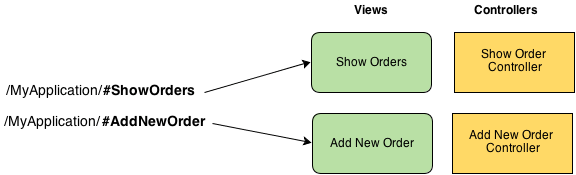
**AngularJS Routing And Views Tutorial With Example**

Welcome to the next tutorial of the ongoing series of tutorials on AngularJS. In previous tutorial we saw[AngularJS Controllers](http://viralpatel.net/blogs/angularjs-controller-tutorial/) and also created a Hello World example using Angular.

In this article we will go through the next useful feature of AngularJS called Routing. Also we will see how we can divide a single page application in multiple views. As we add more and more logic to an app, it grows and soon become difficult to manage. Dividing it in Views and using Routing to load different part of app helps in logically dividing the app and making it more manageable.

Routing helps you in dividing your application in logical views and bind different views to Controllers.



In above diagram we create two Route url /ShowOrders and /AddNewOrder. Each points to a specific view and is managed by a controller. Don’t panic if it does not make any sense. Soon we will see some code and it all will be clear.

**1. Introduction to $routeProvider**

The magic of Routing is taken care by a service provider that Angular provides out of the box called $routeProvider. An [Angular service](http://viralpatel.net/blogs/angularjs-service-factory-tutorial/) is a singleton object created by a service factory. These service factories are functions which, in turn, are created by a service provider. The service providers are constructor functions. When instantiated they must contain a property called $get, which holds the service factory function.

When we use AngularJS’s dependency injection and inject a service object in our Controller, Angular uses $injector to find corresponding service injector. Once it get a hold on service injector, it uses $get method of it to get an instance of service object. Sometime the service provider needs certain info in order to instantiate service object.

Application routes in Angular are declared via the $routeProvider, which is the provider of the $route service. This service makes it easy to wire together controllers, view templates, and the current URL location in the browser. Using this feature we can implement deep linking, which lets us utilize the browser’s history (back and forward navigation) and bookmarks.

**Syntax to add Routing**  
 Below is the syntax to add routing and views information to an angular application. We defined an angular app “sampleApp” using angular.module method. Once we have our app, we can use config()method to configure $routeProvider. $routeProvider provides method .when() and .otherwise() which we can use to define the routing for our app.

|  |
| --- |
| var sampleApp = angular.module('phonecatApp', []);    sampleApp .config(['$routeProvider',    function($routeProvider) {      $routeProvider.        when('/addOrder', {          templateUrl: 'templates/add-order.html',          controller: 'AddOrderController'        }).        when('/showOrders', {          templateUrl: 'templates/show-orders.html',          controller: 'ShowOrdersController'        }).        otherwise({          redirectTo: '/addOrder'        });    }]); |

In above code we defined two urls /addOrder and /showOrders and mapped them with viewstemplates/add-order.html and templates/show-orders.html respectively. When we open http://app/#addOrder url in browser, Angular automatically matches it with the route we configures and load add-order.html template. It then invokes AddOrderController where we can add logic for our view.

**1.1. Hello World AngularJS + Routing**

Let us go through an example in AngularJS and use Routing to load different templates at runtime.

Below sample1.html file is the main html file. It includes AngularJS library and define structure for our app. We have two links: Add New Order and Show Order. Each link loads template in below section.  
*sample1.html*

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">    <head>      <title>AngularJS Routing example</title>    </head>      <body ng-app="sampleApp">        <div class="container">          <div class="row">          <div class="col-md-3">              <ul class="nav">                  <li><a href="#AddNewOrder"> Add New Order </a></li>                  <li><a href="#ShowOrders"> Show Order </a></li>              </ul>          </div>          <div class="col-md-9">              <div ng-view></div>          </div>          </div>      </div>        <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.0.7/angular.min.js"></script>      <script src="app.js"></script>      </body>  </html> |

**ng-view**

One thing worth noting is the ng-view directive. In our angular app, we need to define ng-app directive once. This becomes the placeholder for views. Each view referred by the route is loaded in this section of document.

