

عنوان: توسعه سرویس‌های مبتنی بر REST در AngularJS با استفاده از RestAngular : بخش دوم

نویسنده: مهرداد کاهه

تاریخ: ۱۳۹۴/۰۷/۱۸ ۲۱:۳۰

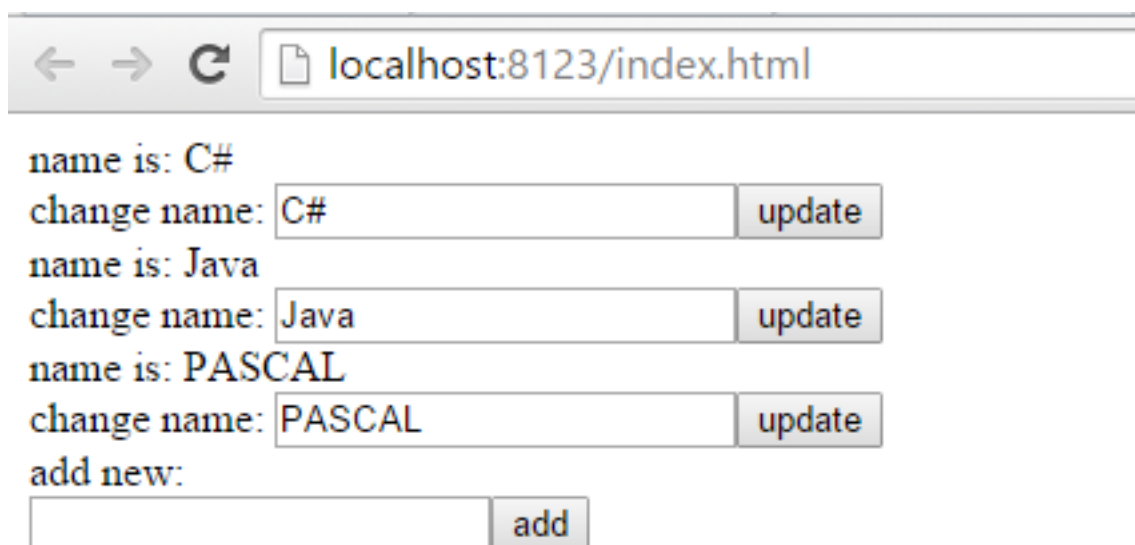
آدرس: [www.dotnettips.info](http://www.dotnettips.info)

گروه‌ها: JavaScript, ASP.NET Web API, AngularJS, json.net

در [بخش پیشین](#) کلیات کتابخانه‌ی [Restangular](#) را بررسی کردیم. در این بخش قصد داریم تا در طی یک پروژه، امکانات و قابلیت‌های بی‌نظیر این سرویس را در یک پروژه‌ی واقعی مشاهده کنیم.

### کلیات پروژه

در این پروژه قصد داریم تا لیست کتاب‌های یک کتابخانه را نمایش دهیم. این کتابها قابلیت ویرایش نام دارند و همچنین شما می‌توانید کتابهای جدیدی را به لیست کتابها اضافه نمایید. تصویر زیر خروجی این پروژه است:



localhost:8123/index.html

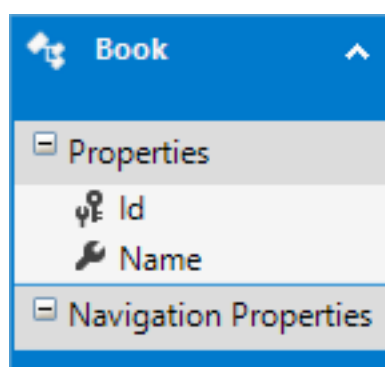
name is: C#  
change name:

name is: Java  
change name:

name is: PASCAL  
change name:

add new:

پایگاه داده‌ی برنامه با نام LibDb درون پوشه‌ی app\_data قرار داده شده‌است. این پایگاه داده تنها دارای یک جدول است؛ با نام Books که دیاگرام آن را در شکل زیر مشاهده می‌کنید:



### پیاده سازی

در ابتدا یک پروژه‌ی Empty را با رفرنس web API ایجاد می‌کنیم. حال درون فایل WebApiConfig.cs تکه کد زیر را اضافه می‌کنیم. این تکه کد فیلدها و آبجکت‌هایی را که از سمت سرور بازگشت داده می‌شوند، به صورت camelCase تبدیل می‌کند.

```
var jsonFormatter = config.Formatters.OfType<JsonMediaTypeFormatter>().First();
```

```
jsonFormatter.SerializerSettings.ContractResolver = new CamelCasePropertyNamesContractResolver();
```

