Black Python formatter

START HERE

`black .`
Format everything here

`black -l <n> .`
Set line length

`black <file.py>`
Format one file

`black -`
Format stdin to stdout

`black --check .`
Verify, change nothing

`black --version`
Which version?

`black --diff .`
Preview the changes

`black -v .`
Show skipped files

Source files .py .pyi .ipynb → parse → **Black** config + CLI flags → reformat → **Equivalence check** AST must match → write → **Formatted files** exit 0 / 1 / 123

CONCEPTS

Black is opinionated: few options, same output for everyone. Pin the version in your project, since style can change between releases.

Precedence: built-in defaults < pyproject.toml < command-line flags.

RUN

`black .`
Format all files under the current dir

`black <file.py> <dir>`
Format given files and folders

`black --check .`
Exit 1 if anything would change

`black --diff .`
Print the changes, write nothing

`black --check --diff .`
Show what a CI failure is about

`black --diff --color .`
Colored diff

`black -q .`
Only print errors

`black -v .`
Report ignored and unchanged files

`black --version`
Print the installed version

`black -W <n> .`
Number of parallel workers

STYLE OPTIONS

`black -l <n> .`
Maximum line length (default 88)

`black -t <py311> .`
Target Python version (repeatable)
Inferred from requires-python in pyproject.toml, else detected per file.

STYLE OPTIONS (cont.)

`black -S .`
Keep string quotes as they are

`black -C .`
Ignore magic trailing commas

`black --pyi <file>`
Treat input as a typing stub

`black --ipynb <file>`
Treat input as a Jupyter notebook

`black -x <file>`
Skip the first line of the source

`black --preview .`
Try upcoming style changes
No stability guarantee between releases.

`black --unstable .`
Preview plus features with known issues

FILES AND STDIN

`black -`
Read stdin, write formatted code to stdout

`black -c "<code>"`
Format a code string

`black --stdin-filename <path> -`
Name stdin so excludes apply

`black --extend-exclude '<regex>' .`
Skip more paths, keep defaults
Regex, not glob. A leading / anchors at the project root.

`black --exclude '<regex>' .`
Replace the default exclusions

`black --force-exclude '<regex>' .`
Exclude even explicitly passed files

`black --include '<regex>' .`
Which files to pick up (default py, pyi, ipynb)

FILES AND STDIN (cont.)

`black --line-ranges <1-20> <file>`
Format only these lines (repeatable)
Single file only; also turns off the safe check.

Black already skips .git, .venv, venv, build, dist, the tool caches and anything in .gitignore.

CONFIG

`[tool.black]`
Settings table in pyproject.toml

`black --config <file> .`
Use exactly this config file

`black --required-version <26> .`
Fail if Black's version differs

`black --no-cache .`
Ignore the per-user cache

`black --cache-dir <dir> .`
Cache location

`black --fast .`
Skip the AST equivalence check

`BLACK_CACHE_DIR / BLACK_NUM_WORKERS`
Environment overrides

Common pyproject keys

Key	Same as
`line-length`	-l
`target-version`	-t (a list)
`skip-string-normalization`	-S
`extend-exclude`	--extend-exclude
`required-version`	--required-version

Black searches upward from the paths you pass and stops at a pyproject.toml with [tool.black], or at .git/.hg. Only one config file is used.

IGNORING CODE

`# fmt: skip`
Leave this line alone

`# fmt: off`
Start an unformatted region

`# fmt: on`
End the region

fmt: off and fmt: on must sit at the same indentation level in the same block.

`# fmt: skip # noqa`
Combine with other pragmas

EXIT CODES

Code	Meaning
0	Done, or nothing to change with --check
1	--check: files would be reformatted
123	Internal error

RECIPES

pyproject.toml

```
[tool.black]
line-length = 100
target-version = ['py311']
extend-exclude = '''
/(\.tox|migrations)/
'''
```

CI gate

```
black --check --diff .
```

Format only changed files

```
git diff --name-only --diff-filter=d -- '*.py' | xargs black
```

Pre-commit hook

```
# .pre-commit-config.yaml
repos:
  - repo: https://github.com/psf/black-pre-commit-mirror
    rev: 26.10.1
    hooks:
      - id: black
```