Corporate Coding Standards

Rachel Theis  
CSD 430

7.27.25

**Introduction**

Coding standards refer to a pre-determined set of rules and conventions within an organization that teams use when writing code. The purpose of implementing coding standards is to achieve uniformity within the organization’s codebase and to enable developers to collaborate seamlessly on projects. Teams should always have an agreed-upon set of coding standards that make sense for the project’s specifics and are driven by the organization’s and broader industry standards and best practices (Codacy, 2023b).

Coding standards are beneficial for teams and their organization for many reasons, and can mitigate common issues that arise from poor software quality related to aggressive deadlines, changing customer requests, and budgetary challenges (Codacy, 2023b). Below is an overview of the benefits of implementing and enforcing coding standards (Dovgal, 2024):

* Collaboration: Teams are more easily able to understand and modify code written by someone else on the team if everyone is following the same standard. This benefit also makes code more readable and maintainable over time.
* Consistency: The consistent nature of code that is being written to follow coding standards makes code higher quality, and makes it easier to switch between tasks and work together. Coding standards also mitigate errors that occur due to the use of conflicting practices.
* Code quality: Coding standards contribute to higher code quality because they enforce best practices for elements like error handling, input validation, and resource management.
* Efficiency: Coding standards make teams work more efficiently because they eliminate unnecessary work like correcting inconsistent or duplicate code.
* Faster Onboarding: It is easier to bring new team members up to speed when code is uniform and follows clear directives. Additionally, if a team member leaves, their workload can be easily reassigned due to the uniformity of their code compared to the larger codebase.
* Security and Compliance: Secure coding practices should be integrated into coding standards and can facilitate compliance with regulations and mitigate known vulnerabilities.
* Scalability: Coding standards can optimize and modularize software and can help manage increasing workflows.
* Reduced Technical Debt: Technical debt occurs when teams choose short-term solutions that band-aid an issue today but push the issue to a later date. Implementing coding standards makes software strong and ready for the future.
* Long-term Savings: Coding standards ultimately save organizations money due to their benefits like efficiency, compliance, and maintainability.

**Open vs. Closed Coding Standards**

Coding Standards can exist publicly or privately. Open coding standards, or public coding standards, are coding standards that leverage community collaboration and input. Open coding standards are dynamic and ever-changing in response to evolving needs and best practices. An example of an open coding standard is CERT, a secure coding standard that is accessible through a public Wiki (Codacy, 2023b). On the other hand, closed, or private, coding standards are enforced only within an organization and are not publicly available. For example, closed coding standards could include an internal style guide that specifies naming conventions, commenting practices, and architecture-specific pattern guidelines (Codacy, 2023b).

Organizations can choose between implementing open coding standards or customizing their own closed coding standards. Factors organizations should consider when choosing between open and closed coding standards include maintainability and consistency, collaboration, security, and resource intensity (Codacy, 2023). For example, closed coding standards lend to greater consistency, maintainability, and security because open coding standards change so frequently (Codacy, 2023b). However, creating an internal set of closed coding standards can be time-consuming and require an expert, so some organizations that do not work with security-critical information may prefer to use existing open coding standards that are externally managed (Codacy, 2023b).

**Implementing Coding Standards**

If an organization chooses to implement closed standards within its organization, it should consider factors like the programming language being used, the type of software being developed, and the size of the development team when creating coding standards (Codacy, 2023a). Coding standards should be clear, specific, and easy to understand. However, organizations can choose to adapt their coding standards from existing publicly available and trustworthy coding standards. When developing coding standards, organizations should leverage as many different perspectives from the team as possible to ensure that the coding standards are correct and accurate, and that everyone on the team is on board with the standards (Codacy, 2023a). Below is an outline of what should be included in coding standards (Manifestly, 2025):

* Naming conventions and code structure: Clear naming conventions and a consistent code structure make code more maintainable and readable. “For instance, variable names should be descriptive and reflect their purpose, methods should be named to clearly indicate their actions, and class names should be nouns that describe what they represent.” (Manifestly, 2025).
* Commenting and documentation processes: Coding standards should clarify expectations regarding how and when to comment and document code. Comments should be used to clarify the code when necessary, such as in the case of complex logic, API usage, and setup instructions.
* Error handling and security guidelines: Coding standards should seek to ensure that code always behaves predictably, even in unusual circumstances. Organizations should use security guidelines, like those provided by OWASP, to mitigate known vulnerabilities. Specific security concerns and behaviors that should be included in coding standards include validating input, sanitizing data, and managing exceptions.

**Examples of Coding Standards**

One example of a publicly available coding standard is Google’s JavaScript Style Guide. Below are a few examples of what this looks like. Here, standards are presented in a clear, specific, and easily understandable way:

Example #1: Source File Structure (Google, 2025):

All new source files should either be a goog.module file (a file containing a goog.module call) or an ECMAScript (ES) module (uses import and export statements).

Files consist of the following, **in order**:

1. License or copyright information, if present
2. @fileoverview JSDoc, if present
3. goog.module statement, if a goog.module file
4. ES import statements, if an ES module
5. goog.require and goog.requireType statements
6. The file’s implementation

**Exactly one blank line** separates each section that is present, except the file's implementation, which may be preceded by 1 or 2 blank lines.

Example #2: Naming rules common to all identifiers (Google, 2025):

Give as descriptive a name as possible, within reason. Do not worry about saving horizontal space as it is far more important to make your code immediately understandable by a new reader. Do not use abbreviations that are ambiguous or unfamiliar to readers outside your project, and do not abbreviate by deleting letters within a word.

errorCount // No abbreviation.

dnsConnectionIndex // Most people know what "DNS" stands for.

referrerUrl // Ditto for "URL".

customerId // "Id" is both ubiquitous and unlikely to be misunderstood.

Disallowed:

n // Meaningless.

nErr // Ambiguous abbreviation.

nCompConns // Ambiguous abbreviation.

wgcConnections // Only your group knows what this stands for.

pcReader // Lots of things can be abbreviated "pc".

cstmrId // Deletes internal letters.

kSecondsPerDay // Do not use Hungarian notation.

**Exception**: Variables that are in scope for 10 lines or fewer, including arguments that are *not* part of an exported API, *may* use short (e.g. single letter) variable names.

**References**

Codacy. (2023a, March 22). *How to implement coding standards in your organization*. Www.linkedin.com. https://www.linkedin.com/pulse/how-implement-coding-standards-your-organization-codacy

Codacy. (2023b, November 14). *Coding Standards: What Are They and Why Are They Important?* Blog.codacy.com. https://blog.codacy.com/coding-standards

Dovgal, A. (2024, December 30). *Coding Standards And Best Practices: Guide & Implementation Tips | Devcom*. DevCom. https://devcom.com/tech-blog/coding-standards-and-best-practices-guide-implementation-tips/

Google. (2025). *Google JavaScript Style Guide*. Github.io. https://google.github.io/styleguide/jsguide.html#introduction

Manifestly. (2025). *Manifestly Checklists | Coding Standards Checklist*. Manifestly Checklists. https://www.manifest.ly/use-cases/software-development/coding-standards-checklist