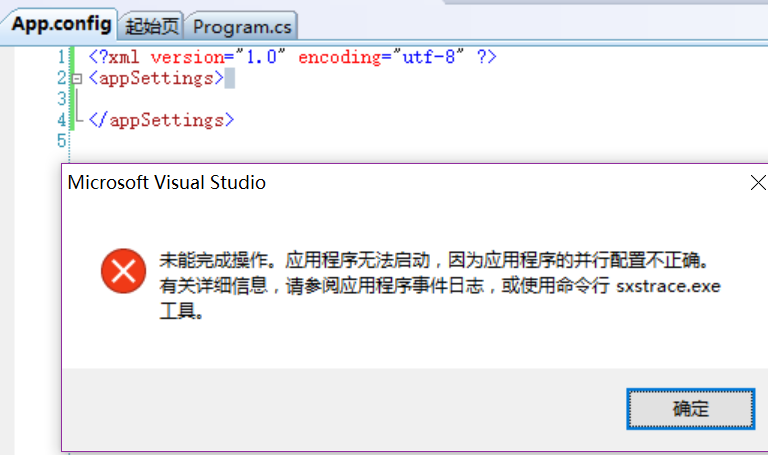
新建一个控制台程序，app.config没有设置对。能生成成功，但是无法运行



物流登陆程序报错，一查是类中的静态成员初始化异常

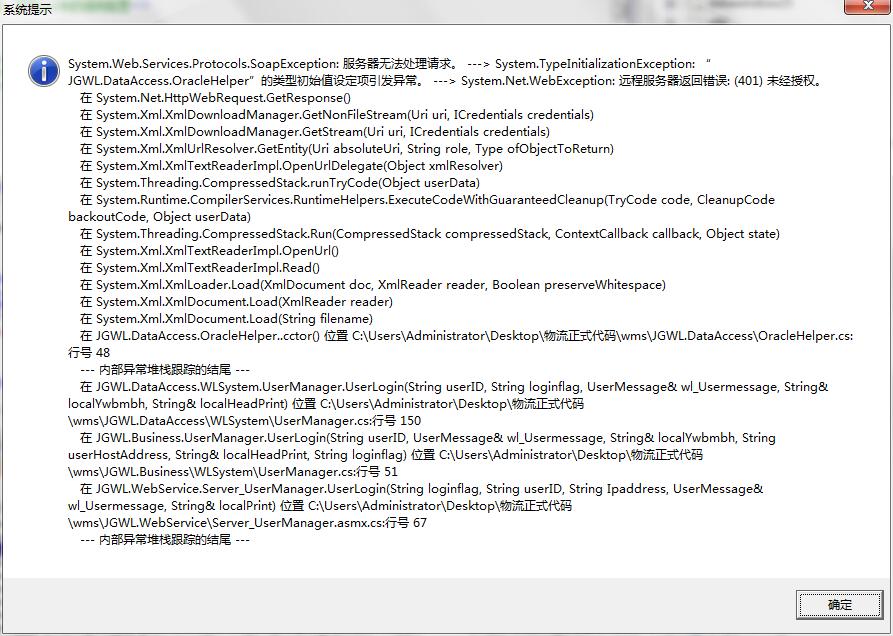
System.Web.Services.Protocols.SoapException: 服务器无法处理请求。 ---> System.TypeInitializationException: “JGWL.DataAccess.OracleHelper”的类型初始值设定项引发异常。 ---> System.Net.WebException: 远程服务器返回错误: (401) 未经授权。

public static XmlDocument dataAccessXml;

