Developer Guide

General Coding Guidelines

1. **Naming conventions for local variables, global variables, constants and functions:**  
Some of the naming conventions are given below:

* Meaningful and understandable variables name helps anyone to understand the reason of using it.
* Local variables should be named using camel case lettering starting with small letter (e.g. **localData**) whereas Global variables names should start with a capital letter (e.g. **GlobalData**). Constant names should be formed using capital letters only (e.g. **CONSDATA**).
* It is better to avoid the use of digits in variable names.
* The names of the function should be written in camel case starting with small letters.
* The name of the function must describe the reason of using the function clearly and briefly.

**2. Indentation:**  
Proper indentation is very important to increase the readability of the code. For making the code readable, programmers should use White spaces properly. Some of the spacing conventions are given below:

* There must be a space after giving a comma between two function arguments.
* Each nested block should be properly indented and spaced.
* Proper Indentation should be there at the beginning and at the end of each block in the program.
* All braces should start from a new line and the code following the end of braces also start from a new line.

3. **Code should be well documented:**  
The code should be properly commented for understanding easily. Comments regarding the statements increase the understandability of the code.

4. **Length of functions should not be very large:**  
Lengthy functions are very difficult to understand. That’s why functions should be small enough to carry out small work and lengthy functions should be broken into small ones for completing small tasks.

5. **Avoid using a coding style that is too difficult to understand:**  
Code should be easily understandable. The complex code makes maintenance and debugging difficult and expensive.

Angular Coding Guidelines

1. Angular Coding Best Practices using Angular Coding Guidelines

* Make sure code does not exceed 400 lines limit per file.
* Ensure that code does not exceed 75 lines for each function.
* If the variables have constant values, declare them with const.
* Name of properties and methods should be used in lower camel cases.
* Always leave an empty line between imports and modules.

2. Single Responsibility Principle

One of the most important practices in Angular development is to make sure that you don’t create more than one instance of any component, directive or service of the same file. Each file should be responsible for one particular function. Doing this help keep your Angular code-clean, manageable , readable and maintainable.

3. Break the large components into manageable sizes

Overall angular app would benefit greatly with increased efficiency in breaking more significant components into smaller manageable ones. One of the angular best practices is to try and break your components into smaller components to the extent that each has only one atomic task.

4. Proper Utilization of Lazy Loading

Utilizing lazy load the modules can enhance your productivity. Lazy load is a built-in feature in angular which helps developer with loading the thing which is required. For example, when you use the feature, it loads the components and other things you needed and stops other and unnecessary files to get loaded.

5. File Naming and Structure

* Use consistent and descriptive file names.
* Organize your files into logical folders such as components, services, modules, etc.
* Use kebab-case for file names (e.g., my-component.component.ts, my-service.service.ts).

6. Component Naming

* Use PascalCase for component names (e.g., MyComponent).
* Append the component type to the name (e.g., HeaderComponent, ProductListComponent)

7. Module Naming

* Use PascalCase for module names (e.g., AppModule, SharedModule).

8. Services

* Use services to encapsulate business logic and data manipulation.
* Ensure that services have a single responsibility.
* Provide services at the module level to ensure singleton instances.

9. Comments and Documentation

* Provide meaningful comments to explain complex logic or algorithms.
* Document your code in word and excel file.

10. Root Module

Root module is needed for starting an angular app. This module loads all the root components and other modules. Root modules are alternatively known as AppModule and are created under the src/app folder.

11. Feature Module

Feature module, as the name suggests, implements a specific feature of your Angular App. All the associated pipes, components and directives that help implement the feature become a part of its module. You can create sub folders for your directives and pipes under the module folder. Make a components folder to place all the components in an effective way.

12. Core Module

A "core module" in Angular is a module that houses essential and reusable components, services, and configurations that are fundamental to your application's functionality. It acts as a central place where you can define and manage services that are shared across different parts of your application.

13. Shared Module

A "shared module" is a module that encapsulates and exports reusable components, directives, pipes, and other features that can be used across multiple parts of your application. It's designed to promote code reusability and maintainability by providing a central place to define and manage these shared elements.

14. Building Reusable Component

Building reusable component is a golden rule that cannot be omitted when discussing Angular.js best practices. If there is a part of your Angular App’s UI that you know you will need in multiple places, build a component for it and use the component. This will help in implementing the UI element consistently throughout the application. And also, if your app goes through redesign process for the some reason, you will not have to change the UI code at 100s of places. You need to change the component, you are good to go.

15. Use Index.ts

Index.ts allows Angular Developers to keep all related things closely packed together, so they don’t need to bother about the source file name. This helps to reduce the size of introductory statements.

16. Avoid using logic in the components

Many developers mix the components and business logic, resulting in a complex and untidy mess that is difficult to comprehend. Having logic in a separate service will allow you to use it in multiple components.