



## Lesson Objectives

- AngularJS Routing Basics
- Understanding Routing Modes



## 5.1: AngularJS Routing

## Routing

- It is very important to navigate from one page view to another in single page application
- We can achieve this by including multiple templates in the view using ng-include directive, but this will be unmanageable and also make it difficult to allow other developers to join in the development
- We can break out the view into a layout and template views and only show the view which we want to show based upon the URL the user is accessing. Routing means loading sub-templates depending upon the URL of the page
- Routes are a way for multiple views to be used within a single HTML page. This enables your page to look more "app-like" because users are not seeing page reloads happening within the browser; but the data in the web page gets updated

To build a small **Single Page Application** (SPA) with multiple views to familiarize ,we need to know routing mechanism in angular applications.

5.1: AngularJS Routing

## AngularJS Routes

- AngularJS routes enable us to create different URLs for different content in our application. Having different URLs for different content enables the user to bookmark URLs to specific content. In AngularJS each such bookmarkable URL is called a route.
- AngularJS routes enables us to show different content depending on what route is chosen. A route is specified in the URL after the # sign
  - <http://capgemini.com/index.html#/training>

5.1: AngularJS Routing

## Setting up page for routing

- To setup a page for routing we need to follow the 4 steps given below
- AngularJS requires the route service, which is not part of the default Angular library. We need to load angular-route.js, as part of your script loading.
- We need to inject the route service in our app module
- Use ngView directive in the HTML tag (use div tag) to display the given route.
- Configure `$routeProvider` in the module's `config()` function via calls to the `when()` and `otherwise()` functions.



Copyright © Capgemini 2015. All Rights Reserved 5

**ngView** : The ngView directive is used to display the HTML templates or views in the specified routes. Every time the current route changes, the included view changes with it according to the configuration of the `$route` service.

**\$routeProvider** : The `$routeProvider` (provider in ngRoute Module) is used to configure the routes. We use the module's `config()` to configure the `$routeProvider`. The `config()` takes a function which takes the `$routeProvider` as parameter and the routing configuration goes inside the function.

`$routeProvider` has a simple API, accepting either the `when()` or `otherwise()` method.

## 5.1: AngularJS Routing

## Setting up page for routing

```
var app = angular.module('routeApp', ['ngRoute']);
app.config(function($routeProvider){
    $routeProvider
        .when('/',
            {
                template: '<h1>Home Page</h1>'
            })
        .when('/company',
            {
                template: '<h1>CAPGEMINI</h1>'
            })
        .otherwise({
            redirectTo: '/'
        })
    });
```

- When the browser loads the Angular app, it will default to the URL set as the default route. Unless we load the browser with a different URL, the default is the '/' route.



Copyright © Capgemini 2015. All Rights Reserved 6

If the `redirectTo` property is set with a string, then the value will change the path and trigger a route change to the new path.


If the `redirectTo` property is set with a function, the result of the function will be set as the value of the new path, triggering a route-change request.


If the `redirectTo` property is a function, Angular will call it with one of the following parameters:

- 1: The route parameters extracted from the current path
- 2: The current path
- 3: The current search

# Demo

- Angular-01-Router





Copyright © Capgemini 2015. All Rights Reserved 7

## 5.1: AngularJS Routing

## Routing Modes

- Routing mode refers specifically to the format of the URL in the browser address bar. It determines the look of the URL. AngularJS has 2 routing modes
- Hashbang Mode
  - The default behavior of the \$location service is to route using the hashbang mode. It provides deep-linking capabilities to Angular apps. URL paths take a prepended '#' character. We can configure hashbang mode in the config function on an app module. We can also configure the hashPrefix, which is part of the fallback mechanism that Angular uses for older browsers.

```
var app = angular.module('routeApp',['ngRoute']);
app.config(function($routeProvider,$locationProvider){
    $locationProvider.html5Mode(false);
    $locationProvider.hashPrefix('!');
});
```



5.1: AngularJS Routing

Routing Modes


■ HTML5 Mode

■ This mode makes URLs look like regular URLs (except that in older browsers they will look like the hashbang URL).

■ \$location service automatically falls back to using hashbang URLs if the browser doesn't support the HTML5 history API and also rewrites the URL.

■ For example, with the tag: <a href="/employee/36?show=true">Employee</a>, a legacy browser's URL will be rewritten to the hashbang URL equivalent: /index.html#! /employee/36?show=true

```
var app = angular.module('routeApp',['ngRoute']);
app.config(function($routeProvider,$locationProvider){
    $locationProvider.html5Mode(true);
});
```

Capgemini  
CONSULTING TECHNOLOGY ENTREPRENEUR

Copyright © Capgemini 2015. All Rights Reserved 9

The back-end server will have to support URL rewriting on the server side. To support HTML5 mode, the server will have to make sure to deliver the index.html page for all apps. That ensures that our Angular app will handle the route.

When writing links inside of our Angular app in html5mode, we'll never want to use relative links. If you are serving your app using the root, it won't be a problem; however, if you are serving in any other base route, our Angular app won't be able to handle it.

Alternatively, you can set the base URL of your app using the <base> tag in the HEAD section of the HTML document: <base href="/base/url" />

## 5.1: AngularJS Routing

## Route Parameters

- AngularJS will parse a route param with a colon (:) and pass it to \$routeParams.

```
var app = angular.module('routeApp', ['ngRoute']);
app.config(function($routeProvider){
    $routeProvider
        .when('/Employees/:id',
            {
                templateUrl: 'partials/employees.html',
                controller: 'EmployeeController'
            });
});
```

- Angular will populate the \$routeParams with the key of :id, and the value of key will be populated with the value of the loaded URL. If the browser loads the URL /Employees/714709, then the \$routeParams object will look like: {id:714709}

## 5.1: AngularJS Routing

## Route Parameters

- Parameter samples
  - '/capgemini' : Matches exactly capgemini
  - '/employee/:id' : Matches employee /714709, employee /desigan
- We can also specify parameters as query parameters following a '?'
  - '/employee/:id' : Matches employee /714709?department=training

```
var app = angular.module('routeApp', ['ngRoute']);
app.config(function($routeProvider){
    $routeProvider.when('/employee/:id',
    {
        redirectTo: function(routeParams, path, search){
            console.log(routeParams); // Object {id: "714709"}
            console.log(path);         // /employee/714709
            console.log(search);       // Object {department: "training"}
            return "/";
        }
    })
});
```

## 5.1: AngularJS Routing


## \$routeParams


- The \$routeParams service allows you to retrieve the current set of route parameters. Controller functions can get access to route parameters via the AngularJS \$routeParams service

```
var app = angular.module('routeApp',['ngRoute']);
app.config(function($routeProvider){
    $routeProvider
        .when('/:company',
            {
                templateUrl:'partials/map.html',
                controller:'RouteController'
            })
});
app.controller("RouteController",function($scope,$routeParams){
    $scope.model = $routeParams.company;
});
```

# Demo

- Lesson05-Route





Copyright © Capgemini 2015. All Rights Reserved 13

## Summary

- \$routeProvider service use to create routes.
- otherwise \$routeProvider function allows us to set a default route.
- Using \$routeParams service we can access the parameters passed on a route.



Add the notes here.

## Review Question

- How to display the given route?
  - ng-view
  - ng-show
  - ng-init
  - ng-app
  
- How to inject the route service in our app module?
  - ngController
  - ngService
  - ngRoute

