ASP.NET程式設計
網站安全管理

講師：林賢達

Peter.lin@imestech.com

課程大綱

- 驗證與授權
- Windows驗證
- 表單驗證
- 登入控制項
- 成員資格管理
- 角色管理

驗證與授權

- 根據使用者輸入的帳號和密碼，對使用者身分進行確認(驗證)，並授予該名使用者存取資源的權限(授權)
- ASP.NET提供三種驗證模式
  - 使用組態檔的authentication元素來設定Web應用程式的驗證模式

```
<system.web>
    <authentication="Windows|Forms|Passport|None">
</system.web>
```

驗證與授權

- ASP.NET提供兩種授權方式
  - 檔案授權：使用NTFS設定本機資源的存取權限根據使用者的Windows帳號來判斷對某個檔案的請求是否有權限。僅對Windows整合式驗證，才有作用
  - URL授權：使用Web組態檔進行授權

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

驗證與授權

- 使用allow和deny元素允許或拒絕某個使用者或角色的存取權限
- 重要屬性
  - user：指定某個使用者，使用星號(*)表示所有使用者，問號(?)表示匿名使用者
  - roles：指定某個使用者群組(具有相同權限的使用者集合)
  - verbs：指定對特定的HTTP請求進行處理

驗證與授權

- 應用範例
  - 只允許某個網域使用者

    <allow users="imestech.com\peter.lin" />
  - 允許所有使用者群組

    <allow roles="*" />
  - 拒絕匿名使用者

    <deny users="?" />
  - 拒絕所有使用者使用Debug請求

    <deny user="*" verbs="DEBUG" />

對目錄或檔案進行授權

- 使用<location>元素對子目錄或子目錄的某個檔案設定授權方式
- 或者在子目錄的組態檔來設定

```
<location path="目錄|檔案">
    <system.web>
        <!--授權方式-->
    </system.web>
</location>
```

HttpContext.Current.User

- 當通過身分驗證後，使用者訊息會被儲存在HttpContext.Current.User
- 重要成員
  - Identity屬性：表示使用者身分，然後使用Name屬性取出使用者名稱；使用IsAuthenticated屬性判斷使用者是否經過驗證；使用AuthenticationType屬性取出驗證模式
  - IsInRole方法：判斷使用者是否屬於某個使用者群組

ASP.NET網站管理工具

- 安全性設定
  - 指定驗證類型：表單驗證或Windows驗證
  - 建立使用者和管理使用者(編輯和刪除)
  - 建立或管理角色(刪除角色和加入/移除使用者)
  - 建立存取規則，設定對整個網站或個別資料夾的存取，允許或拒絕哪些使用者或角色
  - 管理存取規則，顯示和刪除整個網站或個別資料夾的存取規則

ASP.NET網站管理工具

- 提供者
  - 成員資格提供者

    AspNetSqlMemberShipProvider (預設)，定義在 machine.config
  - 角色提供者

    AspNetSqlRoleProvider(預設)，定義在 machine.config

Windows驗證

- 使用WindowsPrincipal類別對使用者身分進行驗證，檢查使用者的Windows帳號是否存在於網域或工作群組，然後根據使用者所屬的群組進行授權
- 使用Windows驗證時，必須搭配IIS的整合式Windows驗證
- 常用於企業的內部網站

Windows驗證

- 將authetication元素的mode屬性設定成Windows驗證模式

```
<authentication mode="Windows" />
<authorization>
        <allow user="domainname\username" />
        <deny user="*" />
</authorization>
```

註：紅色文字為驗證；藍色文字為授權

允許網域使用者登入，然後拒絕其他使用者

表單驗證

- 使用應用程式的資料庫進行使用者身分驗證，並使用Cookie來記錄使用者的登入資訊。當瀏覽其他網頁時，即可使用Cookie得知使用者身分
- 使用Forms驗證時，必須啟用IIS的匿名存取

```
<authentication mode="Forms" >
    <forms loginUrl= "login.aspx" />
</authentication>
<authorization>
    <deny user="?" />
</authorization>
```

