bash

### Auto-loaded (recommended)

```
poetry completions bash >> ~/.bash_completion
```



### Lazy-loaded

```
poetry completions bash > ${XDG_DATA_HOME:-~/.local/share}/bash-completio
```



## Fish

```
poetry completions fish > ~/.config/fish/completions/poetry.fish
```



# Zsh

```
poetry completions zsh > ~/.zfunc/_poetry
```



You must then add the following lines in your `~/.zshrc`, if they do not already exist:

```
fpath+=~/.zfunc  
autoload -Uz compinit && compinit
```



## Oh My Zsh

```
mkdir $ZSH_CUSTOM/plugins/poetry  
poetry completions zsh > $ZSH_CUSTOM/plugins/poetry/_poetry
```

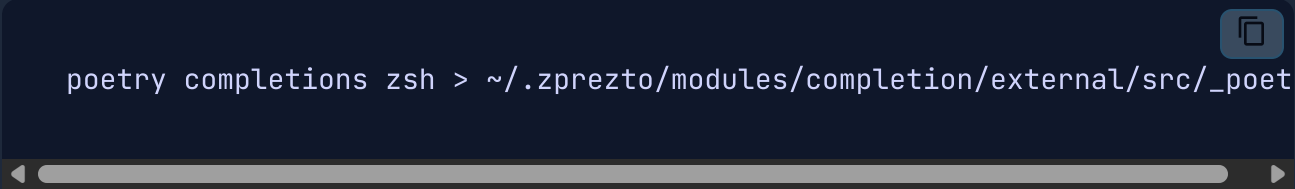


You must then add `poetry` to your plugins array in `~/.zshrc`:

```
plugins(  
  poetry  
  ...  
)
```



## Prezto



```
poetry completions zsh > ~/.zprezto/modules/completion/external/src/_poet
```

If completions still don't work, try removing `~/.cache/prezto/zcompcache` and starting a new shell.



Python packaging and dependency management made easy.



## DOCUMENTATION

[Introduction](#)

[Basic usage](#)

[Managing dependencies](#)

[Libraries](#)

[Commands](#)

[Configuration](#)

[Repositories](#)

[Managing environments](#)

[Dependency specification](#)

[Plugins](#)

[The pyproject.toml file](#)

[Contributing to Poetry](#)

[Community](#)

[FAQ](#)

[pre-commit hooks](#)

[Building extension modules](#)

 [GITHUB](#)

[Project](#)

[Issues](#)

[Discussions](#)

## OTHER PROJECTS

[poetry-core](#)


[install.python-poetry.org](#)

[Bundle plugin](#)

[Export plugin](#)

---

Copyright © 2018-2025. All Rights Reserved.

Powered by  **Vercel**