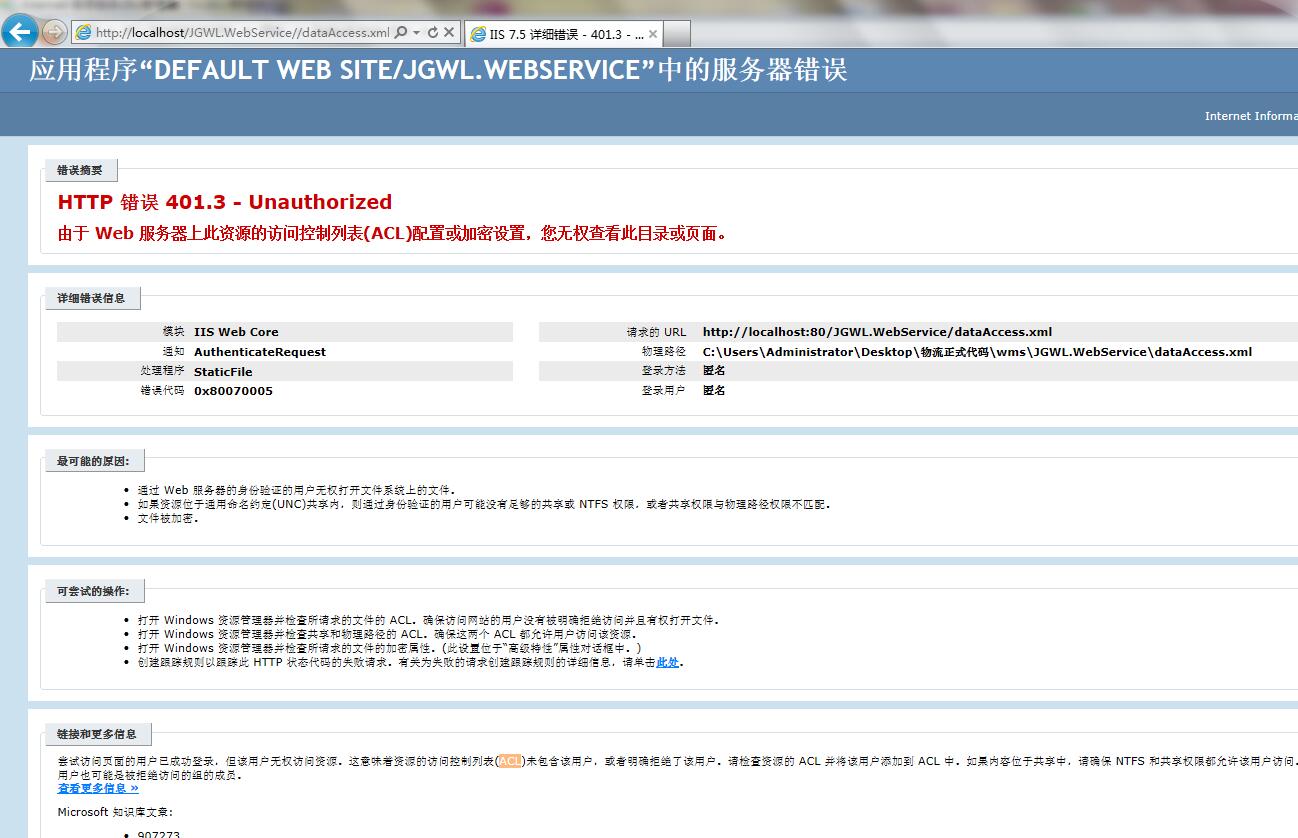
string path = "http://localhost/JGWL.WebService\\dataAccess.xml";

dataAccessXml = new XmlDocument();

dataAccessXml.Load(path);



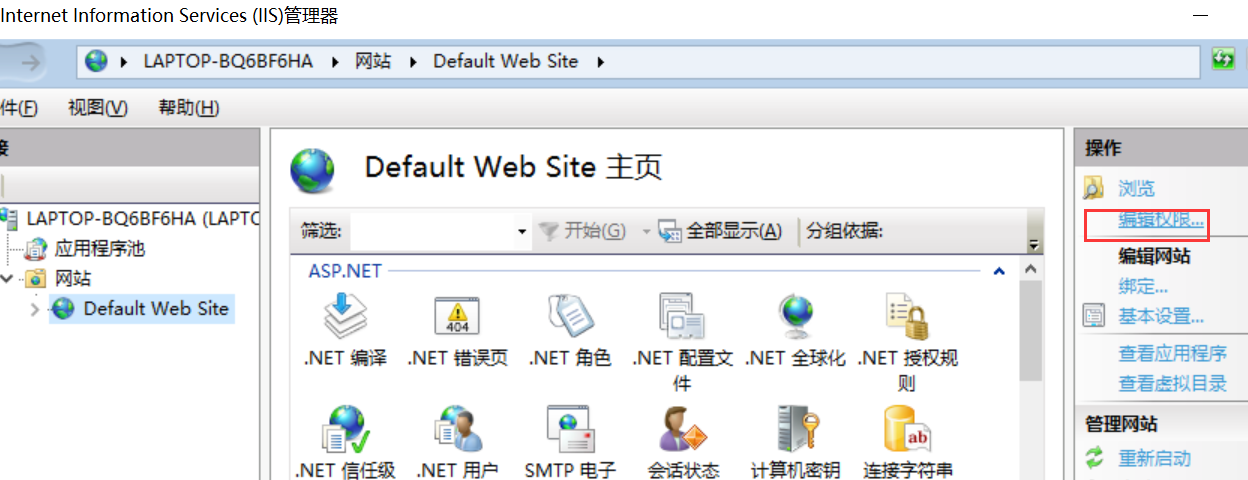
发现在iis中无法访问这个xml文件



网上查

在IIS中 依次执行如下操作：

网站——编辑权限——共享（为了方便可以直接将分享对象设置为everyone）——安全（直接勾选 everyone ）——应用——确定。



同时还右键xml文件，在属性里的安全设置了everyone的可写可读权限。最后可以了。但不知道是哪个方法解决的问题

SQLERRM是一则函数。oracle sqlerrm函数 sqlerrm函数返回指定[错误代码](http://baike.baidu.com/item/%E9%94%99%E8%AF%AF%E4%BB%A3%E7%A0%81)的错误信息。

sqlcode和sqlerrm是不能直接在sql语句中使用，必须先将其赋给变量后，才能在sql语句中使用，如下。

Error: PL/SQL: ORA-00904: "SQLERRM": 标识符无效

Line: 93

Text: update t\_changehw set errtext=sqlerrm where oldhw=c1.oldhw;

