When Can I Check My Code Quality?

You can check your code's quality:

- As you write it
- When it's checked in
- When you're running your tests

It's useful to have linters run against your code frequently. If automation and consistency aren't there, it's easy for a large team or project to lose sight of the goal and start creating lower quality code. It happens slowly, of course. Some poorly written logic or maybe some code with formatting that doesn't match the neighboring code. Over time, all that lint piles up. Eventually, you can get stuck with something that's buggy, hard to read, hard to fix, and a pain to maintain.

To avoid that, check code quality often!

As You Write

You can use linters as you write code, but configuring your environment to do so may take some extra work. It's generally a matter of finding the plugin for your IDE or editor of choice. In fact, most IDEs will already have linters built in.

Here's some general info on Python linting for various editors:

- Sublime Text
- VS Code
- Atom
- Vim
- Emacs

Before You Check In Code

If you're using Git, Git hooks can be set up to run your linters before committing. Other version control systems have similar methods to run scripts before or after some action in the system. You can use these methods to block any new code that doesn't meet quality standards.

While this may seem drastic, forcing every bit of code through a screening for lint is an important step towards ensuring continued quality. Automating that screening at the front gate to your code may be the best way to avoid lint-filled code.

When Running Tests

You can also place linters directly into whatever system you may use for continuous integration. The linters can be set up to fail the build if the code doesn't meet quality standards.

Again, this may seem like a drastic step, especially if there are already lots of linter errors in the existing code. To combat this, some continuous integration systems will allow you the option of only failing the build if the new code increases the number of linter errors that were already present. That way you can start improving quality without doing a whole rewrite of your existing code base.

Conclusion

High-quality code does what it's supposed to do without breaking. It is easy to read, maintain, and extend. It functions without problems or defects and is written so that it's easy for the next person to work with.

Hopefully it goes without saying that you should strive to have such high-quality code. Luckily, there are methods and tools to help improve code quality.

Style guides will bring consistency to your code. PEP8 is a great starting point for Python. Linters will help you identify problem areas and inconsistencies. You can use linters throughout the development process, even automating them to flag lint-filled code before it gets too far.

Having linters complain about style also avoids the need for style discussions during code reviews. Some people may find it easier to receive candid feedback from these tools instead of a team member. Additionally, some team members may not want to "nitpick" style during code reviews. Linters avoid the politics, save time, and complain about any inconsistency.

In addition, all the linters mentioned in this article have various command line options and configurations that let you tailor the tool to your liking. You can be as strict or as loose as you want, which is an important thing to realize.

Improving code quality is a process. You can take steps towards improving it without completely disallowing all nonconformant code. Awareness is a great first step. It just takes a person, like you, to first realize how important high-quality code is.