* **What is SPA (Single Page Application)**

**ngRoute is Not Included by Default**

**AngularJS Route** uses **HashBang**-ed URLs, For example a simple URL looks like http://www. AngularjsTutorial.com/first/page. Whereas In a Single Page AngularJS Application  the URL usually looks like, http://www. AngularjsTutorial.com/#/first/page. Browser treats URL with **#** differently than standard URL and hence an addition request will not be made to the server again.

Steps to enable routing functionality into your application

Step 1 : Include **angular-routes.js**

Step 2: Now we need to add the **‘ngRoute’** as a dependency to our module before we perform anything using angular routes

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

Step 3: Next step is to define which place of the Single Page Application should change when the route changes, this can be done by defining the **‘ng-view’** directive.

**Note :** There can be only one **ng-view** component per application, if we need more views to be defined then we need to go for **ui-router.**

4. Now We need define our routes in AngularJS **config** section using the **$routeProvider**. Angular Config consist of the function definition of the dependency which is injected. Our **$routeProvider** has two methods  
  
1.**when(path, route)**

**Path**: The **Path** is a url or a URL regex specifies when this particular should be active  
**Route** : The second is the configuration object which specifies what needs to performed when the particular route is called.

2.**otherwise(params)** : **otherwise** function on the**$routeProvider** specifies what AngularJS needs to do if the user tries to go to a URL that is not specified in the route configuration. You could see that we have added **redirectTo** key which redirects the unknown URL to the mentioned URL.

routeProvider.when(url, {

template: string,

templateUrl: string,

controller: string, **function** or array,

controllerAs: string,

resolve: object<key, **function**>

});

1. **template**: If the HTML which you want to display is **not very large** then it can be in-lined as a string to the template key. AngularJS directly inserts this template HTML into the **ng-view** directive.
2. **templateUrl**: When the HTML is complex, we need to separate it into individual files and give the URL to the file in the templateUrl. AngularJS loads the separate file from the server which needs to be displayed for the particular route. One important thing is that for the first request only it loads the file from the server, further request are loaded from the local cache.
3. **controller**: controller is an optional argument in **.when** function. It can be defined in two ways
   1. If we have already defined a controller using **ng-controller**, then we can just specify the name of the controller as a string.
   2. The other option is we can declare a controller in-line like below

$routeProvider.when('/Spring, {

template: 'This is the Spring Page,

controller: ['$log', function($log) {

$log.log('Loading Spring Page!');

}]

});

1. **controllerAs** : **controllerAs** can be used when don’t want to name the controller using **‘as’** syntax. The below two route definitions are the same.

$routeProvider.when('/Spring’, {

template: 'This is the Spring Page ',

controller: 'SpringCtrl as ctrl'

});

$routeProvider.when('/Spring’, {

template: 'This is the Spring Page ‘,

controllerAs: 'ctrl'

});

1. **redirectTo** : There are cases when the user keyed in the wrong url or page which got renamed we cannot show the user a **404** page instead we can redirect him to the index page in those cases we can use **redirectTo** option.

$routeProvider.when('/asdf', {

redirectTo: ‘/index’

});

**AngularJS Routing Example using ngRoute**

July 5, 2016 by **[javainterviewpoint](http://www.javainterviewpoint.com/author/javainterviewpoint/)** [**Leave a Comment**](http://www.javainterviewpoint.com/angularjs-routing-example-using-ngroute/#respond)

In this article lets learn about a very useful and most used feature of AngularJS called **Routing**. **[AngularJS](https://www.javainterviewpoint.com/category/angularjs/)**  routes enable us to implement multiple views in a [**Single Page Application**](http://www.javainterviewpoint.com/angularjs-form-validation-ngmessages/)**(SPA)**. Usually in a Single Page Application we always want to avoid the reload of an entire page. We only want to load the relevant data and HTML snippet instead of fetching the entire HTML again and again. When we are using ngRoute of AngularJS the browser does not make any additional requests

**AngularJS Route** uses **HashBang**-ed URLs, For example a simple URL looks like http://www. AngularjsTutorial.com/first/page. Whereas In a Single Page AngularJS Application  the URL usually looks like, http://www. AngularjsTutorial.com/#/first/page. Browser treats URL with **#** differently than standard URL and hence an addition request will not be made to the server again.

**ngRoute is Not Included by Default**

The **$routeProvider** is defined in a module called **ngRoute** which is not present in the current version of **angular.js**. prior to **AngularJS 1.1.6**, the **ngRoute** module was used to be shipped out of the box. But all the subsequent versions don’t include **ngRoute** by default. That’s why we need to include additional **angular-route.js**

Lets take a quick look into AngularJs Route example before getting into the detailed explanation.