在函数中创建的timer也能每次都触发Tick事件

private void Form1\_Load(object sender, EventArgs e)

{

System.Windows.Forms.Timer timer1 = new Timer();

timer1.Interval = 1000;

timer1.Tick += new System.EventHandler(this.timer1\_Tick);

timer1.Enabled = true;

}

private void timer1\_Tick(object sender, EventArgs e)

{

MessageBox.Show("h");

}

在JGWL.WebService下创建Global.asax文件，程序启动时启动定时器。要有访问webservice的操作才会才会启动Application\_Start。访问RFserver这个webservice不会触发JGWL.WebService的Application\_Start。应用程序池的闲置超时时间到了以后工作进程关闭。定时器一起关闭

namespace JGWL.WebService

{

public class Global : System.Web.HttpApplication

{

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

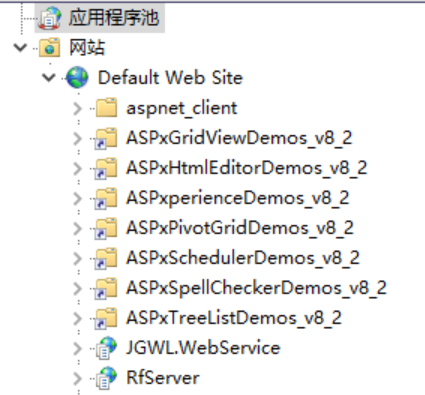
{

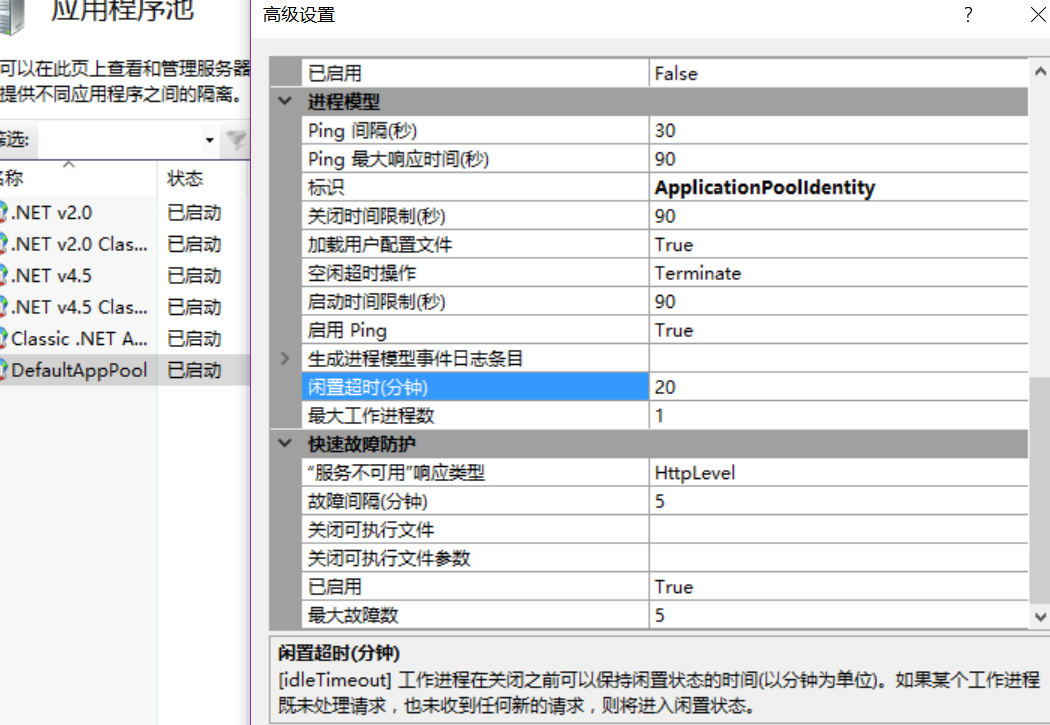
System.Timers.Timer evtTimer = new System.Timers.Timer(2000);

evtTimer.Elapsed += new System.Timers.ElapsedEventHandler(OnEvtTimer);

evtTimer.Enabled = true;

}





**Application.AddMessageFilter 方法 (IMessageFilter)**

Use a message filter to prevent specific events from being raised or to perform special operations for an event before it is passed to an event handler. Message filters are unique to a specific thread.

To prevent a message from being dispatched, the value parameter instance that you pass to this method must override the [PreFilterMessage](https://msdn.microsoft.com/zh-tw/library/system.windows.forms.imessagefilter.prefiltermessage.aspx) method with the code to handle the message. The method must return **false**.

