ASP.NET 2.0验证机制的应用研究

宋建伟,吴清烈
(东南大学 电子商务系,江苏 南京 211189)

摘 要:多数网站都需要验证机制对用户身份进行验证并授予用户相应权限。为了提高开发人员在身份验证方面的效率,ASP.NET 2.0中增加了一套专门用于身份验证的新控件及API。介绍了ASP.NET 2.0中新推出的验证控件和验证API的功能,然后通过解决一个网站开发中遇到的身份验证问题,说明了这些控件和API的使用方法。从中可以看出使用ASP.NET 2.0的验证机制,仅仅书写少量代码便可以完成以往需要大量时间和代码才能完成的任务,从而证明ASP.NET 2.0验证机制的确极具易用性和实用性。

关键词:ASP.NET 2.0;验证机制;控件;实例
中图分类号:TP393 **文献标识码**:A **文章编号**:1673-629X(2009)02-0209-04

Application Research of Authentication Mechanism of ASP.NET 2.0

SONG Jian-wei, WU Qing-lie
(Department of Electronic Business and Commerce, Southeast University, Nanjing 211189, China)

Abstract: Authentication mechanism is needed in most web sites to judge users and grand corresponding rights to them. In order to improve the efficiency of authentication for programmers, ASP.NET 2.0 adds a suite of new authentication controls and APIs intended for authentication. First introduces the functions of the new authentication controls and APIs in ASP.NET 2.0, and then specifies how to use these controls and APIs through solving an authentication problem in building a web, from which can see that with the authentication mechanism of ASP.NET 2.0, just a few codes can do the same thing with the previous way of writing many codes with much time, thus to prove its facility, practicability and expansibility.

Key words: ASP.NET 2.0; authentication mechanism; controls; instance

0 引 言

验证机制是用来判断用户身份并对不同身份用户授予不同权限的一种机制,本质上是权限管理的问题。权限管理是B/S模式信息系统应用中影响安全性的关键环节之一,主要包括用户授权和页面访问控制问题[1]。过去为实现网站的验证功能,开发人员需要建立多个数据表并编写大量代码与数据库交互,整个过程费时耗力,而且易出错。ASP.NET 2.0的出现使这一切变得容易。

ASP.NET是微软为了抗衡JPS而推出的新一代ASP(Active Server Pages)脚本语言,它借鉴了JSP的优点,同时它又具有自身的一些新特点[2]。ASP.NET 2.0是微软在2005年末推出的新一代动态网页开发技术。与ASP.NET 1.1相比,2.0版本增加了大量新控件和新特性,使其不但在功能上有了显著增强,而且更加易学易用,能够显著改善用户体验;而其中新增加的验证机制提供了一套完善的身份验证方案,成为ASP.NET 2.0的一大亮点。

从开发人员角度来看,ASP.NET 2.0验证机制由两部分组成:一组验证控件,一组验证API。前者提供了图形化的验证界面,便于程序员实现无代码快速开发;后者的验证API则是验证控件发挥作用的幕后支持者,是ASP.NET 2.0验证机制的灵魂部分,当验证控件满足不了开发要求的时候,验证API就成为解决问题的有效途径。

1 验证控件概览

ASP.NET 2.0中新增的验证控件集成在Visual Studio 2005的各个版本中。用Visual Studio 2005新建一个网站后,在"工具箱"中展开"登录"面板,可以看到7个与验证相关的控件,它们是:Login, PasswordRecovery, LoginStatus, LoginName, CreateUserWizard, ChangePassword, LoginView控件。通过拖放这组

收稿日期:2008-06-22
作者简介:宋建伟(1982-),男,山东烟台人,硕士,研究方向为电子商务与信息系统;吴清烈,博士,副教授,研究方向为电子商务、管理科学与信息系统。

控件，开发人员不用写一行代码就可以实现基本的验证功能；相比在 ASP. NET 1.1 时代需要亲手建数据库，设计界面，编写数据库连接层代码等一系列繁琐工作，这算得上一个不小的进步。下面简单说明各个控件的功能：

