

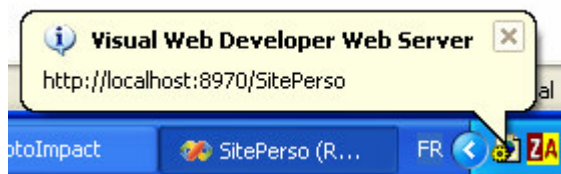
Sécurité ASP.NET 2.0 : l'authentification par formulaire

Cet article a été réalisé à partir de la Beta 1 de Visual Studio 2005. Il a pour but de vous initier à quelques nouveautés d'ASP.NET 2.0.

1. Faire tourner son projet sous IIS

🚩 Un nouveau serveur Web pour le développeur

En lançant notre premier projet Web, nous avons la surprise de découvrir que notre site ne se lance pas sous IIS mais sous un serveur dédié au développeur.

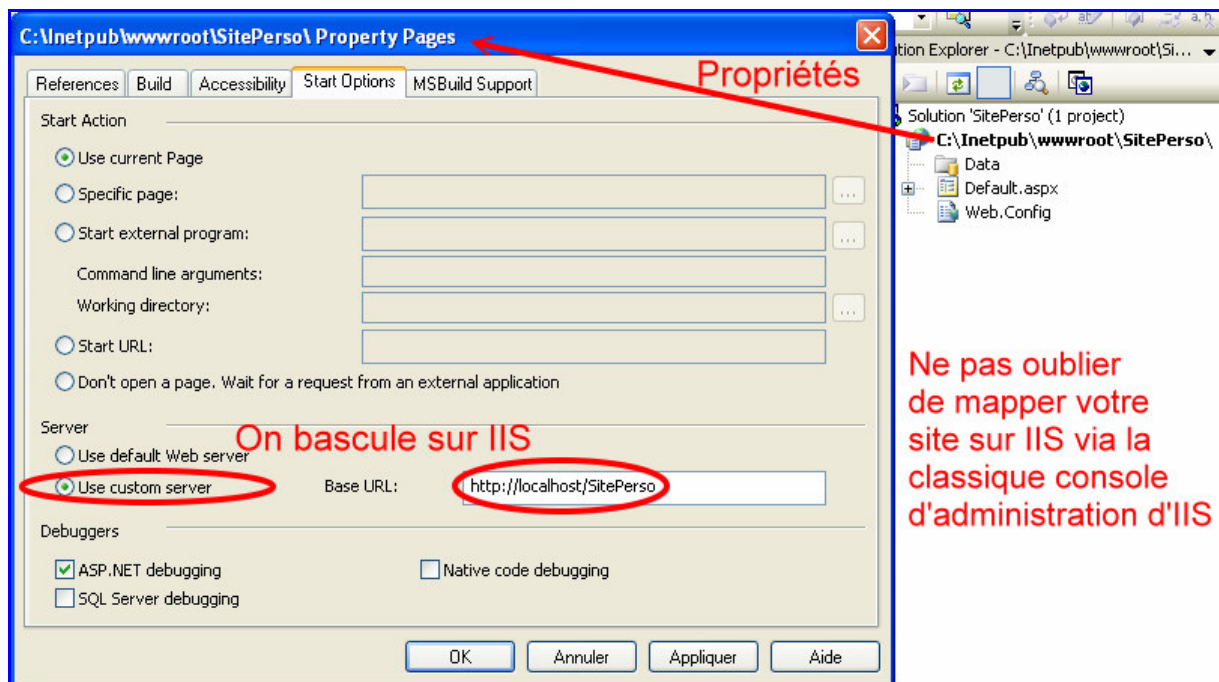


Pourquoi pas ? Après tout, pourquoi lancer la « grosse artillerie » (IIS) quand un petit serveur web suffit. A vrai dire, le déploiement final sous IIS se fait sans problème même si on a travaillé sous « Visual Web Developer Web Server ».

Mais, voyons comment repasser sous IIS car j'ai mes habitudes...

🚩 Faire fonctionner votre projet sous IIS

Cela se fait relativement facilement. L'illustration suivante montre comment faire :



Et maintenant, le lancement de notre site via Visual Studio se fait bien sous IIS.

