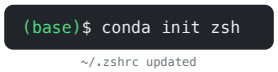
conda / mamba

The everyday conda and mamba commands, for creating environments, managing packages, and sharing your setup.

blog.thecoatlessprofessor.com
updated 05/21/2026

Install & Setup

Get a conda/mamba distribution and initialize your shell.

PURPOSE	COMMAND	RESULT
Install Miniforge.	`brew install miniforge`	
Initialize your shell once.	`conda init zsh`	
Check version & install info.	`conda info`	
Keep conda up to date.	`conda update -n base conda`	
Stop auto-activating base.	`conda config --set auto_activate false`	

Miniforge ships conda + mamba and defaults to the conda-forge channel

Environments

Create, list, activate, and remove isolated prefixes.

PURPOSE	COMMAND	RESULT
Create a named env.	`conda create -n myenv python=3.12`	
List all environments.	`conda env list`	
Activate an environment.	`conda activate myenv`	
Leave the environment.	`conda deactivate`	
Rename an environment.	`conda rename -n myenv proj`	
Delete an environment.	`conda remove -n myenv --all`	

an environment is an isolated prefix directory with its own Python

Installing & Removing Packages

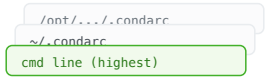
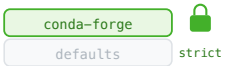
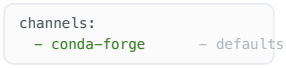
Add, update, and remove packages in the active environment.

PURPOSE	COMMAND	RESULT
Install into the active env.	`conda install numpy pandas`	
Pin an exact version.	`conda install "numpy=1.26"`	
Install from a channel.	`conda install -c conda-forge polars`	
Faster solve via mamba.	`mamba install scipy`	
Update one or all.	`conda update --all`	
Remove a package.	`conda remove pandas`	

conda installs prebuilt binaries, including non-Python dependencies pip cannot

Channels & Config

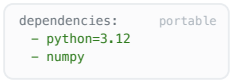
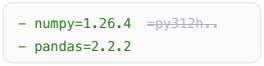
Where packages come from, and how the solver picks.

PURPOSE	COMMAND	RESULT
Show config and its sources.	`conda config --show-sources`	
Make conda-forge top priority.	`conda config --add channels conda-forge`	
Prefer the top channel strictly.	`conda config --set channel_priority strict`	
Use a channel just once.	`conda install -c bioconda samtools`	
Read a config key.	`conda config --get channels`	

conda-forge first + strict priority is the modern default that avoids breakage

Reproduce & Share

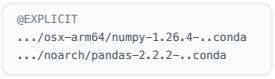
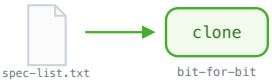
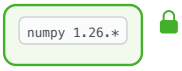
Capture an environment and rebuild it elsewhere.

PURPOSE	COMMAND	RESULT
Export the full active env.	`conda env export > environment.yml`	
Export only what you asked for.	`conda env export --from-history \` `  > environment.yml`	
Export cross-OS (no builds).	`conda env export --no-builds \` `  > environment.yml`	
Create an env from a file.	`conda env create -f environment.yml`	
Sync an env to the file.	`conda env update -f environment.yml \` `  --prune`	

`--from-history` yields a file that solves freshly on any platform

Pinning & Lockfiles

Exactly reproducible environments, byte for byte.

PURPOSE	COMMAND	RESULT
List installed packages.	`conda list`	
Filter the list.	`conda list numpy`	
Write an explicit spec list.	`conda list --explicit > spec-list.txt`	
Rebuild an identical env.	`conda create -n clone \` `  --file spec-list.txt`	
Pin a package permanently.	`edit <env>/conda-meta/pinned`	

`--explicit` is a same-OS lockfile that rebuilds without re-solving

Inspect & Search

Find packages and understand the dependency graph.

PURPOSE	COMMAND	RESULT

Search channels for a package.	`conda search numpy`	`numpy 2.3.2 conda-forge`
		`numpy 2.2.0 conda-forge`
		`numpy 1.26.4 conda-forge`

Search a specific channel.	`conda search -c conda-forge "polars>=1.0"`	`polars 1.20.0` `polars 1.12.0`
What a package depends on.	`mamba repoquery depends numpy`	numpy → python, libblas
What depends on a package.	`mamba repoquery whoneeds numpy`	pandas, scipy → numpy
Fast package search.	`mamba search scipy`	`scipy 1.14.1` `scipy 1.13.0`

repoquery depends/whoneeds walk the dependency graph both ways

Run, Clean & mamba

Run inside an env without activating, free disk space, know your tools.

PURPOSE	COMMAND	RESULT
Run a command in an env.	`conda run -n myenv python app.py`	myenv → `$ app.py`
Clone an environment.	`conda create -n backup --clone myenv`	myenv → backup
Reclaim disk space.	`conda clean --all`	freed 1.2 GB
mamba is a drop-in for conda.	`mamba create -n myenv python=3.12`	myenv (same envs & config)
Check what is installed.	`mamba --version`	mamba 2.1.1, conda 25.5.0

every mamba subcommand mirrors conda and shares its environments