بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

مذاکره محتوا (Content Negotiation) در ASP.NET Web API

مدرس : مهندس افشین رفوآ

دوره آموزشی Web API

طبق تعریفی که مشخصه یHTTP (RFC 2616) ارائه می دهد ، Content negotiation فرآیند انتخاب بهترین نمایش برای پاسخ داده شده از بین چندین نمایش در دسترس می باشد . مکانیزم اولیه Content negotiation در HTTP دارای سربرگ درخواست های(request header) زیر می باشد:

- Accept: انواع مختلف داده ی قابل قبول برای پاسخ مثل application/json ، application/xml یا یک نوع داده شخصی مثل application/vnd.example+xml را مشخص می کند.
- Accept-Charset: کاراکترهای قابل پذیرش را مشخص می کند مثل UTF-8 یا ISO 8859-1.
- Accept-Encoding: کدگذاری محتوای قابل قبول را مشخص می کند مثل gzip.
- Accept-Language: زبان پیش فرض را انتخاب می کند مثل en-us.

همچنین سرور می تواند بخش های دیگر درخواست HTTP را زیر نظر بگیرد. برای مثال، اگر درخواست شامل یک X-Requested-With-Header باشد، یک درخواست AJAX نشان می دهد و اگر accept header وجود نداشته باشد سرور به صورت پیش فرض JSON را قرار می دهد.

در این مقاله، می بینیم چطور Web API از Accept و Accept-Charset استفاده می کند.

ترتیب (Serialization)

اگر Web API Controller یک منبع به عنوان CLR برگرداند، خط لوله (pipeline) مقدار بازگشتی را serial می کند و در پاسخ HTTP می نویسد. برای مثال action کنترولر زیر را در نظر بگیرید:

```
public Product GetProduct(int id)
{
    var item = _products.FirstOrDefault(p => p.ID == id);
    if (item == null)
```

```
    {
        throw new HttpResponseException(HttpStatusCode.NotFound);
    }
    return item;
}
```

کاربر این درخواست HTTP را می فرستد:

```
GET http://localhost.:21069/api/products/1 HTTP/1.1
Host: localhost.:21069
Accept: application/json, text/javascript, */*; q=0.01
```

در پاسخ، سرور باید کدی شبیه به کد زیر ارسال کند:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 57
Connection: Close
{"Id":1,"Name":"Gizmo","Category":"Widgets","Price":1.99}
```

در این مثال کاربر، JSON یا JavaScript و یا هر چیزی (*/*) درخواست می کند. سرور مسئولیت نمایش Product با JSON را بر عهده دارد. توجه داشته باشید که سربرگ نوع محتوا(Content-type header) پاسخ را application/json قرار دهید.

یک Controller می تواند یک مقدار HttpResponseMessage را برگرداند. برای مشخص کردن یک CLR برای پاسخ ، متد CreateResponse را باید فراخوانی کنید:

```
public HttpResponseMessage GetProduct(int id)
{
    var item = _products.FirstOrDefault(p => p.ID == id);
    if (item == null)
    {
        throw new HttpResponseException(HttpStatusCode.NotFound);
    }
    return Request.CreateResponse(HttpStatusCode.OK, product);
}
```

این گزینه کنترول بیشتری روی جزئیات پاسخ به شما می دهد. شما می توانید کد وضعیت را مشخص کنید و headerهایی مثل HTTP و غیره را اضافه کنید.

آن object که منبع را serial می کند قالب داده (media formatter) می نامند. Media formatter از کلاس MediaTypeFormatter مشتق شده است. Web API برای XML و JSON قالب داده در نظر می گیرد و شما می توانید یک قالب شخصی برای پشتیبانی بقیه ی انواع داده ها بسازید.

Content Negotiation چگونه کار می کند؟

ابتدا pipeline، سرویس IContentNegotiator را از HttpConfiguration دریافت می کند. همچنین لیستی از قالب داده ها از مجموعه HttpConfiguration.Formatters **دریافت می کند.**

سپس pipeline ، IContentNegotiatior.Negotiate را فراخوانی می کند و از مقادیر زیر صرف نظر می کند:

- نوع object برای serial کردن
- مجموعه ای از media formatter ها
- درخواست HTTP

متد Negotiate دو نوع اطلاعات بر می گرداند:

- قالبی که باید استفاده شود.
- Media type پاسخ

اگر هیچ قالبی پیدا نشد متد Negotiate مقدار null برمی گرداند و کاربر خطای HTTP 406 (غیر قابل دسترس) را دریافت می کند. کد زیر نشان می دهد چگونه یک Controller می تواند به صورت مستقیم Content negotiation را فراخوانی کند.

