عنوان: **Content Negotiation در WCF**
نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۳/۰۳/۰۶ ۹:۴۵
آدرس: www.dotnettips.info
گروه‌ها: WCF, Serialization, Content Negotiation

Content Negotiation ، مکانیزمی است که طی آن مصرف کننده یک سرویس http تعیین می‌کند که خروجی مورد نظر از سرویس به چه فرمتی در اختیار آن قرار گیرد. این قابلیت بسیار زیبا در Asp.Net Web Api فراهم می‌باشد. اما از آن جا که در WCF به صورت توکار مکانیزمی جهت پیاده سازی این قابلیت در نظر گرفته نشده است می‌توان از طریق یک کتابخانه ثالث به نام WCFRestContrib به این مهم دست یافت.

به صورت معمول برای پیاده سازی Content Negotiation، مصرف کننده باید در Accept هدر درخواست، برای سرویس مورد نظر، نوع Content-Type را نیز تعیین نمایید. از طرفی سرویس دهنده نیز باید معادل Mime Type درخواست شده، یک Formatter جهت سریالایز داده‌ها در اختیار داشته باشد. در WCF از طریق کتابخانه WcfRestContrib می‌توانیم به صورت زیر Content Negotiation را پیاده سازی نماییم:

ابتدا از طریق Nuget کتابخانه زیر را نصب کنید:

```
install-package WcfRestContrib
```

حال فرض کنید سرویسی به صورت زیر داریم:

```
[ServiceContract]
public interface IBooksService
{
    [OperationContract]
    void AddBook(string isbn, Book book);
}
```

کدهای بالا روشی مرسوم برای تعریف Service Contractهای WCF است. برای اینکه سرویس WCF بالا به صورت Rest طراحی شود و از طرفی قابلیت سریالاز داده‌ها به چندین فرمت را داشته باشد باید به صورت زیر عمل نماییم:

```
[ServiceContract]
public interface IBooksService
{
    [WebInvoke(UriTemplate = "/{isbn}", Method=Verbs.Put)]
    [WebDispatchFormatter]
    [OperationContract]
    void AddBook(string isbn, Book book);
    ....
}
```

وظیفه WebDispatchFormatterAttribute تعریف شده برای Operation بالا این است که نوع فرمت مورد نیاز را از Accept هدر درخواست واکشی کرده و با توجه به MimeTypeهای تعریف شده در سرویس، داده‌ها را به آن فرمت سریالاز نماید. در صورتی که MimeType درخواست شده از سوی مصرف کننده، سمت سرور تعریف نشده بود، MimeType پیش فرض انتخاب می‌شود.
گام بعدی مشخص کردن انواع MimeTypeها برای این سرویس است. در WcfRestContrib به صورت پیش فرض چهار Formatter تعبیه شده است:

» Xml : از DataContractSerializer موجود در WCF برای سریالاز و دی سریالایز داده‌ها استفاده می‌کند.
» Json : از طریق DataContactJsonSerializer برای سریالاز و دی سریالایز داده‌ها استفاده می‌کند.
POX : همانند مورد اول از DataContractSerializer استفاده می‌کند با این تفاوت که DataContractها بدون Namesapce و Attribute و DataMemberها نیز بدون Order می‌باشند.
» Form Url Encoded

در صورتی که نیاز به formatter دیگری دارید می‌توانید با استفاده از CustomFormatter موجود در این کتابخانه، Formatter

دلخواه خود را پیاده سازی نمایید.

همان طور که در بالا ذکر شد، در صورتی که MimeType درخواست شده از سوی مصرف کننده، سمت سرور تعریف نشده باشد، MimeType پیش فرض انتخاب می‌شود. برای تعریف MimeType پیش فرض می‌توان از خاصیت WebDispatchFormatterConfigurationAttribute که در فضای نام WcfRestContrib.ServiceModel.Description قرار دارد استفاده کرد. تعاریف سایر MimeTypeها نیز با استفاده از WebDispatchFormatterMimeTypeAttribute انجام می‌شود. به صورت زیر:

```
[WebDispatchFormatterConfiguration("application/xml")]
[WebDispatchFormatterMimeType(typeof(WcfRestContrib.ServiceModel.Dispatcher.Formatters.PoDataContract),
"application/xml",  "text/xml")]
[WebDispatchFormatterMimeType(
typeof(WcfRestContrib.ServiceModel.Dispatcher.Formatters.DataContractJson),  "application/json")]
[WebDispatchFormatterMimeType(
typeof(WcfRestContrib.ServiceModel.Dispatcher.Formatters.FormUrlEncoded), "application/x-www-form-
urlencoded")]
public class Books : IBooksService
{
   public void AddBook(string isbn, Book book)
   {
   }
}
```

همانند سایر تنظیمات WCF می‌توان تمامی این موارد را در فایل Config پروژه سرویس نیز تعریف کرد: برای مثال:

```
<system.serviceModel>
    <extensions>
        <behaviorExtensions>
            <add name="webFormatter"

type="WcfRestContrib.ServiceModel.Configuration.WebDispatchFormatter.ConfigurationBehaviorElement,
WcfRestContrib,
                        Version=x.x.x.x, Culture=neutral, PublicKeyToken=89183999a8dc93b5"/>
        </behaviorExtensions>
    </extensions>
    <serviceBehaviors>
        <behavior name="Rest">
          <webFormatter>
            <formatters defaultMimeType="application/xml">
              <formatter mimeTypes="application/xml,text/xml"
                         type="WcfRestContrib.ServiceModel.Dispatcher.Formatters.PoxDataContract,
WcfRestContrib"/>
              <formatter mimeTypes="application/json"
                         type="WcfRestContrib.ServiceModel.Dispatcher.Formatters.DataContractJson,
WcfRestContrib"/>
              <formatter mimeTypes="application/x-www-form-urlencoded"
                         type="WcfRestContrib.ServiceModel.Dispatcher.Formatters.FormUrlEncoded,
WcfRestContrib"/>
            </formatters>
          </webFormatter>
        </behavior>
    </serviceBehaviors>
</system.serviceModel>
```

نکته:

در صورتی که قصد داشته باشیم که باتوجه به direction مورد نظر (نظیر Outgoing یا Incoming) داده‌ها سریالایز/ دی سریالایز شوند، می‌توان این مورد را در هنگام تعریف OperationContract تعیین کرد:

```
[WebDispatchFormatter(WebDispatchFormatter.FormatterDirection.Outgoing)]
```

مطلب تکمیلی:

مشاهده پیاده سازی Content Negotiation در Asp.Net MVC