Jess Monnier

CSD-430 Module 11 Assignment

12 October 2025

**Coding Standards**

A development team using any programming language would be incomplete without coding standards, or a universally accepted convention for formatting the code in a way that is clean, readable, secure, bug-free, and efficient (Codacy, 2025). While technically a solo programmer could format their code in any way that still allows it to function as intended, even they are likely to suffer if they do not follow a consistent set of standards; in the future, when reviewing old programs they have written, they would be almost certain to find themselves a little (or a lot) lost figuring out how everything fits together.

For many programming languages, there are commonly accepted coding standards that tend to be applied by teams using that language. For example, a lot of Python programmers follow the PEP 8, the official style guide created by Guido van Rossum, Barry Warsaw, and Alyssa Coghlan and maintained by the open-source Python development community (Bader, 2025). However, the coding standards used by a development team don’t have to be official standards accepted by the larger community of that programming language. Their purpose is to help keep code readable by anyone who needs to reference it, a goal that tends to get more difficult the longer it has been since the code was written—especially if coding standards are not used or are poorly followed. Therefore, a team could come up with its own standards specific to their needs and processes. In fact, I would not be surprised if many teams expected programmers to follow guidelines specific to their organization in addition to a widely accepted set of conventions. For example, in this class we have been following the requirement of documenting our name, the assignment, the date, and the purpose of the program at the top of any code submitted for a grade, which could be considered a coding standard specific to this course.

The nature of coding standards means that there is no universal one-size-fits-all guideline for writing code in all programming languages, but there do tend to be commonalities. For example, languages often come with widely accepted conventions for naming variables. What those conventions entail may vary by language, but they do typically exist. According to the PEP-8 style guide for Python, packages and modules should have short, all-lowercase names, classes should use the CapWords convention, and functions and variables should be all lowercase with underscores to improve readability (van Rossum et al, 2001). On the other hand, the Google Java Style Guide prescribes that packages and modules should be all lowercase with no underscores, class names should be in CapsWords (or UpperCamelCase, as they call it), method names should be in lowerCamelCase (2020)… et cetera. There are a *lot* of naming conventions for Java.

Some other common aspects of code that are often prescribed by coding standards include: indentation, commenting, whitespace, code line length, function/method length, error handling, and file organization (Codacy, 2025). Regardless of the aspect of coding being standardized, the end goal is the same: ensure that the code is consistent, clear, secure, and easy to maintain. Some sources may describe this goal slightly differently; for instance, a 2024 Medium article written by Sagar Hudge details five pillars of code quality, which are readability, maintainability, reusability, reliability, and performance efficiency. It is not hard to imagine why organizations creating products with code would prioritize goals like these. Some important advantages of choosing and/or creating coding guidelines and sticking to them include increased software efficiency, reduced development time, earlier error detection, and reduced complexity, all of which add up to a reduction in hidden costs from developing software (GeeksforGeeks, 2024). By extension, the consequences of not following effective coding standards can be many (such as security breaches or greater difficulty detecting and correcting errors), but it all boils down to increased cost of some sort, be it money, time, or reputation.

Selecting coding standards doesn’t have to be an ordeal. It can consist of simply choosing a widely accepted standard used across the community of programmers using a specific language, and then adding any specific conventions that are applicable to your organization. Ideally, newly hired programmers proficient in the given language will already be aware of and prepared to follow the widely accepted standards, making it easier to learn and internalize any specific to the team. But even in the case that a team or new hire has to wade through the initial overhead of learning an unfamiliar set of standards, the time and money saved by effectively implementing these standards will be well worth it in the end.

**References**

Bader, D. (2025, September 25). *PEP 8* [Python Glossary]. Real Python. <https://realpython.com/ref/glossary/pep-8/>

Codacity. (2025, March 11). *Coding Standards: What Are They and Why Are They Important?* <https://blog.codacy.com/coding-standards>

GeeksforGeeks. (2024, May 23). *Coding Standards and Guidelines*. <https://www.geeksforgeeks.org/software-engineering/coding-standards-and-guidelines/>

Google. (2020). Google Java style guide. <https://google.github.io/styleguide/javaguide.html>

van Rossum, G., Warsaw, B., & Coghlan, N. (2001). PEP 8 – Style guide for Python code (PEP 8). Python Software Foundation. <https://peps.python.org/pep-0008/>