Asp.Net 2.0 et Visual Studio.NET 2005

Les nouveautés

Sommaire

I. Présentation

Nous approchons de plus en plus de la date de sortie de Visual Studio .NET 2005 que beaucoup d'entre vous ont certainement déjà essayé en version beta.

Pour eux et pour les autres, voici les principales nouveautés apportées par Visual Studio .NET 2005 en ce qui concerne asp.NET 2.0. Par ailleurs, asp.NET 2.0, la nouvelle version d'asp.NET, amène également toute une série de nouveautés et de facilités dans le monde du développement web. Nous n'en verrons qu'une infime partie tellement elles sont nombreuses.

II. Les nouveautés de Visual Studio .NET 2005 appliquées à asp.NET 2.0

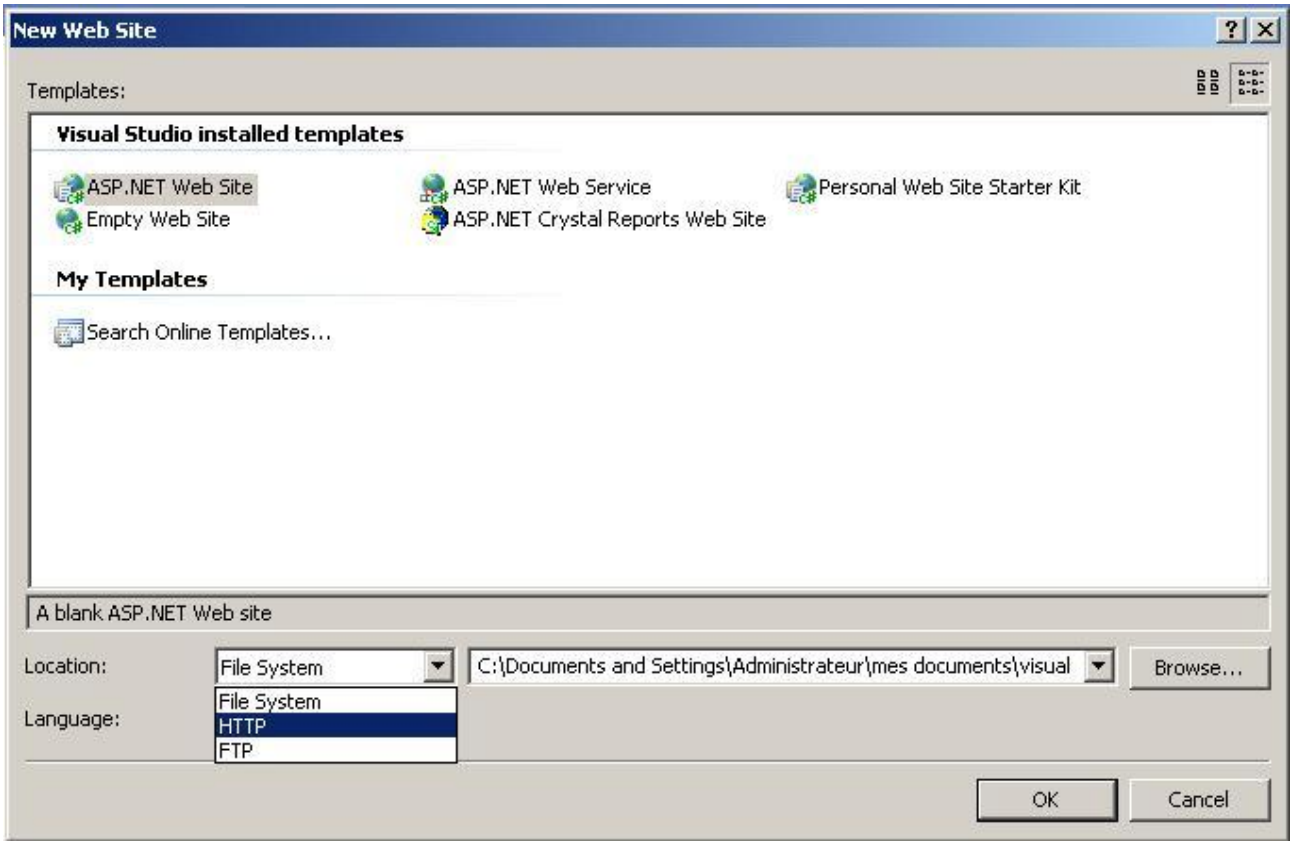
A. Un répertoire au lieu d'une solution...