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire

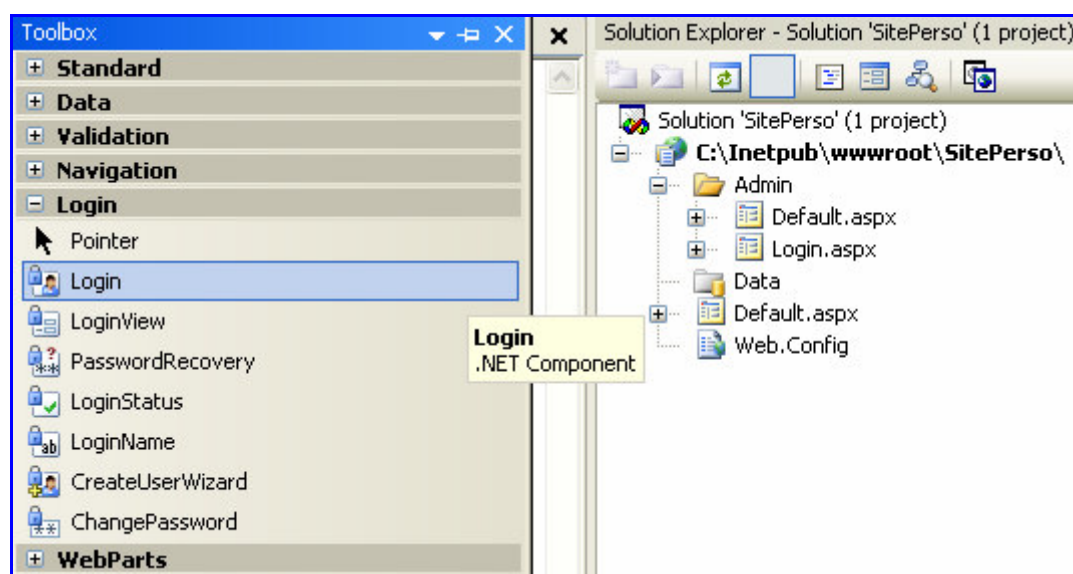
2. Mise en place d'une authentification par formulaire

Des nouveaux composants disponibles

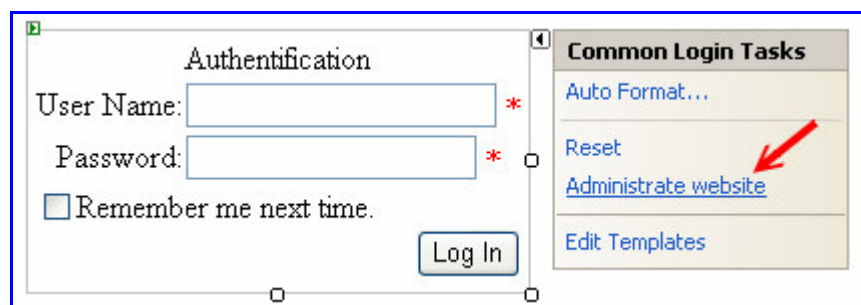
ASP.NET 2.0 fournit un grand nombre de nouveaux contrôles et de composants. Toujours dans le but de nous proposer « en standard » des fonctionnalités nouvelles et nous faire gagner du temps.

Nous allons maintenant restreindre l'accès à notre site. Toute personne se plaçant sous le répertoire « admin » devra nécessairement s'identifier. Nous allons définir une page « login.aspx » qui permettra de s'authentifier.

La boîte à outils nous propose des nouveaux composants relatifs à l'authentification, notamment le composant « Login ».