You can define ng-view in main html file in one of the below way.

|  |
| --- |
| <div ng-view></div>  ..  <ng-view></ng-view>  ..  <div class="ng-view"></div> |

**1.2. Add Routing in AngularJS**

In above sample1.html file we included a javascript file app.js which holds the application logic. Below is the content of app.js.

*app.js*

|  |
| --- |
| //Define an angular module for our app  var sampleApp = angular.module('sampleApp', []);    //Define Routing for app  //Uri /AddNewOrder -> template add\_order.html and Controller AddOrderController  //Uri /ShowOrders -> template show\_orders.html and Controller AddOrderController  sampleApp.config(['$routeProvider',    function($routeProvider) {      $routeProvider.        when('/AddNewOrder', {          templateUrl: 'templates/add\_order.html',          controller: 'AddOrderController'      }).        when('/ShowOrders', {          templateUrl: 'templates/show\_orders.html',          controller: 'ShowOrdersController'        }).        otherwise({          redirectTo: '/AddNewOrder'        });  }]);      sampleApp.controller('AddOrderController', function($scope) {        $scope.message = 'This is Add new order screen';    });      sampleApp.controller('ShowOrdersController', function($scope) {        $scope.message = 'This is Show orders screen';    }); |

We first use .config() method to define $routeProvider configuration. Also in the same file we define two controllers AddOrderController and ShowOrdersController. In a real world application these controllers will hold a lot of logic but for example sake we just define a message property on $scope which later we use to display on view.

Notice how we used otherwise() method to define a default route. In case routeProvider does not matche with any url, it redirects to default route.

|  |
| --- |
| ...  otherwise ({      redirectTo: '/AddNewOrder'  }); |

**1.3. Add HTML template files**

Our app.js is ready. We still needs to define two html templates. These are partial templates of our app.

*templates/add\_order.html*

|  |
| --- |
| <h2>Add New Order</h2>    {{ message }} |

add\_order.html template should have an html form for adding new orders. For sake of simplicity we just show a message.

*templates/show\_orders.html*

|  |
| --- |
| <h2>Show Orders</h2>    {{ message }} |

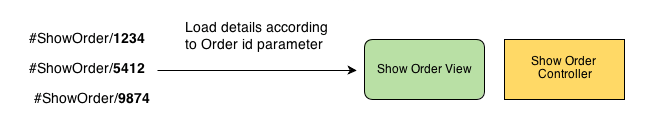
**1.4. Online Demo**

Click links in below example to load different template based on Uri.

Demo link: [Plnkr.co](http://plnkr.co/edit/geb5fh1Ns4fZC5XDOvaV?p=preview)

**2. How to pass Parameters in Route Urls**

We saw how to define route in above example. Now let us see how can we define parameters in route urls.

Consider below scenario. We want to display details of different orders. Based on a parameter order\_id we will define order details in view.  


In angular while define route we can define parameters using orderId in url. For example:

|  |
| --- |
| when('/ShowOrder/:orderId', {        templateUrl: 'templates/show\_order.html',        controller: 'ShowOrderController'  }); |

And we can read the parameter in ShowOrderController by using $routeParams.orderId.

|  |
| --- |
| ...  $scope.order\_id = $routeParams.orderId;  ... |

Let us checkout a sample application. Below sample2.html is main html page which describe the structure. It contains order information data in tabular format. Each order has a “show details” link. This link uses parametrize route url to load order detail screen.