Désormais, il n'y a plus de solution pour les applications asp.NET comme nous l'avions avec Visual Studio .NET 2002 et 2003. Effectivement, Visual Studio .NET 2005 se base sur des répertoires.

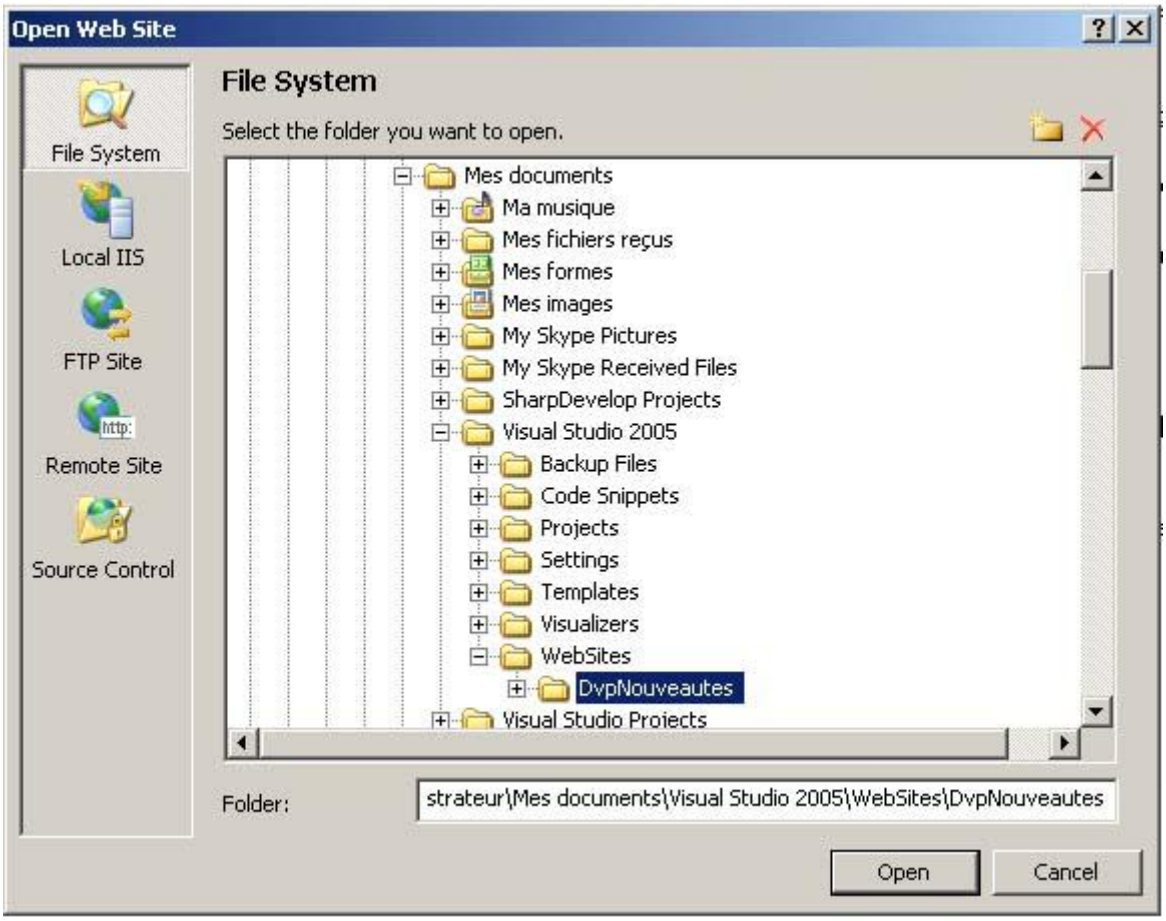
Ainsi, première constatation à l'ouverture de Visual Studio .NET 2005 (VS pour les intimes, on se permettra donc de l'appeler de la sortie car nous allons le fouiller de fond en comble), c'est bien dans File > Open > Web Site... et non pas Project / Solution comme nous en avions l'habitude.

Il en va de même pour créer un projet.

Comme on peut le voir sur l'image suivante, il est possible de créer une application dans un répertoire de Windows ou encore directement par FTP ou HTTP.



