عنوان: **توسعه سرویس‌های مبتنی بر REST در AngularJs با استفاده از RestAngular : بخش اول**
نویسنده: مهرداد کاهه
تاریخ: ۱۳۹۴/۰۷/۱۵ ۱۷:۲۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, AngularJS

برای مطالعه‌ی این مقاله شما باید به مواردی از قبیل کتابخانه‌ی AngularJs ، تعاملات بین کلاینت و سرور و همچنین معماری RESTful تسلط کافی داشته باشید و ما از توضیح و تفصیلی این سرفصل‌ها اجتناب میکنیم.
خیلی خوب بپردازیم به اصل مطلب:

Restangular چیست؟

کتابخانه RestAngular بنا به گفته ناشر در مستندات Github آن، یک سرویس توسعه داده شده AngularJs می‌باشد که کدهای نوشته شده‌ی برای پیاده سازی فرآیندهای Request/Response کلاینت و سرور را به حداقل رسانده است. این فرآیندها در قالب درخواست‌های GET، POST، DELETE و Update صورت می‌پذیرند. این سرویس برای وب اپلیکیشن‌هایی که که داده‌های خود را از APIهای RESTful همانند Microsoft ASP.NET Web API 2 دریافت می‌کنند نیز بسیار کارآمد است.
کد زیر یک فرآیند درخواست GET را نمایش می‌دهد که در آن از سرویس RestAngular استفاده شده است:

```
// Restangular returns promises
Restangular.all('users').getList()  // GET: /users
.then(function(users) {
  // returns a list of users
  $scope.user = users[0]; // first Restangular obj in list: { id: 123 }
})

// Later in the code...

// Restangular objects are self-aware and know how to make their own RESTful requests
$scope.user.getList('cars');  // GET: /users/123/cars

// You can also use your own custom methods on Restangular objects
$scope.user.sendMessage();  // POST: /users/123/sendMessage

// Chain methods together to easily build complex requests
$scope.user.one('messages', 123).one('from', 123).getList('unread');
// GET: /users/123/messages/123/from/123/unread
```

همانطور که ملاحظه میکنید تمام پروسه‌ی دریافت، در یک خط خلاصه شده است. همچنین می‌بینید که restAngular دارای Promise نیز هست. در تکه کد اول، تمامی userها به صورت Get دریافت می‌شوند. در تکه کد دوم مشاهده می‌کنید که عملیات درخواست لیست ماشین‌های کاربر چگونه در یک خط خلاصه می‌گردد. حال فرض کنید قصد داشتیم تا این عملیات را با وابستگی http پیاده سازی نماییم. برای این کار باید چندین خط کد را درون یک سرویس جا می‌دادیم و آنگاه از متد سرویس در کنترلر استفاده می‌کردیم.
در اینجا همچنین قادرید درخواست‌های خود را به صورت سفارشی نیز اعمال کنید (تکه کد سوم) و در آخر مشاهده می‌کنید که پیچیده‌ترین عملیات‌های درخواست کاربر را می‌توان در یک خط خلاصه نمود. برای نمونه کد آخر یک دستور GET تو در تو را به چه سادگی پیاده سازی کرده است.

وابستگی ها

باید توجه داشته باشید که برای استفاده از RestAngular نیاز به کتابخانه‌ی Lodash نیز می‌باشد.

شروع پروژه

برای شروع پروژه و استفاده از RestAngular، پس از اضافه کردن اسکریپت رفرنس‌ها و وابستگی‌ها (lodash و AngularJs) باید وابستگی restAngular را به صورت زیر به angular.module اضافه نمایید:

```
// Add Restangular as a dependency to your app
angular.module('your-app', ['restangular']);

// Inject Restangular into your controller
angular.module('your-app').controller('MainCtrl', function($scope, Restangular) {
  // ...
```

```
});
```

توجه داشته باشید هنگامیکه یک وابستگی به module اضافه می‌گردد، با حروف کوچک یعنی restangular اضافه می‌گردد؛ اما هنگامیکه به کنترلر اضافه می‌شود، به صورت Restangular و با R بزرگ اضافه می‌گردد.

برخی از متدهای RestAngular

در این بخش قصد داریم تا برخی از پر کاربردترین متدهای RestAngular را تشریح کنیم:

نام متد	پارامترهای ارسالی	توضیحات
one	route, id	این متد یک RestAngular object ایجاد میکند که از آدرسی که در route قرار داده شده با id مشخص دریافت میشود.
all	route	این متد یک RestAngular object که لیستی از المنت هایی را که در آدرس route قرار دارد، دریافت می‌نماید.
oneUrl	route, url	این متد یک RestAngular object ایجاد می‌کند که یک المنت از url خاصی را بازگشت میدهد. مانند: Restangular.oneUrl(' googlers ' , ' http://www.google.com/1 ').get();
allUrl	route, url	این متد مانند متد قبل است با این تفاوت که یک مجموعه را بازگشت میدهد.
copy	formElement	این متد یک کپی از المنت‌های یک فرم را ایجاد میکند که ما میتوانیم آنها را تغییر دهیم.
restangularizeElement	parent,element, route, queryParams	یک المنت جدید را به صورت Restangularize تغییر میدهد.
restangularizeCollection	parent, element, route, queryParams	یک کالکشن Collection را به صورت Restangularize تغییر میدهد.

در این قسمت قصد داشتیم تا شما را با این کتابخانه کمی آشنا کنیم. در قسمت بعدی سعی می‌کنیم تا با مثال‌هایی کاربردی، شما را بیشتر با این سرویس خارق العاده آشنا کنیم.