As you will have seen from the .NET Windows namespace, mouse events are provided individually to the form or control that the mouse is over when the event occurs. So how can you detect a mouse event over any control or form regardless of where it occurs? The answer is to filter all messages sent to the application's Message Loop. suffice to say that to use this technique you implement the IMessageFilter interface which then receives notifications of a mouse event over any object.

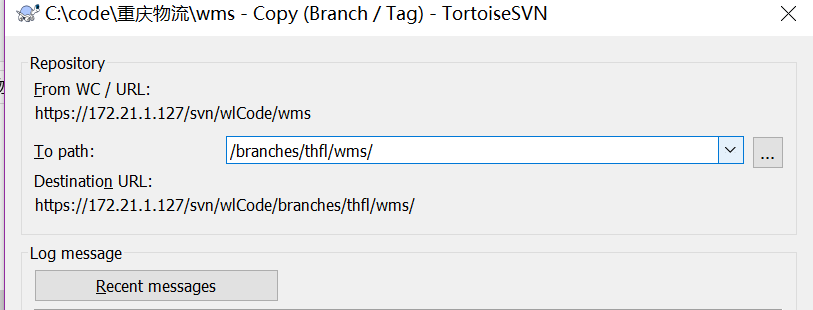
# Control.Bounds 屬性

Gets or sets the size and location of the control including its nonclient elements, in pixels, relative to the parent control.

通过controls.add添加的控件的location是相对父控件。其他new出来的form相对于屏幕？

Svn 的branch。要将wms文件夹 branch到 brances/thfl 中。首先要建立thfl文件夹

然后提交到svn



https://stackoverflow.com/questions/3612443/calling-a-webservice-async

You are correct that Async/Completed will block an asynchronous page. The method by which they "block" the asynchronous page is wrapped up in SynchronizationContext; in particular, each asynchronous page has a pending operation count which is incremented by Async and decremented after Completed.

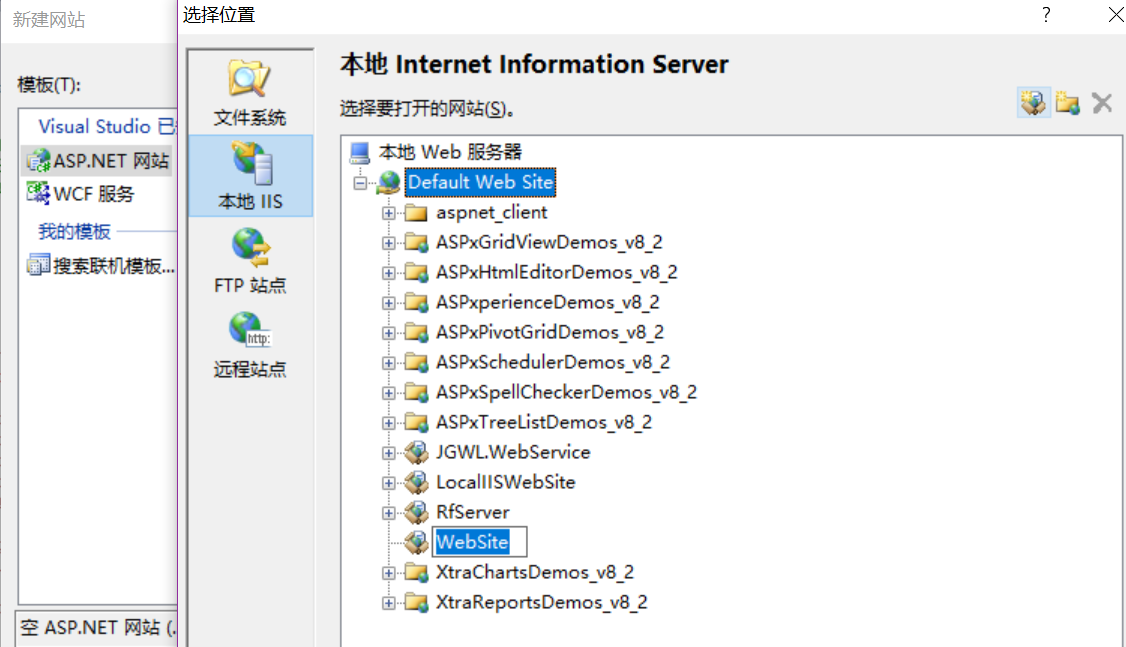
# 演练：在 Visual Web Developer 中创建本地 IIS 网站

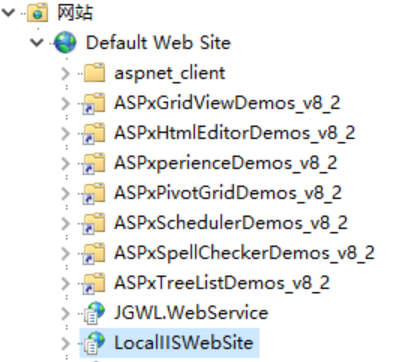
https://msdn.microsoft.com/zh-cn/library/a1zz9df4(v=vs.100).aspx