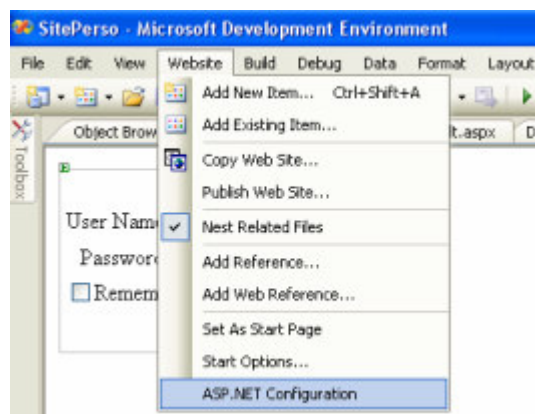
Nous allons nous en servir. Nous pouvons remarquer un « smart tag » sur le composant qui permet de personnaliser celui-ci. Nous allons cliquer sur « administrate Website ».



Nous parvenons alors sur un site d'administration qui va nous permettre de paramétrer notre application web. Nous pouvons aussi accéder à ce site via le menu « ASP.NET configuration » de Visual Studio :

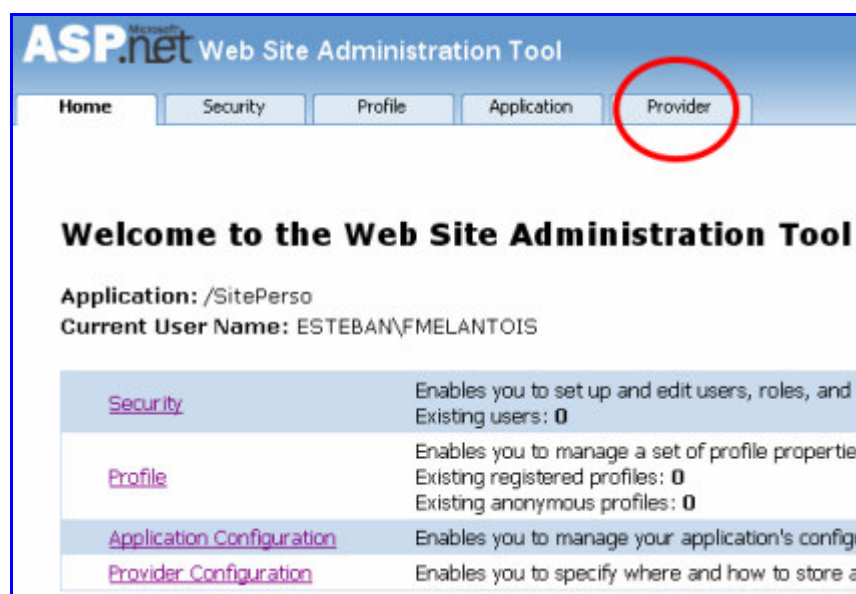
.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire



Les Providers (MembershipProvider)

Nous obtenons une page avec un certain nombre d'onglets. Et nous allons nous attarder à fournir un « provider » à notre composant « Login » qui en attend un (un « Membership Provider »)



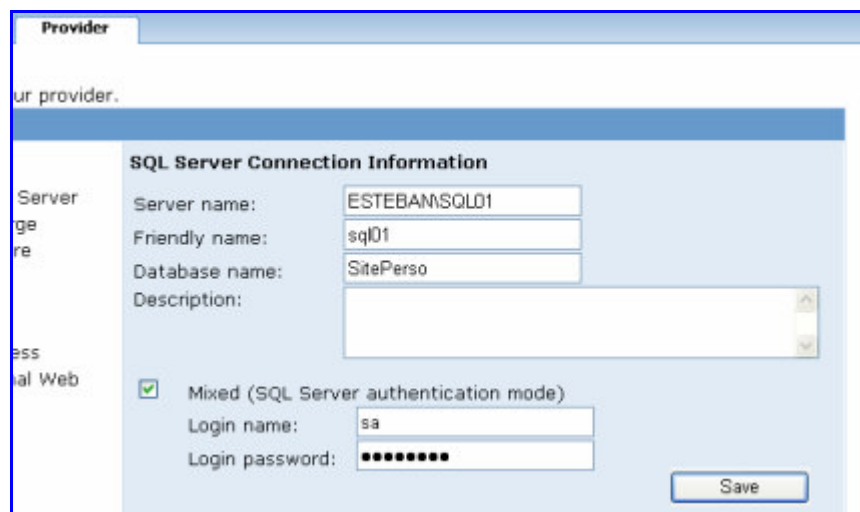
La version beta 1 nous fournit des providers en standard dont l'AspNetAccessProvider qui est rattaché à une base de données Access dans le répertoire « Data » de votre application. Ce provider sera remplacé dans la version finale par un provider rattaché à la base SQL server 2005 Express Edition.

