

Erstellen sicherer ASP.NET-Anwendungen

Authentifizierung, Autorisierung und sichere Kommunikation

Auf der Orientierungsseite finden Sie einen Ausgangspunkt und eine vollständige Übersicht zum *Erstellen sicherer ASP.NET-Anwendungen*.

Zusammenfassung

Im Folgenden wird erläutert, wie die Objekte **GenericPrincipal** und **FormsIdentity** in Verbindung mit der Formularauthentifizierung erstellt und gehandhabt werden.

Vorgehensweise: Erstellen von GenericPrincipal-Objekten bei der Formularauthentifizierung

Bei Anwendungen, die mit Formularauthentifizierung arbeiten, wird häufig die **GenericPrincipal**-Klasse (in Verbindung mit der **FormsIdentity**-Klasse) zum Erstellen eines nicht Windows-spezifischen Autorisierungsschemas unabhängig von einer Windows-Domäne verwendet.

Für die Anwendung bestehen in diesem Zusammenhang die folgenden Möglichkeiten:

- Verwenden der Formularauthentifizierung zur Anforderung der Anmeldeinformationen der Benutzer (Benutzername und Kennwort).
- Überprüfen der übergebenen Anmeldeinformationen anhand eines Datenspeichers, z. B. einer Datenbank oder des Verzeichnisdienstes Microsoft® Active Directory®.
- Erstellen der Objekte **GenericPrincipal** und **FormsIdentity** basierend auf Werten, die aus dem Datenspeicher abgerufen wurden. Diese können Details zur Rollenmitgliedschaft eines Benutzers umfassen.
- Verwenden dieser Objekt zum Treffen von Autorisierungsentscheidungen.

Nachstehend wird erläutert, wie eine formularbasierte Webanwendung erstellt wird, die den Benutzer authentifiziert und ein benutzerdefiniertes Formularauthentifizierungsticket erstellt, das Benutzer- und Rolleninformationen umfasst. Darüber hinaus wird gezeigt, wie diese Informationen den Objekten **GenericPrincipal** und **FormsIdentity** zugeordnet und die neuen Objekte mit dem HTTP-Webanforderungskontext verbunden werden, wodurch es möglich wird, die Objekte in die Autorisierungslogik der Anwendung zu integrieren.

Diese Vorgehensweise konzentriert sich auf die Erstellung der Objekte **GenericPrincipal** und **FormsIdentity** sowie auf die Verarbeitung des Formularauthentifizierungstickets. Einzelheiten über das Authentifizieren von Benutzern anhand von Active Directory und SQL Server 2000 finden Sie in den weiteren, hiermit zusammenhängenden Vorgehensweisen im Abschnitt "Referenz" des vorliegenden Handbuchs.

- "Vorgehensweise: Verwenden der Formularauthentifizierung mit Active Directory"
- "Vorgehensweise: Verwenden der Formularauthentifizierung mit SQL Server 2000"

Anforderungen

Im Folgenden finden Sie eine Liste der empfohlenen Hardware und Software und eine Beschreibung der Netzwerkinfrastruktur, der Fähigkeiten und Kenntnisse sowie der Service Packs, die Sie benötigen.

- Microsoft SQL Server™ 2000
- Microsoft Visual Studio® .NET als Entwicklungssystem

Die in dieser Vorgehensweise erläuterten Verfahren setzen zudem Kenntnisse der ASP.NET-Webentwicklung mit dem Entwicklungstool Microsoft Visual C#™ voraus.

Zusammenfassung

Diese Vorgehensweise enthält folgende Verfahren:

1. Erstellen einer Webanwendung mit einer Anmeldeseite
2. Konfigurieren der Webanwendung für die Formularauthentifizierung
3. Erzeugen eines Authentifizierungstickets für authentifizierte Benutzer
4. Erstellen der Objekte **GenericPrincipal** und **FormsIdentity**
5. Testen der Anwendung

1. Erstellen einer Webanwendung mit einer Anmeldeseite

Mit diesem Verfahren wird eine neue ASP.NET-Webanwendung erstellt. Die Anwendung enthält zwei Seiten, eine Standardseite, auf die nur authentifizierte Benutzer zugreifen können, und eine Anmeldeseite für die Eingabe der Anmeldeinformationen.

► So erstellen Sie eine Webanwendung mit einer Anmeldeseite

1. Starten Sie Visual Studio .NET, und erstellen Sie eine neue C# ASP.NET-Webanwendung mit Namen **GenericPrincipalApp**.
2. Benennen Sie **WebForm1.aspx** in **Logon.aspx** um.
3. Fügen Sie **Logon.aspx** die folgenden Steuerelemente hinzu, um ein Anmeldeformular zu erstellen.

