Mtime
前台逻辑设计文档

版本记录

修改日期	修改人	修改版本	修改内容摘要
2008-05-22	Yuanjian	1.0	初稿

目　录

1 总体说明

请首先阅读：
$/Documents/开发/逻辑设计/08 版/其他/三期开发须知.doc
$/Documents/开发/逻辑设计/08 版/其他/08 管理区页面说明.doc
特别注意：
尽量不要使用正则

2 环境搭建

2.1 概述

测试站点服务器的 IP 是：192.168.1.22，以下的站点可以直接通过修改 Host 为 192.168.1.22 来访问
测试图片服务器的 IP 是：192.168.1.123，如果不需要调试图片服务，请修改 HOST:
###
192.168.1.123 upload2.test.com
192.168.1.123 imageservice2.test.com
192.168.1.123 img21.test.cn
192.168.1.123 img22.test.cn
192.168.1.123 img31.test.cn
192.168.1.123 img32.test.cn
192.168.1.123 upload3.test.com
192.168.1.123 imageservice3.test.com
192.168.1.123 pimg3.test.com
192.168.1.123 cssserver.test.cn
192.168.1.123 cssservice.test.cn
###

2.2 主站：www.test.com

本地调试用，主要是一些页面样式和脚本

2.2.1 解决方案

$/codev3.0/MtimeMovieCommunity.root/MtimeMovieCommunity

2.2.2 搭建步凑

- 配置本地 Host
  127.0.0.1　　www.test.com
- IIS 新建站点：www.test.com
  指向本地 Community 目录，对应的 Vss 路径：
  $/codev3.0/MtimeMovieCommunity.root/MtimeMovieCommunity/Web/Community