Nous allons préférer choisir notre propre base de données (Sql Server 2000) afin de regrouper les tables qui vont y être créées et les tables que nous définirons nous-même pour notre application web.

Nous créons au préalable notre base de données « SitePerso » via l'Enterprise Manager de Sql Server. Nous configurons ensuite très simplement, dans le site d'administration de notre site web, l'accès à notre base via ce formulaire :

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire



ur provider.

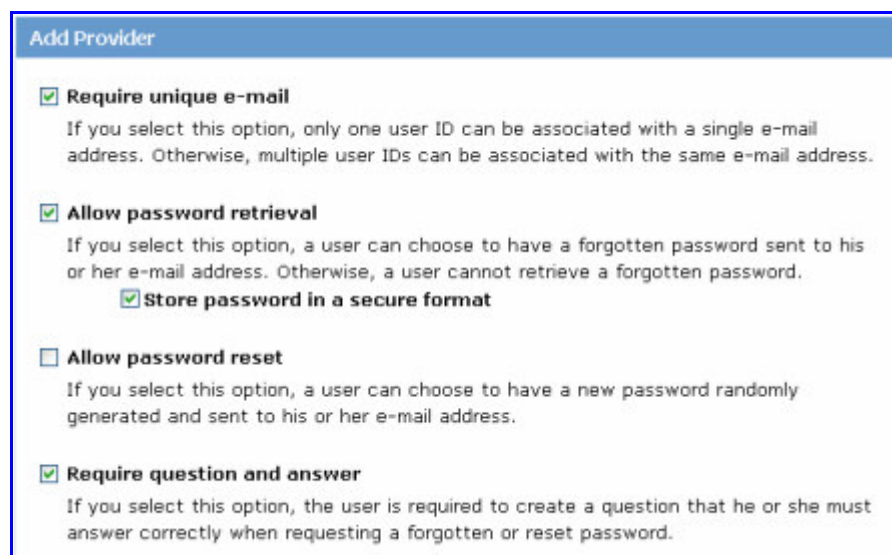
SQL Server Connection Information

Server name: ESTEBANSQL01
Friendly name: sql01
Database name: SitePerso
Description:

Mixed (SQL Server authentication mode)
Login name: sa
Login password: ●●●●●●

Save

En sauvegardant, voici les options proposées :



Add Provider

Require unique e-mail
If you select this option, only one user ID can be associated with a single e-mail address. Otherwise, multiple user IDs can be associated with the same e-mail address.

Allow password retrieval
If you select this option, a user can choose to have a forgotten password sent to his or her e-mail address. Otherwise, a user cannot retrieve a forgotten password.
 Store password in a secure format

Allow password reset
If you select this option, a user can choose to have a new password randomly generated and sent to his or her e-mail address.