Tabelle 1: Die Steuerelemente für **Logon.aspx**

Typ des Steuerelements	Text	ID
Bezeichnungsfeld	User Name:	-
Bezeichnungsfeld	Password	-
Textfeld	-	txtUserName
Textfeld	-	txtPassword
Schaltfläche	Logon	btnLogon

4. Legen Sie die **TextMode**-Eigenschaft des Textfeldsteuerelements **txtPassword** auf **Password** fest.
5. Klicken Sie im Projektmappen-Explorer mit der rechten Maustaste auf **GenericPrincipalApp**, zeigen Sie auf **Hinzufügen**, und klicken Sie dann auf **Web Form hinzufügen**.
6. Geben Sie **default.aspx** als neuen Namen des Formulars ein, und klicken Sie dann auf **Öffnen**.

2. Konfigurieren der Webanwendung für die Formularauthentifizierung

► So bearbeiten Sie die Datei "Web.config" der Anwendung, um die Anwendung für die Formularauthentifizierung zu konfigurieren

1. Öffnen Sie **Web.config** im Projektmappen-Explorer.
2. Suchen Sie das **<authentication>**-Element, und ändern Sie das **mode**-Attribut in **Forms**.
3. Fügen Sie das **<forms>**-Element als untergeordnetes Element des **<authentication>**-Elements hinzu, und legen Sie die Attribute **loginUrl**, **name**, **timeout** und **path** wie nachstehend gezeigt fest:

```
<authentication mode="Forms">
  <forms loginUrl="logon.aspx" name="AuthCookie" timeout="60" path="/">
  </forms>
</authentication>
```

4. Fügen Sie das folgende **<authorization>**-Element unterhalb des **<authentication>**-Elements hinzu. Dies sorgt dafür, dass nur authentifizierte Benutzer auf die Anwendung zugreifen können. Das im Vorfeld eingerichtete **loginUrl**-Attribut des **<authentication>**-Elements leitet nicht authentifizierte Anforderungen auf die Seite **Logon.aspx** um.

```
<authorization>
  <deny users="?" />
  <allow users="*" />
</authorization>
```

3. Erzeugen eines Authentifizierungstickets für authentifizierte Benutzer

In diesem Verfahren wird der Code zum Erzeugen des Authentifizierungstickets für authentifizierte Benutzer geschrieben. Das Authentifizierungsticket ist eine Art Cookie, das vom **FormsAuthenticationModule** von ASP.NET verwendet wird.

Der Authentifizierungscode umfasst in der Regel das Nachschlagen des übergebenen Benutzernamens und des Kennworts, und zwar entweder in einer benutzerdefinierten Datenbank oder in Active Directory.

Weitere Informationen darüber, wie dieses Nachschlagen erfolgt, finden Sie in den folgenden Vorgehensweisen im Abschnitt "Referenz" dieses Handbuchs:

- "Vorgehensweise: Verwenden der Formularauthentifizierung mit Active Directory"
- "Vorgehensweise: Verwenden der Formularauthentifizierung mit SQL Server 2000"

► So erzeugen Sie ein Authentifizierungsticket für authentifizierte Benutzer

1. Öffnen Sie die Datei **Logon.aspx.cs**, und fügen Sie am Anfang der Datei unterhalb der vorhandenen **using**-Anweisungen die folgenden **using**-Anweisungen hinzu:

```
using System.Web.Security;
```

2. Fügen Sie der **WebForm1**-Klasse die folgende private Hilfsmethode mit Namen **IsAuthenticated** hinzu, die verwendet wird, um die Benutzernamen und Kennwörter für die Authentifizierung der Benutzer zu überprüfen. In diesem Code wird davon ausgegangen, dass sämtliche Kombinationen aus Benutzername und Kennwort gültig sind.

```

private bool IsAuthenticated( string username, string password )
{
    // Lookup code omitted for clarity
    // This code would typically validate the user name and password
    // combination against a SQL database or Active Directory
    // Simulate an authenticated user
    return true;
}

```

3. Fügen Sie die nachstehende private Hilfsmethode mit Namen **GetRoles** hinzu, die verwendet wird, um die Rollen abzurufen, denen der Benutzer angehört.

```

private string GetRoles( string username, string password )
{
    // Lookup code omitted for clarity
    // This code would typically look up the role list from a database table.
    // If the user was being authenticated against Active Directory, the
    // Security groups and/or distribution lists that the user belongs to may be
    // used instead

    // This GetRoles method returns a pipe delimited string containing roles
    // rather than returning an array, because the string format is convenient
    // for storing in the authentication ticket / cookie, as user data
    return "Senior Manager|Manager|Employee";
}