这些网站可在不同位置保存网页和其他文件。为定位网页和其他文件，可以有下面的选项：

* 在本地硬盘上的一个文件夹中，这称为文件系统网站。
* 作为 Microsoft Internet 信息服务 (IIS) 的本地副本下的 Web 应用程序，这称为本地 IIS 网站。

新建网站会在iis中建立网站



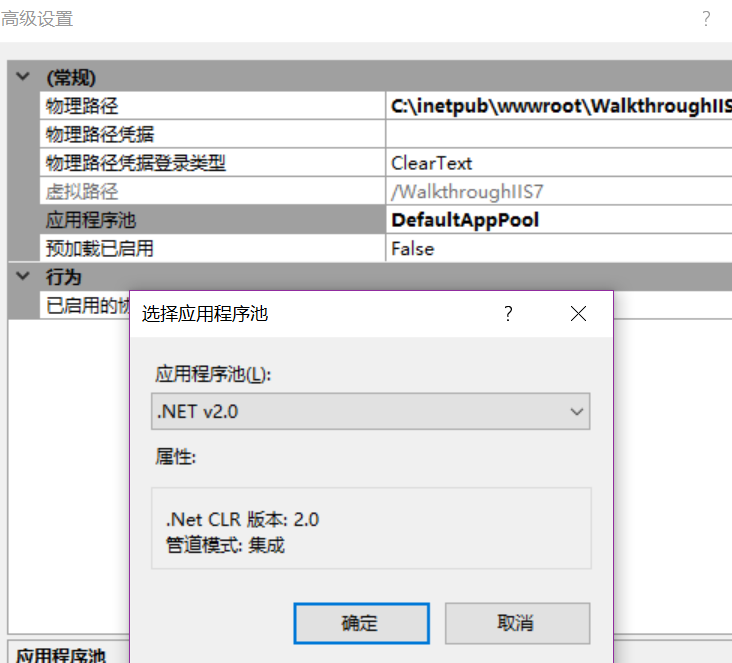


### 测试本地 IIS 网站

1. 在 Visual Web Developer 中，打开 Default.aspx 页。
2. 按 Ctrl+F5 运行该页。

该页在浏览器中打开。请注意，浏览器中的 URL 是“http://localhost/LocalIISWebSite/default.aspx”。页的请求发至 localhost（不带端口号），该请求由 IIS 处理。

尚未在 Web 服务器上注册 ASP.NET 2.0。为了使网站正确运行，可能需要手动将 Web 服务器配置为使用 ASP.NET 2.0，按 F1 可了解更多详细信息。



该模块中定义的自定义标头将显示在浏览器中。在 ASP.NET 页中，无法访问由 IIS 定义的自定义标头，因为此标头信息是在已经将页内容呈现到流之后添加的。但是，使用对 HTTP 通信进行监视的工具（例如，Fiddler）可以确认该标头已设置。

模块中添加的标头



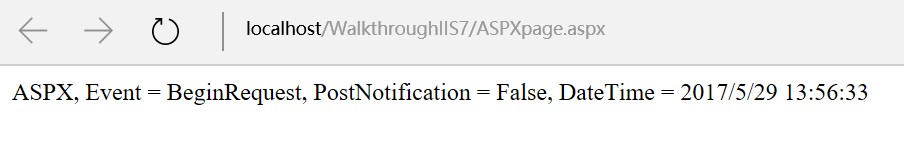
<form id="form1" runat="server">

<div>

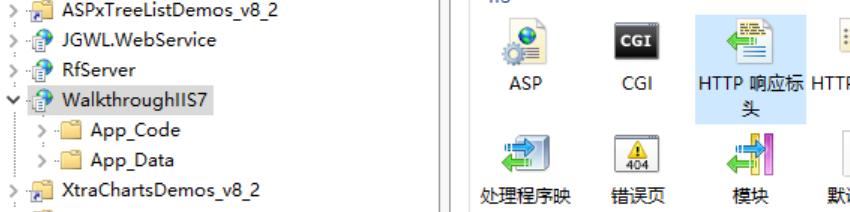
<%= Response.Headers.Get("CustomHeader2").ToString() %>

</div>

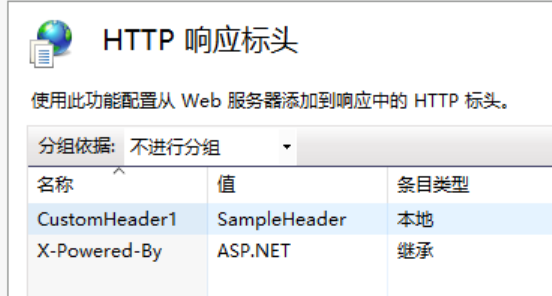
</form>



iis中添加的标头



双击http响应标头进入



本地iis网站，在iis中配置后，会自动更改webconfig？

您已经使用 IIS 管理器执行了配置任务。在此过程中，您将查看应用程序的 Web.config 文件中的更改。

system.webServer 节包含您使用 IIS 管理器进行的配置更改。system.webServer 节具有以下子元素：

* 一个 modules 元素，它为请求处理管道注册自定义模块。
* 一个 httpProtocol 元素，它定义了自定义响应头。
* 一个 urlCompression 元素，它禁用了静态压缩。