```
public HttpResponseMessage GetProduct(int id)
{
    var product = new Product()
        { Id = id, Name = "Gizmo", Category = "Widgets", Price = 1.99M };

    IContentNegotiator negotiator = this.Configuration.Services.GetContentNegotiator();
    ContentNegotiationResult result = negotiator.Negotiate(
        typeof(Product), this.Request, this.Configuration.Formatters);
    if (result == null)
    {
        var response = new HttpResponseMessage(HttpStatusCode.NotAcceptable);
        throw new HttpResponseException(response));
    }
    return new HttpResponseMessage()
    {
        Content = new ObjectContent<Product>(
            product,                // What we are serializing
            result.Formatter,           // The media formatter
            result.MediaType.MediaType  // The MIME type
        )
    };
}
```

این کد معادل چیزی است که pipeline **به صورت اتوماتیک انجام می دهد.**

Content negotiation پیش فرض

کلاس DefaultContentNegotiator یک پیاده سازی پیش فرض از IContentNegotiator ارائه می دهد که از چندین شاخص برای انتخاب یک قالب استفاده می کند.

ابتدا، قالب باید بتواند داده را serial کند. این عمل با فراخوانی MediaTypeFormatter.CanWriteType صورت می گیرد. سپس، Content Negotiation هر قالب محتوا را در نظر می گیرد و چگونگی ارتباط درخواست HTTP را ارزیابی می کند. برای ارزیابی ارتباط، Content Negotiation در قالب به دو چیز توجه می کند:

- مجموعه SupportedMediaTypes که لیستی از انواع داده ها پشتیبانی می کند. توجه داشته باشید Accept header می تواند شامل یک بازه باشد. برای مثال text/plain را می توان در بازه text/* و یا */* قرار داد.
- مجموعه MediaTypeMappings شامل لیستی از object های MediaTypeMapping می باشد. کلاس MediaTypeMapping یک راه عمومی برای ارتباط درخواست های HTTP با انواع داده ها آماده می کند. برای مثال می تواند سربرگ HTTP(custom header) را به media formatter خاص مرتبط کند.

اگر چندین ارتباط وجود داشته باشد، ارتباط با بالاترین کیفیت برنده می شود. برای مثال:

```
Accept: application/json, application/xml; q=0.9, */*; q=0.1
```

در این مثال، application/json دارای یک عامل موفقیت ضمنی از 1.0 می باشدو درنتیجه بر application/xml ترجیح داده می شود.

اگر هیچ ارتباطی پیدا نشود، Content Negotiator سعی می کند نوع داده را در صورت وجود به درخواست، ارتباط دهد. برای مثال، اگر درخواست شامل داده ی JSON باشد Content Negotiator دنبال قالب JSON می گردد. اگر باز هم ارتباطی پیدا نکرد، Content Negotiator اولین قالبی که بتواند داده را serial کند انتخاب می کند.

انتخاب کردن یک کدگذاری کاراکتری

بعد از آنکه یک قالب انتخاب شد، Content Negotiator بهترین کدگذار کاراکتری را انتخاب می کند. در صورت وجود یک درخواست، با در نظر گرفتن خاصیت SupportedEncodings در یک قالب، Accept-Charset header را مرتبط می کند.