```

4. Zeigen Sie das Formular **Logon.aspx** im Entwurfsmodus an, und doppelklicken Sie auf die Schaltfläche **Logon**, um einen Ereignishandler für das Klickereignis zu erstellen.
5. Fügen Sie einen Aufruf der **IsAuthenticated**-Methode hinzu, und übergeben Sie den Benutzernamen und das Kennwort, der bzw. das im Anmeldeformular eingegeben wurde. Weisen Sie den Rückgabewert einer Variablen vom Typ **bool** zu, die angibt, ob der Benutzer authentifiziert werden konnte oder nicht.

```

bool isAuthenticated = IsAuthenticated( txtUserName.Text,
                                        txtPassword.Text );

```

6. Fügen Sie einen Aufruf der **GetRoles**-Methode hinzu, um für den Fall, dass der Benutzer authentifiziert wird, die Rollenliste des Benutzers abzurufen.

```

if (isAuthenticated == true )
{
    string roles = GetRoles( txtUserName.Text, txtPassword.Text );
}

```

7. Erstellen Sie ein neues Formularauthentifizierungsticket, das den Benutzernamen, eine Ablaufzeit und die Liste der Rollen enthält, denen der Benutzer angehört. Beachten Sie, dass die **UserData**-Eigenschaft des Authentifizierungstickets verwendet wird, um die Rollenliste des Benutzers zu speichern. Beachten Sie zudem, dass mit dem folgenden Code ein nicht persistentes Ticket erstellt wird, obwohl es vom Anwendungsszenario abhängig ist, ob das Ticket/Cookie persistent ist oder nicht.

```

// Create the authentication ticket
FormsAuthenticationTicket authTicket = new
    FormsAuthenticationTicket(1, // version
        txtUserName.Text, // user name
        DateTime.Now, // creation
        DateTime.Now.AddMinutes(60), // Expiration
        false, // Persistent
        roles ); // User data

```

8. Fügen Sie Code zum Erstellen einer verschlüsselten Zeichenfolgendarstellung des Tickets hinzu, und speichern Sie dieses als Datenwert in einem **HttpCookie**-Objekt.

```

// Now encrypt the ticket.
string encryptedTicket = FormsAuthentication.Encrypt(authTicket);
// Create a cookie and add the encrypted ticket to the
// cookie as data.
HttpCookie authCookie =
    new HttpCookie(FormsAuthentication.FormsCookieName,
        encryptedTicket);

```

9. Fügen Sie das Cookie der Cookieauflistung hinzu, die an den Browser des Benutzers zurückgegeben wird.

```

// Add the cookie to the outgoing cookies collection.
Response.Cookies.Add(authCookie);

```

10. Leiten Sie den Benutzer auf die ursprünglich angeforderte Seite um.

```

// Redirect the user to the originally requested page
Response.Redirect( FormsAuthentication.GetRedirectUrl(
    txtUserName.Text,
    false ));
}

```

4. Erstellen der Objekte **GenericPrincipal** und **FormsIdentity**

Mit diesem Verfahren wird ein Ereignishandler für die Anwendungsauthentifizierung implementiert, und basierend auf den Informationen im Authentifizierungsticket werden die Objekte **GenericPrincipal** und **FormsIdentity** erstellt.

► So erstellen Sie die Objekte **GenericPrincipal** und **FormsIdentity**

1. Öffnen Sie **gobal.aspx** im Projektmappen-Explorer.
2. Wechseln Sie in die Codeansicht, und fügen Sie am Anfang der Datei die folgenden **using**-Anweisungen hinzu:

```

using System.Web.Security;
using System.Security.Principal;

```

3. Suchen Sie den Ereignishandler **Application_AuthenticateRequest**, und fügen Sie den folgenden Code hinzu, um aus der mit der Anforderung übergebenen Cookieauflistung das Formularauthentifizierungscookie abzurufen.

```

// Extract the forms authentication cookie
string cookieName = FormsAuthentication.FormsCookieName;
HttpCookie authCookie = Context.Request.Cookies[cookieName];

if(null == authCookie)
{
    // There is no authentication cookie.
    return;
}

```

4. Fügen Sie den folgenden Code hinzu, um das Authentifizierungsticket aus dem Formularauthentifizierungscookie zu extrahieren und zu entschlüsseln.

```

FormsAuthenticationTicket authTicket = null;
try
{
    authTicket = FormsAuthentication.Decrypt(authCookie.Value);
}
catch(Exception ex)
{
    // Log exception details (omitted for simplicity)
    return;
}

if (null == authTicket)
{
    // Cookie failed to decrypt.
    return;
}