2.3 频道页：www.test.com

2.3.1 解决方案

$/codev3.0/MtimeMovieCommunity.root/MtimeChannel.root

2.3.2 搭建步凑

参考：$/Documents/开发/逻辑设计/2011 版/2011 版频道页.docx

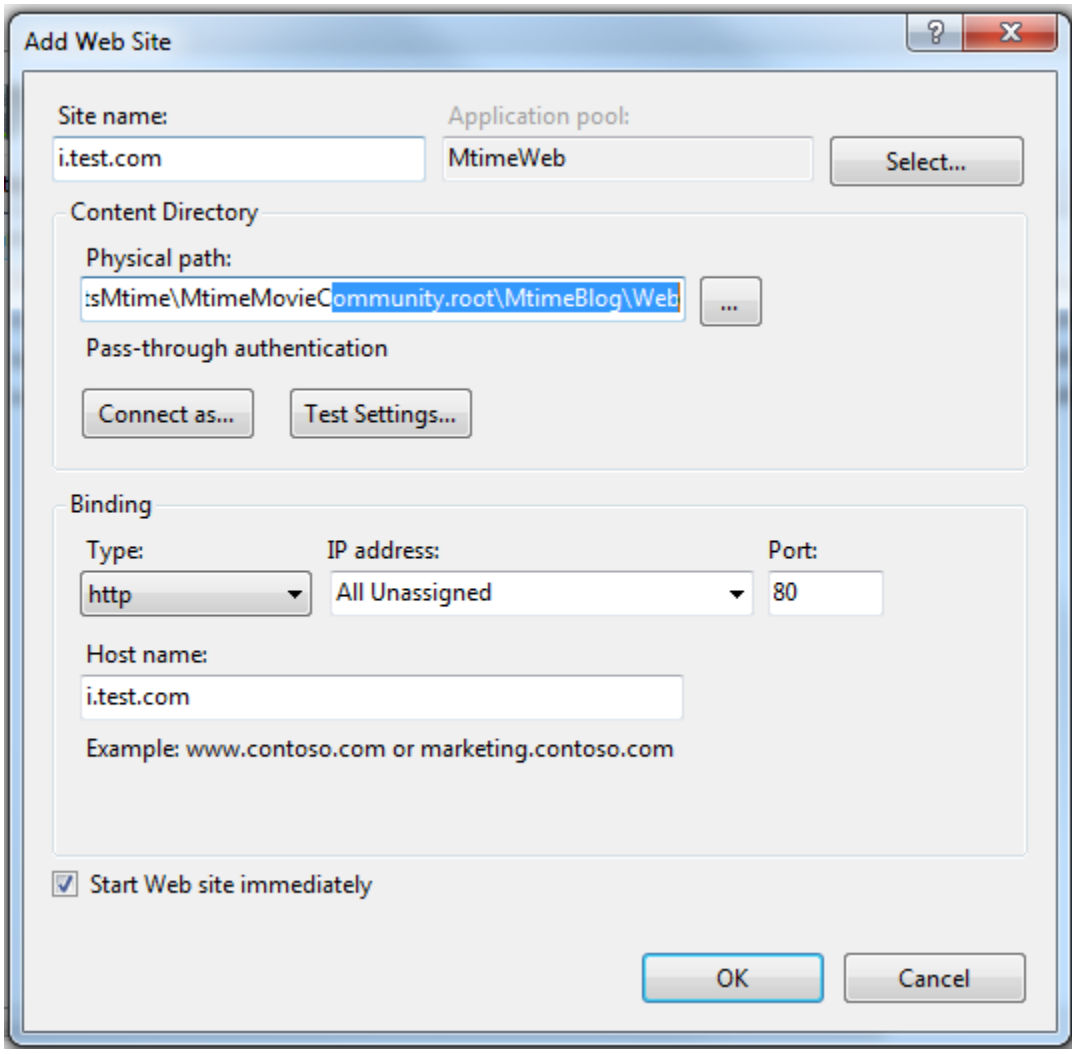
2.4 博客：http://i.test.com/gszx

2.4.1 解决方案

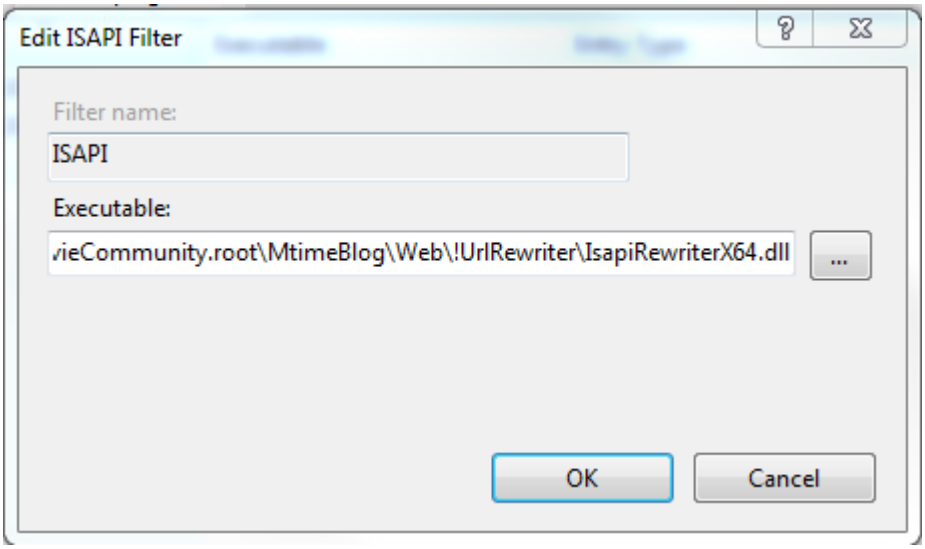
$/codev3.0/MtimeMovieCommunity.root/MtimeBlog

2.4.2 搭建步凑

- 配置本地 Host
  127.0.0.1　　i.test.com
- IIS 新建站点：i.test.com
  指向本地 Web 目录，对应的 Vss 路径：
  $/codev3.0/MtimeMovieCommunity.root/MtimeBlog/MtimeBlog/Web



- 设定 ISAPI UrlRewriter
  根据 32 位或者 64 位系统分别指定：\MtimeBlog\Web\!UrlRewriter\ IsapiRewriter.dll 或 IsapiRewriterX64.dll



