

تا پیش از این به احتمال زیاد با `Interceptor` ها در `IOC Container` ها متفاوت آشنا شدید و برای `AOP` از آن‌ها استفاده کرده‌اید. در این جا نیز دقیقاً همان مفهوم و هدف را دنبال خواهیم کرد؛ اضافه کردن و تزریق کدهای نوشته شده به منطق برنامه. کاربرد `Interceptor` ها در انگولار، زمانی است که قصد داشته باشیم یک سری تنظیمات عمومی را برای درخواست‌های `$http` انجام دهیم. همچنین می‌توان انجام برخی مراحل مشترک، نظیر اعتبارسنجی یا مدیریت خطاها را نیز توسط `Interceptor` ها انجام دهیم. سرویس `$http` در `Angular` جهت ارتباط و تبادل اطلاعات با دنیای `Backend` مورد استفاده قرار می‌گیرد. حالت‌هایی بنابر نیاز به وجود می‌آیند که بخواهیم ارسال اطلاعات به سرور و هم چنین پاسخ دریافتی را `capture` کنیم و قبل از این که داده‌ها در اختیار `App` قرار گیرد، آن را مورد بررسی قرار دهیم (برای مثال لاگ اطلاعات) یا حتی نوشتن یک `HTTP error handling` جهت مدیریت خطاهای به وجود آمده حین ارتباط با سرور (برای مثال خطای 404). حال با ذکر مثالی این موارد را بررسی می‌کنیم. برای نوشتن یک `Interceptor` می‌توان با استفاده از سرویس `factory` این کار را به صورت زیر انجام داد.

```
module.factory('myInterceptor', ['$log', function($log) {
    $log.debug('data');

    var myInterceptor = {
        ....
        ....
        ....
    };

    return myInterceptor;
}]);
```

کد بالا یک `Interceptor` بسیار ساده است که وظیفه آن لاگ اطلاعات است. در انگولار چهار نوع `Interceptor` برای سرویس `$http` داریم:

« **request** : قبل از هر فراخوانی سرویس‌های سمت سرور، ابتدا این `Interceptor` فراخوانی می‌شود و `config` سرویس `$http` در اختیار آن قرار می‌گیرد. می‌توان این تنظیمات را با توجه به نیاز، تغییر داد و نمونه ساخته شده جدید را در اختیار سرویس `$http` قرار دهیم.

« **response** : هر زمان که عملیات فراخوانی سرویس‌های سمت سرور به درستی انجام شود و همراه با آن پاسخی از سرور دریافت شود، این `Interceptor` قبل از فراخوانی تابع `success` سرویس `$http`، اجرا خواهد شد.

« **requestError** : از آنجا که سرویس `$http` دارای مجموعه‌ای از `Interceptor` ها است و آن‌ها نیز یکی پس از دیگری حین انجام عملیات اجرا می‌شوند، اگر در `Request Interceptor` قبلی خطایی رخ دهد بلافاصله این `Interceptor` فراخوانی می‌شود.

« **responseError** : درست مانند حالت `requestInterceptor` است؛ فقط خطای مربوطه باید در تابع `response` باشد.

با توجه به توضیحات بالا کد قبلی را به صورت زیر تعمیم می‌دهیم.

```
module.factory('myInterceptor', ['$q', '$log', function($q, $log) {
    $log.debug('data');

    return {
        request: function(config) {
            return config || $q.when(config);
        },
        requestError: function(rejection) {
```

```
return $q.reject(rejection);
},
response: function(response) {
return response || $q.when(response);
},
responseError: function(rejection) {
return $q.reject(rejection);
}
}
}]]);
```

برای رجیستر کردن Interceptor بالا به سرویس‌های \$http باید به صورت زیر عمل نمود.

```
angular.module('myApp')
.config(function($httpProvider) {
$httpProvider.interceptors.push('myInterceptor');
});
```