表單驗證

- forms元素的重要屬性
  - name屬性：記錄使用者訊息的Cookie名稱，預設為.ASPXAUTH，不同應用程式需要個別指定
  - loginUrl屬性：指定登入網頁，預設為Login.aspx
  - defaultUrl屬性：指定網站的首頁
  - cookieless屬性：指定是否使用Cookie來記錄使用者訊息
  - timeout屬性：指定Cookie的過期時間，預設為30分鐘

表單驗證

- forms元素的重要屬性(續)
  - slidingExpiration屬性：設定是否從最近一次存取時間開始計算timeout時間
  - protection屬性：指定Cookie的儲存方式，可以是All (Validation + Encryption), None, Validation 和 Encryption，其中

    Validation，使用數位簽章確保在傳送過程中不會被監聽或竄改，但不加密

    Encryption，使用DES加密，但不確保在傳送過程中不會被監聽或竄改

表單驗證

- 使用credentials元素儲存使用者帳號和密碼
  - 使用passwordForm設定密碼的加密方式，clear表示不加密，MD5使用MD5 hash加密，SHA1使用SHA1 hash加密

```
<authentication mode="Forms" >
    <forms loginUrl= "login.aspx" >
        <credentials passwordFormat="Clear|MD5|SHA1">
            <user name="peter" password="Ab12345" />
            <user name="neo" password="1qaz@WSX" />
        </credentials>
    </forms>
</authentication>
```

Passport驗證模式

- 使用PassPort SDK來開發登入Microsoft Passport Network的驗證機制，例如MSN Messenger、MSN Hotmail 及其他網站和服務
- 提供單一簽入認證功能(Single Sing-On)
- Microsoft Passport Network

  https://accountservices.passport.net/ppnetworkhome.srf?vv=500&lc=1028

登入控制項

- Login控制項在網頁上顯示登入畫面
- 重要成員
  - UserName：取出輸入的使用者名稱
  - Password：取出輸入的密碼
  - Authenticate事件：實作自己的驗證邏輯
  - TitleText：設定登入畫面的標題文字
  - SubmitButtonText：設定[登入]按鈕的說明文字
  - CreateUserUrl屬性：指定建立使用者網頁的URL
  - PasswordRecoveryUrl：指定密碼復原網頁的URL

登入控制項

- 重要成員(續)
  - CreateUserText設定建立使用者連結的說明文字
  - DisplayRememberMe顯示記憶密碼的核取方塊?
  - InstructionText：提供指示的顯示文字
  - LoggedIn事件：當登入成功時所觸發的事件
  - LoggingIn事件：按下登入按鈕時所觸發的事件
  - LoginError事件：當登入失敗時所觸發的事件