注意：如果在 Windows2003 下需要指定通配符映射

2.5 我的时光：my.test.com

2.5.1 解决方案

$/codev3.0/MtimeMovieCommunity.root/MC

2.5.2 搭建步凑

- 配置本地 Host
  127.0.0.1 my.test.com
  127.0.0.1 sandbox.my.test.com
- IIS 新建站点：my.test.com
  指向本地 Web 目录，对应的 Vss 路径：
  $/codev3.0/MtimeMovieCommunity.root/MC/Web
- 在 my.test.com 里设定 ISAPI UrlRewriter
  根据 32 位或者 64 位系统分别指定：MC/Web\!UrlRewriter\ IsapiRewriter.dll 或 IsapiRewriterX64.dll
  注意：如果在 Windows2003 下需要指定通配符映射

- IIS 新建站点：sandbox.my.test.com
  指向本地 MC.Proxy.Web 目录，对应的 Vss 路径：
  $/codev3.0/MtimeMovieCommunity.root/MC/MC.Proxy.Web

2.6 群组：group.test.com

2.6.1 解决方案

$/codev3.0/MtimeMovieCommunity.root/MTimeGroup

2.6.2 搭建步凑

- 配置本地 Host
  127.0.0.1　　group.test.com
- IIS 新建站点：group.test.com
  指向本地 MTime.Web 目录，对应的 Vss 路径：
  $/codev3.0/MtimeMovieCommunity.root/MTimeGroup/MTime.Web

2.7 基础

2.7.1 开发流程

- 仔细阅读需求文档及页面
- 整理并撰写逻辑设计文档（文档中应该包含设计思路，页面各个模块说明，所使用的页面子区域名，存储过程等）
- 根据需求提出存储过程需求并统一发送给开发组长
- 根据存储过程名和 MtimeCodeGen 创建数据访问层代码
- 开发具体页面

注意：
1、如果页面中有指定 img 元素并使用了本地 images 下的图片请让页面制作人员修改为 Css 背景图的方式。
2、前台性能是第一位的，开发之前一定要仔细斟酌，尽量避免无谓的消耗。

2.7.2 引用

- 服务器端引用 Css

```
RegisterClientStyleBlock ( "IndexCss", Globals.LoadCss ( "/css/index.css" ) );
```

- //TODO:客户端引用 Css

```
$loadCss( cssPath );
```

- 服务器端引用 Js

RegisterClientScriptBlock ("IndexJs", Globals.LoadJs ("/index.js"));

- //TODO:客户端引用 js

$loadJs(cssPaths, callback);

2.7.3 公共脚本 API

- 基于 Prototype.js 1.5
- 初始化加载：

/js/prototype.js
/js/global.NativeExt.js
/js/global.Mtime.js
/js/global.PrototypeExt.js
/js/global.Utility.js
/js/global.3rdExt.js

- 某对象可用时回调

$FuncReady(functionName, callback)

- DOM 可用时回调

$DOMReady(callback)

- 页面中某 DOM 元素可用时回调

$IdReady(id, callback)

- 页面跳转

$redirect(url)

- 延迟执行某方法

$defer(functionObj/*方法*/, obj/*绑定对象*/, args/*参数*/, appendArgs/*参数*/)，例如：
原方法：objectX.functionX(paramX);
延迟执行该方法：$defer(objectX.functionX, objectX, [paramX]);

- 弹出窗口

对话框：$dialog(title, description, okCallback, cancelCallback)
提示框：$alert(title, content, callback)
特殊浮动窗口从：Windows 派生，\Community\Js\08\Window.js
Tooltip 使用：TooltipWindow，\Community\Js\08\TooltipWindow.js

2.8 具体页面开发

2.8.1 注册插件

位置在解决方案 MtimeMovieCommunity\Plugin\TemplateGenerater 目录下
运行Register.cmd

2.8.2 创建配置

打开在 Web.MovieCommunity 的 Controls 项目下 Pages 目录下的 CommunityPageConfigs.xml 文件，新建一个配置项用于处理请求的 Http 命令，例如：

```
<!--
name:请求标志
ClassName:处理该请求的页面类名
OutputCache:是否使用页面内容缓存
Duration:缓存时间,单位：180秒
VaryByParam:缓存依赖的请求参数列表，用","分割，如果为none则不依赖请求参数
IsXml:是否按照XML标准输出，默认是false
-->
<!--写网志-->
<CommunityPages name="Blogs/Manages/PostBlog" ClassName="PostBlog"
OutputCache="false" Duration="60" VaryByParam="none"  IsAsyn="false" IsXml="false" />
```

