pre-commit:
A framework for managing and maintaining multi-language pre-commit hooks.

Miguel Bernal Marin

Pythonistas GDL

miguel.bernal.marin@gmail.com
Telegram: *@miguelinux*

25 de julio de 2025

¡Hola!

Miguel Bernal Marin

Profesor:

- Instituto Tecnológico Superior de Jalisco (Zapopan)
- ITESO

Contenido

Contenido

What is a Linter?

Definition

A linter is a software tool that **analyzes source code** for errors, style issues, and vulnerabilities. Its goal is to improve code quality.

The name comes from the original tool called **Lint**, which was designed to check C code for errors and inefficiencies.

Linter functions

- Identify syntax errors, logical errors, typos, and potential security vulnerabilities

Linter functions

- Identify syntax errors, logical errors, typos, and potential security vulnerabilities
- Comply with coding standards and style guidelines

Linter functions

- Identify syntax errors, logical errors, typos, and potential security vulnerabilities
- Comply with coding standards and style guidelines
- Suggest or apply corrections

Linter functions

- Identify syntax errors, logical errors, typos, and potential security vulnerabilities
- Comply with coding standards and style guidelines
- Suggest or apply corrections
- Format code

Linter functions

- Identify syntax errors, logical errors, typos, and potential security vulnerabilities
- Comply with coding standards and style guidelines
- Suggest or apply corrections
- Format code
- Enforce coding standards.

Importance of lintering

- Identify errors before they become real problems at runtime

Importance of lintering

- Identify errors before they become real problems at runtime
- Maintain code more consistently

Importance of lintering

- Identify errors before they become real problems at runtime
- Maintain code more consistently
- Simplify tasks

Importance of lintering

- Identify errors before they become real problems at runtime
- Maintain code more consistently
- Simplify tasks
- Ensure code quality.

Contenido

Ruff

- An extremely fast Python linter and code formatter, written in Rust.
- https://docs.astral.sh/ruff/

Ruff

- An extremely fast Python linter and code formatter, written in Rust.
- https://docs.astral.sh/ruff/
- Ruff aims to be orders of magnitude faster than alternative tools while integrating more functionality behind a single, common interface.

Ruff

- An extremely fast Python linter and code formatter, written in Rust.
- https://docs.astral.sh/ruff/
- Ruff aims to be orders of magnitude faster than alternative tools while integrating more functionality behind a single, common interface.
- Ruff can be used to replace Flake8 (plus dozens of plugins), Black, isort, pydocstyle, pyupgrade, autoflake, and more, all while executing tens or hundreds of times faster than any individual tool.

Black

- The uncompromising code formatter
- https://black.readthedocs.io/en/stable/

Black

- The uncompromising code formatter
- https://black.readthedocs.io/en/stable/
- Black gives you speed, determinism, and freedom from pycodestyle nagging about formatting

Black

- The uncompromising code formatter
- https://black.readthedocs.io/en/stable/
- Black gives you speed, determinism, and freedom from pycodestyle nagging about formatting
- Black makes code review faster by producing the smallest diffs possible.

Black

- The uncompromising code formatter
- https://black.readthedocs.io/en/stable/
- Black gives you speed, determinism, and freedom from pycodestyle nagging about formatting
- Black makes code review faster by producing the smallest diffs possible.
- Blackened code looks the same regardless of the project you're reading.

Bandit

- Bandit is a tool designed to find common security issues in Python code.
- https://bandit.readthedocs.io/en/latest/

Reorder Python imports

- Tool for automatically reordering python imports
- https://github.com/asottile/reorder-python-imports

Reorder Python imports

- Tool for automatically reordering python imports
- https://github.com/asottile/reorder-python-imports
- Similar to isort but uses static analysis more.

Contenido

Pre-commit

- A framework for managing and maintaining multi-language pre-commit hooks.
- https://pre-commit.com/

Pre-commit

- A framework for managing and maintaining multi-language pre-commit hooks.
- https://pre-commit.com/
- Git hook scripts are useful for identifying simple issues before submission to code review

Pre-commit

- A framework for managing and maintaining multi-language pre-commit hooks.
- https://pre-commit.com/
- Git hook scripts are useful for identifying simple issues before submission to code review
- We run our hooks on every commit to automatically point out issues in code such as missing semicolons, trailing whitespace, and debug statements.

Pre-commit installation

Using pip:

```
pip install pre-commit
```

RPM based Linux:

```
dnf install pre-commit
```

DEB based Linux:

```
apt install pre-commit
```

Pre-commit configuration file

- create a file named **.pre-commit-config.yaml**

Pre-commit configuration file

- create a file named **.pre-commit-config.yaml**
- you can generate a very basic configuration using **pre-commit sample-config** or copy from our repo ;-)

```yaml
repos:
  - repo: https://github.com/pre-commit/pre-commit-hooks
    rev: v5.0.0
    hooks:
      - id: check-merge-conflict
      - id: debug-statements
      - id: fix-byte-order-marker
      - id: trailing-whitespace
      - id: end-of-file-fixer
      - id: check-yaml
      - id: check-added-large-files
      - id: detect-private-key
```

Install the git hook scripts

- run **pre-commit install** to set up the git hook scripts

```
$ pre-commit install
pre-commit installed at .git/hooks/pre-commit
$
```

- now **pre-commit** will run automatically on **git commit**!

Run against all the files (optional)

- it's usually a good idea to run the hooks against all of the files when adding new hooks
- usually **pre-commit** will only run on the changed files during git hooks

```
$ pre-commit run --all-files
[INFO] Initializing environment for https://github.com/pre-commit/pre-commit-hooks.
[INFO] Initializing environment for https://github.com/psf/black.
[INFO] Installing environment for https://github.com/pre-commit/pre-commit-hooks.
[INFO] Once installed this environment will be reused.
[INFO] This may take a few minutes...
[INFO] Installing environment for https://github.com/psf/black.
[INFO] Once installed this environment will be reused.
[INFO] This may take a few minutes...
Check Yaml...............................................................Passed
Fix End of Files.........................................................Passed
Trim Trailing Whitespace.................................................Failed
- hook id: trailing-whitespace
- exit code: 1

Files were modified by this hook. Additional output:

Fixing sample.py

black....................................................................Passed
```

Agradecimientos

¡Gracias!

El código fuente de esta presentación la puedes encontrar en

- https://github.com/miguelinux/slides {platicas/pyGDL250725/}