pre-commit Git hook manager

CONCEPTS

> Hooks only see staged files on commit. A fresh config does nothing until you run --all-files once.

git commit staged files
↓ hook script
pre-commit reads config
↓ filters files
Hooks cached envs
↓ exit code
Commit or abort

> A hook that edits files counts as a failure: review the changes, git add them and commit again.

INSTALL & SETUP

pip install pre-commit
Install with pip

pre-commit install
Install the git pre-commit hook

RUNNING

pre-commit run
Run hooks on staged files

pre-commit run --all-files
Run on every tracked file

pre-commit run *<hook-id>*
Run a single hook

pre-commit run *<hook-id>* **--all-files**
One hook on all files

pre-commit run --from-ref *<ref>* **--to-ref** *<ref>*
Run on files changed between refs

SKIPPING

SKIP=*<id>*,*<id>* **git commit**
Skip hooks by id

git commit --no-verify
Skip all git hooks

UPDATING & CACHE

pre-commit autoupdate
Bump every rev to the latest tag

CONFIG FILE

Top-level keys

Key	Meaning
`repos`	list of hook sources (required)
`default_stages`	stages hooks use by default
`default_install_hook_types`	what install sets up
`default_language_version`	e.g. python: python3.12
`files / exclude`	global regex filters
`fail_fast`	stop after first failing hook
`minimum_pre_commit_version`	refuse older versions

Hook keys

Key	Meaning
`id`	which hook (required)
`args`	extra arguments before filenames
`files / exclude`	regex on file paths
`types / types_or`	filter by file type
`stages`	git stages it runs on
`additional_dependencies`	extra packages for the env
`always_run`	run even without matching files
`pass_filenames`	local hooks: pass files or not
`language_version`	override the runtime version

LOCAL & META HOOKS

repo: local
Hooks defined in your own repo

CI

pre-commit run --all-files --show-diff-on-failure
Typical CI step