这里的 Name 代表请求的命令（如请求：http://localhost/Blogs/Manages/PostBlog.m,则命令就为 Blogs/Manages/PostBlog），ClassName 标明处理该请求的页面类名。

2.8.3 配置 UrlRewrite

修改 Url.08.config，增加访问配置，如：

```
<!-- 首页 -->
<RewriterRule>
    <LookFor>~/</LookFor>
    <SendTo>~/Index08.m</SendTo>
</RewriterRule>
```

2.8.4 创建页面类

在 Controls 项目 Pages 目录下相应模块的目录下面新建一个类，类名与在 CommunityPageConfigs.xml 配置的类名相同，名字空间为：Mtime.Community.Controls.CommunityPages，这里需要注意的是基类，我们目前主要有 5 个基类，分别是：

CommunityTemplatePage08：普通页面

CommunitySimpleTemplatePage：简单页面，如登录注册页等

CommunityManageTemplatePage ：主要用于管理区页面

CommunityUtilityTemplatePage：工具页面

CommunityServerTemplatePage：主要用于做服务的页面，不需要具体的页面文件

拷贝前台解决方案下Solution Items\Doc\CommunityPage.zip和PageSubArea.zip到My

Documents\Visual Studio 2005\Templates\ItemTemplates\Visual C#下面，则可直接创建CommunityPage和页面子区域类。

例如：

```
    sealed class Index08 : CommunityTemplatePage08
    {
        public Index08()
        {
            //页面文件路径
            pageFile = "~/Index08.html";
            //指定页面所使用的主要Css名，默认为main
            //mainCssName = "index";
            //注册Ajax方法，必须要有AjaxMethod指定的方法
            //AjaxManager.Register ( this, "Index08" );
        }

        public override void Load()
        {
            base.Load ();
            //标题及页面关键字、描述
            //this.title = GlobalResourceManager.Current.GetString( "Index_Title" );
            //RegisterClientMetaBlockWithMetaStatement ( "Keywords",
GlobalResourceManager.Current.GetString ( "Index_Meta_Keywords" ) );
            //RegisterClientMetaBlockWithMetaStatement ( "Description",
GlobalResourceManager.Current.GetString ( "Index_Meta_Description" ) );
        }

        public override void Render()
        {
            base.Render ();
        }

        /*
        [AjaxMethod]
        public string HelloWorld()
        {
        }
        */
    }
```

2.8.5 添加具体页面文件

根据在页面类中指定的文件路径，Web.MovieCommunity项目建立具体页面文件，格式如下，注意大小写，特别需要注意的是该**文件必须为UTF-8编码**，如：

```
<CommunityPage>
<Title>标题</Title>
<MetaContent>
<![CDATA[
<!--Meta内容-->
 ]]>
</MetaContent>
<StyleContent>
<![CDATA[
 ]]>
</StyleContent>
<ScriptContent>
<![CDATA[
<!--脚本内容-->
//-->
</script>
 ]]>
</ScriptContent>
<Content>
<![CDATA[
<!--实际的页面内容-->
 ]]>
</Content>
</CommunityPage>
```

2.8.6 服务器端页面方法

- 页面构造函数（请尽量让构造轻，因为 Ajax 回发时也会调用它）

```
//指定页面文件路径
pageFile = "~/Index08.html";
//指定主要使用的Css文件
mainCssName = "default";
//注册改页需要Ajax方法
AjaxManager.Register ( this, "Index08" );
```

- 输出内容

```
ReplaceTagString( string tag, string controlString )
```

如：ReplaceTagString("<!--Tag1-->", "替换后的值")

- 输出 Js：尽量在服务器端不要输出执行脚本

RegisterClientScriptBlock(string scriptName, string scriptContent)

- 输出 Css

RegisterClientStyleBlock(string styleName, string styleContent)

- 输出 Meta

RegisterClientMetaBlock(string metaName, string metaContent)

- 输出广告块

RegisterAdvertisementBlock(string adContent)