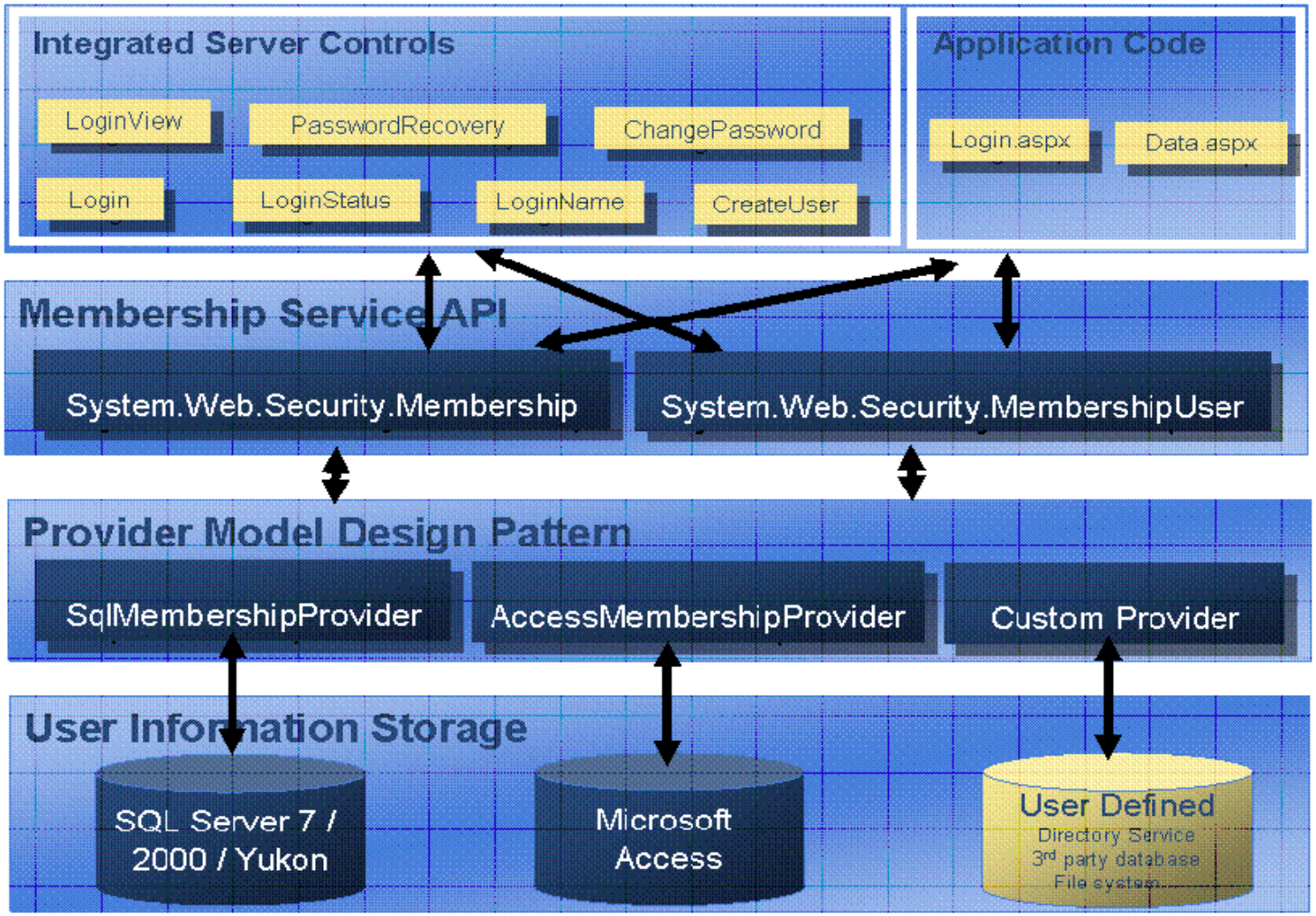
登入控制項

- LoginName控制項
  - 在網頁上顯示使用者名稱
- LoginView控制項
  - 根據登入狀態和角色顯示不同的檢視
- PasswordRevovery控制項
  - 當使用者忘記密碼時，要求使用者回答密碼問題
- ChangePassword控制項
  - 提供修改密碼的功能

登入控制項

- LoginStatus登入控制項
  - 在網頁上顯示目前使用者的登入狀態，如果尚未經過驗證，顯示Login，否則顯示Logout
  - 如果按下Login連結，則導向登入頁面；按下Logout連結，則執行登出功能
- CreateUserWizard控制項
  - 在網頁上顯示使用者註冊畫面，建立新的使用者帳號
  - 預設的密碼強度要求7位以上的長度，而且至少必須包含一個非字母數字的特殊字元

成員資格服務


Integrated Server Controls
LoginView
PasswordRecovery
ChangePassword
Login
LoginStatus
LoginName
CreateUser
Application Code
Login.aspx
Data.aspx
Membership Service API
System.Web.Security.Membership
System.Web.Security.MembershipUser
Provider Model Design Pattern
SqlMembershipProvider
AccessMembershipProvider
Custom Provider
User Information Storage
SQL Server 7 / 2000 / Yukon
Microsoft Access
User Defined
Directory Service
3rd party database
File system

