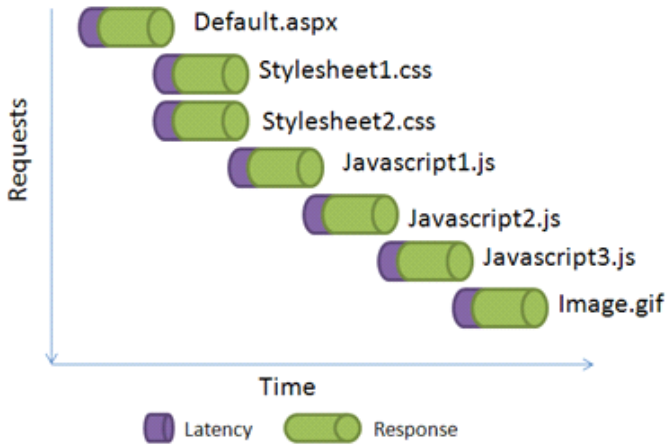
HttpMultiFileHandler 소개

리얼웹	개발본부	배성혁	2010-09-02 오후 12:42

원본 : Http handler to combine multiple files, cache and deliver compressed output for faster page load

아이디어는 이렇습니다. 일반적으로 웹 페이지를 구성하는 것은 페이지 자체의 HTML과 각종 이미지, 그 다음에 부가적으로 CSS 등의 Style 관련 파일과 Javascript 파일들입니다.

이 때 브라우저가 특정 페이지를 로드할 때, 다음과 같은 일이 벌어집니다.

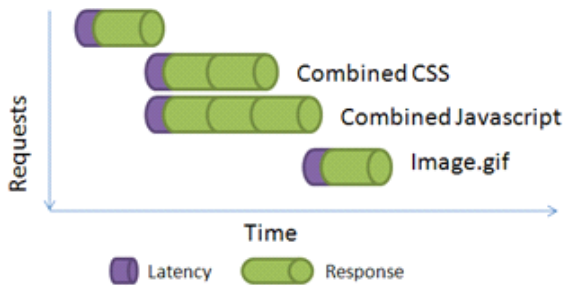


일반적인 Page Load 시의 통신 Timeline

자 보시다시피, Page의 본문이 다 다운되고 나면, 관련된 리소스들 (linked resources) 을 다운받기 위해 network round-trip을 수행합니다. 이때, 관련 리소스들이 아주 많을 경우, 한번에 최대 2개씩 다운로드를 받습니다. 그러니 여러 파일을 다운로드 받아야 할 경우에는 페이지 로딩 속도가 저하되는 것은 너무나 당연합니다.

이를 해결하기 위한 방안은 여러가지가 있습니다. (회사 도서 중에 **Ultra-Fast ASP.NET** 에 많은 방법이 있습니다.)
DNS를 모두 적어준다던가 하는 간단한 방법도 있습니다만,
여기서는 HttpHandler를 통해, **필요한 파일들을 한꺼번에 묶어서 보내는 방식**을 써보기로 합시다. 즉 다운로드 받을 파일의 갯 수를 줄여서, round-trip을 최소화 하자는 얘기입니다. (이 것만으로도 속도가 빨라지는데, 압축까지 하게 되면 더욱 빨라질 것입니다.)

아래 그림에서 보듯이, CSS 두 개가 한꺼번에 내려 받고, 3개의 Javascript 파일도 동적으로 하나로 묶여서 다운로드가 됩니다.
이렇게 되면, Page Download시간이 기존보다 2배 이상 빨라질 것입니다.



HttpMultiFileHandler를 이용하여, 여러파일을 하나로 묶어 전송한다.

자 그럼 본격적으로 어떻게 사용하는지부터 살펴봅시다.

원 저작자는 일반적으로 다운로드 할 파일들을 묶음으로 정의하는 것은 환경설정에서 수행하였습니다. 대부분의 Page가 거의 같은 파일들을 다운로드 받으므로 그렇게 해도 무방합니다.

```
    <appSettings>
        <add key="customError.Mail.From" value="sender@realweb21.com" />
        <add key="customError.Mail.To" value="admin@realweb21.com" />
        <add key="Application.Copyright" value="Copyright &amp;copy; RealWeb corp. All rights reserved." />

        <!-- HttpMultiFileHandler에서 여러 파일을 다운로드 할 수 있도록 묶음 처리했음.-->
        <add key="Set.Css" value="~/App_Themes/Default/Css1.css|~/App_Themes/Default/Css2.css" />
        <add key="Set.Javascript" value="~/Scripts/Js1.js|~/Scripts/JS2.js|~/Scripts/jquery-1.4.1.js"/>
    </appSettings>
```

