**[优化网站设计（一）：减少请求数](http://www.cnblogs.com/chenxizhang/archive/2013/04/29/3050839.html)**

**前言**

网站设计的优化是一个很大的话题,有一些通用的原则,也有针对不同开发平台的一些建议。这方面的研究一直没有停止过，我在不同的场合也分享过这样的话题。

作为通用的原则，雅虎的工程师团队曾经给出过35个最佳实践。这个列表请参考

**Best Practices for Speeding Up Your Web Site** <http://developer.yahoo.com/performance/rules.html>

同时，他们还发布了一个相应的测试工具Yslow <http://developer.yahoo.com/yslow/>

我强烈推荐所有的网站开发人员都应该学习这些最佳实践，并结合自己的实际项目情况进行应用。

接下来的一段时间，我将结合ASP.NET这个开发平台，针对这些原则，通过一个系列文章的形式，做些讲解和演绎，以帮助大家更好地理解这些原则，并且更好地使用他们。

**准备工作**

为了跟随我进行后续的学习，你需要准备如下的开发环境和工具

1. Google Chrome 或者firefox ，并且安装 Yslow这个扩展组件.请注意，这个组件是雅虎提供的，但目前没有针对IE的版本。
   1. <https://chrome.google.com/webstore/detail/yslow/ninejjcohidippngpapiilnmkgllmakh>
   2. <https://addons.mozilla.org/en-US/firefox/addon/yslow/>
   3. 你应该对这些浏览器的开发人员工具有所了解，你可以通过按下F12键调出这个工具。
2. Visaul Studio 2010 SP1 或更高版本，推荐使用Visual Studio 2012
   1. <http://www.microsoft.com/visualstudio/eng/downloads>
3. 你需要对ASP.NET的开发基本流程和核心技术有相当的了解，本系列文章很难对基础知识做普及。

**本文要讨论的话题**

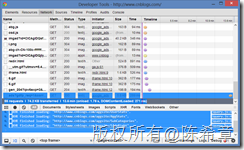
这一篇文章讨论的是第一个原则：应该尽可能加减少请求数。这个原则的说明请参考  <http://developer.yahoo.com/performance/rules.html#num_http>

我们的网页在加载的时候，为了提供更加丰富的内容和效果，除了页面本身这个请求之外，总是需要加载其他一些资源的，例如我们常见的Javascript文件，css文件，图片，甚至还会有一些Flash组件等等。这本无可厚非，但如果过多的外部请求，会很直接地降低页面加载的速度。

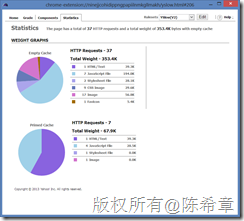
我们来看一个例子吧。例如我们经常访问的博客园的首页

[](http://images.cnitblog.com/blog/9072/201304/29153611-1b05b2754a524aa3adba2313ccc5acf5.png)

这个网页的加载需要多少次请求呢？（如果不考虑缓存的话）

[](http://images.cnitblog.com/blog/9072/201304/29153623-fb9c0379883b49ab9e958d6298e3ae1c.png)

我们看到，请求数为55个。我们进一步通过Yslow来分析，可以得到综合的报表

[](http://images.cnitblog.com/blog/9072/201304/29153630-4008698231c74e3298ba66a3177fe64b.png)[](http://images.cnitblog.com/blog/9072/201304/29153631-ec01217395764dcdb91a7210af40c57f.png)[](http://images.cnitblog.com/blog/9072/201304/29153635-5cf681a6fa3245409341ef3adcd3fd73.png)

应该说博客园的设计已经比较注意很多细节了。他们得到了B级的评分。我们再来看看其他一些主要的门户的表现吧（从左至右，依次是新浪，搜狐，腾讯，淘宝），他们都只得到D级的评分。

【备注】这些评分只是作为一个参考，不做任何的结论和推论。

[](http://images.cnitblog.com/blog/9072/201304/29153641-b320cc27fbbe4437aef313a87e119507.png)[](http://images.cnitblog.com/blog/9072/201304/29153647-9cba1ead37974097a5f16f8dc672d372.png)[](http://images.cnitblog.com/blog/9072/201304/29153653-55d7ba6b6fbf4a51b5444440c270b7bb.png)[](http://images.cnitblog.com/blog/9072/201304/29153700-246ed6ba4b164485a957a237d06e81b9.png)