system.webServer 节类似于下面的示例：

<system.webServer>

<modules>

<add name="CustomModule" type="CustomModule" preCondition="" />

</modules>

<httpProtocol>

<customHeaders>

<add name="CustomHeader1" value="SampleHeader" />

</customHeaders>

</httpProtocol>

<urlCompression doStaticCompression="false" />

</system.webServer>

可以使 ASP.NET Web 应用程序作为本地 IIS 网站运行。若要使 Visual Studio 可以创建并使用本地 IIS 网站，必须启用元数据库兼容性。这样使 Visual Studio 可以与 IIS 元数据和 IIS 7.0 配置存储区进行交互。

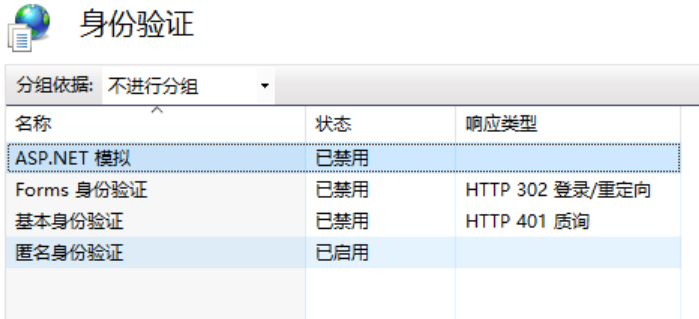
与 Microsoft Internet 信息服务 (IIS) 一起使用的 ASP.NET 可以使用以下任意一种身份验证方法来验证用户凭据（如用户名和密码）：

* Windows：基本、摘要式或集成 Windows 身份验证（NTLM 或 Kerberos）。
* Forms 身份验证，您可以通过该身份验证在您的应用程序中创建登录页并管理身份验证。
* 客户证书身份验证

通过将已验证凭据或它们的表示形式与某些内容进行比较，ASP.NET 可以控制对站点信息的访问；这些内容可以是 NTFS 文件系统权限，也可以是列出了已授权用户、已授权角色（组）或已授权 HTTP 谓词的 XML 文件。

## 方案 2 - Forms 身份验证

在 Forms 身份验证方案中，应用程序直接从用户那里收集名字和密码等凭据，并自己判断这些信息的真实性。应用程序不使用 IIS 身份验证，但 IIS 身份验证设置可以影响 Forms 身份验证。作为一项原则，当您使用 Forms 身份验证时，请在 IIS 中启用匿名访问。否则，如果用户未通过 IIS 身份验证，他们甚至无法访问应用程序以便向 Forms 身份验证提供用户名和密码。



# ASP.NET 身份验证

https://msdn.microsoft.com/zh-cn/library/eeyk640h(v=vs.100).aspx

身份验证是从用户处获取标识凭据（如用户名和密码）并通过某些授权机构验证那些凭据的过程。如果这些凭据有效，则将提交这些凭据的实体视为通过身份验证。在身份得到验证后，授权进程将确定该身份是否可以访问给定资源。

ASP.NET 通过身份验证提供程序（即包含验证请求方凭据所需代码的代码模块）来实现身份验证。本节中的主题描述内置于 ASP.NET 中的身份验证提供程序。

在 ASP.NET 应用程序中，Windows 身份验证将 Microsoft Internet 信息服务 (IIS) 所提供的用户标识视为已经过身份验证的用户。IIS 提供了大量用于验证用户标识的身份验证机制，其中包括匿名身份验证、Windows 集成的 (NTLM) 身份验证、Windows 集成的 (Kerberos) 身份验证、基本（base64 编码）身份验证、摘要式身份验证以及基于客户端证书的身份验证。

在 ASP.NET 中，使用 [WindowsAuthenticationModule](https://msdn.microsoft.com/zh-cn/library/system.web.security.windowsauthenticationmodule%28v=vs.100%29.aspx) 模块来实现 Windows 身份验证。该模块根据 IIS 所提供的凭据构造一个 [WindowsIdentity](https://msdn.microsoft.com/zh-cn/library/system.security.principal.windowsidentity%28v=vs.100%29.aspx)，并将该标识设置为该应用程序的当前 [User](https://msdn.microsoft.com/zh-cn/library/system.web.httpcontext.user%28v=vs.100%29.aspx) 属性值。

Windows 身份验证是 ASP.NET 应用程序的默认身份验证机制，并指定作为使用 [authentication](https://msdn.microsoft.com/zh-cn/library/532aee0e%28v=vs.100%29.aspx) 配置元素的应用程序的身份验证模式，如下面的示例所示。