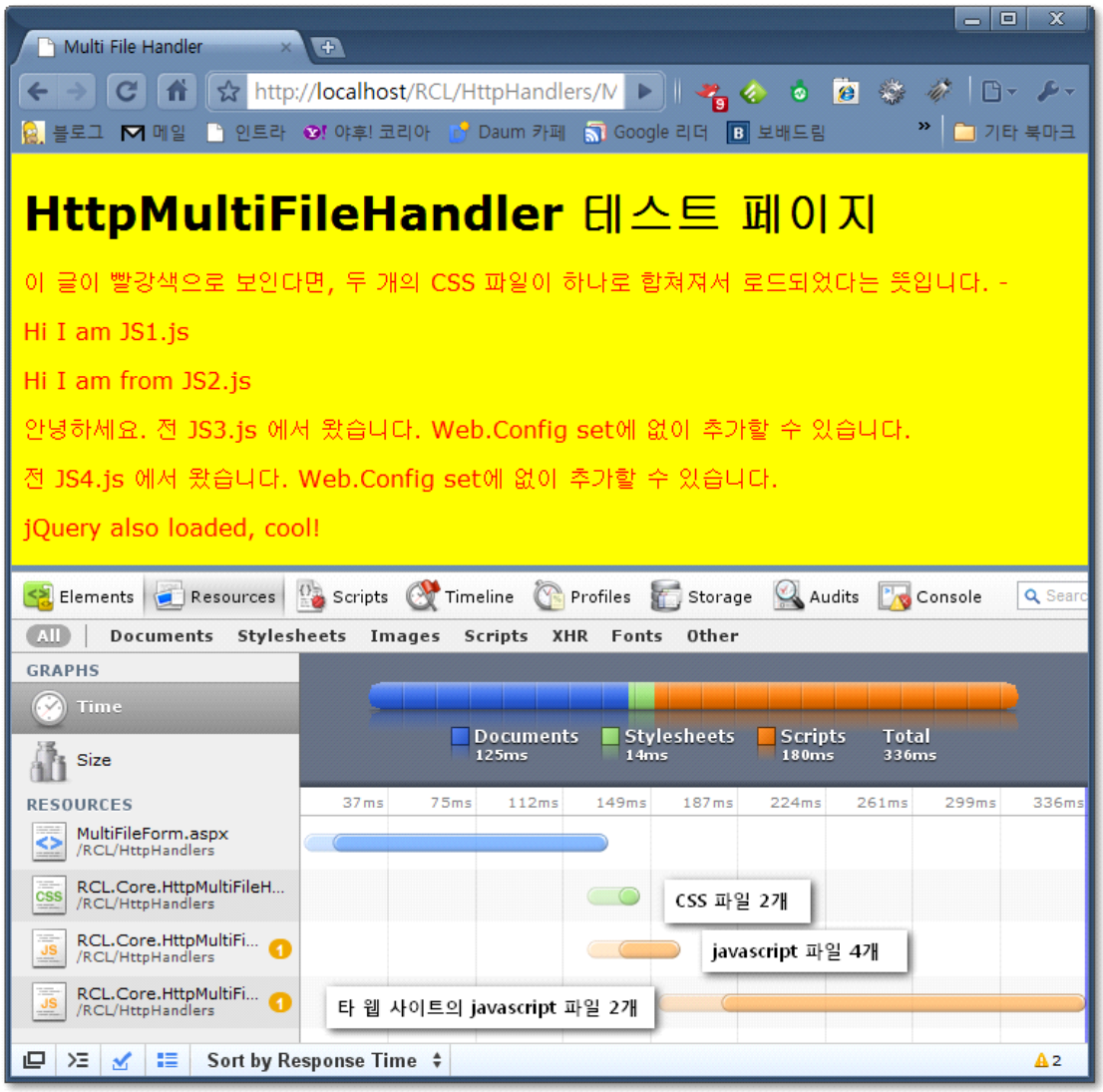
환경설정에서 appSettings 에 CSS 파일 묶음, Javascript 묶음을 설정합니다.

다음으로 실제로 위의 파일들을 하나로 묶어서 다운로드 할 수 있는 HttpHandler 를 등록합니다.

```
        <pages validateRequest="false">
        </pages>
        <httpHandlers>
            <!-- 여러 파일을 동시에 다운로드 받을 수 있도록 하는 Handler 입니다.-->
            <add verb="*" path="RCL.Core.HttpMultiFileHandler.axd" type="RCL.Core.HttpMultiFileHandler, RCL.Core" validate="false"/>
        </httpHandlers>
        <httpModules>
            <!-- Page 처리 성능 측정을 위한 모듈 -->
            <add name="PagePerformanceModule" type="RCL.Core.PagePerformanceModule, RCL.Core" />
            <add name="PageAccessLogModule" type="RCL.Core.PageAccessLogModule, RCL.Core" />
            <!-- AsyncAccessLogModule 이 다른 IHttpAsyncHandler와 충돌이 난다. -->
            <add name="AsyncAccessLogModule" type="RCL.Core.AsyncAccessLogModule, RCL.Core" />
        </httpModules>
    </system.web>
```

자 이제 모든 환경 설정은 끝났습니다.

