# The Apex Developer Manual

“Any fool can write code that a computer can understand. Good programmers write code that humans can understand.” -Martin Fowler”

This document lists a working set of coding standards used as a reference when developing on the [Company] Salesforce org. Understanding and following these guidelines will be essential for submitting Pull Requests that get approved and merged, as they lay down the foundational review criteria.

The following descriptions are representative, not exhaustive, as many common sense variations can be derived. Understand the principals and apply them wisely.

## Development Process

We follow the development process below so that [describe the intended outcome].

1. [each step of the process]

## Design Considerations

### Trigger Architecture

### Scheduled Jobs

### Integrations

### Practiced Design Patterns

### [additional design considerations]

## Naming Conventions

### Classes

[Describe how classes should be named.]

#### Class Variations

[Include specialized class naming conventions, if desired.]

### Variables

[Describe how variables should be named.]

### Constants

[Describe how constants should be named.]

### Methods

[Describe how methods should be named.]

## Code Layout and Formatting

### The Editor Config

[todo: create an Editor Config and add it to your code repository. Adjust wording below as needed.]

Well formatted code increases readability, understanding and ultimately maintainability of the code base. A developer should strive to use a consistent layout and format. It will make the lives of other developers (who want to read, review, or maintain your code) a lot easier.

If you are using IntelliJ/IlluminatedCloud, an idea-editor-config.xml has been provided, which can easily be imported to conform to our general layout and formatting. If you are using VSCode, an .editorconfig file has been provided in the repo. To set it up, start with the following steps:

1. Pull from main and make sure .editorconfig is in the root directory.
2. Download the VSCode extension EditorConfig for VS Code.
3. Navigate to VSCode Settings->Extensions->EditorConfig->Template
4. Change from default to local the path for .editorconfig file.

### Line Wrapping

[Include specifications regarding line-wrapping and code structure statements.]

### Placement

[Include specifications regarding placement of variables, code blocks, etc.]

### Class and Method Declarations

[Include specifications regarding class and method declarations.]

### if, if-else, else-if, else Statements

[Include specifications regarding if/else statements.]

### for Loops

[Include specifications regarding for loops.]

### try-catch Statements:

[Include specifications regarding try-catch statements.]

### Blank Lines

[Include specifications about the usage of blank lines, if desired. If not, remove section.]

## Unit Tests

Unit Tests are important, so they get their own section. Well-formed, consistent, and rich unit tests are as valuable as the code itself. If they are conceived and created as an afterthought or annoyance that must be completed, it is likely that the intended value they can provide is greatly diminished.

### Naming Unit Tests

[How should unit tests be named?]

### The Structure of a Unit Test

[How should unit tests be structured?]

### Test Variable Names

[If you have any expectation regarding test variables names, include them here.]

### Assertions

[List any specific requirements regarding test assertions.]

## Opinionated Practices

These are key coding principles and practices that aim to produce efficient and scalable code.

### [name]

[List any practices that are expected for extending and maintaining the org.]

## Other Conventions

### [name]

[List any other convention you want maintained in the org.]

## Logging

[Logging practices]