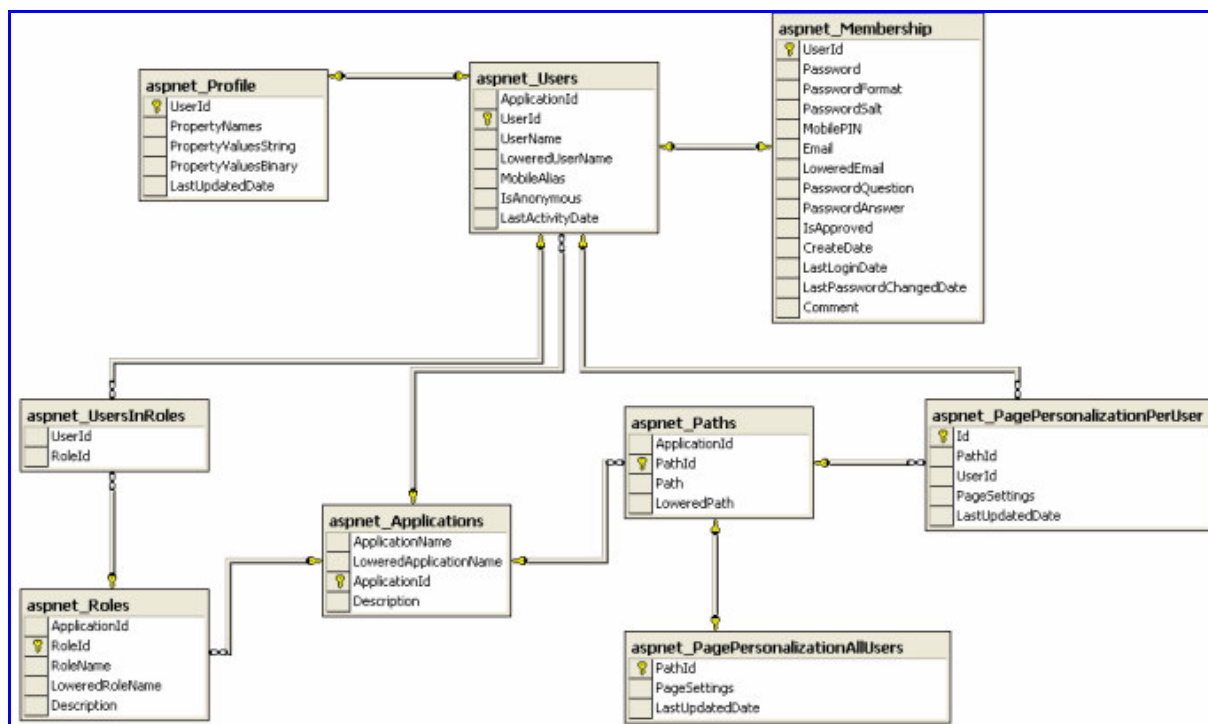
Require question and answer
If you select this option, the user is required to create a question that he or she must answer correctly when requesting a forgotten or reset password.

Comme vous le voyez un certain nombre d'options intéressantes sont disponibles. On peut par exemple s'assurer qu'à un email corresponde un membre, qu'un utilisateur puisse retrouver son mot de passe (un peu normal, non ?), avoir recours à une question - réponse pour s'assurer qu'il s'agit bien de l'utilisateur ou encore faire en sorte que le mot de passe soit automatiquement crypté dans la base de données.

Cliquons : Nous avons alors créé notre provider « sql01 ». Jetons un œil à notre base de données via un diagramme pour avoir un aperçu des tables et des relations créées.

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire



Le nommage des tables permet de comprendre que l'organisation est assez simple. Les opérations sur ces tables sont gérées par des procédures stockées.

Puisque nous avons la structure générale, rien ne nous interdit à l'avenir d'ajouter nos propres tables et de réaliser des liaisons avec celles représentant les utilisateurs, leurs rôles ou leurs profils.

Retournons dans « l'Administrateur Web », nous allons y définir des membres. Tout se passe dans l'onglet « sécurité ». Il nous faut définir au moins un membre et un rôle pour pouvoir tester notre authentification par formulaire.

ASP.NET Web Site Administration Tool

Home Security Profile Application Provider

You can use the Web Site Administration Tool to manage all the security settings for your application. You can set up users and passwords (authentication), create roles (groups of users), and create permissions (rules for controlling access to parts of your application).

By default, user information is stored in a Microsoft Access database in the Data folder of your Web site. If you want to store user information in a different database, use the Provider tab to select a different provider.

[To configure security step by step, use the Security Setup Wizard.](#)

Click the links in the table to manage the settings for your application.