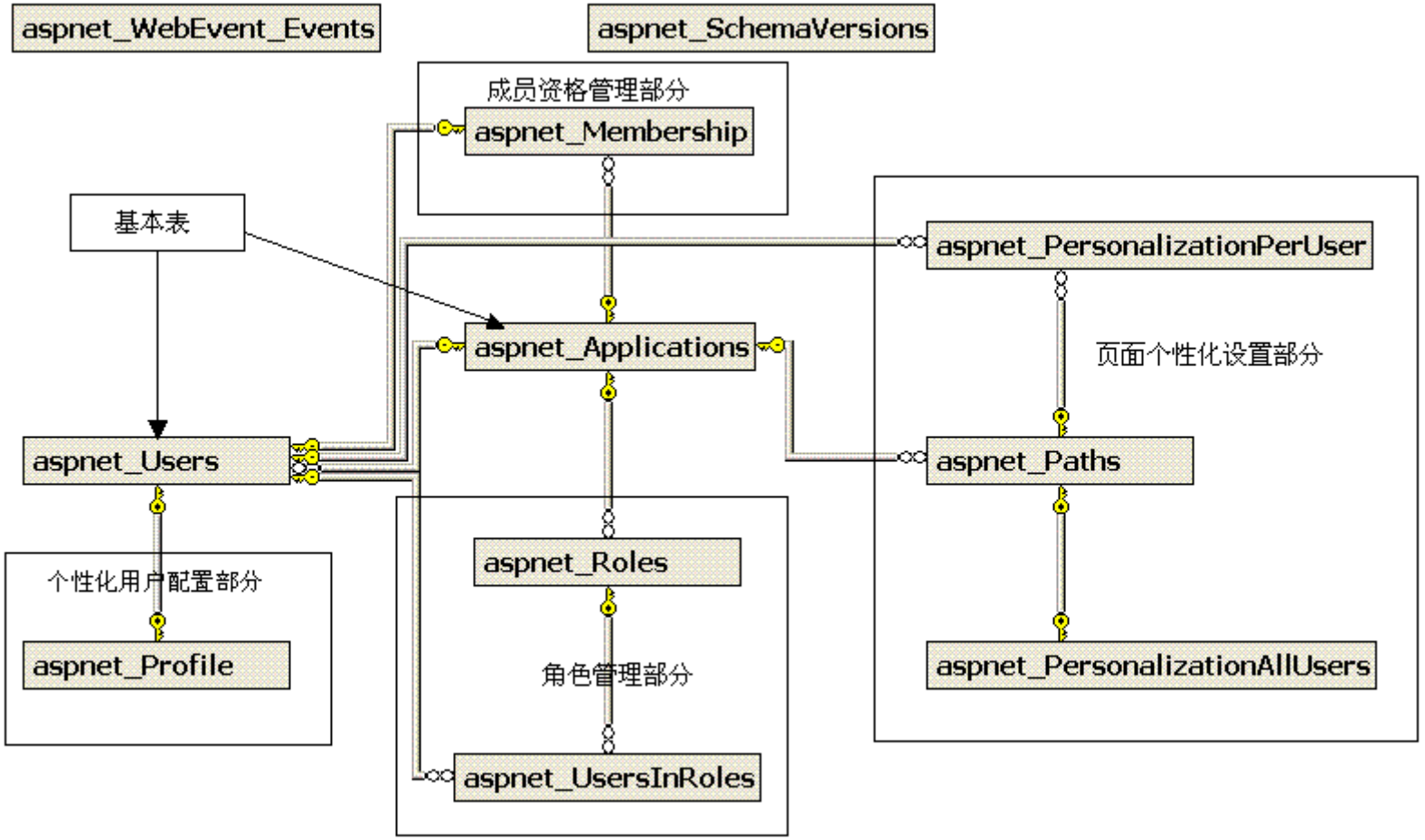
ASP.NET網站管理工具

- 提供者設定
  - 成員資格提供者：AspNetSqlMembershipProvider(預設)
  - 角色提供者：AspNetSqlRoleProvider(預設)

```
<system.web>
  <membership>
    <providers>
      <add name="AspNetSqlMembershipProvider" type="System.Web.Security.SqlMembershipProvider"
        connectionStringName="LocalSqlServer" />
    </providers>
  </membership>
</system.web>


<configuration>
  <connectionStrings>
    <add name="LocalSqlServer" connectionString="Server=(local);Database=aspnetdb;"
      providerName="System.Data.SqlClient"/>
  </connectionStrings>
</configuration>
```

aspnetdb資料庫



變更密碼強度

- 在web.config加入自訂的成員資格提供者或修改預設的AspNetSqlMemberShipProvider
- 修改下列屬性
  - minRequiredPasswordLength：建立有效密碼時，要求的最小長度
  - minRequiredNonalphanumericCharacters：建立有效密碼時，必須輸入的最少特殊非字母數字的字元數
- 使用ASP.NET網站管理工具變更成員資格提供者名稱(如果需要)

ASP.NET網站安全性

- ASP.NET 1.x 提供FormsAuthentication類別進行身分驗證和存取驗證Cookie的方法
- ASP.NET 2.0
  - 成員資格管理API – MemberShip類別
  - 角色管理API – Roles類別
  - 使用ASP.NET網站管理工具
- ASP.NET 2.0 Quick Start Tutorial

  http://quickstarts.asp.net/quickstartv20/aspnet/

成員資格管理

- 使用System.Web.Security.MemberShip類別和MemberShipUser類別來進行成員資格管理
- MemberShip主要負責管理MemberShipUser
