**Poetry** **Dokumentation** ist auch [unter dem Link](https://python-poetry.org/docs/cli/) abrufbar. Dependencies management + [project environment isolation](https://python-poetry.org/docs/managing-environments/) are the key functions of Poetry.

AVAILABLE COMMANDS

**about** Shows information about Poetry.

**add** Adds a new dependency to pyproject.toml. ***--source***-flag adds an [alternative repo](https://python-poetry.org/docs/repositories/) than PyPI

**build** [Builds](https://python-poetry.org/docs/libraries/#packaging) a package from a library, as a tarball and a wheel by default. The package can be [published](#Publish)

**cache** Interact with Poetry's cache

**check** Checks the validity of the pyproject.toml file.

**config** Manages Poetry [configuration](https://python-poetry.org/docs/configuration/) [settings](https://python-poetry.org/docs/configuration/#available-settings). ***--local***-flag configures Poetry for the current project

**debug** Debug various elements of Poetry.

**env** Interact with Poetry's project environments.

***poetry env info --path***

***{path\_to\_venv}/bin/activate.ps1*** *//* [*alternative*](https://python-poetry.org/docs/basic-usage/#activating-the-virtual-environment) *to* ***“poetry shell****” to activate a shell in a specific venv.*

**export** Exports the lock file to alternative formats.

**help** Display the manual of a command

**init** Creates a basic [pyproject.toml file](https://python-poetry.org/docs/pyproject/) in the current directory (if the project already exists)

**install** Installs the project dependencies, [with or without](https://python-poetry.org/docs/basic-usage/#installing-dependencies) checking the .lock file restrictions

*poetry install --sync //* [*makes sure*](poetry%20install%20--sync) *that .lock-file contains only dependencies from pyproject.toml and removes anything that is not necessary*

**lock** Locks the project dependencies.

**new** Creates a new Python project at <path>.

**publish** [Publishes](https://python-poetry.org/docs/libraries/" \l "publishing-to-pypi) my own created library package to a remote repository (f.e. [***PyPI***](https://realpython.com/what-is-pip/#installing-packages-with-pip) or ***private repo***).

**remove** Removes a package from the project dependencies.

**run** Runs a command in the appropriate environment. *// poetry run python main.py*

**search** Searches for packages on remote repositories.

**self** Interact with Poetry directly.

**shell** [Spawns a shell](https://python-poetry.org/docs/basic-usage/#activating-the-virtual-environment) within the current virtual environment. *// Poetry is also a python app! See link*

**show** Shows information about packages.

**update** [Update](https://python-poetry.org/docs/basic-usage/#updating-dependencies-to-their-latest-versions) the dependencies as according to the pyproject.toml file + upd .lock-file.

**version** Shows the version of the project or bumps it when a valid bump rule is provided.

ARGUMENTS

**<command>** The command to execute

**<arg>** The arguments of the command

GLOBAL OPTIONS

**-h (--help)** Display this help message

**-q (--quiet)** Do not output any message

**-v (--verbose)** Increase the verbosity of messages: "-v" for normal output, "-vv" for more verbose output and "-vvv" for debug

**-V (--version)**  Display this application version

**--ansi** Force ANSI output

**--no-ansi** Disable ANSI output

**-n (--no-interaction)** Do not ask any interactive question