- 标题及页面关键字、描述（如果是静态内容可以直接写在页面上）

this.title = GlobalResourceManager.Current.GetString("Index_Title");
RegisterClientMetaBlockWithMetaStatement ("Keywords",
GlobalResourceManager.Current.GetString ("Index_Meta_Keywords"));
RegisterClientMetaBlockWithMetaStatement ("Description",
GlobalResourceManager.Current.GetString ("Index_Meta_Description"));
注意这里的：ResourceManager 需要根据模块选择

2.8.7 创建 View

为了便于开发和维护，把针对 HTML 的拼接单独处理，故开发一个轻量级的 HTML 生成工具。

- 首先注册插件，见 1.2.1
- 在 Controls 项目下，对应的使用类下新建 HTML 文件，如：M08_R_Index_TopRecommend_View.htm，并设置该文件的属性中：Custom Tool 为 Mtime.HtmlTemplate，如果设置 CustomTool 的 Namespace 为?^，则可不倒入默认的名字空间
- 输入 html 代码后保存，则会生成对应文件 M08_R_Index_Movie_LeftRecommend_View.cs，该文件主要是提供一个 Render 方法，便于生成“生成 HTML”的代码。

如：

```
public static string Render()
{
    StringBuilder buider = new StringBuilder ( 828 );
  ......
    return buider.ToString ();
}
```

- HTML 代码规则如下：

1、命令

<%= %>：用与显示的 c#命令，生成工具会它的内容作为 HTML 的一部分
<% %>：直接的 c#命令，生成工具不会对它的内容进行处理

<%@ %>：提供给 render 的参数，生成工具会把它的内容作为 render 的参数
<%? %>：用于导入命名空间，生成工具会把它的内容作为生成类的名字空间
<%: %>：设定生成类的名字空间
注意：<%@ %>和<% %><%? %>都必须单独作为一行编写

2、例子：

```
<%@ CommonSpecialObjectInfo mainRecommend %>
<%@ CommonSpecialObjectCollection listRecommend %>
<%@ string imageServer %>
<div class="mb10">
    <div class="clearfix">
        <a href="<%= mainRecommend.Url %>" target="_blank" class="img_outter">
            <img src="<%= imageServer + mainRecommend.Pic1Url %>"" alt="<%=
mainRecommend.Title %>" /></a>
        <h3>
            <a href="<%= mainRecommend.Url %>" target="_blank" ><%=
mainRecommend.Title %></a>
        </h3>
        <p class="lh180 mr10">
            <%= mainRecommend.Memo %>
        </p>
    </div>
    <% for ( int i = 0, count = listRecommend.Count; i < count; i++ ) %>
    <% { %>
    <%      CommonSpecialObjectInfo recommend = listRecommend [i]; %>
    <h5>
        <a href="<%= recommend.Url %>" target="_blank"><%= recommend.Title %></a>
    </h5>
    <p class="bgline">
        <%= recommend.Memo %>
    </p>
    <% } %>
</div>
```

生成的代码：

```
        public static string Render( CommonSpecialObjectInfo mainRecommend,
CommonSpecialObjectCollection listRecommend, string imageServer )
        {
            StringBuilder buider = new StringBuilder ( 828 );
            buider.Append ( @"<div class=""mb10"">" );
            buider.Append ( @"<div class=""clearfix"">" );
            buider.AppendFormat ( @"<a href=""{0}"" target=""_blank"" class=""img_outter"">",
mainRecommend.Url );
            buider.AppendFormat ( @"<img src=""{0}""" alt=""{1}"" /></a>", imageServer +
mainRecommend.Pic1Url, mainRecommend.Title );
            buider.Append ( @"<h3>" );
            buider.AppendFormat ( @"<a href=""{0}"" target=""_blank"" >{1}</a>",
mainRecommend.Url, mainRecommend.Title );
            buider.Append ( @"</h3>" );
            buider.Append ( @"<p class=""lh180 mr10"">" );
            buider.AppendFormat ( @"{0}", mainRecommend.Memo );
```

```
            buider.Append ( @"</p>" );
            buider.Append ( @"</div>" );
            for ( int i = 0, count = listRecommend.Count; i < count; i++ )
            {
                CommonSpecialObjectInfo recommend = listRecommend [i];
                buider.Append ( @"<h5>" );
                buider.AppendFormat ( @"<a href=""{0}"" target=""_blank"">{1}</a>",
recommend.Url, recommend.Title );
                buider.Append ( @"</h5>" );
                buider.Append ( @"<p class=""bgline"">" );
                buider.AppendFormat ( @"{0}", recommend.Memo );
                buider.Append ( @"</p>" );
            }
            buider.Append ( @"</div>" );
            return buider.ToString ();
        }
```