- MemberShipUser描述成員資格儲存中的一個註冊的使用者訊息
  - 使用MemberShip的GetUser或CreateUser取傳回一個MemberShipUser物件
  - 用於取出，修改和重設密碼

MemberShip類別

- 建立使用者
  - 使用Create靜態方法來註冊新的使用者
- 使用者身分驗證

```
If MemberShip.ValidateUser(UserName.Text, Password.Text) Then
    RedirecFromLoginPage(UserName.Text, RememberMe.Checked)
End If
```

- 取出所有使用者
  - 使用GetAllUsers靜態方法

MemberShip類別

- 取出使用者資訊
  - 使用GetUser靜態方法，傳入使用者名稱，傳回MemberShipUser物件
- 更新使用者的註冊資訊
  - 使用ObjectDataSource和DetailsView控制項
- 產生隨機的密碼
  - 使用GeneratePassword靜態方法

MemberShip類別

- 刪除使用者
  - 使用DeleteUsuer靜態方法刪除使用者，傳入使用者名稱，傳回是否完成刪除
- 解除被鎖定的使用者
  - 使用IsLockout屬性檢查使用者是否被鎖定，如果是，可以使用UnLockUser靜態方法解除鎖定

角色管理

- 使用System.Web.Security.Roles類別來管理角色、加入/刪除角色中的使用者和頁面授權
- 啟用角色管理

```
<system.web>
    <roleManager enabled="true" />
</system.web>
```

Roles類別

- 建立角色
  - 使用CreateRole靜態方法來新增角色

```
If Not Roles.RoleExists("Developers") Then
   Roles.CreateRole("Developers")
End If
```

- 刪除角色
  - 使用DeleteRole靜態方法來刪除不再使用的角色

Roles類別

- 加入使用者到角色中
  - 使用AddUserToRole靜態方法來新增使用者

```
If Not Roles.IsUserInRole("Developers") Then
    Roles.AddUser(MemberShip.GetUser().UserName,"Developers")
End If
```

- 移除角色中的某個使用者
  - 使用RemoveUserFromRole靜態方法刪除使用者
- 取出角色中的使用者
  - 使用GetUsersInRole靜態方法