<system.web>

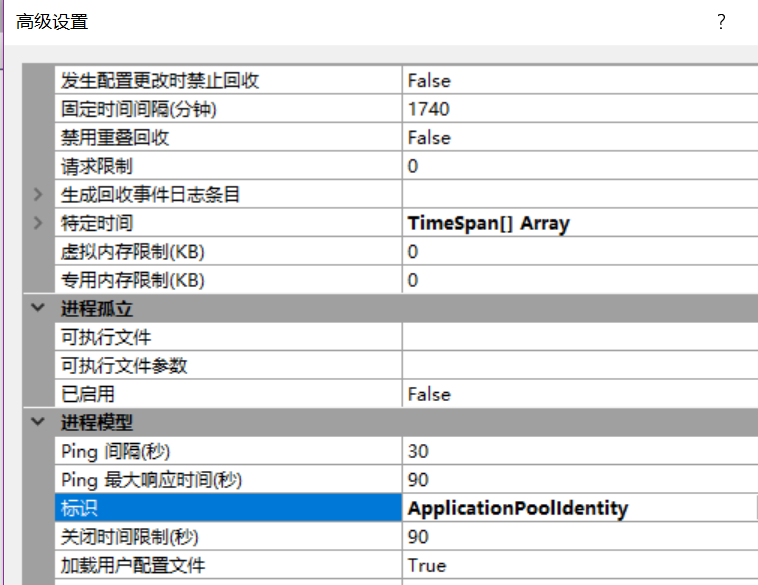
<authentication mode="Windows"/>

</system.web>

## 模拟 Windows 标识

尽管 Windows 身份验证模式根据 IIS 所提供的凭据将当前 [User](https://msdn.microsoft.com/zh-cn/library/system.web.httpcontext.user%28v=vs.100%29.aspx) 属性值设置为 [WindowsIdentity](https://msdn.microsoft.com/zh-cn/library/system.security.principal.windowsidentity%28v=vs.100%29.aspx)，但它不修改提供给操作系统的 Windows 标识。 默认情况下，此 Windows 标识是 ASP.NET 进程的标识。在 Microsoft Windows 2000 和 Windows XP Professional 上，此标识是 ASP.NET 辅助进程的标识，即本地 ASPNET 帐户。在 Windows Server 2003 上，此标识是 ASP.NET 应用程序所属的 IIS 应用程序池的标识。默认情况下，此标识是 NETWORK SERVICE 帐户。

（Windows 标识：用户账户，ASP.NET 进程的标识：运行asp.net程序的账户）



不使用模拟时，将使用运行应用程序池的账户的权限访问资源。使用模拟时，使用iis传递过来的通过验证的windows账户的权限

通过启用模拟功能，可以将 ASP.NET 应用程序的 Windows 标识配置为 IIS 所提供的 Windows 标识。也就是说，指示 ASP.NET 应用程序模拟 IIS 为 Windows 操作系统验证的所有任务（包括文件和网络访问）提供的标识。

若要为 Web 应用程序启用模拟功能，请在该应用程序的 Web.config 文件中将 [identity](https://msdn.microsoft.com/zh-cn/library/72wdk8cc%28v=vs.100%29.aspx) 元素的 impersonate 属性设置为 true，如下面的代码示例所示。

<system.web>

<authentication mode="Windows"/>

<identity impersonate="true"/>

</system.web>

有关 ASP.NET 进程标识的更多信息，请参见[配置 ASP.NET 进程标识](https://msdn.microsoft.com/zh-cn/library/dwc1xthy%28v=vs.100%29.aspx)。有关模拟的更多信息，请参见 [Impersonate](https://msdn.microsoft.com/zh-cn/library/system.security.principal.windowsidentity.impersonate%28v=vs.100%29.aspx) 方法。

# ASP.NET Forms 身份验证概述

https://msdn.microsoft.com/zh-cn/library/7t6b43z4(v=vs.100).aspx

Forms 身份验证使您可以使用自己的代码对用户进行身份验证，然后将身份验证标记保留在 Cookie 或页的 URL 中。您可以使用 [CookieMode](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication.cookiemode%28v=vs.100%29.aspx) 属性，指定将身份验证票证包含在 URL 中，而不是包含在 Cookie 中。Forms 身份验证通过 [FormsAuthenticationModule](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthenticationmodule%28v=vs.100%29.aspx) 类参与到 ASP.NET 页的生命周期中。可以通过 [FormsAuthentication](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication%28v=vs.100%29.aspx) 类访问 Forms 身份验证信息和功能。

通过使用 [authentication](https://msdn.microsoft.com/zh-cn/library/532aee0e%28v=vs.100%29.aspx) 配置元素。在配置文件中，指定一个 URL 以将未经身份验证的请求重定向到登录页。然后在 Web.config 文件或单独的文件中定义有效的凭据（设置用户名密码）。下面的示例演示配置文件的一部分，其中为 [Authenticate](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication.authenticate%28v=vs.100%29.aspx) 方法指定了登录页和身份验证凭据。密码已经使用 [HashPasswordForStoringInConfigFile](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication.hashpasswordforstoringinconfigfile%28v=vs.100%29.aspx) 方法进行加密。

