SwiftLint Swift style linter

START HERE

swiftlint
Lint the current directory

`swiftlint` **rules** *<rule>*
Explain one rule

`swiftlint` **--fix**
Auto-correct violations

`swiftlint` **--config** *<file>*
Use a specific config

`swiftlint` **lint --strict**
Treat warnings as errors

// swiftlint:disable:next *<rule>*
Silence the next line

`swiftlint` **rules**
List all rules

`swiftlint` **--reporter json**
Machine-readable output

.swiftlint.yml rules + paths → `select` **Swift files** included minus excluded → `lint` **Violations** warning or error → `report` **Reporter** xcode, json, ...

CONCEPTS

SwiftLint reads .swiftlint.yml from the directory you run it in; rules run on every .swift file below it.

Analyzer rules need a full compiler log and run only under swiftlint analyze; ordinary lint rules never run there.

--fix rewrites files on disk and skips linting. Commit or stash first, then run swiftlint again to see what is left.

RUN

swiftlint
Lint all Swift files below here

`swiftlint` **lint** *<path>*
Lint given files or folders

`swiftlint` **--fix**
Auto-correct what can be fixed

`swiftlint` **lint --strict**
Fail on warnings too

`swiftlint` **lint --quiet**
Print violations only

`swiftlint` **lint --lenient**
Downgrade errors to warnings

`swiftlint` **--fix --format**
Also reformat like Xcode

`swiftlint` **lint --force-exclude** *<file>*
Skip excluded files even if named
Needed for editor and pre-commit integrations that pass file paths explicitly.

`swiftlint` **lint --use-stdin**
Lint standard input

`swiftlint` **lint --no-cache**
Ignore the lint cache

`swiftlint` **lint --only-rule** *<rule>*
Run just this rule

`swiftlint` **lint --enable-all-rules**
Run every rule, even opt-in

`swiftlint` **lint --working-directory** *<dir>*
Run as if started in <dir>

`swiftlint` **--version**
Print the version

RULES

`swiftlint` **rules**
List rules and their status

`swiftlint` **rules** *<rule>*
Show one rule's description

RULES (cont.)

`swiftlint` **rules --enabled**
Only rules enabled by config

`swiftlint` **rules --disabled**
Only rules that are off

`swiftlint` **rules --correctable**
Only rules --fix can fix

`swiftlint` **rules --default-config**
Print every rule's defaults

`swiftlint` **rules --config-only**
Print only the YAML config

`swiftlint` **docs**
Open the rule reference

CONFIG

disabled_rules: [*<rule>***]**
Turn off default rules

opt_in_rules: [*<rule>***]**
Turn on extra rules

included: [Sources]
Lint only these paths

excluded: [Pods, .build]
Never lint these paths

only_rules: [*<rule>***]**
Run exactly these rules
Cannot be combined with disabled_rules or opt_in_rules.

analyzer_rules: [*<rule>***]**
Rules for swiftlint analyze

opt_in_rules: [all]
Enable every opt-in rule

strict: true
Warnings fail the run

reporter: "*<name>***"**
Default output format

<rule>**: warning**
Set a rule's severity

line_length: 120
Warning threshold for a rule

file_length: {warning: 500, error: 1200}
Warning and error limits

type_name: {min_length: 4}
Tune a rule's options

parent_config: *<path or url>*
Inherit a shared config

child_config: *<path or url>*
Layer a higher-priority config

`swiftlint` **--config** *<a>* **--config** *<b>*
Stack configs, last wins
Passing --config turns off nested .swiftlint.yml lookup.

CONFIG (cont.)

baseline: *<file>*
Ignore violations saved in a baseline

lenient: true
Errors become warnings

INLINE CONTROL

// swiftlint:disable *<rule>*
Off until enabled or end of file

// swiftlint:enable *<rule>*
Turn the rule back on

// swiftlint:disable:next *<rule>*
Off for the next line

// swiftlint:disable:this *<rule>*
Off for this line

// swiftlint:disable:previous *<r>*
Off for the previous line

// swiftlint:disable all
Silence every rule

// swiftlint:disable *<r1>* *<r2>*
Several rules at once

OUTPUT

`swiftlint` **reporters**
List available reporters

`swiftlint` **lint --reporter json**
Choose the output format

`swiftlint` **lint --reporter github-actions-logging**
Annotate GitHub PRs

`swiftlint` **lint --reporter sarif**
SARIF for code scanning

`swiftlint` **lint --reporter summary**
Count violations per rule

`swiftlint` **lint --output** *<file>*
Write the report to a file

Common reporters

Name	Use for
`xcode`	Xcode build issues
`json`	scripts and jq
`github-actions-logging`	PR annotations
`junit`	CI test reports
`html`	browsable report
`summary`	rule counts

BASELINE & ANALYZE

`swiftlint` **lint --write-baseline** *<file>*
Save current violations

`swiftlint` **lint --baseline** *<file>*
Report only new violations

`swiftlint` **baseline report** *<file>*
Show what a baseline hides

`swiftlint` **analyze --compiler-log-path** *<log>*
Run analyzer rules

INSTALL & CI

brew install swiftlint
Install with Homebrew

mint install realm/SwiftLint
Install with Mint

swift package plugin swiftlint
Run via the SwiftLintPlugins package

Xcode 15+ sandboxes script phases: set ENABLE_USER_SCRIPT_SANDBOXING to NO so the lint phase can read your sources.

RECIPES

Minimal .swiftlint.yml

```
disabled_rules:
  - trailing_whitespace
opt_in_rules:
  - empty_count
  - closure_spacing
included:
  - Sources
excluded:
  - Pods
line_length: 120
type_name:
  min_length: 4
```

Xcode run script phase

```
if command -v swiftlint >/dev/null 2>&1; then
  swiftlint
else
  echo "warning: swiftlint not installed"
fi
```

Adopt on a messy codebase

```
swiftlint lint --write-baseline .swiftlint.baseline.json
# then add to .swiftlint.yml:
baseline: .swiftlint.baseline.json
swiftlint lint --strict
```

pre-commit hook

```
repos:
  - repo: https://github.com/realm/SwiftLint
    rev: <version>
    hooks:
      - id: swiftlint
        entry: swiftlint --fix --strict
```