🧰 Python Code Quality & Performance Toolkit

> **Objetivo:** Ferramentas para análise de qualidade, performance, complexidade, segurança e estilo em projetos Python.

🧠 1. Métricas de Complexidade e Manutenibilidade

Ferramenta	Comando principal	Descrição
Radon	`radon cc . -a`	Calcula **complexidade ciclomática**, **Halstead metrics** e **índice de manutenibilidade**.
Xenon	`xenon . --max-absolute B --max-modules A`	Extensão do Radon, **enforce limites de complexidade** (CI/CD friendly).
Wily	`wily build . && wily report`	Mede **evolução da complexidade** com base no histórico do Git.
Lizard	`lizard .`	Análise rápida de **complexidade e tamanho de funções**.

⚙️ 2. Lint e Análise Estática

Ferramenta	Comando principal	Descrição
Flake8	`flake8 .`	Verifica **erros de estilo** e **más práticas** (PEP8).
Pylint	`pylint my_module.py`	Avalia **qualidade geral do código** com pontuação de 0–10.
Ruff	`ruff check .`	Linter moderno, **muito rápido**; substitui flake8 + isort + parte do mypy.
Bandit	`bandit -r .`	Detecta **falhas de segurança** em código Python.

🧮 3. Tipagem e Contratos

Ferramenta	Comando principal	Descrição
Mypy	`mypy .`	Verifica **tipos estáticos** usando type hints (`PEP484`).
Pyright	`pyright`	Alternativa rápida para análise de tipos (VSCode usa internamente).
Deal	decorators `@pre`, `@post`, `@inv`	Implementa **Design by Contract** (pré/pós-condições).
Typeguard	`@typechecked`	Faz **checagem de tipos em runtime**.

🚀 4. Performance e Profiling

Ferramenta	Comando principal	Descrição
cProfile	`python -m cProfile -o output.prof script.py`	Profiler padrão de CPU.
pstats	`python -m pstats output.prof`	Analisa resultados do cProfile.
line_profiler / lineviz	`kernprof -l script.py && python -m line_profiler script.py.lprof`	Mede tempo **por linha de código**.
memory_profiler	`python -m memory_profiler script.py`	Mede **uso de memória** por linha.
py-spy	`py-spy top --pid <PID>`	Profiler externo, ideal para **produção**.
scalene	`scalene script.py`	Perfil **CPU + memória + tempo por linha** com visualização colorida.
perf	`python -m perf timeit 'code'`	Benchmarks **precisos e reprodutíveis**.

🧪 5. Testes e Cobertura

Ferramenta	Comando principal	Descrição
Pytest	`pytest`	Framework de testes mais usado e extensível.
Coverage.py	`coverage run -m pytest && coverage report`	Mede **cobertura de código**.
pytest-cov	`pytest --cov=package_name`	Integra pytest + coverage.
Mutmut	`mutmut run`	Faz **mutation testing**, testa robustez dos testes.

🛡️ 6. Segurança e Dependências

Ferramenta	Comando principal	Descrição
pip-audit	`pip-audit`	Detecta **vulnerabilidades (CVE)** em dependências.
Safety	`safety check`	Verifica **dependências inseguras**.
Deptry	`deptry .`	Detecta **dependências não usadas ou faltando**.
pip-check-reqs	`pip-missing-reqs .`	Valida se `requirements.txt` está correto.

🗒️ 7. Estilo e Formatação

Ferramenta	Comando principal	Descrição
Black	`black .`	Formata código automaticamente (PEP8).
Isort	`isort .`	Ordena imports de forma consistente.
Docformatter	`docformatter -r .`	Formata docstrings (PEP257).
Pre-commit	`pre-commit install`	Executa linters e formatadores antes de commits.

🧩 8. Manutenibilidade e Visualização

Ferramenta	Comando principal	Descrição
Vulture	`vulture .`	Encontra **código morto** (funções/imports não usados).
Pydeps	`pydeps my_package`	Gera **gráfico de dependências** entre módulos.
Code2Flow	`code2flow my_module.py`	Gera **diagrama de fluxo de execução**.
SonarQube	CI/CD integration	Análise avançada de qualidade e dívida técnica.

🧱 9. Pipeline Ideal de Qualidade

```
# Instalação
pip install black ruff mypy pytest pytest-cov radon scalene

# Fluxo recomendado
black .             # Formata código
ruff check .        # Lint e estilo
mypy .              # Tipagem estática
radon cc . -a       # Complexidade ciclomática
pytest --cov        # Testes + cobertura
scalene main.py     # Profiling detalhado
```

🗒️ Referências Rápidas

- 📘 PEP8 – Estilo de Código: https://peps.python.org/pep-0008/
- 📗 PEP484 – Tipagem: https://peps.python.org/pep-0484/
- 📙 PEP257 – Docstrings: https://peps.python.org/pep-0257/