Bien entendu, il est possible de faire de même pour l'ouverture (ca pourrait être embêtant de pouvoir créer des sites web sans pouvoir en faire la maintenance). Du coup, l'interface d'ouverture est toute nouvelle et propose différentes options. L'interface est adapté au type d'ouverture choisi. Dans le cas d'un répertoire sur un disque local à la machine de développement, on reconnaîtra le composant classique de sélection des répertoires.



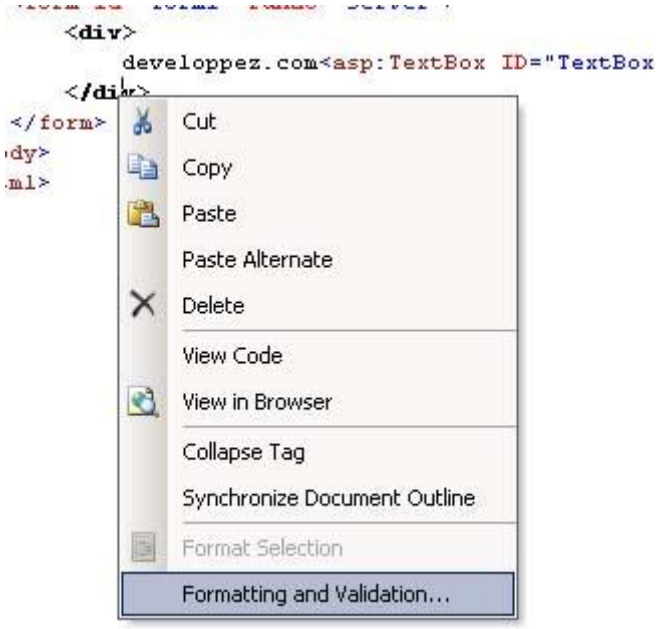
Associé au Serveur web décrit dans le point suivant, cela évite de devoir recréer un répertoire virtuel lorsque l'on change de machine etc.

B. Visual Studio et son éditeur

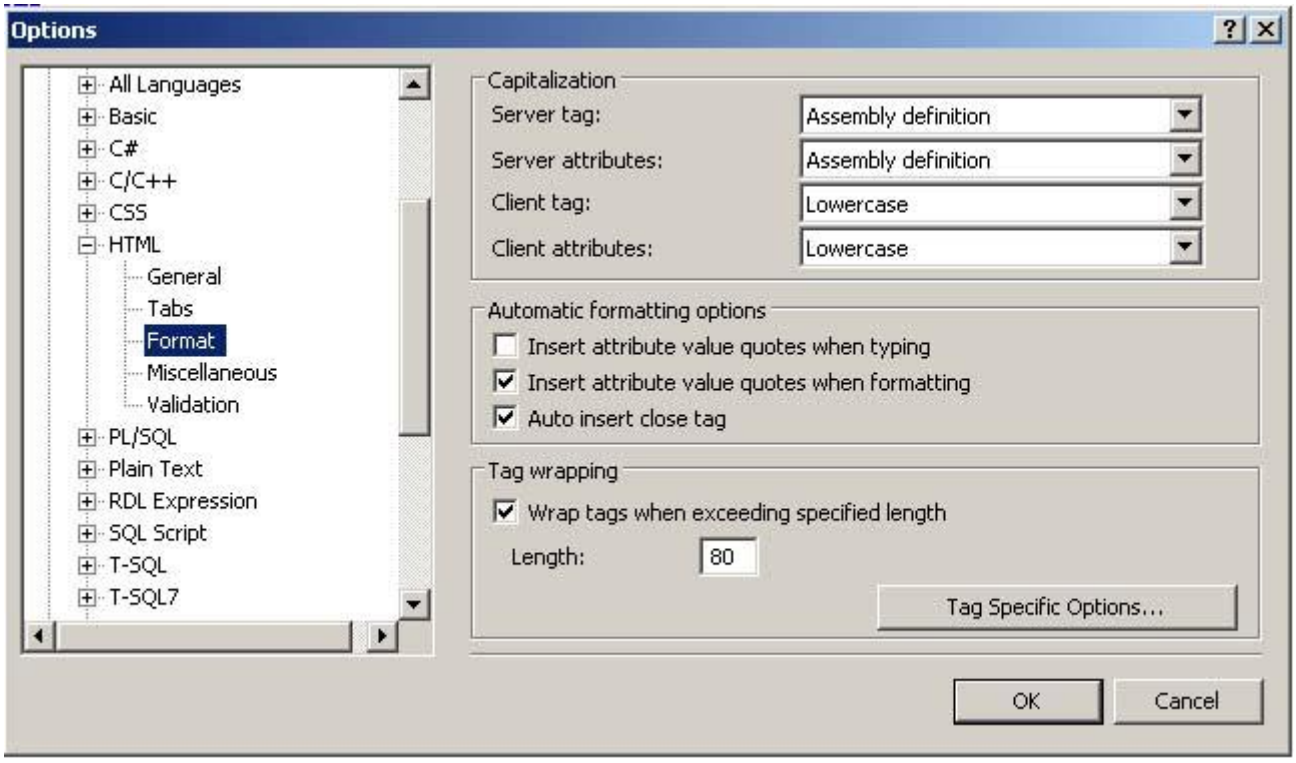
1. Coloration syntaxique des tags de composants web

