عنوان: **$inject در AngularJs**
نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۰۹/۲۳ ۸:۴۵
آدرس: www.dotnettips.info
گروه‌ها: AngularJS

همان طور که در پست‌های قبلی ذکر شده بود در angular تزریق وابستگی به صورت پیش فرض وجود دارد. کافیست نام سرویس مورد نظر با نام‌های پیش فرض تعبیه شده در angular یا با نام سرویس‌های ساخته شده توسط خودتان مطابقت داشته باشد. به عنوان مثال برای تزریق سرویس $scope در توابع سازنده کنترلر کافیست یک پارامتر به همین نام را به عنوان آرگومان در این توابع در نظر بگیرید. همچنین برای استفاده از سرویس $http باید یک پارامتر دیگر به همین نام در این توابع در نظر داشته باشید و همچنین برای $location. در مورد سرویس‌های ساخته شده توسط خودتان نیز باید همین قانون را پیاده کنید. در این پست قصد دارم از injector تعبیه شده در angular برای تغییر رفتار فریم ورک در هنگام شناسایی پارامترهای توابع استفاده کنم.
ابتدا مثال زیر را به روش‌های قبلی پیاده سازی می‌کنیم:

```
var app = angular.module('myApp', []);

app.factory('bookService', function () {
    var books = [
        { name: 'A' },
        { name: 'B' },
        { name: 'C' }
    ];
    return books;
});
app.controller('bookCtrl', function ($scope, bookService) {
    $scope.books = bookService;
})
```

view مورد نظر نیز به صورت زیر خواهد بود:

```
<script type="text/javascript" src="~/scripts/Modules/module5.js"></script>

<div ng-app="myApp">
    <div ng-controller="bookCtrl">
        <table>
            <tr ng-repeat="book in books">
                <td>
                    {{book.name}}
                </td>
            </tr>
        </table>
    </div>
</div>
```

نیاز به توضیح نیست که در هنگام تعریف تابع سازنده کنترلر bookCtrl باید نام پارارمترهای وروردی تابع در هنگام تزریق وابستگی دقیقا مانند مثال بالا باشد. (بعنی $scope برای دسترسی به سرویس scope و bookService برای دسترسی به سرویس ساخته شده توسط factory - *ترتیب پارامترها در اینجا اهمیتی ندارد*). حال مثال بالا را با استفاده از injector موجود در angular برای تزریق وابستگی‌ها پیاده سازی می‌کنم. ابتدا تابع کنترلر bookCtrl را به صورت زیر ایجاد می‌کنیم:

```
var bookCtrl = function (sc,bs) {
    sc.books = bs;
};
```

از پارامتر sc به جای $scope و از bs به عنوان bookService در این تابع استفاده شده است. سپس کنترلر موجود را به ماژول مورد نظر نسبت می‌دهیم:

```
app.controller('bookCtrl',bookCtrl);
```

اگر برنامه را به همین صورت اجرا کنید خروجی مورد نظر حاصل نخواهد شد. زیرا آرگومان‌های sc و bs برای angular تعریف نشده است. کافیست وابستگی‌های تابع کنترلر را به صورت زیر برای angular مشخص نماییم:

```
bookCtrl.$inject = ['$scope','bookService'];
```

در نتیجه تعریف کنترلر بالا به صورت کامل زیر خواهد بود:

```
var app = angular.module('myApp', []);

app.factory('bookService', function () {
    var books = [
        { name: 'A' },
        { name: 'B' },
        { name: 'C' }
    ];
    return books;
});

var bookCtrl = function (sc,bs) {
    sc.books = bs;
};

bookCtrl.$inject = ['$scope','bookService'];

app.controller('bookCtrl',bookCtrl);
```

از این پس در هنگام فراخوانی تابع کنترلر bookCtrl سرویس‌های $scope و bookService به ترتیب به عنوان آرگومان‌های اول و دوم در اختیار کنترلر قرار می‌گیرند. می‌توان به جای فراخوانی مستقیم $inject، تزریق وابستگی‌ها را در هنگام تعریف توابع سازنده به صورت زیر نیز فراهم ساخت:

```
app.controller('bookCtrl', ['$scope', 'bookService', function (sc, bs) {
    sc.books = bs;
}])
```

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۲۴ ۲۳:۰

پس با این حساب AngularJS به minification اسکریپت‌ها حساس است. چون در حین minification تمام نام پارامترها با a و b و c جایگزین می‌شوند. در این مورد چه پیشنهادی وجود دارد؟

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۹/۲۴ ۲۳:۱۰

درسته.چون سیستم تزریق وابستگیش با نام متغیرها کار می‌کند با minification نام متغیرها تغییر می‌کند و در نتیجه برنامه از کار می‌افتد. راه‌های معین کردن صریح وابستگی‌ها در مقاله‌ی فوق ذکر شده.
روش اول:

```
bookCtrl.$inject = ['$scope','bookService'];
```

روش دوم:

```
app.controller('bookCtrl', ['$scope', 'bookService', function (sc, bs) {
    sc.books = bs;
}])
```

در این روش وابستگی‌های کنترلرها صریحا ذکر شده و با تغییر نام متغیرها انگولار می‌داند که چه وابستگی هایی را باید تزریق کند.

نویسنده: وحید
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۶:۲۳

لطفا توضیحی در مورد $Watch دهید ممنون

نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۱۰/۱۴ ۲۱:۱

به صورت کلی با استفاده از $watch می‌توان تمامی تغییراتی را که به خواص ViewModel اعمال می‌شوند مشاهده کرد.
تعریف کلی آن به صورت زیر است:

```
$watch(watchExpression, listener, objectEquality)
```

»watchExpression : می‌توان نام خاصیت مورد نظر در ViewModel یا یک تابع را که قصد مشاهده تغییرات آن را داریم تعیین کنیم.
»Listener : با تغییر در مقدار watchExpression اگر مقدار قبلی این عبارت با مقدار فعلی آن برابر نباشد این تابع فراخوانی می‌شود.
» objectEquality : به صورت پیش فرض Angular مقادیر مورد نظر برای تغییرات را فقط از نظر Reference Equal بودن چک می‌کند. اگر بخواهیم که Angular به صورت عمقی و درختی مقادیر ابجکت‌ها را بررسی کند مقدار این پارامتر باید true شود.

در فریم ورک Angular هر زمان که عمل مقید سازی خواص ViewModel به عناصر DOM انجام می‌گیرد در واقع یک نمونه از $watch به لیستی به نام watch list $ اضافه می‌شود. دقت کنید که صرفا تعریف در محدوده کنترلر کافی نیست بلکه باید خاصیت مورد نظر حتما مقید شود. برای مثال

```
app.controller('MainCtrl', function($scope) {
```

```
  $scope.foo = "Foo";
  $scope.world = "World";
});
```

در View نیز

```
Hello, {{ World }}
```

در کنترلر بالا دو خاصیت تعریف شده است، در حالی که در View فقط یک خاصیت مقید شده است. درنتیجه فقط یک $watch به لیست مورد نظر اضافه شده است.

و به عنوان نکته آخر، در Angular نسخه 1.1.4 تابعی به نام watchCollection اضافه شده است که برای ردیابی تغییرات یک مجموعه مورد استفاده قرار می‌گیرد.

یک مثال در این مورد