(1)Login 控件：提供通用的用户登录界面；

(2)PasswordRecovery 控件：显示恢复密码界面，在正确回答出提示问题后网站将密码以邮件方式发送给用户；

(3) LoginStatus 控件：显示用户登录状态，点击它还可以实现登录/注销功能；

(4) LoginName 控件：显示当前登录用户名称；

(5)CreateUserWizard 控件：用于创建新用户，提供了一些必要注册信息供用户输入，如用户名、密码、邮箱、密码恢复提示问题等；

(6)ChangePassword 控件：提供密码更改功能；

(7)LoginView 控件：一个模板控件，通过对不同身份的用户设置不同的模板，可以使不同身份的用户看到不同的内容。

上述控件中除了 LoginStatus 和 LoginName 控件外，其他控件都是由标签、文本框、按钮等控件组合而成的复合控件。如果对默认外观不满意，开发者可以在其属性中选择“转换为模板”将其拆散，然后自行安排控件外观；在属性中选择“重置”，则可以将控件再次拼合成一个整体。

2 验证 API 概览

上述 7 个验证控件使用起来虽然很方便，但缺点也很明显：功能单一，缺乏扩展性。尤其是 CreateUserWizard 控件，可供用户输入的信息有限，往往满足不了开发者要求。比如网站的管理员需要一个审核机制，需要用户在注册时输入其真实姓名作为管理员的审批依据，此时 CreateUserWizard 控件对新增加的需求就无能为力了。

ASP. NET 2.0 提供的验证 API 可以用来处理这种情况。事实上这些 API 也是 7 个验证控件发挥作用的幕后支持者，比如点击 Login 控件的“登录”按钮时，触发的事件会自动调用对应 API 对用户信息进行验证。若要充分理解并灵活运用 ASP. NET 2.0 的验证机制以实现更丰富的功能，研究验证 API 是很有必要的，下面就介绍这组 API。

2.1 成员资格管理 API：Membership

Membership API 主要用于创建、管理、验证用户账号相关信息。CreateUserWizard 和 Login 控件就是依靠这个 API 来实现其功能的。Membership 主要包括两个类：Membership 类和 MembershipUser 类。Membership 类用于创建、管理、验证用户账号相关信息；MembershipUser 类用于管理特定用户信息，其实例需要通过 Membership 类的一些方法来创建。开发过程中将这两个类结合使用，可以解决很多用户验证方面的问题。

2.2 角色管理 API：Role

Role API 主要用于角色管理，通过将每个用户设定成不同角色可以很好地实现用户权限管理。Role API 有一个类：Roles 类，此类提供的一系列属性和方法可以方便地对用户所属角色进行操作。

2.3 个性化用户设置 API：Profile

Membership 中存储的用户信息只有用户名、密码、邮箱等与用户验证有关的信息，而实际应用中需要的用户信息往往不止这些。ASP. NET 2.0 提供的 Profile API 是对 Membership 的有益补充，它可以存放任意数量的额外用户信息，因此具有很强的扩展性和实用性。Profile API 主要包括两个类：Profile 和 ProfileCommon，二者间的关系类似 Membership 和 MembershipUser 间的关系。用 Profile API 存储用户额外信息很简单，只需要在网站配置文件 web. config 中的＜system. web＞＜/system. web＞节中的＜profile＞＜/profile＞节中增加字段即可。以上面要求用户注册时输入真实姓名为例，再加上其他一些用户信息，要将这些信息存储在 Profile 中的配置方法如下：

```
<profile>
  <properties>
    <add name="真实姓名" type="System.String" />
    <add name="研究方向" type="System.String" />
    <add name="年级" type="System.String" />
    <add name="联系电话" type="System.String" />
    <add name="研究方向" type="System.String" />
  </properties>
</profile>
```

配置完成后就可以在代码中通过智能列表方便地使用这些配置条目，为用户添加额外信息[3,4]。

3 一个实例

介绍完上述知识后，就可以用验证控件和验证 API 来解决上面提出的基于真实姓名的审核问题。首先，在注册页 Register. aspx 中，用户注册时除了输入 CreateUserWizard 控件固有的信息外，还要输入真实姓名。为此先将此 CreateUserWizard 控件打散，然后在其中加入一个标签，一个文本框和一个验证控件(不可见)，如图 1 所示。

