عنوان: **مسیریابی در AngularJs #بخش سوم**
نویسنده: حمید صابری
تاریخ: ۱۳۹۲/۱۰/۲۱ ۰:۱۰
آدرس: www.dotnettips.info
گروه‌ها: AngularJS

در بخش‌های پیشین (بخش اول و بخش دوم) به خوبی با اصول و روش مسیریابی (Routing) در AngularJS آشناشدیم. در این بخش می‌خواهم به برخی جزئیات درباره مسیریابی بپردازم. اولین موضوع، تغییراتی است که از نسخه 1.2 به بعد در روش استفاده از سرویس مسیریابی در AngularJS بوجود آمده است. از نسخه 1.2 سرویس مسیریابی از هسته اصلی AngularJS خارج شد و برای استفاده از امکانات این سرویس باید فایل angular-route.js و یا angular-route.min.js را به صفحه خود بیفزاییم:

```
<script src="~/Scripts/angular.min.js"></script>
<script src="~/Scripts/angular-route.min.js"></script>
```

سپس باید هنگام تعریف ماژول، ngRoute را به عنوان وابستگی تزریق کنیم:

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

روش Controller as در AngularJS که از نسخه 1.2 به بعد امکان استفاده از آن وجود دارد قبلا معرفی شده است. با پاس کردن خصوصیت controllerAs به متد when می‌توان از viewای استفاده کرد که در آن از این روش استفاده شده است.

```
.when('/controllerAS', {
    controller:     'testController',
    controllerAs:   'tCtrl',
    template:       '<div>{{tCtrl.Title}}</div>'
})
```

باقی ماجرا مانند گذشته است.
موضوع دیگری که پرداختن به آن می‌تواند مفید باشد، بررسی بیشتر متد when است. وقتی در متد config ماژول از $routeProvider استفاده می‌کنیم، داریم سرویس $route را تنظیم، مقداردهی اولیه، و نمونه گیری می‌کنیم. درواقع با استفاده از متدهای when و otherwise داریم سرویس $route را مقداردهی اولیه می‌کنیم (برای آشنایی با تفاوت factory، service و provider کلیک کنید). خوب! جریان این مقادیری که به عنوان پارامتر به این متدها پاس می‌کنیم چیست؟
متد when به این صورت تعریف شده است:

```
when(string path, object route)
```

پارامتر path در بخش‌های قبل به اندازه کافی معرفی شده است. پارامتر route یک شی است شامل اطلاعاتی که با تطبیق آدرس صفحه با پارامتر path، به $route.current مقداردهی می‌شود (حالا باید متوجه شده باشید که روال افزودن داده‌های سفارشی به سیستم مسیریابی و دسترسی به آن‌ها که در بخش دوم مطرح شد به چه شکل کار می‌کند). این شی می‌تواند خصوصیات از قبل تعریف شده‌ای داشته باشد که در ادامه آن‌ها را مرور می‌کنیم: **controller:** می‌تواند یک رشته شامل نام کنترلر از قبل تعریف شده، یا یک تابع به عنوان تابع کنترلر باشد. **controllerAs:** رشته‌ای شامل نام مستعار کنترلر. **template:** رشته‌ای شامل قالب html، و یا تابعی که قالب html را بازمی‌گرداند. این خصوصیت بر templateUrl اولویت دارد. اگر مقدار این خصوصیت یک تابع باشد، $routeParams به عنوان پارامتر ورودی به آن پاس می‌شود. **templateUrl:** رشته‌ای شامل مسیر فایل قالب html، و یا تابعی که این رشته را بازمی‌گرداند. اگر مقدار این خصوصیت یک تابع باشد، $routeParams به عنوان پارامتر ورودی به آن پاس می‌شود. **redirectTo:** مقداری برای به روز رسانی $location، و فراخوانی روال مسیر یابی. این مقدار می‌تواند یک رشته، و یا تابعی که یک رشته را بازمی‌گرداند باشد. اگر مقدار این خصوصیت یک تابع باشد، این پارامترها به آن پاس می‌شود:
$routeParams برای دسترسی به پارامترهای آمده در آدرس صفحه جاری.