Users	Roles	Access Rules
Existing users: 0 Create user Manage users Select authentication type	Roles are not enabled Enable roles Create roles Manage roles	Create access rules Manage access rules

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire

Active	User ID	Roles
<input checked="" type="checkbox"/>	fred	Add "fred" to roles: <input checked="" type="checkbox"/> administrateurs

Allons voir le fichier web.config de notre application web :

```
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <appSettings/>
  <connectionStrings>
    <add name="webAdminConnection632309431819052448"
          connectionString="server=ESTEBAN\SQL01;user ID=sa;password=password;database=SitePerso" />
  </connectionStrings>
  <system.web>
    <!--
      Set compilation debug="true" to insert debugging symbols into the compiled page.
      Because this affects performance, set this value to true only during development.
    -->
    <membership defaultProvider="sql01">
      <providers>
        <add connectionStringName="webAdminConnection632309431819052448"
              applicationName="/SitePerso" description="" requiresUniqueEmail="true"
              enablePasswordRetrieval="false" enablePasswordReset="false" requiresQuestionAndAnswer="true"
              passwordFormat="Hashed" name="sql01" type="System.Web.Security.SqlMembershipProvider, System.Web" />
      </providers>
    </membership>
    <profile defaultProvider="sql01">
      <providers>
        <add connectionStringName="webAdminConnection632309431819052448"
              applicationName="/SitePerso" description="" name="sql01" type="System.Web.Profile.SqlProfileProvider, System.Web" />
      </providers>
    </profile>
    <roleManager enabled="true" defaultProvider="sql01" domain="localhost">
      <providers>
        <add connectionStringName="webAdminConnection632309431819052448"
              applicationName="/SitePerso" description="" name="sql01" type="System.Web.Security.SqlRoleProvider, System.Web" />
      </providers>
    </roleManager>
  </system.web>
</configuration>
```

Le fichier a été modifié. On y retrouve des sections consacrées à nos différents providers (MemberShip, Role ou Profile). Une section particulière « connectionStrings » permet de rassembler toutes les chaînes de connexion, ce qui est relativement utile pour s'y retrouver.

Finalisation de l'authentification par formulaire

Ajoutons ces lignes de code dans le fichier web.config :

```
<authentication mode="Forms">
  <forms name=".ASPXUSERAUTH" loginUrl="Admin/login.aspx" />
</authentication>
```

et

```
<location path="admin">
  <system.web>
```

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire

```
<authorization>
  <deny users="?" />
</authorization>
</system.web>
</location>
```

A la page « Login.aspx », fournissons le « provider » au composant Login :

```
<asp:Login ID="Login1" Runat="server" DestinationPageUrl="-/Admin/Default.aspx"
MembershipProvider="sql01"/>
```

Compilons le tout et accédons au répertoire « admin », nous sommes bien redirigé vers la page « Login.aspx ». Vous constaterez quelques erreurs javascript, c'est normal nous sommes en beta 1 ! Si nous regardons la source via notre navigateur web, nous identifions la ligne à incriminer :

```
<script
src="WebResource.axd?a=s&r=WebUIValidation.js&t=632263671396796896"
type="text/javascript"></script>
```

Par curiosité, ouvrez une nouvelle instance de votre navigateur web, et copiez ceci

```
http://localhost/SitePerso/WebResource.axd?a=s&r=WebUIValidation.js
```

Pour faire fonctionner correctement, nous avons remplacé « & ; » par « & ».

Nous obtenons du javascript. Le composant Login génère du code en appelant l'« Http Handler » WebResource.axd.

Refermons cette parenthèse, remplissons le formulaire Login de notre page par le login et le mot de passe que vous aurez défini précédemment. L'authentification se passe bien puisque nous sommes redirigés sur la page « Admin/Default.aspx ».

Vous aurez sans doute remarqué que nous n'avons pas écrit la moindre ligne de code !

Vous allez pouvoir maintenant commencer à réaliser votre « back-office ». Pour gérer les membres, les différents composants de la famille « Login » de la boîte à outils vont pouvoir vous aider à le faire très vite. Vous allez pouvoir jouer avec les rôles aussi pour établir des niveaux d'accès.

Bien évidemment, si une grande partie se fait « by design », les codeurs auront accès aux différents API relatives sans problème.

Vous pouvez même redéfinir vos propres providers grâce aux classes :

```
System.Web.Security.MembershipProvider
System.Web.Security.RoleProvider
```

C'est une architecture bien pensée puisque notre application (son code) ne dépend pas du type de base de données (on change de provider via le web.config).

