Semgrep Static code scanning

CONCEPTS

Semgrep matches code structure, not text: a pattern like foo($X) finds the call however it is spaced or nested.

Rules YAML / registry
↓ `--config`
Targets files, ignores applied
↓ `scan`
Findings severity, location
↓ `--json` / `--sarif`
Report exit code, files

Without --error, scan exits 0 even when it finds issues. Add it (or use semgrep ci) to fail a pipeline.

SCANNING

`semgrep scan`
Scan the current directory

`semgrep scan --config auto`
Registry rules chosen for your code

`semgrep scan --config` *`<file|dir>`*
Run local rule files

`semgrep scan --config p/`*`<ruleset>`*
Run a registry ruleset

`semgrep scan` *`<path>`*
Scan only a file or folder

`semgrep scan --severity` *`<level>`*
Filter by INFO, WARNING, ERROR

`semgrep scan --include` *`<glob>`*
Only paths matching glob

`semgrep scan --exclude` *`<glob>`*
Skip paths matching glob

AD-HOC PATTERNS

`semgrep -e '`*`<pattern>`***`' -l`** *`<lang>`* *`<path>`*
Search with a pattern

`semgrep -e 'eval(...)' -l python .`
Find any eval call

AD-HOC PATTERNS (cont.)

Pattern syntax

Syntax	Matches
`...`	any arguments or statements
`$X`	any expression (metavariable)
`<... p ...>`	p nested anywhere inside

OUTPUT

`semgrep scan --json`
JSON to stdout

`semgrep scan --sarif -o` *`<file>`*
SARIF report to a file

`semgrep scan --json-output` *`<file>`*
JSON file, text on screen

CI & BASELINE

`semgrep ci`
Scan in CI, fail on blocking findings

`semgrep ci --baseline-commit` *`<ref>`*
Only findings new since ref

`semgrep scan --baseline-commit` *`<ref>`*
Same for plain scan

`semgrep scan --error`
Exit 1 when findings exist

IGNORING

`# nosemgrep`
Ignore this line or the next

`# nosemgrep:` *`<rule-id>`*
Ignore one rule only

`.semgrepignore`
Paths to skip, gitignore syntax

RULES

Exit codes

Code	Meaning
0	ok, or findings without --error
1	findings with --error
2	fatal error
4	invalid pattern in rule
7	invalid rule or missing config