بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

مدیریت خطا(Exception) در ASP.NET Web API

مدرس : مهندس افشین رفوآ

دوره آموزشی Web API

HttpResponseException

اگر برای Web API Controller یک خطا رخ دهد چه اتفاقی می افتد؟به صورت پیش فرض، اکثر خطاها در یک پاسخ HTTP با کد ۵۰۰(خطای درونی سرور) نمایش داده می شود.

HttpResponseException یک مورد ویژه محسوب می شود. این exception هر گونه کد های حالت(status code) HTTP که شما در Exception constructor مشخص ساختید بر می گرداند. برای مثال، اگر پارامتر id معتبر نباشد، متد زیر کد ۴۰۴(پیدا نکردن) را بر می گرداند.

```
public Product GetProduct(int id)
{
    Product item = repository.Get(id);
    if (item == null)
    {
        throw new HttpResponseException(HttpStatusCode.NotFound);
    }
    return item;
}
```

برای کنترل بیشتر روی پاسخ (Response)، شما همچنین می توانید یک Response را بسازید و به آن HttpResponseException اضافه کنید:

```
public Product GetProduct(int id)
{
    Product item = repository.Get(id);
    if (item == null)
    {
        var resp = new HttpResponseMessage(HttpStatusCode.NotFound)
        {
            Content = new StringContent(string.Format("No product with ID = {0}", id)),
            ReasonPhrase = "Product ID Not Found"
        }
```

```
        throw new HttpResponseException(resp);
    }
    return item;
}
```

فیلتر های خطا(Exception filters)

شما می توانید با نوشتن Exception filter مدیریت خطاها را سفارشی سازی کنید. Exception filter ها زمانی اجرا می شوند که یک متد Controller به هر خطایی به غیر از خطای HttpResponseException برخورد کند. HttpResponseException یک مورد خاص می باشد زیرا این خطا ،صرفا برای برگرداندن پاسخ HTTP طراحی شده است.

Exception filter ها رابط System.Web.Http.Filters.IExceptionFilter را پیاده سازی می کنند. راحت ترین راه برای ساخت یک exception filter این است که از System.Web.Http.Filters.ExceptionFilterAttribute مشتق شود و متد OnException را override کند.

Exception filter ها در ASP.NET Web API شبیه ASP.NET MVC می باشد ولی با این حال در namespace ها و توابع جداگانه تعریف می شوند. به عنوان یک مورد خاص می توان به کلاس HandleErrorAttribute در MVC اشاره کرد که مدیریت آن در Web API Controller امکان پذیر نیست.

کد زیر فیلتری را نشان می دهد که خطای NotImplementedException را به کد ۵۰۱ HTTP(عدم پیاده سازی) تبدیل می کند.

```
namespace ProductStore.Filters
{
    using System;
    using System.Net;
    using System.Net.Http;
    using System.Web.Http.Filters;
    public class NotImplExceptionFilterAttribute : ExceptionFilterAttribute
    {
        public override void OnException(HttpActionExecutedContext context)
        {
            if (context.Exception is NotImplementedException)
            {
                context.Response = new HttpResponseMessage(HttpStatusCode.NotImplemented);
            }
        }
    }
}
```

خاصیت Response در HttpActionExecutedContext دربردارنده پاسخ HTTP ارسالی به سمت کاربر می باشد.

ثبت فیلتر های خطا(registering exception filters)

چند راه برای register کردن exception filter های Web API وجود دارد:

- توسط action
- توسط controller
- به صورت سراسری

برای اجرا کردن فیلتر به یک action خاص، filter را به عنوان attribute به یک Action اضافه کنید:

```
public class ProductsController : ApiController
{
    [NotImplExceptionFilter]
    public Contact GetContact(int id)
    {
        throw new NotImplementedException("This method is not implemented");
    }
}
```

