Simple AJS app {{step-by-step}} approach

1. Define structure of the project
   1. Folder js for controllers, services, filters, directives, etc.
   2. Folder view for HTML templates
   3. index.html for entry point for the project
      1. include basic AJS libraries like

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>  
<script

src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular-route.js"></script>

1. Set directive ng-app in index.html on <html> or <body> i.e. root level
2. If you assign name on directive ng-app like ng-app=”app” you have to define module named “app”
   1. Include in index.html

<script src="js/app.js"></script>

with implementation like

(**function** () {  
 'use strict';  
  
 angular.module('app', ['app.controllers', 'ngRoute'  
 ]);  
  
})();

The dependencies in brackets could vary.

1. Using a directive in HTML component like

<div ng-controller="app.controllers as hello">  
 <input ng-model="hello.greeting.text">  
  
 <p>{{hello.greeting.text}}, World</p>  
</div>

* 1. Use controllerAs Controller Syntax
  2. Define controller in folder js like controllers.js
  3. Include in index.html

<script src="js/controllers.js"></script>

* 1. The implementation of the controller should be like

(**function** () {  
 'use strict';  
  
 angular.module('app', [])  
 .controller('app.controllers', HelloController);  
  
 HelloController.$inject = ['$scope'];  
  
 **function** HelloController($scope) {  
 **var** vm = **this**;  
 vm.greeting = {text: 'Hello ...'};  
 // use $location for something good here...  
  
 console.log('HelloController', $scope);  
  
 }  
  
})();

* + 1. Use a capture variable for this when using the controllerAs syntax. Choose a consistent variable name such as vm, which stands for ViewModel.

Why?: The this keyword is contextual and when used within a function inside a controller may change its context. Capturing the context of this avoids encountering this problem.