.NET passionnément, tout simplement

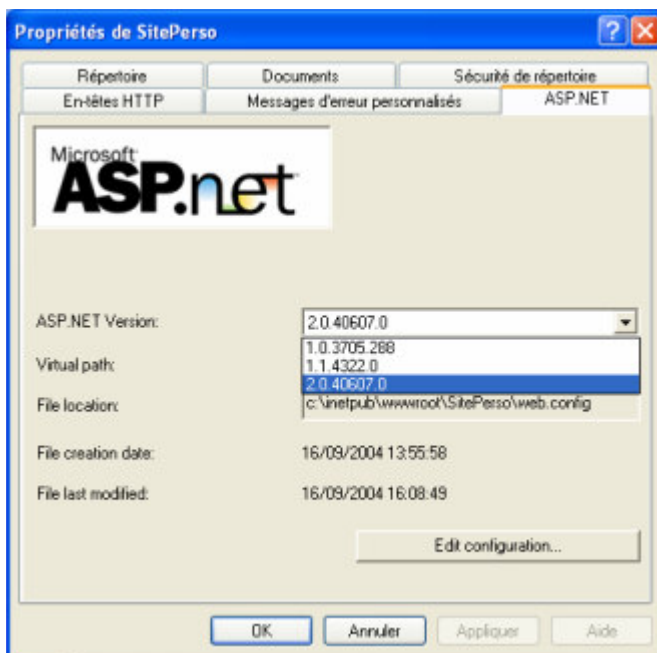
Sécurité ASP.NET 2.0 : l'authentification par formulaire

Par exemple, si vous avez une base « MySQL », rien ne s'oppose à son utilisation. Bien évidemment, il vous faudra coder les providers (la communauté Open Source en fournira sans doute très rapidement) en héritant de `MembershipProvider`, de `RoleProvider`...

3. Un regard dans IIS

Vous vous doutez bien que si j'ai insisté pour faire la présentation en me servant d'IIS plutôt que du serveur Web dédié au développeur, ce n'est pas seulement parce que j'ai du mal à perdre certaines habitudes. Nous avons des découvertes à faire ;-)

Lançons la console d'administration d'IIS. Et nous constatons que nous avons un nouvel onglet dédié à ASP.NET :



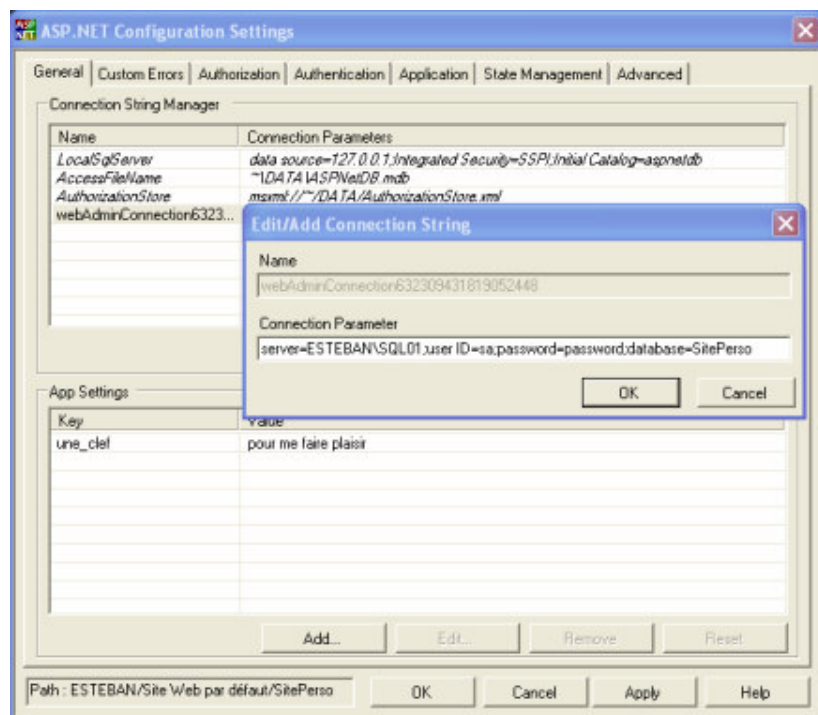
Celui-ci permet de spécifier sur quelle version du framework fonctionne votre application web.

