Code Repository Exercises

Security, Membership, and Profiles

Purpose: The purpose of this exercise is to familiarize you with the concepts of application authentication and authorization, as well as how to use ASP.NET's built-in membership and role provider functions to accomplish these tasks. You will see how to set up a membership database and use the Website Administration Tool (WAT) to create users and roles for your application. Finally, you will learn to use ASP.NET's various membership controls, including Login, LoginName, LoginStatus, LoginView, ChangePassword, CreateUserWizard, and PasswordRecovery.

Directory Name: Membership

Instructions:

In this exercise, you will use the ASP.NET membership provider to handle user authentication. Do the following:

1. Use the Website Administration Tool (WAT) to enable role management in the repository application. Create two roles, "User" and "Admin". Create a new user with username *admin* and password *admin1!*. Assign this user to both the "User" and "Admin" roles.
    a. Go to your Web.Config file and notice that <roleManager enabled="true" /> has been added by the WAT.
    b. Refresh the App_Data folder of your application. You should now see a new SQL Express database called ASPNETDB.mdf. This is the default authentication database created by ASP.NET when forms authentication is enabled. You can open this database in the server explorer window and browse through it. The *admin* user you created is stored in this database.
2. In the directory created for this part of the exercise, create a new.aspx page called *Login.aspx*. Place a login control on this page with an attractive formatting template. Set the *CreateUserUrl* property to "CreateUser.aspx", the *PasswordRecoveryUrl* property to "RecoverPassword.aspx", and the *DestinationPageUrl* property to "Welcome.aspx" (you will create these pages in the next steps). Set the *CreateUserText* and *PasswordRecoveryText* properties to appropriate text for each link. Your login control should look something like the one shown on page 660 of the MacDonald book.
3. Create a new .aspx page called *CreateUser.aspx*. Add a CreateUserWizard control to this page and apply a formatting template. In response to the *CreatedUser* event of the CreateUserWizard, you should add the new user to the "User" role (see page 672 of the MacDonald book for an example). Set the *ContinueDestinationPageUrl* property to "Welcome.aspx" (created later).
4. Create a new .aspx page called *RecoverPassword.aspx*. Add a PasswordRecovery control to this page and apply a formatting template. The default behavior of a PasswordRecovery control is to email the password to the user. However, this requires configuration of an SMTP server.

Modify the PasswordRecovery control to simply display the password on the page by handling the *SendingMail* event of the control (see page 668 of the MacDonald book for an example). Note, you still need to specify a "from" email address in the Web.config file even though the email is not being sent. Do this by adding the following somewhere within the <configuration> element of the Web.Config file, but outside of any sub-element of this element:

```
<system.net>
        <mailSettings>
            <smtp from="your_email_address">
            </smtp>
        </mailSettings>
    </system.net>
```

5. Create a new .aspx page called *Welcome.aspx*. This page should contain a LoginView control that does the following:
    a. For anonymous users, displays the message "You are not logged in. Please login here". The "here" should be a link to the *Login.aspx* page.
    b. For users in the "Admin" role, displays hyperlinks to two pages (created in the next steps): *User.aspx* and *Admin.aspx*. Also displays a Logout link (use a LoginStatus control).
    c. For users in the "User" role, displays hyperlinks to only the *User.aspx* page. Also displays a Logout link (use a LoginStatus control)
6. Create a new .aspx page called *User.aspx*. Restrict anonymous access to this page in the Web.Config file, and allow only those in the "User" OR "Admin" roles to have access (see page 623-625 for an example of how to restrict access in the Web.Config file).
7. Create a new .aspx page called *Admin.aspx*. Restrict anonymous access to this page in the Web.Config file, and allow only those in the "Admin" role to have access.
8. Add profile fields to the Web.Config file to store user information, including first name, last name, birth date, and address information including street, city, state, zip code, and country. All fields should be of type string except for birth date, which should be of type System.DateTime. **Group the address fields together** by using a profile group (as shown on page 686 of the MacDonald book), **or** using a custom Address class (as shown on pages 687-689).
9. Create a user control called *CurrentUserInfo.ascx*. This user control should have the following content:
    a. The message, "Welcome, *username*!", where *username* is the username of the currently logged-in user (use a LoginName control).
    b. An appropriately labeled hyperlink back to the *Welcome.aspx* page.
    c. A ChangePassword control that allows the user to change his/her password. Apply a formatting template to the ChangePassword control.
    d. A set of appropriately labeled textboxes that allow the currently logged in user to add or modify his/her own profile information from step 8. When the user control is initially loaded, the current profile information should appear in the Textboxes. Create a button that allows the user to save any changes to the Textbox values in his/her profile.
10. Add an instance of the *CurrentUserInfo.ascx* control to the *User.aspx* and *Admin.aspx* pages.

11. At this stage, the *User.aspx* and *Admin.aspx* pages have identical content (the *CurrentUserInfo.ascx* control). Add additional functionality to *Admin.ascx* that allows the currently logged in administrator to see a list of all of the applications users, and allow the administrator to grant or revoke administrative privileges for each user (i.e. assign them to the "Admin" role or remove this assignment). You can choose how you implement this functionality, but the user should be able to see a list of users (by username), see whether they are already assigned to the "Admin" role or not, and allow them to add or remove users to the "Admin" role as necessary. (Hint: one option would be to use a Gridview to display a list of all MembershipUsers of the application, as shown on pages 653-654 of the MacDonald Textbook. You would need to add an additional field to the Gridview, like a checkbox, that allows the user to assign or remove the "Admin" role.
12. Further enhance the functionality on the *Admin.aspx* page to allow administrators to select any of the application's users and view all profile information for the selected user (i.e., if you used a Gridview to display all users, you can enable selection on the Gridview and then retrieve the profile for the selected user). **Note that this functionality is <u>separate from</u> and <u>in addition to</u> the functionality created in the *CurrentUserInfo.ascx* user control that allows the user to view and update his/her own profile information.** The profile information for the selected user should be viewable only, not editable.

Header on Default.aspx: Membership

Pages linked from Default.aspx: *Login.aspx*