<authentication mode="Forms">

<forms name="SavingsPlan" loginUrl="/Login.aspx">

<credentials passwordFormat="SHA1">

<user name="Kim"

password="07B7F3EE06F278DB966BE960E7CBBD103DF30CA6"/>

<user name="John"

password="BA56E5E0366D003E98EA1C7F04ABF8FCB3753889"/>

</credentials>

</forms>

</authentication>

在身份验证成功之后，[FormsAuthenticationModule](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthenticationmodule%28v=vs.100%29.aspx) 模块会将 [User](https://msdn.microsoft.com/zh-cn/library/system.web.httpcontext.user%28v=vs.100%29.aspx) 属性的值设置为对经过身份验证的用户的引用。

string authUser2 = User.Identity.Name;

使用 Forms 身份验证的简便方法是，使用 ASP.NET 成员资格和 ASP.NET 登录控件。ASP.NET 成员资格可让您存储和管理用户信息，并包含对用户进行身份验证的方法。ASP.NET 登录控件与 ASP.NET 成员资格一起使用，将提示用户输入凭据、验证用户、恢复或更换密码等操作的逻辑进行封装。

public static void RedirectFromLoginPage(

string userName,

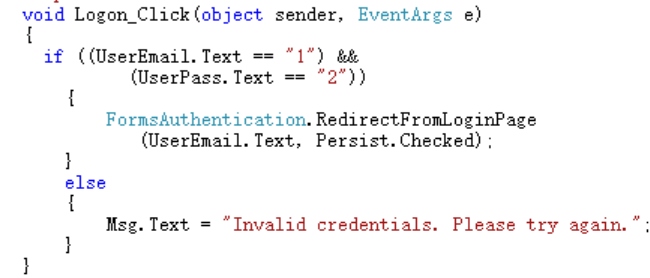
bool createPersistentCookie

)

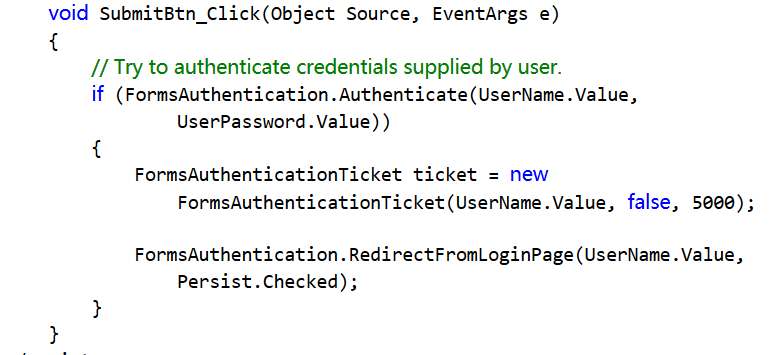
Redirects an authenticated user back to the originally requested URL or the default URL.

the [RedirectFromLoginPage](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication.redirectfromloginpage%28v=vs.100%29.aspx) method issues an authentication ticket and places it in the [QueryString](https://msdn.microsoft.com/zh-cn/library/system.web.httprequest.querystring%28v=vs.100%29.aspx) property.

With forms authentication, you can use the SetAuthCookie method when you want to authenticate a user but still retain control of the navigation with redirects.



需要对 HTTP Cookie 属性进行精确控制的应用程序可以构造票证，并执行自定义代码中的重定向。在这些情况下，应使用 [FormsAuthentication](https://msdn.microsoft.com/zh-cn/library/system.web.security.formsauthentication%28v=vs.100%29.aspx) 类的加密方法来加密身份验证票证。



# 配置 ASP.NET 应用程序以使用成员资格

使用应用程序的 Web.config 文件中的 [membership](https://msdn.microsoft.com/zh-cn/library/1b9hw62f%28v=vs.100%29.aspx) 元素配置 ASP.NET 成员资格。membership 元素是 system.web 节的一个子元素

使用 membership 元素的 defaultProvider 属性来指定默认的成员资格提供程序。如果没有显式指定默认的提供程序，计算机配置将指定一个名为“AspNetSqlMembershipProvider”的 [SqlMembershipProvider](https://msdn.microsoft.com/zh-cn/library/system.web.security.sqlmembershipprovider%28v=vs.100%29.aspx) 实例（它被标识为默认的提供程序）。“AspNetSqlMembershipProvider”连接到本地 SQL Server 中的 aspnetdb 数据库。

您还可以通过配置 membership 节中的提供程序来为此提供程序指定默认的提供程序实例和选项。

# Web 应用程序项目与网站项目

<https://msdn.microsoft.com/zh-cn/library/dd547590(v=vs.100).aspx>

# 如何：演示成员资格提供程序实现

<https://msdn.microsoft.com/zh-cn/library/6tc47t75(v=vs.100).aspx>