```

5. Fügen Sie den folgenden Code hinzu, um die per Pipe getrennte Liste der Rollennamen auszulesen, die bei der ursprünglichen Authentifizierung des Benutzers an das Ticket angehängt wurde.

```

// When the ticket was created, the UserData property was assigned a
// pipe delimited string of role names.
string[] roles = authTicket.UserData.Split(new char[]{'|'});

```

6. Fügen Sie den folgenden Code hinzu, um das **FormsIdentity**-Objekt mit dem Benutzernamen zu erstellen, der dem Ticketnamen entnommen wurde, und um ein **GenericPrincipal**-Objekt zu erstellen, das diese Identität zusammen mit der Rollenliste des Benutzers enthält.

```

// Create an Identity object
FormsIdentity id = new FormsIdentity( authTicket );

// This principal will flow throughout the request.
GenericPrincipal principal = new GenericPrincipal(id, roles);
// Attach the new principal object to the current HttpContext object
Context.User = principal;

```

5. Testen der Anwendung

Mit diesem Verfahren wird der Seite **default.aspx** Code hinzugefügt, um Informationen aus dem **GenericPrincipal**-Objekt anzuzeigen, das an das aktuelle **HttpContext**-Objekt angefügt wurde. So wird sichergestellt, dass das Objekt ordnungsgemäß erstellt und der aktuellen Webanforderung zugewiesen wurde. Anschließend erstellen und testen Sie die Anwendung.

► So testen Sie die Anwendung

1. Doppelklicken Sie im Projektmappen-Explorer auf **default.aspx**.
2. Doppelklicken Sie auf das Webformular **default.aspx**, um den Ereignishandler zum Laden der Seite anzuzeigen.
3. Führen Sie einen Bildlauf zum Anfang der Datei durch, und fügen Sie unterhalb der vorhandenen **using**-Anweisungen die folgende **using**-Anweisung hinzu.

```
using System.Security.Principal;
```

4. Kehren Sie zum Ereignishandler für das Laden der Seite zurück, und fügen Sie den folgenden Code hinzu, um den Identitätsnamen anzuzeigen, der dem **GenericPrincipal**-Objekt zugewiesen wurde, das mit der aktuellen Webanforderung zusammenhängt.

```
IPrincipal p = HttpContext.Current.User;  
Response.Write( "Authenticated Identity is: " +  
                p.Identity.Name );  
Response.Write( "<p>" );
```

5. Fügen Sie den folgenden Code hinzu, um die Rollenmitgliedschaft der aktuell authentifizierten Identität zu prüfen.

```
if ( p.IsInRole("Senior Manager") )  
    Response.Write( "User is in Senior Manager role<p>" );  
else  
    Response.Write( "User is not in Senior Manager role<p>" );  
  
if ( p.IsInRole("Manager") )  
    Response.Write( "User is in Manager role<p>" );  
else  
    Response.Write( "User is not in Manager role<p>" );  
  
if ( p.IsInRole("Employee") )  
    Response.Write( "User is in Employee role<p>" );  
else  
    Response.Write( "User is not in Employee role<p>" );  
  
if ( p.IsInRole("Sales") )  
    Response.Write( "User is in Sales role<p>" );  
else  
    Response.Write( "User is not in Sales role<p>" );
```

6. Klicken Sie im Projektmappen-Explorer mit der rechten Maustaste auf **default.aspx**, und klicken Sie dann auf **Als Startseite festlegen**.
7. Klicken Sie im Menü **Erstellen** auf **Projektmappe erstellen**. Beheben Sie sämtliche Buildfehler.
8. Drücken Sie **Strg+F5**, um die Anwendung auszuführen. Da **default.aspx** als Startseite konfiguriert ist, ist dies auch die zuerst angeforderte Seite.

9. Wenn Sie auf die Anmeldeseite umgeleitet werden (da Sie anfänglich ja nicht über ein Authentifizierungsticket verfügen), geben Sie irgendeinen Benutzernamen und ein beliebiges Kennwort ein, und klicken Sie dann auf **Logon**.
10. Vergewissern Sie sich, dass Sie auf die Seite **default.aspx** umgeleitet werden und dass die Benutzeridentität und die korrekten Rollendetails angezeigt werden. Der Benutzer sollte ein Mitglied der Rollen **Senior Manager**, **Manager** und **Employee** sein, jedoch kein Mitglied der Rolle **Sales**.

Weitere Ressourcen

Weitere Informationen über diesen Themenbereich finden Sie in den folgenden Vorgehensweisen im Abschnitt "Referenz" dieses Handbuchs:

- "Vorgehensweise: Verwenden der Formularauthentifizierung mit Active Directory"
- "Vorgehensweise: Verwenden der Formularauthentifizierung mit SQL Server 2000"