예제 페이지를 작성하고, 페이지 구동이 어떻게 되는지 살펴봅시다.




```aspx
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="MultiFileForm.aspx.cs" Inherits="RCL.Core.WebApp.HttpHandlers.MultiFileForm" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Multi File Handler</title>
    <link type="text/css" rel="Stylesheet" href="RCL.Core.HttpMultiFileHandler.axd?S=Set.Css&T=text/css&V=1" />

    <!-- 외부사이트에서 javascript 정보를 다운로드 받습니다 -->
    <script type="text/javascript" src="RCL.Core.HttpMultiFileHandler.axd?T=type/javascript&V=3&F=http://alexgorbatchev.com/pub/sh/current/scripts/shCore.j
    </script>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1>HttpMultiFileHandler 테스트 페이지</h1>
        <p>이 글이 빨강색으로 보인다면, 두 개의 css 파일이 하나로 합쳐져서 로드되었다는 뜻입니다. - </p>

        <script type="text/javascript" src="RCL.Core.HttpMultiFileHandler.axd?S=Set.Javascript&T=type/javascript&V=2&F=~/Scripts/Js3.js|~/Scripts/Js4.js">
        </script>

        <p id="jquery">
        </p>

        <script type="text/javascript">
            $(function() {
                $("#jquery").html("jQuery also loaded, cool!");
            });
        </script>
    </div>
    </form>
</body>
</html>
```

Page 소스를 보면, 3가지 멀티 파일 묶음 다운로드가 있습니다.

1번은 web.config에서 설정된 CSS 파일들을 묶어서 다운로드 합니다. 그래서 웹페이지 화면이 노란 바탕에 빨강 글씨가 나타나게 된 것이구요.

2번은 web.config에서 설정된 javascript 파일 묶음에 추가로 필요한 파일을 "F" 인자에 할당하여 (Js3.js, Js4.js) 총 5개의 javascript 파일을 하나로 묶어서 다운로드 받습니다.

3번은 좀 특이하게 리소스 위치가 현재 웹 응용프로그램이 아닌, 외부에 있을 경우에도 리소스 들을 다운로드 받아서 하나의 파일로 묶어서 제공해 준다는 것입니다.

이제 대강 어떻게 돌아가는지는 아시겠죠?

그럼 원저작자가 성능을 위해 두 가지를 더 했는데 다음과 같은 기능입니다.

1. **전송할 리소스를 압축하여 전송한다.**
2. **서버 메모리 캐시 (HttpContext.Current.Cache)**에 파일 통합본 정보를 저장해 놓고, 다음 요청 시에 사용한다.
   물론 유효기간을 두어, 일정시간이 지나면, 폐기되도록 한다. - 이렇게 해야 어느 정도 최신 정보를 볼 수 있습니다.

위 두 가지 일은 서버 모듈을 개발하는 개발자라면 꼭 공부해 두고, 자기 것으로 만들어 보시기 바랍니다.

그럼 RCL에서는 원작자와 달리 뭘 더 추가했을까요?

1. **비동기 IO 처리를 수행**
   처리할 내용이 파일을 읽어서 응답 스트림에 쓰는 것이므로, 파일 읽기를 비동기 방식으로 수행한다면, 확장성이 보장됩니다. 특히 외부 리소스에 대해서는 WebClient 를 이용하여, 비동기 방식으로 리소스를 다운로드 받도록 하여, 확장성 및 속도를 향상 시켰습니다.

2. **병렬 처리**
   처리할 파일이 복수 개이므로, 병렬로 파일을 읽게 하여, 응답 스트림에 쓴다면, 속도 향상이 클 것입니다. 특히 CPU가 많은 서버라면 속도 향상에 많은 기여를 할 것입니다.

3. HttpHandler 자체를 **IHttpAsyncHandler**를 구현하여, 웹 응용프로그램의 확장성을 보장했습니다.
4. 파일 묶음 정의를 고정시키지 않고, 추가할 수 있도록 했습니다. (원작자는 예제니까 그런거고)
5. **멀티바이트 언어에 대한 대처**
   원저자는 영어권이라 문제가 안되지만, 멀티바이트 언어를 사용하는 환경에서는 여러 파일을 하나의 Stream으로 묶을 때 스트림의 선두번지에 멀티바이트임을 나타내는 prefix 를 제거해줘야 합니다. 이 것 때문에 한 두 시간 헤맸습니다.