بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

محدوده (Scope) در AngularJS

مدرس : مهندس افشین رفوآ

دوره آموزشیAngularJS

محدوده (Scope) در AngularJS

Scope را می توان به چند طریق تعریف کرد:

به قسمتی که HTML (view) و JavaScript (Controller) را بهم پیوند می دهد ، Scope می گویند.

به یک object با ویژگی ها و متدهای قابل دسترس نیز Scope می گویند. Scope هم برای view و هم برای Controller قابل دسترس می باشد.

چگونه از Scope استفاده کنیم؟

زمانی که شما یک Controller در AngularJS ایجاد می کنید، یک object `$scope` به عنوان argument ایجاد می شود. در مثال زیر، ویژگی های ساخته شده در Controller، می توانند به view ارجاع داده شوند.

```
<!DOCTYPE html>
<html>
<head>
    <title>داده تحلیل آموزشگاه</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
</head>
<body>
    <div ng-app="myApp" ng-controller="myCtrl">
        <h1>{{carname}}</h1>
    </div>
    <script>
        var app = angular.module('myApp', []);
        app.controller('myCtrl', function ($scope) {
            $scope.carname = "Volvo";
        });
    </script>
```

```
    <p>The property "carname" was made in the controller, and can be referred to in the view
by using the {{ }} brackets.</p>
</body>
</html>
```

هرگاه به `$scope` در Controller ویژگی هایی اضافه کنیم، view(HTML) می تواند به این ویژگی ها دسترسی داشته باشد. شما نباید از پیشوند `$scope` در view استفاده کنید بلکه فقط باید به نام آن ویژگی ارجاع دهید، به عنوان مثال باید به صورت `{{carname}}` بنویسید.

درک کردن Scope

یک برنامه کاربردی AngularJS شامل بخش های زیر می باشد:

- View که شامل کدهای Html است.
- Model که view مربوط به آن دسترسی دارد.
- Controller که با استفاده از توابع JavaScript، اطلاعات را ایجاد کند، تغییر دهد، حذف کند و یا کنترل کند.

Scope یک object با ویژگی ها و متدهایی است که هم برای view و Controller قابل دسترسی باشد. به عنوان مثال، هر گاه شما در view تغییراتی ایجاد کنید، model و Controller به روز رسانی می شوند.

```
<!DOCTYPE html>
<html>
<head>
    <title>داده تحلیل آموزشگاه</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
</head>
<body>
    <div ng-app="myApp" ng-controller="myCtrl">
        <input ng-model="name">
        <h1>My name is {{name}}</h1>
    </div>
    <script>
        var app = angular.module('myApp', []);
        app.controller('myCtrl', function ($scope) {
            $scope.name = "John Doe";
        });
    </script>
    <p>When you change the name in the input field, the changes will affect the model, and it
will also affect the name property in the controller.</p>
</body>
</html>
```

Scope خود را بشناسید

این نکته بسیار مهم است که شما بدانید از کدام Scope باید استفاده کنید. در دو مثال قبلی تنها یک Scope وجود داشت و شناختن Scope کار دشواری نبود اما برای برنامه های بزرگتر Section هایی از HTML DOM وجود دارد که برای هر Scope یکتای خود قابل دسترسی می باشد.

به عنوان مثال، هرگاه از دایرکتیو `ng-repeat` استفاده می کنیم، هر تکرار به همان object تکرار شده دسترسی دارد.

```
<!DOCTYPE html>
<html>
<head>
    <title>داده تحلیل آموزشگاه</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
</head>
<body>
    <div ng-app="myApp" ng-controller="myCtrl">
        <ul>
            <li ng-repeat="x in names">{{x}}</li>
        </ul>
    </div>
    <script>
        var app = angular.module('myApp', []);
        app.controller('myCtrl', function ($scope) {
            $scope.names = ["Emil", "Tobias", "Linus"];
        });
    </script>
    <p>The variable "x" has a different value for each repetition, proving that each
repetition has its own scope.</p>
</body>
</html>
```

هر المان `<li>` به همان object تکرارشونده دسترسی دارد. در این مورد یک رشته با x نشان داده شده است.

Scope ریشه

هر برنامه ی کاربردی، یک `$rootScope` دارد که Scope ساخته شده در المان Html آن دارای دایرکتیو `ng-app` می باشد. rootScope در کل برنامه قابل دسترسی است و اگر یک متغیر دارای نام های مشابه در Current Scope و rootScope باشد، برنامه یکی از آنها را استفاده می کند.

به عنوان مثال، یک متغیر به نام color هم در Controller Scope و هم در rootScope وجود دارد.

```
<!DOCTYPE html>
```

```
<html>
<head>
    <title>داده تحلیل آموزشگاه</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
</head>
<body ng-app="myApp">
    <p>The rootScope's favorite color:</p>
    <h1>{{color}}</h1>
    <div ng-controller="myCtrl">
        <p>The scope of the controller's favorite color:</p>
        <h1>{{color}}</h1>
    </div>
    <p>The rootScope's favorite color is still:</p>
    <h1>{{color}}</h1>
    <script>
        var app = angular.module('myApp', []);
        app.run(function ($rootScope) {
            $rootScope.color = 'blue';
        });
        app.controller('myCtrl', function ($scope) {
            $scope.color = "red";
        });
    </script>
    <p>Notice that controller's color variable does not overwrite the rootScope's color
value.</p>
</body>
</html>
```