Restons sur la version 2.0 et cliquons sur « Edit Configuration ».

Nous obtenons le tableau de bord suivant :

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire



Ce tableau est mappé sur le fichier « web.config » et inversement ! Je vous laisse découvrir toutes les possibilités de gérer votre application Web de manière plus conviviale que l'édition du fichier web.config, en particulier, surtout pour l'administrateur de la machine en production !

Terminons par un autre aspect de sécurité : je vous ai sans doute agacé en prétendant parler de sécurité et en laissant en clair le mot de passe de ma base de données dans le web.config. Eh bien, je l'ai fait exprès ;-)

L'administrateur de la machine en production va s'en charger. Et comment ?

Grâce à une option de la commande en ligne d'aspnet_regiis.exe :

Aspnet_regiis -pe connectionStrings -app /SitePerso

-pe permet d'encrypter une section du Web.config

Regardons notre fichier web.config :

```
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <protectedData>
    <protectedDataSections>
      <add name="connectionStrings" provider="RSAProtectedConfigurationProvider" />
    </protectedDataSections>
  </protectedData>
  <appSettings/>
  <connectionStrings>
    <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
      xmlns="http://www.w3.org/2001/04/xmlenc#"
      <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripleDES-cbc" />
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#"
        <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#"
          <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
          <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#"
            <KeyName>RSA Key</KeyName>
          </KeyInfo>
          <CipherData>
            <CipherValue>dDxYcayX+ltad19F90p25LTZPaGBNrIk+LutPG7ee4M6/kPdUNHom5wkGgeDPc6seA0t3M3g7/tV31tIQX
          </CipherValue>
          </CipherData>
        </EncryptedKey>
        </KeyInfo>
        <CipherData>
          <CipherValue>HoEHgmsUW51P2EjOkk7DborFAFmvg6hFmJfQz8GtQObTERBb121hPqSoyLdro2/RQPez3cVC9f1XfCJxkAt9
          </CipherValue>
        </CipherData>
      </EncryptedData>
    </connectionStrings>
```

.NET passionnément, tout simplement

Sécurité ASP.NET 2.0 : l'authentification par formulaire

Ce n'est pas merveilleux ?

Vous allez me dire comment revient-on en arrière ?

Aspnet_regiis -pd connectionString -app /SitePerso

-pd permet de décrypter une section du Web.config

4. Conclusion

Il reste encore beaucoup de choses à dire et à découvrir sur le sujet. Pour cette première approche, je m'en suis tenu à l'essentiel.

La version finale d'ASP.NET 2.0 devrait nous faire gagner du temps sur des tâches aussi récurrentes que l'authentification.

Les méthodes « à l'ancienne » que vous avez pu appliquer avec ASP.NET 1.1 fonctionnent encore, je vous rassure. Vos « back-offices » que vous avez rendus plus ou moins génériques pourront encore vivre... Mais petit à petit, vous prendrez de nouvelles habitudes ;-)

5. En savoir plus

La documentation sur le sujet est en perpétuel incrémentation.

<http://msdn.microsoft.com/asp.net/whidbey/default.aspx>

New security Features in ASP.NET 2.0

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/SecFeatNT2.asp>

Un numéro spécial de MSDN Magazine : ASP.NET 2.0 Revealed !

<http://msdn.microsoft.com/msdnmag/issues/04/06/default.aspx>

Une série de Webcasts sur ASP.NET 2.0 à suivre :

<http://msdn.microsoft.com/training/webcasts/#ASP>

Sécurisez vos applications .NET partie 1 et 2 (Elise Dupont)

<http://www.dotnet-fr.org/sections.php3?op=viewarticle&artid=71>

<http://www.dotnet-fr.org/sections.php3?op=viewarticle&artid=75>

N'hésitez pas à me contacter :

Frédéric Mélantois

Email : fmelantois@free.fr