.NET Framework 4 - ASP.NET

**Forms Authentication Provider**

Forms authentication enables you to authenticate the user name and password of your users using a login form that you create. Unauthenticated requests are redirected to a login page, where the user provides credentials and submits the form. If the application authenticates the request, the system issues a ticket that contains a key for reestablishing the identity for subsequent requests.

The topics in this section describe how to use forms authentication to create a custom login system.

|  |
| --- |
| **Description: NoteNote** |
| A convenient way to use forms authentication is to use ASP.NET membership (which stores user credentials) and the ASP.NET login controls (which you can use to create a login page). |

.NET Framework 4 - ASP.NET

**Forms Authentication Control Flow**

The flow of control for ASP.NET forms authentication is shown in the following table.

|  |  |
| --- | --- |
| **Browser and HTTP operation** | **Server reply** |
| Requests a protected resource from a server. The HTTP operation is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl01_code');" \o "Copy Code)  GET /default.aspx | If there is no authentication cookie, redirects the request to a logon page to collect credentials. Information about the originating page is placed in the query string using RETURNURL as the key. The server HTTP reply is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl02_code');" \o "Copy Code)  302 Found  Location: http://samples.microsoft.com/logon.aspx?RETURNURL=/default.aspx |
| Follows the redirection to the logon page. The HTTP operation is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl03_code');" \o "Copy Code)  GET /logon.aspx?RETURNURL=/default.aspx | Returns the logon page. For security, we recommend that you use Secure Sockets Layer (SSL) for the logon page to keep the user's credentials from being sent in clear text. The server HTTP reply is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl04_code');" \o "Copy Code)  200 OK |
| After user enters credentials into the logon page, submits the page. The HTTP operation is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl05_code');" \o "Copy Code)  POST /logon.aspx?RETURNURL=/default.aspx | Validates user credentials and, if the credentials are authenticated, redirects the browser to the original URL specified in the [QueryString](http://msdn.microsoft.com/en-us/library/system.web.httprequest.querystring.aspx) as the RETURNURL variable. By default, the authentication ticket is issued as a cookie.  Description: Note**Note**  You can specify that the authentication ticket be included in the URL instead of a cookie using the [CookieMode](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.cookiemode.aspx) property.  The server HTTP reply is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl10_code');" \o "Copy Code)  302 Found  Location: /default.aspx |
| Follows the redirection and requests the original resource again. The HTTP operation is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl11_code');" \o "Copy Code)  GET /default.aspx | If the user is authenticated, grants access and grants the authentication cookie, which contains an authentication ticket. Future requests by the same browser session will be authenticated when the module inspects the cookie. It is possible to create a persistent cookie that can be used for future sessions, but only until the cookie's expiration date. The server HTTP reply is:  [Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl12_code');" \o "Copy Code)  200 OK  Set-Cookie: ASPXTICKET=ABCDEFG12345;Path=/  Note that the cookie path is set to /. Because cookie names are case-sensitive, this prevents inconsistent case in URLs on the site. For example, if the path were set to /SavingsPlan and a link contained /savingsplan, the user would be forced to re-authenticate because the browser would not send the cookie. |

.NET Framework 4 - ASP.NET

**Forms Authentication Credentials**

Forms authentication credentials that are used to validate users at logon can be stored in an external data source or in the application configuration file.

|  |
| --- |
| **Description: NoteNote** |
| ASP.NET membership is the preferred method for storing and managing user credentials in forms-authenticated applications. For more information, see [Managing Users by Using Membership](http://msdn.microsoft.com/en-us/library/tw292whz.aspx). |

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifStoring Users in the Application Configuration File

When using forms authentication, you can validate users from user/password pairs in the [credentials](http://msdn.microsoft.com/en-us/library/e01fc50a.aspx) section of the Web site's configuration file. You can use the [Authenticate](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.authenticate.aspx) method to compare the credentials collected from the user to the list of user/password pairs in the [credentials](http://msdn.microsoft.com/en-us/library/e01fc50a.aspx) section to determine whether access should be granted. In the following example, users Kim and John can log on if they provide the correct password.

