Page 289-294

1. ASP.NET membership providers enable us to run multiple applications against the same back end by employing the concept of an *ApplicationName* value.
2. This value is part of the users and membership tables, and enables the creation of multiple identical user names in the same physical database, as long as each user name has a different application name.
3. We also need to ensure that our queries include the *ApplicationName* value to get the appropriate user’s information.
4. Required properties and members for creating a custom membership provider:
   1. *ApplicationName* property=====🡺 The name of the application using the membership information.
   2. ChangePassword method==🡺 Takes a user name, a current password, and a new password as input; updates the password.
   3. ChangePasswordQuestionAndAnswer method ==🡺 Takes a user name, a password, a password question, and a password answer as input; updates the password question and answer
   4. CreateUser method =🡺 Takes the name of a new user, a password, and an email address as input;
   5. The CreateUser method returns a MembershipUser object.
   6. DeleteUser method =🡺 Takes the name of a user as input and deletes that user’s information.
   7. Description property (from ProviderBase)==🡺 A string that describes the provider.
   8. EnablePasswordReset property=🡺 A Boolean value. users can use the ResetPassword method to overwrite their current password
   9. EnablePasswordRetrieval property ==🡺 users can retrieve their password using the GetPassword method.
   10. FindUsersByEmail method ==🡺 Returns a list of membership users in which the user name contains a match of the supplied emailToMatch
   11. FindUsersByName method=🡺 Returns a list of membership users in which the user name contains a match of the supplied usernameToMatch.
   12. GetAllUsers method=🡺 Returns a MembershipUserCollection.
   13. GetNumberOfUsersOnline method==🡺 Returns an integer value.
   14. The UserIsOnlineTimeWindow property is an integer value specifying the number of minutes to use when determining whether a user is online.
   15. GetPassword method=🡺 Takes a user name and a password answer as input, and retrieves the password.
   16. If the EnablePasswordRetrieval flag is not set, this method will throw a NotSupportedException.
   17. GetUser methods=🡺 The GetUser method returns a MembershipUser object.
   18. GetUserNameByEmail method ==🡺 Takes an email address as input and returns the first user name.
   19. Initialize method (from ProviderBase) ==🡺 Takes the name of the provider and a NameValueCollection of configuration settings as input.
   20. MaxInvalidPasswordAttempts property==🡺 Works with the PasswordAttemptWindow to guard against an unwanted source guessing the password or password answer of a membership user through repeated attempts.
   21. Name property (from ProviderBase)==🡺 A string that names the provider.
   22. PasswordAttemptWindow property===🡺 Works with the MaxInvalidPasswordAttempts property to determine the time period (in minutes)
   23. PasswordFormat property==🡺 Indicates the format that passwords are stored in: Clear, Encrypted, or Hashed.
   24. RequiresQuestionAndAnswer property=🡺 Indicates whether users must provide a password answer to retrieve their password using the GetPassword method, or reset their password using the ResetPassword method.
   25. RequiresUniqueEmail property==🡺 Indicates whether users must supply a unique email address value when creating a user. If a user already exists in the data source for the current ApplicationName, the CreateUser method returns null and a status value of DuplicateEmail.
   26. ResetPassword method.
   27. UnlockUser method==🡺 Takes a user name as input; stores the IsLockedOut property to false.
   28. UpdateUser method.
   29. ValidateUser method.
5. IIS is the primary mechanism for authentication because it comes bundled with seven providers.
6. starting with IIS 7, only the Anonymous authentication provider comes installed by default.
7. Anonymous authentication does not require users to input login credentials.
8. Basic authentication requires credentials.
9. Digest authentication is similar to Basic authentication, but the credentials are sent hashed.
10. Windows authentication uses credentials from Windows logged-in users and sends them with the HTTP request.
11. The Authorize attribute tells the system that any users calling the controller or the action need to be authenticated.
12. Forms authentication is one of the most commonly used authentication mechanisms.
13. The AllowAnonymous attribute tells the system that it is permissible for the users to not be authenticated.
14. Membership providers have been a part of ASP.NET since .NET 2.0.
15. What type of authentication accepts login credentials that will be checked against the domain or local server and are sent in a hashed format?
    1. Basic authentication
    2. Digest authentication
    3. Forms authentication
    4. Windows authentication
16. Forms authentication enables you to write code to validate user credentials. After it is complete, you can register the authentication cookies for use throughout the user’s visit by using which of the following?
    1. FormsAuthentication.SetAuthCookie
    2. FormsAuthenticationCookie = new FormsAuthenicationCookie();
    3. FormsAuthentication.ClearAuthCookie
    4. MembershipProvider.User =
17. What default attributes or inline checks would you use to create an accepted-list scenario in ASP.NET MVC 4? (Choose all that apply.)
    1. Authorize attribute
    2. RequireHttps attribute
    3. WebSecurity.IsAuthenticated
    4. AllowAnonymous attribute
18. What interfaces or classes should be implemented or inherited when creating custom authentication that is based on a non-Windows, third-party provider? (Choose all that apply.)
    1. ActiveDirectoryMembershipProvider
    2. IIdentity
    3. SqlMembershipProvider
    4. IPrincipal
19. What kind of helper methods does WebSecurity provide? (Choose all that apply.)
    1. Login
    2. ResetPassword
    3. CreateAccount
    4. ChangePassword
    5. DeleteAccount
20. Authorization is the process of giving a user permission to take an action on something, such as create, read, update, or delete.
21. Authorization is also the process of comparing a user’s capability to interact with items in the system against the user’s request to determine whether the user should be granted that permission.
22. The ASP.NET membership system is based on a provider model that acts as a framework.