بسم اللّه الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

Action Result ها در Web API 2

مدرس : مهندس افشین رفوآ

دوره آموزشی Web API

در این قسمت قصد داریم نشان دهیم چگونه Asp.net Web API مقدار برگشتی از controller را به پاسخ HTTP تبدیل می کند. یک کنترولر Web API قابلیت برگرداندن هر کدام از مقادیر زیر را دارد:

1-void

2-HttpResponseMessage

3-IHttpActionResult

4-هر نوع مقدار دیگر

Web API بر اساس نوع مقادیر بازگشتی که در بالا گفته شد، از مکانیزم خاصی برای ساخت پاسخ HTTP استفاده می کند.

نوع بازگشتی	چگونگی ساخت پاسخ توسط Web API
void	پاسخ 204 را بر می گرداند(بدون محتوا)
HttpResponseMessage	به صورت مستقیم به یک پاسخ HTTPتبدیل می کند.
IHttpActionResult	ExecuteAsync را برای ساخت HttpResponseMessage فراخوانی می کند و سپس آن را به یک پاسخ HTTP تبدیل می کند.

چگونگی ساخت پاسخ توسط Web API	نوع بازگشتی
مقدار بازگشتی سریال شده را در قالب پیام پاسخ قرار می دهد و ok(200) بر می گرداند.	مقادیر دیگر

خب حالا در ادامه بحث موارد بالا را توضیح می دهیم.

Void

اگر مقدار برگشتی void باشد، web API یه مقدار پاسخ خالی HTTP با کد 204(بدون محتوا) ارسال می کند.

Controller به صورت زیر است:

```
public class ValuesController : ApiController
{
   public void Post()
   {
   }
}
```

پاسخ HTTP:

```
HTTP/1.1 204 No Content
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 02:13:26 GMT
```

HttpResponseMessage

اگر Action یک مقدار HttpResnponseMessage برگرداند، Web API مقدار بازگشتی را به طور مستقیم به یک پاسخ HTTP تبدیل می کند و با استفاده از ویژگی های HttpResponseMessage آن object محتوای پاسخ را پی ریزی می کند.

این ویژگی قابلیت کنترل زیادی به پاسخ می دهد. برای مثال، Action زیر، مقدار Cache-Contol header را مقداردهی می کند.

```
public class ValuesController : ApiController
{
   public HttpResponseMessage Get()
   {
       HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, "value");
       response.Content = new StringContent("hello", Encoding.Unicode);
       response.Headers.CacheControl = new CacheControlHeaderValue()
       {
           MaxAge = TimeSpan.FromMinutes(20)
       };
       return response;
   }
}
```

پاسخ:

```
HTTP/1.1 200 OK
Cache-Control: max-age=1200
Content-Length: 10
Content-Type: text/plain;charset=utf-16
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 08:53:35 GMT
Hello
```

اگر شما یک مدل را به متد CreateResponse ارسال کنید، Web API برای سریال کردن model از یک پیام media formatter استفاده می کند.

```
public HttpResponseMessage Get()
{
   // Get a list of products from a database.
   IEnumerableproducts = GetProductsFromDB();
   // Write the list to the response body.
   HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, products);
   return response;
}
```

Web API از accept header برای انتخاب formatter استفاده می کند.

IHttpActionResult

رابط IHttpActionResult در Web API 2 معرفی شده است. اساسا، این رابط یک HttpResponseMessage را تعریف می کند. در زیر، به چند نمونه از مزایای استفاده از رابط IHttpActionResult اشاره می کنیم:

- ساده سازی آزمایش Controller ها
- انتقال منطق معمول برای ساخت پاسخ های HTTP در قالب کلاس های جداگانه
- آشکارتر کردن معنی action کنترولر با استفاده از مخفی سازی جزئیات ساخت پاسخ

IHttpActionResult شامل متد واحد می باشد، ExecuteAsync به صورت غیر همسان یک مثال HttpResponseMessage تولید می کند.

```
public interface IHttpActionResult
{
   TaskExecuteAsync(CancellationToken cancellationToken);
}
```

اگر action یک Controller یک مقدار IHttpActionResult بر گرداند، Web API متد ExecuteAsync را برای ساخت یک HttpResponseMessage فراخوانی می کند و سپس HttpResponseMessage را به یک پاسخ HTTP تبدیل می کند.

در زیر، یک پیاده سازی ساده از IHttpActionResult که یک جواب ساده را می سازد مشاهده خواهیم کرد:

```
public class TextResult : IHttpActionResult
{
```

```
    string _value;
    HttpRequestMessage _request;
    public TextResult(string value, HttpRequestMessage request)
    {
        _value = value;
        _request = request;
    }
    public TaskExecuteAsync(CancellationToken cancellationToken)
    {
        var response = new HttpResponseMessage()
        {
            Content = new StringContent(_value),
            RequestMessage = _request
        };
        return Task.FromResult(response);
    }
}
```

و action کنترولر به صورت زیر است:

```
public class ValuesController : ApiController
{
    public IHttpActionResult Get()
    {
        return new TextResult("hello", Request);
    }
}
```

و پاسخ:

```
HTTP/1.1 200 OK
Content-Length: 5
Content-Type: text/plain;charset=utf-8
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 08:53:35 GMT
Hello
```

گاهی اوقات، شما لازم است برای پیاده سازی IHttpActionResult از System.Web.Http.Results برای namespace استفاده کنید. کلاس ApiController یک متد helper تعریف می کند که نتایج action های built-in را برمی گرداند.

در مثال بعد، اگر درخواست با ProductID همخوانی نداشته باشد ، Controller کلاس ApiController.NotFound را برای ساخت پیام 404(یافت نشد) فراخوانی می کند. در غیر این صورت کنترولر، ApiController.OKرا برای ساخت پیام 200 (ok) فراخوانی می کند.

```
public IHttpActionResult Get (int id)
{
    Product product = _repository.Get (id);
    if (product == null)
    {
        return NotFound();// Returns a NotFoundResult
    }
```

```
    return Ok(product); // Returns an OkNegotiatedContentResult
}
```

مقادیر بازگشتی دیگر

برای تمامی مقادیر بازگشتی دیگر، Web API از media formatter برای serial کردن مقدار بازگشتی استفاده می کند. Web API مقدار serial شده را درون پاسخ می نویسد و وضعیت آن را 200 (ok) قرار می دهد.

```
public class ProductsController : ApiController
{
    public IEnumerableGet()
    {
        return GetAllProductsFromDB();
    }
}
```

یکی از ایرادات این روش این است که شما نمی توانید یک خطا را مستقیما بر گردانید مثل پیام 404 . با این حال شما می توانید HttpResponseException را به کدهای خطا بسط دهید. Web API از Accept header برای انتخاب formatter استفاده می کند.

مثالی از یک درخواست:

```
GET http://localhost/api/products HTTP/1.1
User-Agent: Fiddler
Host: localhost:24127
Accept: application/json
```

مثالی از پاسخ:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=utf-8
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 08:53:35 GMT
Content-Length: 56

[{"Id":1,"Name":"Yo-yo","Category":"Toys","Price":6.95}]
```