**[ASP.NET MVC 4 RC的JS/CSS打包压缩功能 Scripts.Render和Styles.Render](http://www.cnblogs.com/hiflora/p/3966311.html)**

打包(Bundling)及压缩(Minification)指的是将多个js文件或css文件打包成单一文件并压缩的做法，如此可减少浏览器需下载多个文件案才能完成网页显示的延迟感，同时通过移除JS/CSS文件案中空白、批注及修改JavaScript内部函数、变量名称的压缩手法，能有效缩小文件案体积，提高传输效率，提供使用者更流畅的浏览体验。

在ASP.NET MVC 4中可以使用BundleTable捆绑多个css文件和js文件，以提高网络加载速度和页面解析速度。更为重要的是通过捆绑可以解决IE浏览器的31个CSS文件连接的限制。

ASP.NET捆绑是ASP.NET 4.5的新功能，是System.Web.Optimization命名空间下。他提供了一些ASP.NET运行性能方面的优化，比如，一个页面可能有很多CSS/JS/图片,通过灵活的应用BundleTable类，他可以帮你将文件合并压缩代码优化成一个最理想的文件，然后输出到客户端，从而提高了浏览器下载速度。

**ASP.NET MVC 4 Beta 解决方法**

在ASP.NET MVC 4 Beta时代便已内建打包压缩功能，做法是在global.asax.cs的Application\_Start加入

BundleTable.Bundles.EnableDefaultBundles();

如此，便可使用以下写法一口气将整个Scripts目录下的JS及Contents目录下所有CSS打包并压缩成单一文件案，改善网页载入效率: ([参考](http://theshravan.net/bundling-and-minification-support-in-asp-net-mvc-4/))

<script src="@Url.Content("~/Scripts/js")" type="text/javascript"></script>   
<script src="@Url.Content("~/Content/CSS")" type="text/javascript"></script>

**ASP.NET MVC 4 RC**

在ASP.NET MVC 4 RC，发现RC版在打包压缩做法上又有所革新，变得更加弹性有条理。

原本打包规则被藏在global.asax.cs Application\_Start中，RC版起则多了一个新目录App\_Start，其中包含RouteConfig.cs、FilterConfig.cs、BundleConfig.cs三个类，做法上改为[通过WebActivator启动](http://blog.davidebbo.com/2011/02/appstart-folder-convention-for-nuget.html)。新的系统配置将路由规则、过滤器及打包规则等注册逻辑由Application\_Start中拆出来，各自放在独立文件案中，管理及修改起来一目了然，架构上更漂亮。

跟Beta时代很大的差异是将JS与CSS加以群组化，分别定义出jquery, jqueryui, jqueryval, modernizr, css及themes/base/css等群组，让网页可以视需要只加载必要的JS及CSS文件群组，不像先前每次得打包整个目录，对于JS文件的加载顺序及相依性也能做较精准的调控。

而在.cshtml中，则使用Styles.Render及Scripts.Render载入BundleConfig.cs所定义的JS及CSS群组，

[复制代码](javascript:void(0);)

@Styles.Render("~/Content/themes/base/css", "~/Content/css")

@Scripts.Render("~/bundles/modernizr")

@Scripts.Render("~/bundles/jquery", "~/bundles/jqueryui","~/bundles/jqueryval")

@RenderSection("scripts", required: false)

[复制代码](javascript:void(0);)

**实例效率参考**

以MVC示例，我们在cshtml页面需要引入许多css和js文件

[复制代码](javascript:void(0);)

<script src="~/Scripts/jquery-1.8.2.js"></script>

<link href="~/Content/themes/base/jquery.ui.core.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.resizable.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.selectable.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.accordion.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.autocomplete.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.button.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.dialog.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.slider.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.tabs.css" rel="stylesheet" />

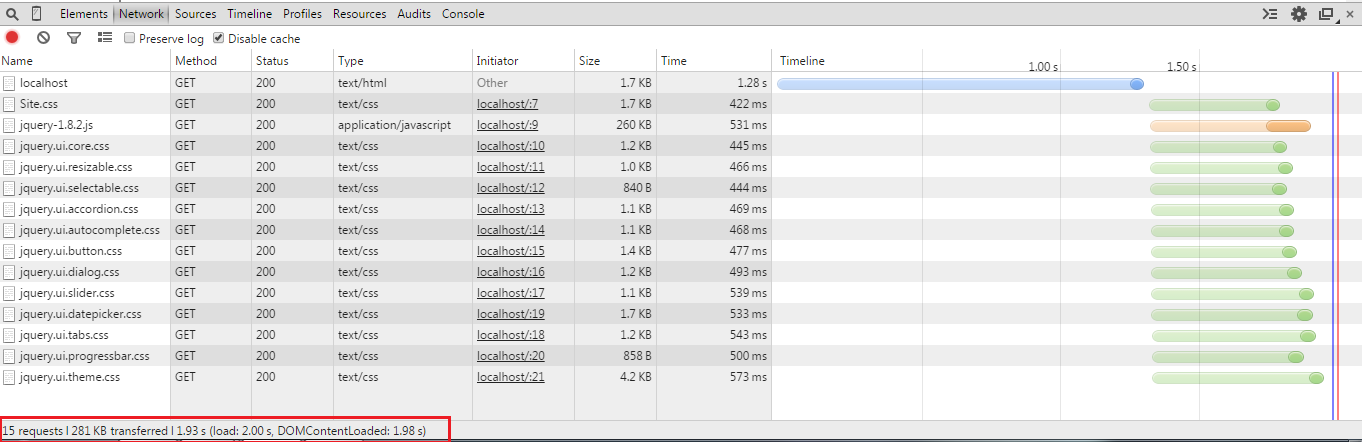
<link href="~/Content/themes/base/jquery.ui.datepicker.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.progressbar.css" rel="stylesheet" />