参考：Controls\PageSubArea\08\Indexs\View 下的文件

2.8.8 访问规则

修改 Host 文件，增加：127.0.0.1　　m08.test.com
使用 m08.test.com 来访问

3 推荐位及页面子区域

3.1 枚举定义

\MtimeMovieCommunity\Constants\v3\CommendBulletin\PageSubAreaName.cs

3.2 使用

3.2.1 创建页面子区域代码

位置：\MtimeMovieCommunity\Controls\PageSubArea\

3.2.2 页面调用

- 根据页面返回子区域列表

PageSubAreas:GetPageSubAreas(PageNameEN pageName)

- 根据页面和子区域名返回子区域列表

PageSubAreas:GetPageSubAreas(PageNameEN pageName, params PageSubAreaName [] pageSubAreaNames)

- 根据子区域名返回子区域列表

PageSubAreas:GetPageSubAreas(params PageSubAreaName [] pageSubAreaNames)

3.2.3 创建页面子区域时调用

- 根据页面子区域的名称获取对应的推荐对象和推荐对象的关联对象列表

PageSubAreas:
GetCommonSpecialObjects(string pageSubAreaName, out CommonSpecialObjectCollection commonSpecialObjects, out CommonSpecialObjectDetailCollection commonSpecialObjectDetails)
GetCommonSpecialObjects(string pageSubAreaName, int topN, out CommonSpecialObjectCollection commonSpecialObjects, out CommonSpecialObjectDetailCollection commonSpecialObjectDetails)

3.2.4 通用获取对象的方法

Components:RelatedObjs
注意：这些方法不要在前台页面或组件中使用

4 广告位

4.1 普通广告位

4.1.1 说明

编辑在后台输入 HTML，创建该页面子区域时需要设定类型为普通广告位。
如：

```
<!--banner-->
<div class="mtb5">
    <div class="outter">
    <div class="inner"><a href="http://www.mtime.com/my/help/blog/1187054" target="_blank"><img src="http://www.mtime.com/act/narnia2/narnia2.jpg" alt="Narnia2" /></a></div>
    </div>
</div>
```

```
<!--end banner-->
```

或

```
<object width="500" height="60"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#versi
on=5,0,0,0" classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000">
          <param value="http://www.mtime.com/tg/BD/Disney/gjbz2_500_60.swf"
name="movie"/>
          <param value="high" name="quality"/>
          <param value="opaque" name="wmode"/>
          <embed width="500" height="60"
pluginspage="http://www.macromedia.com/go/getflashplayer"
type="application/x-shockwave-flash" swliveconnect="true" quality="high"
src="http://www.mtime.com/tg/BD/Disney/gjbz2_500_60.swf" wmode="transparent"/>
          </object>
```

前台在页面加载完之后再加载其具体内容或执行JS。

4.1.2 使用

前台使用基类中的 RegisterAdvertisementBlock (area.Name, area.HtmlContent);方法来注册，推荐使用基类中的：

```
protected void RenderPageSubAreaHtmls( PageSubAreaCollection areas )
```

或者

```
protected void RenderPageSubAreaHtml(PageSubAreaCollection areas, params
PageSubAreaName[] names )
```

它能自动判断推荐位和广告位来分别输出

4.2 特殊广告位

4.2.1 说明

编辑在后台输入可执行脚本，创建该页面子区域时需要设定类型为普通广告位。

如：

```
<script type="text/javascript">
        <!--
            window.open( "http://www.mtime.com" );
        //-->
        </script>
```