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl21_ctl00_ctl03_code');" \o "Copy Code)

<credentials passwordFormat="SHA1" >

<user name="Kim"

password="07B7F3EE06F278DB966BE960E7CBBD103DF30CA6"/>

<user name="John"

password="BA56E5E0366D003E98EA1C7F04ABF8FCB3753889"/>

</credentials>

The credential pairs in the example are encrypted using the Secure Hash Algorithm-1 (SHA1) password-hashing format. The [PasswordFormat](http://msdn.microsoft.com/en-us/library/system.web.configuration.formsauthenticationcredentials.passwordformat.aspx) attribute is required. Values for this property are listed in the following table.

|  |  |
| --- | --- |
| **Value** | **Description** |
| [Clear](http://msdn.microsoft.com/en-us/library/system.web.configuration.formsauthpasswordformat.aspx) | Passwords are stored in clear text. The user password is compared directly to this value without further transformation. |
| [MD5](http://msdn.microsoft.com/en-us/library/system.web.configuration.formsauthpasswordformat.aspx) | Passwords are stored using a Message Digest 5 (MD5) hash digest. To validate credentials, the user password is hashed using the MD5 algorithm and compared to the stored value. The clear-text password is never stored or compared when using this value. This algorithm produces better performance than SHA1. |
| [SHA1](http://msdn.microsoft.com/en-us/library/system.web.configuration.formsauthpasswordformat.aspx) | Passwords are stored using the SHA1 hash digest. To validate credentials, the user password is hashed using the SHA1 algorithm and compared to the stored value. The clear-text password is never stored. Use this algorithm for improved security over the MD5 algorithm. |

The .NET Framework includes classes and methods that make it easy for you to create hashed values programmatically for persistent storage. One class that can be helpful for programming this task is the [FormsAuthentication](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.aspx) class. Its [HashPasswordForStoringInConfigFile](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.hashpasswordforstoringinconfigfile.aspx) method can do the hashing. For more precise control, you can use the [System.Security.Cryptography](http://msdn.microsoft.com/en-us/library/system.security.cryptography.aspx) classes as well.

Hashed passwords stored in a text file cannot be used to regenerate the original password, but they are potentially vulnerable to a dictionary attack. In this type of attack, the attacker, after gaining access to the password file, attempts to guess passwords by using software to iteratively hash all words in a large dictionary and compare the generated hashes to the stored hash. If you store hashed passwords in any way, you should require your users to choose passwords that are not common words and that contain some numbers and non-alphanumeric characters to help prevent dictionary attacks. Additionally, you can make credentials management easier by storing them using ASP.NET membership. For more information, see [Managing Users by Using Membership](http://msdn.microsoft.com/en-us/library/tw292whz.aspx).

.NET Framework 4 - ASP.NET

**Forms Authentication Utilities**

To manage forms authentication, you can use static methods of the [FormsAuthentication](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.aspx) class. The following table lists the methods.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [Authenticate](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.authenticate.aspx) | Attempts to validate the credentials from the configured credential store, given the supplied credentials. |
| [Decrypt](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.decrypt.aspx) | Returns an instance of the [FormsAuthenticationTicket](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthenticationticket.aspx) class, given an encrypted authentication ticket obtained from an HTTP cookie. |
| [Encrypt](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.encrypt.aspx) | Given a [FormsAuthenticationTicket](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthenticationticket.aspx), produces a string containing an encrypted authentication ticket suitable for use in an HTTP cookie. |
| [GetAuthCookie](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.getauthcookie.aspx) | Retrieves an encrypted authentication cookie as an [HttpCookie](http://msdn.microsoft.com/en-us/library/system.web.httpcookie.aspx) instance. The cookie is not added to the [Cookies](http://msdn.microsoft.com/en-us/library/system.web.httpresponse.cookies.aspx) collection. |
| [GetRedirectUrl](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.getredirecturl.aspx) | Returns the redirection URL for the request that caused the redirect to the logon page. |
| [HashPasswordForStoringInConfigFile](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.hashpasswordforstoringinconfigfile.aspx) | Given a password and a string identifying the hash type, produces a hash password suitable for storing in a configuration file. |
| [Initialize](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.initialize.aspx) | Initializes the [FormsAuthentication](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.aspx) class by reading configuration settings and getting the cookie values and encryption values for the current application. |
| [RedirectFromLoginPage](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.redirectfromloginpage.aspx) | Redirects an authenticated user to the originally requested URL. |
| [RenewTicketIfOld](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.renewticketifold.aspx) | Updates the sliding expiration on a [FormsAuthenticationTicket](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthenticationticket.aspx). |
| [SetAuthCookie](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.setauthcookie.aspx) | Creates an authentication ticket and attaches it to the cookie collection of the outgoing response. |
| [SignOut](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.signout.aspx) | Removes the authentication ticket by setting the authentication cookie or URL text to an empty value. This removes both durable and session cookies.  Description: Important note**Important**  Although the [SignOut](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.signout.aspx) method clears the ticket from the authenticated browser session, your application can still be susceptible to a replay attack from an unwanted source that has "sniffed" an authentication ticket. For information on mitigating against a replay attack with forms authentication, see [SignOut](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.signout.aspx). |