در ادامه مدل edmx را ساخته و پس از آن Books Web Api را درون پوشه‌ی کنترلر ایجاد می‌کنیم. این کنترلر به صورت Default web Api Scaffold ساخته شده است و به دلیل اینکه به بحث ما مرتبط نیست؛ از توضیح بیشتر آن می‌گذریم. حال پوشه‌ای را با نام app برای نگه داری فایل‌های AngularJs اضافه می‌کنیم و درون آن فایل app.js را ایجاد می‌کنیم:

```
var angularExample = angular.module('angularexample', ["restangular"])
angularExample.config(["RestangularProvider", function (RestangularProvider) {
    //RestangularProvider.setRestangularFields({
    //    id: "id"
    //});
    RestangularProvider.setBaseUrl('/api');
}]);

angularExample.controller("MainCtrl", ["Restangular", "$scope", function (Restangular, $scope) {
    var resource = Restangular.all('books');
    resource.getList().then(function (response) {
        $scope.books = response;
    });
    $scope.add = function () {
        resource.post($scope.newBook).then(function (newResource) {
            $scope.books.push(newResource);
        })
    }
}]);
```

محتویات فایل app.js را مشاهده می‌کنید. در ادامه درباره‌ی این قسمت بیشتر صحبت می‌کنیم.  
حال در روت پروژه فایل index.html را ایجاد می‌کنیم:

```
<!DOCTYPE html>
<html ng-app="angularexample" xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Rest Angular Sample</title>
<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
<script src="http://cdn.jsdelivr.net/underscorejs/1.5.1/underscore-min.js"></script>
<script src="/Scripts/restangular.min.js"></script>
<script src="/app/app.js"></script>
</head>
<body>
<div ng-controller="MainCtrl">
<div ng-repeat="item in books">
    name is: {{item.name}}<br />
    change name: <input type="text" ng-model="item.name" /><button type="submit" ng-
click="item.put();">update</button>
</div>
<div>
    add new: <br />
<input type="text" ng-model="newBook.name" /><button type="submit" ng-
click="add()">add</button>
</div>
</div>
</body>
```

و در نهایت در پوشه‌ی Scripts فایل‌های سرویس Restangular را قرار می‌دهیم.  
تا به اینجای کار تمامی کارهای مورد نیاز تشکیل پروژه را انجام داده‌ایم. حال به بررسی بیشتر سرویس‌های این پروژه می‌پردازیم؛ یعنی کدهای درون فایل app.js.  
ببینیم که برای دریافت لیست تمامی کتابها، ما چه کارهایی را انجام داده‌ایم! به تکه کد زیر دقت کنید:

```
var resource = Restangular.all('books');
resource.getList().then(function (response) {
    $scope.books = response;
});
```

این تمامی آن چیزی است که ما برای دریافت لیست تمامی کتابها انجام داده‌ایم. به همین سادگی! در خط اول از متد all که در

بخش قبل توضیح مختصری راجع به آن داده بودم برای دریافت لیست تمامی کتابها استفاده کردم که پارامتر درون آن آدرس Web Api است. البته همانطور که می‌دانید در ASP.NET Web Api ما همیشه به base address یک api نیز اضافه می‌کنیم. حال اینکه چرا api در اینجا نیامده در ادامه و در بخش تنظیمات کلی Restangular توضیح می‌دهم. در خط دوم نیز از اشاره‌گر resources متد getList را فراخوانی کرده و لیست کتابها را در response دریافت می‌کنیم.

پس از آن می‌خواهیم ببینیم که عملیات ایجاد یک کتاب جدید چگونه انجام می‌گردد. تکه کد زیر این عملیات را انجام می‌دهد:

```
$scope.add = function () {  
    resource.post($scope.newBook).then(function (newResource) {  
        $scope.books.push(newResource);  
    })  
}
```

ما یک متد با نام add ایجاد کرده‌ایم که در سمت View توسط دایرکتیو ng-click آن را فراخوانی می‌کنیم. همانطور که مشاهده می‌کنید درون app.js متدی برای update موارد قبلی نیست. بیاید سری به View بزنیم. به المنت div زیر توجه کنید. همانطور که می‌بینید تمامی عملیات update با یک دستور ساده item.put حل و فصل شده.

```
<div ng-repeat="item in books">  
    name is: {{item.name}}<br />  
    change name: <input type="text" ng-model="item.name" /><button type="submit" ng-  
click="item.put();">update</button>  
</div>
```

تمامی آنچه که گفته شد تنها بخشی از قابلیت‌های شگفت انگیز این افزونه بود. امیدوارم که مطلب مفید واقع شده باشد. سورس این پروژه را می‌توانید از لینک زیر دریافت کنید.

[سورس پروژه: RestangularSample.rar](#)