Qu'entend-t-on par là? Simplement, vous pouvez décider de mettre tous les contrôles relatifs à l'authentification de l'utilisateur, à savoir LoginView, Login, ..., dans une couleur précise, par exemple en bleu, afin qu'ils ressortent mieux au milieu des différents tags qui composent la page web.

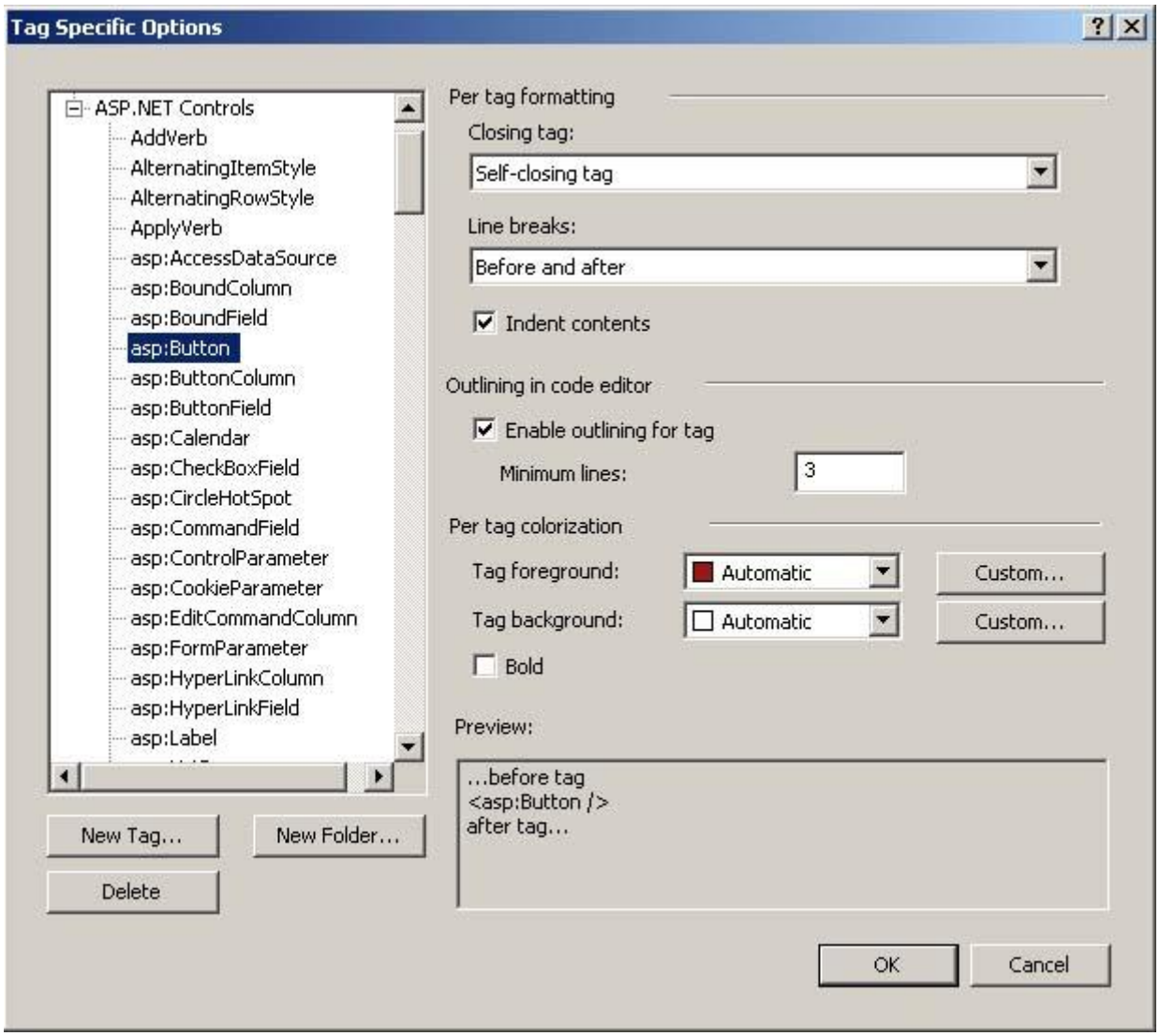
Pour cela, il suffit de définir des propriétés d'affichage du tag. Ceci se fait de la manière suivante:
- Ouverture de "Formatting and validation":