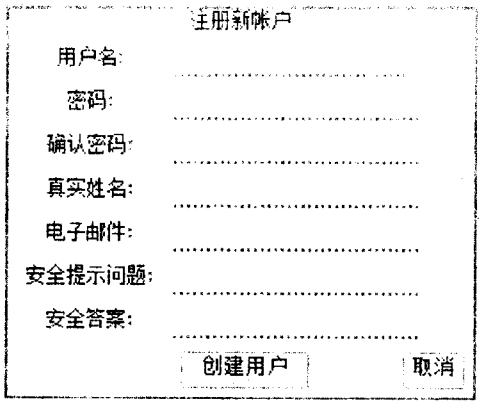

图 1　用户注册界面

当用户点击“创建用户”按钮后,会触发 CreateUserWizard 控件的 CreatedUser 事件,此时除“真实姓名”外,其他注册项均由控件自动调用 API 来创建,因此需要在 CreatedUser 事件中编写代码将用户“真实姓名”存储于 Profile 中:

```
//从 CreateUserWizard 控件中获取用户账号
string newUserName = ((TextBox)(CreateUserWizard1.CreateUserStep.ContentTemplateContainer.FindControl("UserName"))).Text;
//通过用户账号获取其 Profile
ProfileCommon userProfile=Profile.GetProfile((newUserName));
//设置用户 Profile
userProfile.真实姓名 = ((TextBox)(CreateUserWizard1.CreateUserStep.ContentTemplateContainer.FindControl("RealName"))).Text;
//保存设置
userProfile.Save();
```

通过上面代码就将用户真实姓名存储在其 Profile 中。

第二步,在 ActiveUsers.aspx 页中实现管理员的审核功能,用一个 GridView 控件来显示用户的账号、真实姓名和审核情况,界面如图 2 所示。

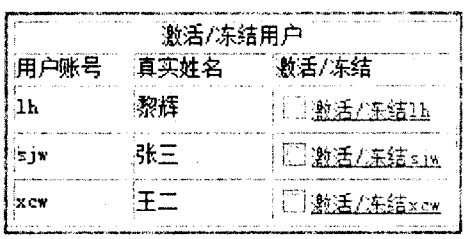

图 2　用户审核界面

在 ActiveUsers.aspx 页的 Page_Load 事件中编写代码使得页面加载后 GridView 显示用户信息,代码如下:

```
//动态创建表格,作为 GridView 的数据源
DataTable dt=new DataTable();
//为表格创建三列数据
dt.Columns.Add("用户账号", typeof(string));
dt.Columns.Add("真实姓名", typeof(string));
dt.Columns.Add("激活", typeof(bool));
//获取所有用户
MembershipUserCollection Users=Membership.GetAllUsers();
//遍历用户
foreach (MembershipUser eachUser in Users)
{
//判断用户身分是不是管理员
if (! Roles.IsUserInRole(eachUser.UserName, "Administrators"))
{
//从表中新建一列用于存放用户信息
DataRow singleRow=dt.NewRow();
//获取用户名
singleRow["用户账号"]=eachUser.UserName;
//通过用户名取得用户 Profile
ProfileCommon userProfile = Profile.GetProfile(eachUser.UserName);
//通过 Profile 获取用户真实姓名
singleRow["真实姓名"]=userProfile.真实姓名;
//获取用户激活情况
singleRow["激活"]=eachUser.IsApproved;
//将用户信息加入表中
dt.Rows.Add(singleRow);
}
}
//遍历完毕,将表作为 GridView 的数据源
GridView1.DataSource=dt;
//绑定数据
GridView1.DataBind();
```

最后,在 GridView 控件的 Row_Command 事件中编写代码,使得管理员点击 GridView 右侧的链接按钮时可以更改用户的激活状态,代码如下:

```
//获取点击行的行号
int index=Convert.ToInt32(e.CommandArgument);
//通过行号获得点击行
GridViewRow selectedRow=GridView1.Rows[index];
//通过点击行获取点击行的用户名
TableCell cellUserName=selectedRow.Cells[0];
//通过用户名获取用户信息
MembershipUser eachUser = Membership.GetUser(cellUserName.Text);
//更改用户验证信息
eachUser.IsApproved=! eachUser.IsApproved;
//更新用户信息
Membership.UpdateUser(eachUser);
```

通过以上少量代码就实现了管理员根据用户真实姓名审批的功能,整个过程没有任何数据库操作[5],从

而显著地简化了开发过程。

4 注意问题

(1)虽然使用 ASP.NET 2.0 提供的验证控件和 API 可以避免数据库操作,但是用户信息仍然是存储在数据库中的,ASP.NET 2.0 验证机制的好处在于屏蔽了数据库操作细节,使开发者专注于业务逻辑而不必关心底层数据连接,从而提高了开发效率。