**如何减少请求数?**

我们可以通过如下的几个方法来减少请求数:

1. 合并外部资源文件(如javascript,css,图片文件）
   1. 图片的合并,是将多个图片合并为一个图片,然后采用css的一些设置（background-image,background-position) 来使用它们。这个很简单实用(但是效果也是显著的）。本文将不做演示。
   2. javascript和css文件的合并，这个可能大家不太经常注意到。本文的后续部分将对此进行演示。
2. 使用Inline images 这种方式**.** 这个方式可能依赖于浏览器的实现，目前并不是所有的浏览器都支持。所以本文也不做演示。

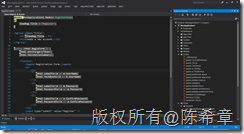
**合并javascript文件和css文件**

对于这两种文件的合并而言，我们当然可以手工去做(copy,paste)，把一个文件的内容粘贴在另外一个文件内容的底部即可。但这种方式有几个缺点：

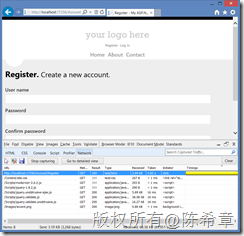
1. 破坏了原有文件的结构
2. 不同页面需要的文件组合可能是不一样的，这种情况下会需要产生多个不同的文件，而且需要比较小心地维护它们
3. 如果文件的内容发生变化，就需要重新再来一次

所以，我并不是很推荐用这种手工合并的方法，事实上，我们有更好的工具来实现, 并且在ASP.NET的一些框架(例如ASP.NET MVC)里面已经有了内置的实现。

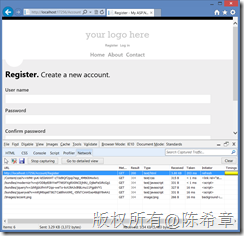
我们先来看一个例子,下面是一个典型的ASP.NET MVC项目

[](http://images.cnitblog.com/blog/9072/201304/29153706-7cdd0019277847ec948664f99b502263.png)

我找了其中的一个用户注册页面，在IE中进行查看

[](http://images.cnitblog.com/blog/9072/201304/29153711-02402a257b3b44f3b723f1c05a31e1ee.png)

我们看到默认情况下，完成这个页面其实要执行8个请求。但经过简单的处理（添加一行代码）之后，我们可以看到如下的效果

[](http://images.cnitblog.com/blog/9072/201304/29153715-15d2ee3efbfd4060b1540c69280d1dd4.png)

而且如果你细心看的话，在这个页面中请求的javascript文件似乎看起来经过了特殊的处理（路径比较特殊）。那么，这是如何实现的呢？

原来,在MVC项目中,默认会有一个所谓的BundleConfig的类，它提供了一个方法RegisterBundles，如下所示

using System.Web;

using System.Web.Optimization;

namespace MvcApplication1

{

public class BundleConfig

{

// For more information on Bundling, visit http://go.microsoft.com/fwlink/?LinkId=254725

public static void RegisterBundles(BundleCollection bundles)

{

bundles.Add(new ScriptBundle("~/bundles/jquery").Include(

"~/Scripts/jquery-{version}.js"));

bundles.Add(new ScriptBundle("~/bundles/jqueryui").Include(

"~/Scripts/jquery-ui-{version}.js"));

bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(

"~/Scripts/jquery.unobtrusive\*",

"~/Scripts/jquery.validate\*"));

// Use the development version of Modernizr to develop with and learn from. Then, when you're

// ready for production, use the build tool at http://modernizr.com to pick only the tests you need.

bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(

"~/Scripts/modernizr-\*"));

bundles.Add(new StyleBundle("~/Content/css").Include("~/Content/site.css"));

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"));

}

}

}

这个方法会在Global.asax文件中调用

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using System.Web.Optimization;

using System.Web.Routing;

namespace MvcApplication1

{

// Note: For instructions on enabling IIS6 or IIS7 classic mode,

// visit http://go.microsoft.com/?LinkId=9394801

public class MvcApplication : System.Web.HttpApplication

{

protected void Application\_Start()

{

AreaRegistration.RegisterAllAreas();

WebApiConfig.Register(GlobalConfiguration.Configuration);

FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);

RouteConfig.RegisterRoutes(RouteTable.Routes);

**BundleConfig.RegisterBundles(BundleTable.Bundles);**

//BundleTable.EnableOptimizations = false;//启用这一行，则使用Bundle的机制进行文件打包

AuthConfig.RegisterAuth();

}

}

}

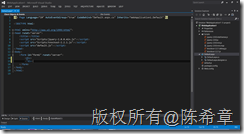
在具体的页面中，如果需要用到上述的脚本组合，则可以使用下面这样的语法来调用

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

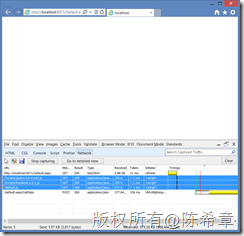
这就是所有的秘密。

那么，这个技术是不是只能使用在MVC中，在我们另外一种开发框架（ASP.NET Web Forms）中是否也可以使用呢？

答案是：可以的。而且这个技术，确实是最早就是用在ASP.NET Web Forms里面，只不过，因为这方面的文档较少，所以可能大家用的不多而已。下面是我作为演示用的一个简单的ASP.NET Web Forms的项目：

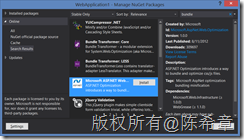
[](http://images.cnitblog.com/blog/9072/201304/29153719-4875ad192a584629b4e762f3fb3f4cb7.png)

我们看到，在页面中，我们添加了三个脚本引用，这样的话，自然在打开页面的时候，需要单独请求这三个脚本文件。

[](http://images.cnitblog.com/blog/9072/201304/29153723-63f59b5844ea4c4198d2e5dc1d30fcfe.png)

我们是否可以将这三个文件合并成一个请求呢？请跟随我来进行如下的操作

首先，添加一个组件,这是微软官方发布的System.Web.Optimization，顾名思义，这就是为了优化网络开发用的

[](http://images.cnitblog.com/blog/9072/201304/29153724-d4eae72d3f064e7eb53c724e6948bd4d.png)

按照一般的惯例，我们在项目中添加一个文件，来进行Bundle的注册

using System.Web.Optimization;

namespace WebApplication1

{

public class BundleConfig

{

public static void RegisterBundle(BundleCollection config)

{

config.Add(new ScriptBundle("~/default").Include("~/scripts/jquery-2.0.0.min.js", "~/scripts/knockout-2.2.1.js", "~/default.js"));

}

}

}

然后，我们修改Global.asax文件，添加如下的代码

using System;

using System.Web.Optimization;

namespace WebApplication1

{

public class Global : System.Web.HttpApplication

{

protected void Application\_Start(object sender, EventArgs e)

{

BundleConfig.RegisterBundle(BundleTable.Bundles);

BundleTable.EnableOptimizations = true;

}

}

}

最后，我们在页面上也做相应的修改，如下所示(请注意粗体部分）

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="WebApplication1.Default" %>

**<%@ Import Namespace="System.Web.Optimization" %>**

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

**<%= Scripts.Render("~/default") %>**

</head>

<body>

<form id="form1" runat="server">

<div>

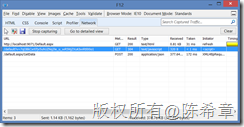
</div>

</form>

</body>

</html>

很不错，我们现在可以查看一下页面运行起来的效果

[](http://images.cnitblog.com/blog/9072/201304/29153725-f02c16e5bad34f128ba60d405e52ad63.png)

很明显，原先的三个请求，现在变成了一个请求。需要注意的是,如果我们去计算文件大小，这个合并之后的文件，体积会比之前三个文件体积总和略小一些。所以你可以理解为这里存在一定的压缩，但这个压缩比是不大的（尤其是原有的javascript文件本身就经过了压缩的情况下）。关于javascript文件或者css文件的压缩，后续会有专门的文章介绍。

上面的例子，演示的是javascript文件的合并。其实，css文件的合并也是类似的做法，区别在于要使用StyleBundle : <http://msdn.microsoft.com/en-us/library/system.web.optimization.stylebundle.aspx>

**总结**

本文介绍了网站优化的第一个原则（尽量减少请求数），我带领大家分析了为什么需要考虑这个原则，以及具体如何实现（包括在MVC和Web Forms的做法）