4.2.2 使用

前台使用基类中的 RegisterSpecalAdvertisementBlock (area.HtmlContent);方法来注册，推荐使用基类中的：

```
protected void RenderPageSubAreaHtmls( PageSubAreaCollection areas )
```

或者

```
protected void RenderPageSubAreaHtml(PageSubAreaCollection areas, params PageSubAreaName[] names )
```

它能自动判断推荐位和广告位来分别输出

5 客户端脚本及控件

请参见：$/Documents/开发/新员工必读/Mtime 前端-脚本开发规范.doc

6 问题及注意事项

6.1 一些常用方法

6.1.1 过滤客户端输入的字符串

StringHelper.CleanInputString(string inputString)

6.1.2 过滤客户端输入的 HTML 字符串

StringHelper.GetCleanHTMLStringFromHtml(string inputString)

6.2 操作权限问题

6.2.1 问题描述

当用户对数据进行添删改时，我们一定要对用户是否对该数据拥有可以更改的权限进行判断，不能简单的认为“用户没权限，那么他在页面上就会看不到”，问题在于他可以操作别的用户的数据，以下是简单的分析，
对于页面上的接口，如下加红的代码：

```
<script type="text/javascript">
var UserIndexBlog = {
```

```
"GetBlogsAndReplies": function(param0, param1, param2, param3, param4, clientCallBack) {
        return Mtime.Component.Ajax.callBack('Mtime.Community.Controls.UserIndexBlog',
'GetBlogsAndReplies', [param0,param1,param2,param3,param4], clientCallBack,
'/my/goodidea/?Ajax_CallBack=true', 'post', '6000');
},

"DeleteBlog": function(param0, param1, param2, param3, clientCallBack) {
        return Mtime.Component.Ajax.callBack('Mtime.Community.Controls.UserIndexBlog',
'DeleteBlog', [param0,param1,param2,param3], clientCallBack,
'/my/goodidea/?Ajax_CallBack=true', 'post', '6000');
},

"DeleteBlogComment": function(param0, param1, clientCallBack) {
        return Mtime.Component.Ajax.callBack('Mtime.Community.Controls.UserIndexBlog',
'DeleteBlogComment', [param0,param1], clientCallBack, '/my/goodidea/?Ajax_CallBack=true', 'post',
'6000');
}

};
</script>
```

用户可以通过 POST 访问：**http://www.mtime.com/my/goodidea/?Ajax_CallBack=true 传递相应的参数上来调用 DeleteBlog 方法**

对于该请求，我们实际的处理是：

```
        [AjaxMethod]
        public bool DeleteBlog ( int blogId , int blogType , int relatedId , int relatedObjId )
        {
            bool isSuccess = false;
            if ( CommunityContext.GetCurrent ().IsUserLogin )
            {
                string typeString = Convert.ToString ( blogType );
                Blogs.DeleteBlog ( userId, blogId.ToString (), categoryId, typeString, out
isSuccess );
                if ( isSuccess )
                {
                    //删除点评或一句话影评时需要同步 Cache 库的点评数
                    if ( blogType == ( int )
Mtime.Community.Components.Constants.BlogType.Comment || blogType == ( int )
Mtime.Community.Components.Constants.BlogType.Opinion )
                    {
                        Blogs.RemoveMovieOrPersonComment ( relatedId, relatedObjId );
                    }
                }
            }
            return isSuccess;
        }
```

跟踪一下调用删除日志的方法，如下：

Commponents::Blogs:

```
        public static void DeleteBlog ( int userId, string blogIds, int blogCategoryId, out bool
isSuccess )
```

```
    {
        SQLServerDAL.Blogs.Blog_BlogDelById (  blogIds,out isSuccess );
```

SQLServerDAL::Blogs：

```
    public static void Blog_BlogDelById( string blogIds, out bool isSuccess )
    {
        SqlParameter [] parms = new SqlParameter []
        {
            new SqlParameter( "@blogids", SqlDbType.NVarChar ),
            new SqlParameter( "@IsSuccess", SqlDbType.Bit )
        };
        parms [0].Value = blogIds;
        parms [1].Direction = ParameterDirection.InputOutput;
        SqlHelper.ExecuteNonQuery ( SqlHelper.CONN_STRING_CORE,
CommandType.StoredProcedure, SPNAME_m3_Blog_BlogDelById, parms );
        isSuccess = SafeConvert.ToBoolean ( parms [1].Value );
    }
```

