Nox Python session runner

CONCEPTS

Sessions are plain Python functions in noxfile.py; each gets its own fresh virtualenv under .nox/.

noxfile.py @nox.session
↓ `nox -s`
Virtualenv .nox/<name>
↓ `session.install`
Dependencies pip / uv
↓ `session.run`
Commands tests, lint, docs

Without -s, every session in nox.options.sessions (or all, if unset) runs in file order.

RUNNING

`nox`
Run all default sessions

`nox -s <session> [<session>...]`
Run named sessions

`nox -l`
List sessions and their Pythons

`nox -- <args>`
Pass args to sessions as posargs

`nox -p <version>`
Only sessions using this Python

`nox -f <file>`
Use another noxfile

`nox -t <tag> [<tag>...]`
Run sessions with a tag

`nox -k "<expr>"`
Select by keyword expression

`nox -s "tests(dj='5')"`
Run one parametrized variant

`nox -x`
Stop after the first failure

`nox -j <N|auto>`
Run sessions in parallel

ENVIRONMENTS & EXECUTION

`nox -r`
Reuse existing virtualenvs

`nox -R`
Reuse venvs and skip installs

`nox --install-only`
Install deps, skip session.run

`nox -db uv`
Default venv backend (uv, virtualenv...)

NOXFILE

`import nox`
Load the library

`@nox.session`
Define a session function

`@nox.session(python=["3.11", "3.12"])`
Run once per Python

`session.install("<pkg>")`
Install into the session venv

`session.install("-e", ".")`
Install the project editable

`session.run("<cmd>", "<arg>")`
Run a command in the venv

`session.posargs`
Args given after --

`session.run("pytest", *session.posargs)`
Forward posargs to a tool

`@nox.session(tags=["<tag>"])`
Group sessions for -t

`@nox.parametrize("<arg>", [<v1>, <v2>])`
Run once per value

`nox.options.sessions = [...]`
Default sessions when none given

`nox.options.default_venv_backend = "uv"`
Set backend for the whole file