برای اجرا کردن filter روی همه ی actionهای یک controller، filter را به عنوان attribute به Controller اضافه کنید:

```
[NotImplExceptionFilter]
public class ProductsController : ApiController
{
    // ...
}
```

برای اجرا کردن filter به صورت سراسری به controller های Web API، یک مورد filter به مجموعه GlobalConfiguration.Configuration.Filters اضافه کنید. Exception filterهای در این مجموعه روی تمام Action های Web API Controller اجرا خواهد شد.

```
GlobalConfiguration.Configuration.Filters.Add(
new ProductStore.NotImplExceptionFilterAttribute());
```

اگر شما از ASP.NET MVC 4 Web Application برای ساخت پروژه استفاده کنید، کد های تنظیمات Web API خود را داخل کلاس WebApiConfig که در فولدر App_Start می باشد قرار دهید:

```
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        config.Filters.Add(new ProductStore.NotImplExceptionFilterAttribute());
        // Other configuration code...
    }
}
```

HttpError

HttpError یک راه دائمی برای برگرداندن اطلاعات خطا در response ارائه می دهد. مثال زیر نشان می دهد چگونه پاسخ کد وضعیت 404 (پیدا نشدن) با یک HttpError برگردانده می شود.

```
public HttpResponseMessage GetProduct(int id)
{
    Product item = repository.Get(id);
    if (item == null)
    {
        var message = string.Format("Product with id = {0} not found", id);
        return Request.CreateErrorResponse(HttpStatusCode.NotFound, message);
    }
    else
    {
        return Request.CreateResponse(HttpStatusCode.OK, item);
    }
}
```

CreateErrorResponse یک method extension در کلاس System.Net.Http.HttpRequestMessageExtensions می باشد. CreateErrorResponse ابتدا یک HttpError و سپس یک HttpResponseMessage که در بردارنده HttpError است می سازد.

در این مثال، اگر متد موفقیت آمیز باشد، product را در قالب پاسخ HTTP برمی گرداند. اما اگر product درخواست کننده پیدا نشود، پاسخ HTTP شامل HttpError در درخواست می فرستد و پاسخ چیزی شبیه به کد زیر است:

```
HTTP/1.1 404 Not Found
Content-Type: application/json; charset=utf-8
Date: Thu, 09 Aug 2012 23:27:18 GMT
Content-Length: 51
{
  "Message": "Product with id = 12 not found"
}
```

توجه داشته باشید در این مثال HttpError به فرمت JSON ، serial شده است.

HttpError و اعتبارسنجی Model

شما می توانید برای اعتبارسنجی model، حالت مدل (model state) را CreateErrorResponse قرار دهید تا شامل اعتبارسنجی خطاها در پاسخ شود:

```
public HttpResponseMessage PostProduct(Product item)
{
    if (!ModelState.IsValid)
    {
        return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ModelState);
    }
    // Implementation not shown...
}
```

این مثال باید مقداری شبیه به کد های زیر بر گرداند:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Content-Length: 320
{
  "Message": "The request is invalid.",
  "ModelState": {
    "item": [
      "Required property 'Name' not found in JSON. Path '', line 1, position 14."
    ],
    "item.Name": [
      "The Name field is required."
    ],
    "item.Price": [
      "The field Price must be between 0 and 999."
    ]
  }
}
```

استفاده کردن از HttpError با HttpResponseException

مثال های قبل یک پیغام HttpResponseMessage از action کنترولر بر می گرداند ولی شما می توانید از HttpResponseException برای برگرداندن HttpError استفاده کنید. این کار به شما اجازه می دهد در حالی که خطای HttpError وجود دارد، یک مدل Strongly-typed با مقدار success برگردانید:

```
public Product GetProduct(int id)
{
    Product item = repository.Get(id);
    if (item == null)
    {
        var message = string.Format("Product with id = {0} not found", id);
        throw new HttpResponseException(
            Request.CreateErrorResponse(HttpStatusCode.NotFound, message));
    }
```

```
    else
    {
        return item;
    }
}
```