(2)默认情况下,成员资格管理,角色管理与个性化用户配置 API 使用的数据表都存储在"App_Data"文件夹下名称为 ASPNETDB .MDF 的 SQL Server 2005 Express 实例文件中。若使用 SQL Server 2000 做数据库,需要先在 SQL Server 2000 中建立一个目标数据库,然后通过运行.NET Framework 2.0 文件夹下的 aspnet_regsql 命令启动数据库注册向导,通过该向导将目标数据库配置为数据源。配置完成后可以看到目标数据库中多了如图 3 所示的几张表,所有用户信息都存放在这几张表中。

5 结束语

ASP.NET 2.0 的验证机制还有很多值得称道的应用和技巧,为开发人员提供了诸多便利,需要在实际应用中进一步体会。

另外,虽然 ASP.NET 2.0 的登录验证很好用,但是开发人员不必为了适应这种新技术而改变自己以往的验证机制,要根据实际情况来选择是否使用这一新技术。

aspnet_Applications	dbo	2008/8/5
aspnet_Membership	dbo	2008/8/5
aspnet_Profile	dbo	2008/8/5
aspnet_Roles	dbo	2008/8/5
aspnet_SchemaVersions	dbo	2008/8/5
aspnet_Users	dbo	2008/8/5
aspnet_UsersInRoles	dbo	2008/8/5

图 3 用户信息数据库表

参考文献:

[1] Microsoft. Microsoft Developer Network(MSDN)[EB/OL]. [2008-06-15]. www.Microsoft.com/msdn/.

[2] 杨 剑, 闪四清. ASP.NET 环境下基于角色的权限控制的实现[J]. 计算机技术与发展,2007,17(5):234-237.

[3] 郝 刚, 袁永刚. ASP.NET 2.0 开发指南[M]. 北京:人民邮电出版社,2006.

[4] 奚江华. 圣殿祭司的 ASP.NET 2.0 开发详解[M]. 北京:电子工业出版社,2006.

[5] 郑宇军. C# 2.0 程序设计教程[M]. 北京:清华大学出版社,2005.

(上接第 208 页)

$X_i) \geqslant 0$,则认为 LX_i 与 LX_j 相容,否则不相容。

(2) 可能抽到许多套试题。

根据前面相容性检测可知,在相容性满足的前提下,可能得到许多套试卷,那到底选谁弃谁,另外,现行许多考试都有备用卷或者说是 A,B 卷等,就涉及到优先选题策略问题,如有 $C(I_i,J) \geqslant C(I_j,J) \geqslant 0$,则先选 I_j 题。例:将上例中的 $M'(D,T)$ 做如下修改:

$$M'(D,T) \equiv \begin{matrix} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{matrix} \begin{matrix} \begin{matrix} d_1 & d_2 & d_3 & d_4 & d_5 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix}$$

$C(T_1,D) = 60$, $C(T_2,D) = 20$, $C(T_3,D) = 35$, $C(T_4,D) = 60$,则 $C(T,D) = 20$,因此,若确定次序为 $T \rightarrow D$,则最好先选题型为 2 的题;同样 $C(D_1,T) = 20$, $C(D_2,T) = 80$, $C(D_3,T) = 40$, $C(D_4,T) = 15$, $C(D_5,T) = 65$,则 $C(D,T) = 15$,若确定次序为 $D \rightarrow T$,则最好先选难度档为 4 的题,根据这种方法,可抽取多套试题,并由此产生多套试卷。

4 结束语

通过对智能组卷过程中成卷模式的研究,给出生成模式的生成过程,以及为了提高组卷效率,有可能出现的考生级别转化而进行的相容性检测等。

参考文献:

[1] 丁仕虹. 网络试题库系统选题组卷策略优化及算法设计、实现[D]. 广州:华南理工大学,2003.

[2] 周红晓. 试题库组卷系统的研究与实现[D]. 金华:浙江师范大学,2003.

[3] 潘志文. 网络试题库系统的设计及其成卷算法的设计与实现[D]. 广州:华南理工大学,2002.

[4] 洪潮兴,陈凤平,徐永汉. 试题库通用软件成卷系统中的数学模型[J]. 华南理工大学学报:自然科学版,1995,23(9):87-92.

[5] 张 敏. 网络试题库系统成卷理论的研究及成卷质量的评价[D]. 广州:华南理工大学,2003.

[6] 林复华,陈光中,洪潮兴,等. 求解基于知识的成卷问题的算法[J]. 华南理工大学学报:自然科学版,1995,23(9):58-64.

[7] 柳 超. 一个网上在线考试系统的设计与实现[D]. 武汉:华中科技大学,2006.