Angularjs

It can be use for single page application. SPA

Sample code:

|  |
| --- |
| <!doctype html>  <html ng-app>    <head>  <script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.3/angular.min.js"></script>  </head>    <body>  <div>  <label>Name:</label>  <input type = "text" ng-model = "yourName" placeholder = "Enter a name here">  <hr />    <h1>Hello {{yourName}}!</h1>  </div>    </body>  </html> |

Include angularjs reference

<head>

<script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.5.2/angular.min.js"></script>

</head>

Point to angularjs app

<body ng-app = "myapp">

</body>

View:

<div ng-controller = "HelloController" >

<h2>Welcome {{helloTo.title}} to the world of Tutorialspoint!</h2>

</div>

*ng-controller* tells AngularJS what controller to use with this view. *helloTo.title*tells AngularJS to write the "model" value named helloTo.title to the HTML at this location.

Controller

<script>

angular.module("myapp", [])

.controller("HelloController", function($scope) {

$scope.helloTo = {};

$scope.helloTo.title = "AngularJS";

});

</script>

This code registers a controller function named *HelloController* in the angular module named *myapp*

The $scope parameter passed to the controller function is the *model*. The controller function adds a *helloTo* JavaScript object, and in that object it adds a*title* field.

* HTML document is loaded into the browser, and evaluated by the browser. AngularJS JavaScript file is loaded, the angular *global* object is created. Next, JavaScript which registers controller functions is executed.
* Next AngularJS scans through the HTML to look for AngularJS apps and views. Once view is located, it connects that view to the corresponding controller function.
* Next, AngularJS executes the controller functions. It then renders the views with data from the model populated by the controller. The page is now ready.

 consists of following three important parts −

* **ng-app** − This directive defines and links an AngularJS application to HTML.
* **ng-model** − This directive binds the values of AngularJS application data to HTML input controls.
* **ng-bind** − This directive binds the AngularJS Application data to HTML tags.

**My first angularjs app**



## How AngularJS integrates with HTML

* **ng-app directive** indicates the start of AngularJS application. It automatically initializes or bootstraps the application when web page containing AngularJS Application is loaded. It is also used to load various AngularJS modules in AngularJS Application.
* **ng-model directive** then creates a model variable named "name" which can be used with the html page and within the div having ng-app directive.
* **ng-bind** **directive** then uses the name model to be displayed in the html span tag whenever user input something in the text box.
* **ng-repeat directive** repeats html elements for each item in a collection.

It is use like foreach loop.

<ol>

<li ng-repeat = "country in countries">

{{ 'Country: ' + country.name + ', Locale: ' + country.locale }}

</li>

</ol>

Where countries is list of objects & country is single reference to it.

* **ng-init directive** initializes an AngularJS Application data. It is used to put values to the variables to be used in the application. Json is placesd to initialize data in ng-init.
* **ng-controller directive** defines controller. Each controller accepts $scope as a parameter which refers to the application/module that controller is to control.

Example:

|  |
| --- |
| <script>  var mainApp = angular.module("mainApp", []);  mainApp.controller('studentController', function($scope) {  $scope.student = {  firstName: "Mahesh",  lastName: "Parashar",    fullName: function() {  var studentObject;  studentObject = $scope.student;  return studentObject.firstName + " " + studentObject.lastName;  }  };  });  </script> |

studentController defined as a JavaScript object with $scope as argument. $scope refers to application which is to use the studentController object.

$scope.student is property of studentController object. firstName and lastName are two properties of $scope.student object. We've passed the default values to them.

fullName is the function of $scope.student object whose task is to return the combined name.

we can also define the controller object in separate JS file and refer that file in the html page.