我们可以看到在实际的处理中我们并未对该用户的实际权限做任何判断，换句话说，只要我登陆了，我可以删除任何用户的任何日志！

6.2.2 解决方案

把用户登录后的 UserId（不能让客户端传递 UserId）传到服务器端或存储过程中进行判断，确认发起者的权限。

6.3 如何直接调试线上页面的 JavaScript 和 CSS

由于我们线上的 js 和 css 都经过了压缩和混淆，调试起来不太方便，大家可以安装 Fiddler（http://www.fiddlertool.com/dl/Fiddler2Setup.exe）并利用它的 AutoResponder 功能来帮助调试。

6.3.1 问题描述

报错：首页有样式及脚本错误 http://www.mtime.com

6.3.2 解决方案

1、初步怀疑是以下两个文件导致：
http://i.mtime.cn/20090312155638/08/home.js
http://i.mtime.cn/20090312155638/css/default.css
2、设置 Fiddler 的 AutoResponder，把对上述两个文件的请求转发到本机的开发文件

exact:http://i.mtime.cn/20090312155638/08/home.js -> G:\ProjectsMtime\MtimeMovieCommunity.root\MtimeMovieCommunity\Web\Community\08\home.js
exact:http://i.mtime.cn/20090312155638/css/default.css –> G:\ProjectsMtime\MtimeMovieCommunity.root\MtimeMovieCommunity\Web\Community\Css\default.css
3、跟开发时一样修改本机的 home.js，default.css 后保存并刷新首页

以上实例请用 IE 测试，更详细的描述及 Firefox 下的使用见：http://lifesinger.org/blog/?p=40

6.4 跨域问题

6.4.1 解决方案

（1）服务器端：跨域的请求在 AjaxMethod 里增加属性：CrossDomain = true
如：

```
#region 新建日志
[AjaxMethod(CrossDomain = true)]
public void RemoveCreateBlogCache(int userId, int categoryId )
{
    RemoveCaches.CreateBlog ( userId, categoryId );
}
#endregion
```

（2）客户端：跨域的请求使用 Mtime.Component.Ajax.get 方法，其余参数同 Mtime.Component.Ajax.callBack
如：

```
server: {
    post: function( blogId, title, body, forewordBody, postTime, author, permission, allowComment, isTop, referObjId, referObjType, referObjTitle, referObjAlignment, isComment, categorys, tags, groups, groupTopics, voteId, approveStatus, clientCallBack, postButton ) {
        return Mtime.Component.Ajax.callBack( 'Mtime.MemberCenter.Pages.BlogService', 'Post', [blogId, title, body, forewordBody, postTime, author, permission, allowComment, isTop, referObjId, referObjType, referObjTitle, referObjAlignment, isComment, categorys, tags, groups,
```

```
groupTopics, voteId, approveStatus], clientCallBack, '/service/blog.mc', 'post', '20000', postButton );
            },

        //清除发日志的缓存
            removeCreateBlogCache: function( userId, categoryId, clientCallBack ) {
                return Mtime.Component.Ajax.get( 'Mtime.Community.Controls.CommunityPages.RemoveCacheService', 'RemoveCreateBlogCache', [userId,categoryId], clientCallBack, '/utility/RemoveCacheService.m', 'get', '20000' );
            },

        //清除编辑日志的缓存
            removeEditBlogCache: function( userId, categoryId, blogId, clientCallBack ) {
                return Mtime.Component.Ajax.get( 'Mtime.Community.Controls.CommunityPages.RemoveCacheService', 'RemoveEditBlogCache', [userId,categoryId,blogId], clientCallBack, '/utility/RemoveCacheService.m', 'get', '20000' );
            }
```

调用方式：

```
            this.server.removeCreateBlogCache( userVariables.userData.UserId, this.lastBlog.categorys, function() {
                //默认所有的跨域请求的相应内容的变量名为result，其余同普通AJAX请求
                if ( result.value ) {
                    alert( "清除完成" );
                }
            } );
```