**AngularJS Routing Example**

<!DOCTYPE html>

<html ng-app="routingApp">

<head>

<title>AngularJS Routing</title>

<script

src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.19/angular.js"></script>

<script

src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.19/angular-route.js"></script>

</head>

<body>

<ul>

<li><a href="#/">Default Route</a></li>

<li><a href="#/Spring">Spring</a></li>

<li><a href="#/Angularjs">Angularjs</a></li>

<li><a href="#/Unknown">Unknown</a></li>

</ul>

<div ng-view></div>

<script type="text/javascript">

angular

.module('routingApp', [ 'ngRoute' ])

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

{

$routeProvider

.when('/',

{

template : 'This is Default page'

})

.when('/Spring',

{

template : 'This is Spring page'

})

.when('/Angularjs',

{

template : 'This is Angularjs page'

}).otherwise({

redirectTo : '/'

});

}]);

</script>

</body>

</html>

1. As a first step we have included the Javascript file of the angular routing module (**angular-routes.js**)
2. Now we need to add the **‘ngRoute’** as a dependency to our module before we perform anything using angular routes
3. Next step is to define which place of the Single Page Application should change when the route changes, this can be done by defining the **‘ng-view’** directive.

**Note :** There can be only one **ng-view** component per application, if we need more views to be defined then we need to go for **ui-router.**

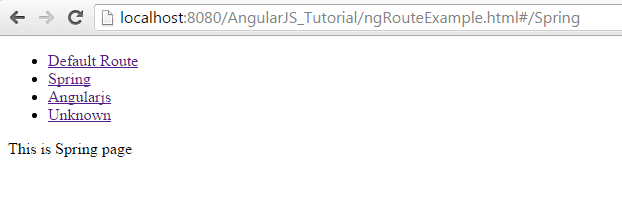
4. Now We need define our routes in AngularJS **config** section using the **$routeProvider**. Angular Config consist of the function definition of the dependency which is injected. Our **$routeProvider** has two methods

1.**when(path, route)**

**Path**: The **Path** is a url or a URL regex specifies when this particular should be active  
**Route** : The second is the configuration object which specifies what needs to performed when the particular route is called.

2.**otherwise(params)** : **otherwise** function on the**$routeProvider** specifies what AngularJS needs to do if the user tries to go to a URL that is not specified in the route configuration. You could see that we have added **redirectTo** key which redirects the unknown URL to the mentioned URL.

**Output :**

**[](http://javainterviewpoint-7ac9.kxcdn.com/wp-content/uploads/2016/07/AngularJS-Route-Example.png)**

**Routing Configurations :**

In the above code we saw a very simple route configuration. We simply loaded different templates for different route calls. The **.when** method **$routeProvider** can the below different options which can be specified.

$routeProvider.when(url, {

template: string,

templateUrl: string,

controller: string, **function** or array,

controllerAs: string,

resolve: object<key, **function**>

});

1. **template**: If the HTML which you want to display is **not very large** then it can be in-lined as a string to the template key. AngularJS directly inserts this template HTML into the **ng-view** directive.
2. **templateUrl**: When the HTML is complex, we need to separate it into individual files and give the URL to the file in the templateUrl. AngularJS loads the separate file from the server which needs to be displayed for the particular route. One important thing is that for the first request only it loads the file from the server, further request are loaded from the local cache.
3. **controller**: controller is an optional argument in **.when** function. It can be defined in two ways
   1. If we have already defined a controller using **ng-controller**, then we can just specify the name of the controller as a string.
   2. The other option is we can declare a controller in-line like below

$routeProvider.when('/Spring, {

template: 'This is the Spring Page,

controller: ['$log', function($log) {

$log.log('Loading Spring Page!');

}]

});

1. **controllerAs** : **controllerAs** can be used when don’t want to name the controller using **‘as’** syntax. The below two route definitions are the same.

$routeProvider.when('/Spring’, {

template: 'This is the Spring Page ',

controller: 'SpringCtrl as ctrl'

});

$routeProvider.when('/Spring’, {

template: 'This is the Spring Page ‘,

controllerAs: 'ctrl'

});

1. **redirectTo** : There are cases when the user keyed in the wrong url or page which got renamed we cannot show the user a **404** page instead we can redirect him to the index page in those cases we can use **redirectTo** option.

$routeProvider.when('/asdf', {

redirectTo: ‘/index’

});

**AngularJS $routeParams Example**

There are times we need pass the parameter to the controller, for example we need to pass **message** or **id** in the URL which the controller uses for a Ajax call. Those parameters can be accessed by using the **$routeParams**.

Redirect to particular url from controller use $location.path('/login');