<link href="~/Content/themes/base/jquery.ui.theme.css" rel="stylesheet" />

[复制代码](javascript:void(0);)

这种写法在浏览页面时，效率如图，google浏览器查看：



如果使用bundling技术，使用如下代替正常的引用路径

@Scripts.Render("~/bundles/jquery")

@Styles.Render("~/Content/themes/base/css")

@RenderSection("scripts", required: false)

具体的文件需要在App\_Start->BundleConfig.cs中，把要添加的js文件include进来：

[复制代码](javascript:void(0);)

bundles.Add(new StyleBundle("~/Content/themes/base/css").Include(

"~/Content/themes/base/jquery.ui.core.css",

"~/Content/themes/base/jquery.ui.resizable.css",

"~/Content/themes/base/jquery.ui.selectable.css",

"~/Content/themes/base/jquery.ui.accordion.css",

"~/Content/themes/base/jquery.ui.autocomplete.css",

"~/Content/themes/base/jquery.ui.button.css",

"~/Content/themes/base/jquery.ui.dialog.css",

"~/Content/themes/base/jquery.ui.slider.css",

"~/Content/themes/base/jquery.ui.tabs.css",

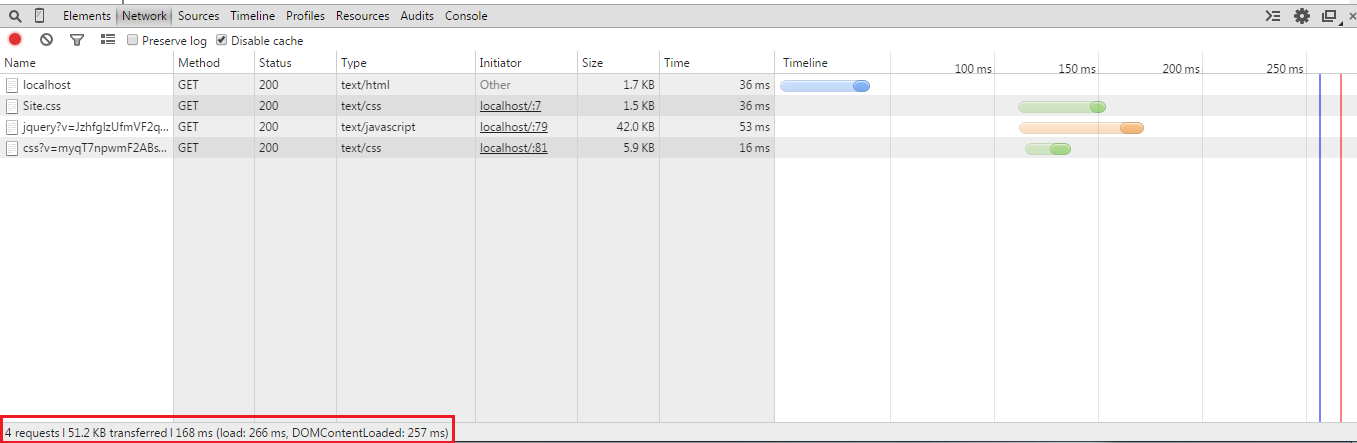
"~/Content/themes/base/jquery.ui.datepicker.css",

"~/Content/themes/base/jquery.ui.progressbar.css",

"~/Content/themes/base/jquery.ui.theme.css"));

[复制代码](javascript:void(0);)

页面访问效率如下：



对比上下的结果，可以发现

请求数：15->4

下载资源数：281K->51.2K

响应时间也会缩短。

需要注意的是，webconfig中的设置： <compilation debug="false" targetFramework="4.5" />，这里一定要设置为false，如果设置为true，在预览页面的时候，不会压缩这些文件，和正常引入文件的写法是一样的效果，主要是为了方便调试。vs2013里默认就是false的，不用做特别的设置。

参考：http://www.cnblogs.com/shanyou/archive/2012/06/22/2558580.html