*sample2.html*

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">    <head>      <title>AngularJS Routing example</title>    </head>      <body ng-app="sampleApp">        <div class="container">      <div class="row">      <div class="col-md-9">          <table class="table table-striped">              <thead>                <tr>                  <th>#</th><th>Order No.</th><th>Details</th><th></th>                </tr>              </thead>              <tbody>                <tr>                  <td>1</td><td>1234</td><td>15" Samsung Laptop</td>                  <td><a href="#ShowOrder/1234">show details</a></td>                </tr>                <tr>                  <td>2</td><td>5412</td><td>2TB Seagate Hard drive</td>                  <td><a href="#ShowOrder/5412">show details</a></td>                </tr>                <tr>                  <td>3</td><td>9874</td><td>D-link router</td>                  <td><a href="#ShowOrder/9874">show details</a></td>                </tr>              </tbody>            </table>            <div ng-view></div>      </div>      </div>      </div>    <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.0.7/angular.min.js"></script>  <script src="app.js"></script>      </body>  </html> |

*app.js*

|  |
| --- |
| var sampleApp = angular.module('sampleApp', []);    sampleApp.config(['$routeProvider',    function($routeProvider) {      $routeProvider.        when('/ShowOrder/:orderId', {      templateUrl: 'templates/show\_order.html',      controller: 'ShowOrderController'        });  }]);      sampleApp.controller('ShowOrderController', function($scope, $routeParams) {        $scope.order\_id = $routeParams.orderId;    }); |

**Note:** Don’t forget to inject $routeParam parameter in controller. Otherwise you wont be able to use it.

*templates/show\_order.html*

|  |
| --- |
| <h2>Order #{{order\_id}}</h2>    Here are the details for order <b>#{{order\_id}}</b>. |

**2.1 Online Demo**

Click the show details links on different orders.

Demo link: [Plnkr.co](http://plnkr.co/edit/MxJkCpJDKXDNyYc1web8?p=preview)

**3. How to Load local views (Views within <script> tag)**

It is not always that you want to load view templates from different files. Sometimes the view templates are small enough that you might want them ship with main html instead of keeping them in separate html files.

**3.1 ng-template directive**

You can use ng-template to define small templates in your html file. For example:

|  |
| --- |
| <script type="text/ng-template" id="add\_order.html">      <h2> Add Order </h2>      {{message}}  </script> |

Here we defined a template “add\_order.html” inside <script> tag. Angular will automatically load this template in ng-view whenever add\_order.html is referred in route.

Let us quickly go through a sample app where we use local view definitions.

sample3.html defines structure of app. It is similar to the first example that we saw (sample1.html) with a bit of change. We defined two views add\_order.html and show\_orders.html within sample1.html as <script> tag.  
*sample3.html*

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">    <head>      <title>AngularJS Routing example</title>      <link href="http://netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css" rel="stylesheet">    </head>      <body ng-app="sampleApp">        <div class="container">          <div class="row">          <div class="col-md-3">             <ul class="nav">              <li><a href="#AddNewOrder"> Add New Order </a></li>              <li><a href="#ShowOrders"> Show Order </a></li>              </ul>          </div>          <div class="col-md-9">              <div ng-view></div>          </div>          </div>      </div>        <script type="text/ng-template" id="add\_order.html">            <h2> Add Order </h2>          {{message}}        </script>        <script type="text/ng-template" id="show\_orders.html">            <h2> Show Orders </h2>          {{message}}        </script>          <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.0.7/angular.min.js"></script>      <script src="app.js"></script>      </body>  </html> |

The app.js is similar to first sample. There is no change in app.js. Refer to first demo example if you wanna check app.js.

**3.2 Online Demo**

Click the show details links on different orders.

Demo link: [Plnkr.co](http://plnkr.co/edit/nYsORXGahmGOVyNmYWaF?p=preview)

**4. Add Custom Data to RouteProvider**

The $routeProvider provides methods when() and otherwise() which we used to define url routes. Sometime we might want to pass custom data based on certain route. For example you might use same Controller in different routes and use some custom data. For example:

|  |
| --- |
| when('/AddNewOrder', {      templateUrl: 'templates/add\_order.html',      controller: 'CommonController',      foodata: 'addorder'  }).  when('/ShowOrders', {      templateUrl: 'templates/show\_orders.html',      controller: 'CommonController',      foodata: 'showorders'  });    sampleApp.controller('CommonController', function($scope, $route) {      //access the foodata property using $route.current      var foo = $route.current.foodata;        alert(foo);    }); |

In above code we defined a new property “foodata” while defining route. This property is then accessed in controller using $route.current.foodata.