## ng-disabled directive is use to bind DOM object. It is use to disable control.

## ng-show show control.

## ng-hide hides a given control.

## ng-click represents a AngularJS click event.

Expression:

Expressions are used to bind application data to html. Expressions are written inside double braces like {{ expression}}. Expressions behaves in same way as ng-bind directives. AngularJS application expressions are pure javascript expressions

Using numbers:

<p>Expense on Books : {{cost \* quantity}} Rs</p>

Using string

<p>Hello {{student.firstname + " " + student.lastname}}!</p>

Using object

<p>Roll No: {{student.rollno}}</p>

Using array

<p>Marks(Math): {{marks[3]}}</p>

Filters:

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Name** | **Description** |
| 1 | uppercase | converts a text to upper case text. |
| 2 | lowercase | converts a text to lower case text. |
| 3 | currency | formats text in a currency format. |
| 4 | filter | filter the array to a subset of it based on provided criteria. |
| 5 | orderby | orders the array based on provided criteria. |

Modules:

AngularJS supports modular approach. Modules are used to separate logics say services, controllers, application etc. and keep the code clean.

* **Application Module** − used to initialize an application with controller(s).
* **Controller Module** − used to define the controller.

mainApp.controller("studentController", function($scope) {

Here we've declared a controller **studentController** module using mainApp.controller function.

Forms:

ng\_click: to run any validation on form. $dirty & $invalid flag to return validation result. Novalidation to disable any browser based validation.

Following are supported events in Angular JS.

* ng-click
* ng-dbl-click
* ng-mousedown
* ng-mouseup
* ng-mouseenter
* ng-mouseleave
* ng-mousemove
* ng-mouseover
* ng-keydown
* ng-keyup
* ng-keypress
* ng-change

Includes:

HTML does not support embedding html pages within html page. To achieve this functionality following ways are used −

Using AngularJS, we can embedded HTML pages within a HTML page using ng-include directive.

<div ng-app = "" ng-controller = "studentController">

<div ng-include = "'main.htm'"></div>

<div ng-include = "'subjects.htm'"></div>

</div>

View:

## ng-view

ng-view tag simply creates a place holder where a corresponding view (html or ng-template view) can be placed based on the configuration.

<div ng-app = "mainApp">

...

<div ng-view></div>

</div>

## ng-template

ng-template directive is used to create an html view using script tag. It contains "id" attribute which is used by $routeProvider to map a view with a controller.

<div ng-app = "mainApp">

...

<script type = "text/ng-template" id = "addStudent.htm">

<h2> Add Student </h2>

{{message}}

</script>

</div>

## $routeProvider

$routeProvider is the key service which set the configuration of urls, map them with the corresponding html page or ng-template, and attach a controller with the same.

Note: it is use as routing.comfig in MVC .net.

var mainApp = angular.module("mainApp", ['ngRoute']);

mainApp.config(['$routeProvider', function($routeProvider) {

$routeProvider.

when('/addStudent', {

templateUrl: 'addStudent.htm', controller: 'AddStudentController'

}).

when('/viewStudents', {

templateUrl: 'viewStudents.htm', controller: 'ViewStudentsController'

}).

otherwise({

redirectTo: '/addStudent'

});

}]);

$routeProvider is defined as a function under config of mainApp module using key as '$routeProvider'.

$routeProvider.when defines a url "/addStudent" which then is mapped to "addStudent.htm". addStudent.htm should be present in the same path as main html page.If htm page is not defined then ng-template to be used with id="addStudent.htm". We've used ng-template.

Note: to work with view we also need to include new library from angularjs

<script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular-route.min.js"></script>

Scope:

Scope joins controller to view.

$scope is passed as first argument to controller during its constructor definition.

We can define functions as well in $scope.

## Scope Inheritance

Scope are controllers specific. If we defines nested controllers then child controller will inherit the scope of its parent controller.

Services:

Services are javascript functions and are responsible to do a specific tasks only. These are individual testable tasks. Services are injucted using dependency injections. AngularJS provides many inbuilt services for example, $http, $route, $window, $location etc. inbuilt services are always prefix as $

There are two ways to create a service.

* factory
* service

Using factory method, we first define a factory and then assign method to it.

var mainApp = angular.module("mainApp", []);

mainApp.factory('MathService', function() {

var factory = {};

factory.multiply = function(a, b) {

return a \* b

}

return factory;

});

Using service method, we define a service and then assign method to it

mainApp.service('CalcService', function(MathService){

this.square = function(a) {

return MathService.multiply(a,a);

}

});

Dependency injection: is a software design pattern in which components are given their dependencies instead of hard coding them within the component. This helps in making components reusable, maintainable and testable.