$location.path() جاری به صورت یک رشته.

()$location.search جاری به صورت یک شی.

caseInsensitiveMatch: یک مقدار منطقی است که مشخص می‌کند بزرگ و کوچک بودن حروف در تطبیق آدرس صفحه با پارامتر route در نظر گرفته بشود یا نه. مقدار پیشفرض این خصوصیت false است. یعنی در همه مثالهایی که تا کنون زده شده، اگر بزرگ و کوچه بودن حروف آدرس صفحه با مقدار مشخص شده برای پارامتر route متفاوت باشد، روال مسیریابی انجام نخواهد شد. برای رفع این مشکل کافی است مقدار این خصوصیت را true قرار دهیم. برای مثال، مسیر 'controllerAS/' که بالاتر تعریف کرده‌ایم را درنظر بگیرید. اگر www.mySite.com/#/ControllerAS را وارد کنیم، هیچ اتفاقی نخواهد افتاد و در واقع این آدرس با route مشخص شده تطبیق پیدا نمی‌کند. اگر بخواهیم کوچک و بزرگ بودن حروف در نظر گرفته نشود، کافیست به این ترتیب عمل کنیم:

```
.when('/controllerAS', {
    controller:      'testController',
    controllerAs:    'tCtrl',
    template:        '<div>{{tCtrl.Title}}</div>',
    caseInsensitiveMatch: true
})
```

resolve: نگاشتی از وابستگی‌هایی که می‌خواهیم به کنترلر تزریق شود. قبلا مفهوم promise توضیح داده شده است . اگر هر یک از این وابستگی‌ها یک promise باشد، مسیریاب تا resolve شدن همه آن‌ها یا reject شدن یکی از آن‌ها منتظر می‌ماند. در صورتی که همه promiseها resolve شوند، رخداد $routeChangeSuccess، و در صورتی که یکی از آن‌ها reject شود رخداد $routeChangeError اجرا می‌شود. یکی از کاربردهای resolve زمانیست که بخواهید جلوی تغییر محتویات صفحه، پیش از بارگذاری داده‌ای که از سمت سرور درخواست کرده‌اید را بگیرید.

```
$routeProvider
    .when('/resolveTest',
        {
            resolve: {
                // این وابستگی بلافاصله بازمی‌گردد
                person: function () {
                    return {
                        name: "Hamid Saberi",
                        email: "Hamid.Saberi@Gmail.com"
                    }
                },
                // این وابستگی یک promise بازمی‌گرداند
                // پس تغییر مسیر تا resolve شدن آن به تاخیر می‌افتد
                currentDetails: function ($http) {
                    return $http({
                        method: 'Get',
                        url: '/current_details'
                    });
                },
                //می‌توانیم از یک وابستگی در وابستگی دیگر استفاده کنیم
                facebookId: function ($http, currentDetails) {
                    $http({
                        method: 'GET',
                        url: 'http://facebook.com/api/current_user',
                        params: {
                            email: currentDetails.data.emails[0]
                        }
                    })
                },
                // بارگذاری فایل‌های اسکریپت مورد نیاز
                fileDeps:function($q, $rootScope){
                    var deferred = $q.defer();
                    var dependencies =
                    [
                        'controllers/AboutViewController.js',
                        'directives/some-directive.js'
                    ];

                    //$Script.js بارگذاری وابستگی‌ها با استفاده از
                    $script(dependencies, function(){
                        // همه وابستگی‌ها بارگذاری شده اند
                        $rootScope.$apply(function(){
                            deferred.resolve();
                        });
                    });

                    return deferred.promise;
```

```
                }
            },
            controller: function ($scope, person, currentDetails, facebookId) {
                this.Person = person;
            },
            controllerAs: 'rtCtrl',
            template: '<div>{{rtCtrl.Person.name}}</div>',
            caseInsensitiveMatch: true
        })
```

نظرات خوانندگان

نویسنده: مسعود رمضانی
تاریخ: ۱۳۹۲/۱۲/۲۰ ۱۱:۰

دوست عزیز خیلی مطلب مفیدی بود.

ممنونم :)