The following table lists helpful properties for managing forms authentication tickets.

|  |  |
| --- | --- |
| **Property** | **Description** |
| [FormsCookieName](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.formscookiename.aspx) | Gets the cookie name for the current application. |
| [FormsCookiePath](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.formscookiepath.aspx) | Gets the cookie path for the current application. |
| [CookiesSupported](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.cookiessupported.aspx) | Gets a value that indicates whether the application is configured to support cookieless forms authentication. |
| [CookieMode](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.cookiemode.aspx) | Gets a value that indicates whether the application is configured for cookieless forms authentication. |
| [CookieDomain](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.cookiedomain.aspx) | Gets the value of the domain of the forms authentication cookie. |
| [DefaultUrl](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.defaulturl.aspx) | Gets the URL that forms authentication will redirect to if no redirect URL is specified. |
| [LoginUrl](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.loginurl.aspx) | Gets the URL for the logon page that forms authentication will redirect to. |
| [RequireSSL](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.requiressl.aspx) | Gets a value indicating whether cookies must be transmitted using Secure Sockets Layer (SSL). |
| [SlidingExpiration](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.slidingexpiration.aspx) | Gets a value indicating whether sliding expiration is enabled. |
| [EnableCrossAppRedirects](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.enablecrossappredirects.aspx) | Gets a value indicating whether authenticated users can be redirected to URLs in other Web applications when the forms authentication ticket is not stored in a cookie. |

You can use the methods of the [FormsAuthentication](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.aspx) class to customize the way forms authentication works. You can also use them in the logon page handler to avoid having to explicitly code the redirection. The following code example shows an ASP.NET Web page that authenticates the user and redirects to the requested page.

Visual Basic

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl34_code');" \o "Copy Code)

<html>

<head>

<script language="VB" runat=server>

Sub SubmitBtn\_Click(Source As Object, e As EventArgs)

' Try to authenticate credentials supplied by user.

If FormsAuthentication.Authenticate \_

(UserName.Value, UserPassword.Value) Then

Dim ticket As New FormsAuthenticationTicket \_

(UserName.Value, False, 5000)

FormsAuthentication.RedirectFromLoginPage \_

(UserName.Value, Persist.Checked)

End If

End Sub

</script>

</head>

<body>

<form method=post runat=server>

<table>

<tr>

<td>Name:</td>

<td><input type="text" id="UserName" runat=server/>

</tr>

<tr>

<td>Password:</td>

<td><input type="password" id="UserPassword" runat=server/>

</td>

</tr>

</table>

<input type="checkbox" id="Persist" runat=server/>

<!-- Use persistent cookie -->

<br>

<input type="submit" OnServerClick="SubmitBtn\_Click" runat=server/>

</form>

</body>

</html>

C#

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl35_code');" \o "Copy Code)

<html>

<head>

<script language="C#" runat=server>

void SubmitBtn\_Click(Object Source, EventArgs e)

{

// Try to authenticate credentials supplied by user.

if (FormsAuthentication.Authenticate(UserName.Value,

UserPassword.Value))

{

FormsAuthenticationTicket ticket = new

FormsAuthenticationTicket(UserName.Value, false, 5000);

FormsAuthentication.RedirectFromLoginPage(UserName.Value,

Persist.Checked);

}

}

</script>

</head>

<body>

<form method=post runat=server>

<table>

<tr>

<td>Name:</td>

<td><input type="text" id="UserName" runat=server/></td>

</tr>

<tr>

<td>Password:</td>

<td><input type="password" id="UserPassword" runat=server/>

</td>

</tr>

</table>

<input type="checkbox" id="Persist" runat=server/>

<!-- Use persistent cookie. -->

<br>

<input type="submit" OnServerClick="SubmitBtn\_Click" runat=server/>

</form>

</body>

</html>

Applications that need detailed control over the HTTP cookie properties can construct the ticket and perform the redirection in custom code. In those cases, you should use encryption methods of the [FormsAuthentication](http://msdn.microsoft.com/en-us/library/system.web.security.formsauthentication.aspx) class to encrypt the authentication ticket.