# Coding Guidelines | IONIC Framework

**1.) Folder Structure :**

Ionic folder structure must be like this :

*root/ hooks/ platforms/*

*ios/ android/*

*plugins/ resources/ scss/*

*ionic.app.scss mobile.scss tablet.scss desktop.scss*

*www/*

*js/*

*controllers/ lib/*

*css/*

*img/*

*lib/*

*index.html config.xml*

**2.) Ionic SCSS :**

Write all your CSS in SCSS, so it’ll easy to maintain and you can use ionic variables in your SCSS. You can modify ionic color or other variables accordingly.

If you need different CSS for tablet or mobile write them in separate files. Your CSS must be commented so it’ll easy to maintain.

Don’t change anything in ionic SCSS files, write your separate CSS.

You can write some custom mixins/variables so that you can make your CSS reusable. Make sure all color code must be defined in variables.

**3.) Brake Controllers/Modules/Services :**

Break large Controllers/Modules/Services in a seperate files so it’ll easy to manage the code.

*Example :*

*app*

*.controllers(‘mainCtrl’, mainCtrl)*

*.controllers(‘appCtrl’, appCtrl);*

*/\*\* write this code in a separate file\*/ function mainCtrl(){*

*/\*Your code here\*/*

*}*

*/\*\* write this code in a separate file\*/ Function appCtrl(){*

*/\*Your code here\*/*

*}*

**4.) Web service URLs:**

Base url define on top of file so it’ll easy to change url for different servers(like : dev,qa,staging)

All web service URLs must be in separate files with proper comments(with get, post and result data).

Example :

*var baseUrl = "https://dev.server.com/api/";//Dev*

*//var baseUrl = "https://qa.server.com/api/";//QA*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1.) Login User*

*--------------------------------*

*--------------------------------*

*//POST Request*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

\* *Request : =>>>*

\* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*{*

*"email":"*[*dheeraj.piplani@appster.in*](mailto:dheeraj.piplani@appster.in)*", "password":"12345",*

*"deviceToken":"",*

*"deviceType":"I"*

*}*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

\* *Response : =>>>*

\* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*{*

*"data":{ "userId" : 8, "firstName" :*

*"Jaswant",*

*"lastName" : "Dhayal",*

*"dob" : "09-05-1990 00:00:00",*

*"address" : "Gurgaon, Haryana 122008, India", "email" : "*[*jaswant.dhayal@appster.in*](mailto:jaswant.dhayal@appster.in)*",*

*"sessionToken" : "2d9d2892-fa72-4e75-8466-97eeededdb10"*

*},*

*"Success":true, "Message":"", "responseCode":200*

*}*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ var loginUser = baseUrl+"user/auth/signInUser";*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2.) Get Current User Profile*

*--------------------------------*

*--------------------------------*

*//GET Request*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

* *Response : =>>>*

\* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

\*

*{*

*"data": true, "success": true,*

*"message": "This is msg", "responseCode": 200*

*}*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*var getBasicUserProfile = baseUrl+"user/profile/getBasicUserProfile?userId=";*

**5.) Static JSONs:**

All static JSON data must in separate files so that it’ll easy to maintain.

**6.) Use directive and includes :**

Reuse your code by using directives, includes, services etc.

If you need to render same html again and again you need to create a directive or include as per requirement. If you need to use same function on many places, you can create a service or factory as per requirement

**7.) Use Single factory/service for call web services :**

Use same factory/service for call all web services(if strictly needed then you can create a different factory/service).

By this way you can handle all errors by their error code at the same place and show message to user or manage app functionality accordingly.

If you need a access token for call web services then this way you can manage access token easily. For a specific error if you need to logout user then you can handle it easily.

**8.) Use Scopes properly :**

There are three type of scopes in your app - $scope(controller scope), $rootScope(app scope), Isolated Scope(directive scope). Use scopes as per requirement, ignore to declare huge data in rootScope.

Declare only the data which you need whole app in rootScope.

All controller functions and variables declare in the controller’s scope

Ignore to use isolated scope, because by this way you bind the controller, not to use the parent controller’s scope and use the own scope however if necessary you can use it.

**9.) Variable Names :**

We use **camelCase** for identifier names (variables and functions). All names start with a **letter**.

Example :

*var firstName = "John"; var lastName = "Doe"; var price = 19.90;*

*var tax = 0.20;*

*var fullPrice = price + (price \* tax);*

**10.) Spaces Around Operators :**

Always put spaces around operators ( = + - \* / ), and after commas:

Example :

*var x = y + z;*

*var values = ["Volvo", "Saab", "Fiat"];*

**11.) Statement Rules :**

**General rules for simple statements:**

Always end a simple statement with a semicolon. General rules for complex (compound) statements:

Example :

*var values = ["Volvo", "Saab", "Fiat"]; var person = {*

*firstName: "John", lastName: "Doe", age: 50, eyeColor: "blue"*

*};*

Put the opening bracket at the end of the first line. Use one space before the opening bracket.

Put the closing bracket on a new line, without leading spaces.

Short objects can be written compressed, on one line, using spaces only between properties, like this:

Example :

*var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};*

**12.) Avoid Global Variables**

Minimize the use of global variables.

This includes all data types, objects, and functions.

Global variables and functions can be overwritten by other scripts. Use local variables instead.

**13.) Compress your code:**

Don't forget to compress your ionic code, you can use this npm tool to compress your code : https://[www.npmjs.com/package/ionic-minify](http://www.npmjs.com/package/ionic-minify)