- Ouverture de "Tag options":



- Modification des paramètres d'affichage



2. Formatage des données dans le designer amélioré

Qui n'a pas juré en passant du mode design au mode html en voyant le "foutoir" (c'est le mot qui convient le mieux selon moi) mis par Visual Studio 2002 et 2003. Merci à la personne qui a dit d'arrêter de faire joujou avec ce que nous encodions en html... Effectivement, il est dorénavant possible de passer d'un mode à l'autre sans pour autant perdre la mise en page html.

Voici un exemple. En mode html, ajoutons un div. Cela donne:

```
<body>
    <form id="form1" runat="server">
                <div>
        </div>
    </form>
```

Passons en mode design pour y ajouter du texte

En revenant en mode html, nous voyons que le tag d'ouverture du div a gardé son indentation tandis que le texte a bel et bien été ajouté.

```
<body>
    <form id="form1" runat="server">
        <div>
            developpez.com
        </div>
```

3. Drag and drop des contrôles web directement dans le source HTML

Auparavant il était uniquement possible d'effectuer un drag and drop des composants en mode design. Désormais il est possible d'effectuer cette opération directement en mode html.

```
z.com<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

4. Facilitation du passage du mode design en mode html

Le mode design est idéal pour gagner du temps. Parfois, on ne se rend pas très bien compte où on se trouve dans la page. Ainsi, le fait de sélectionner un élément dans le mode design et de passer en mode html fait que cet élément est toujours sélectionné. Dans notre exemple il s'agit d'un simple texte mais cela pourrait être d'autres composants bien plus complexes.

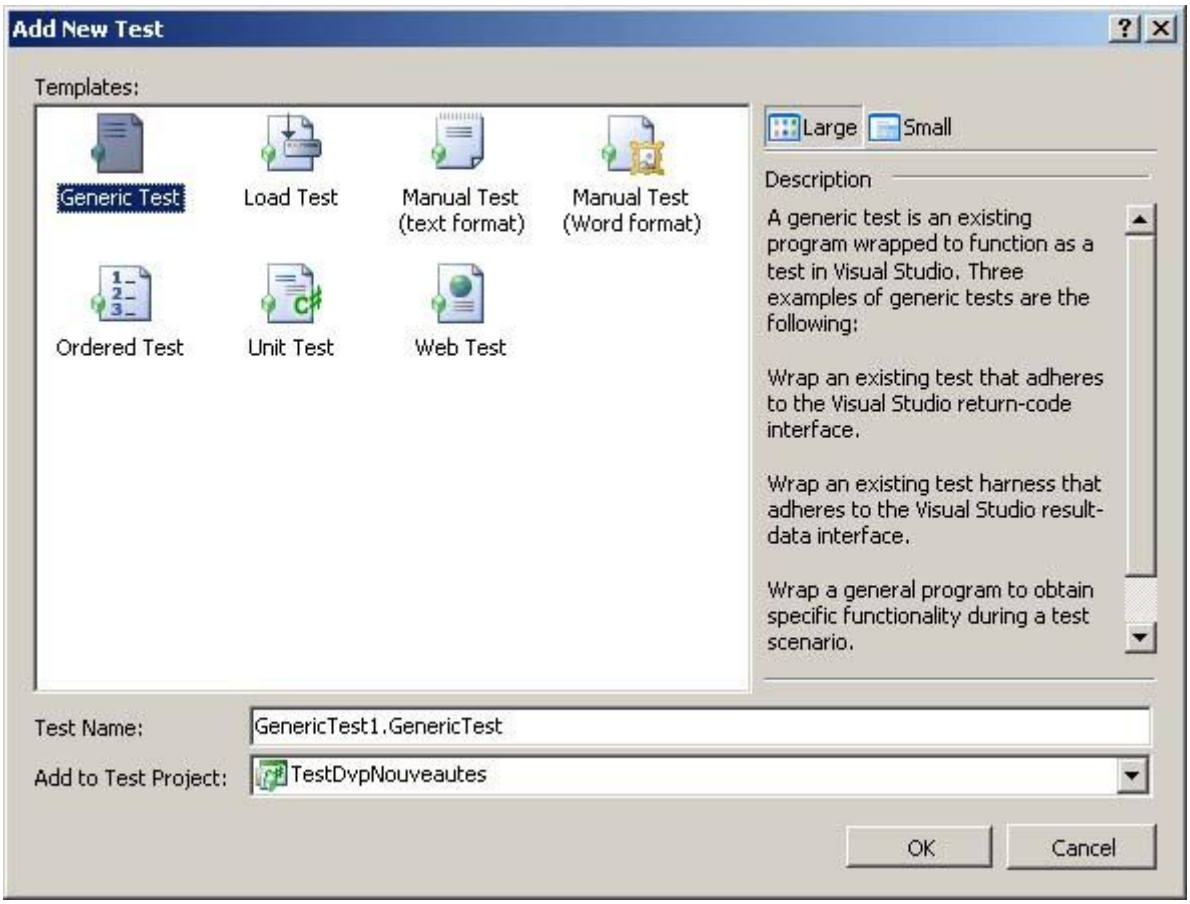
```
        <div>
            developpez.com
</div>
```

developpez.com

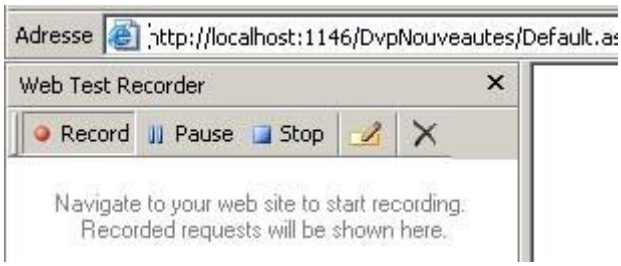
```
        <div>
            developpez.com
</div>
```

C. Test des performances

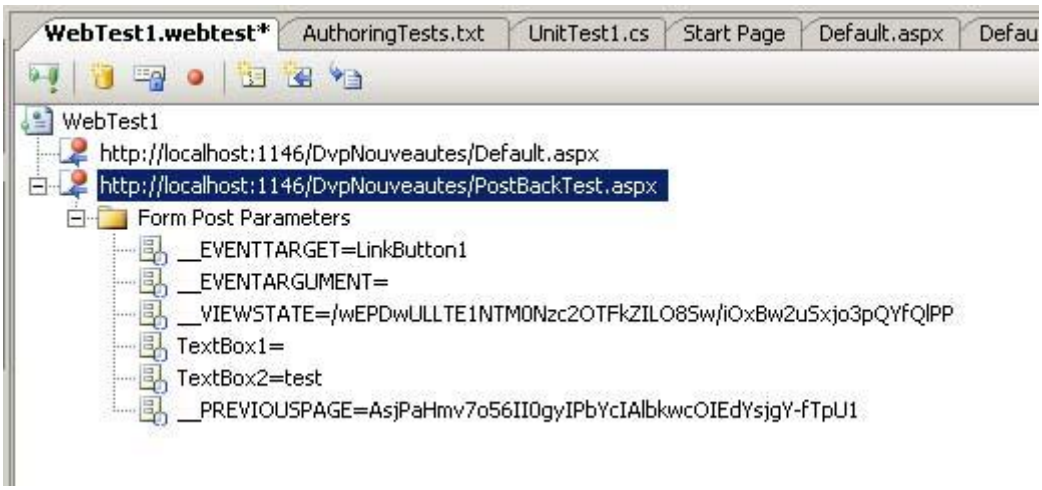
Visual Studio permet de tester les performances des sites web en asp.NET 2.0. Il suffit de spécifier que l'on souhaite faire un test dans le menu "Test > New test..." ...



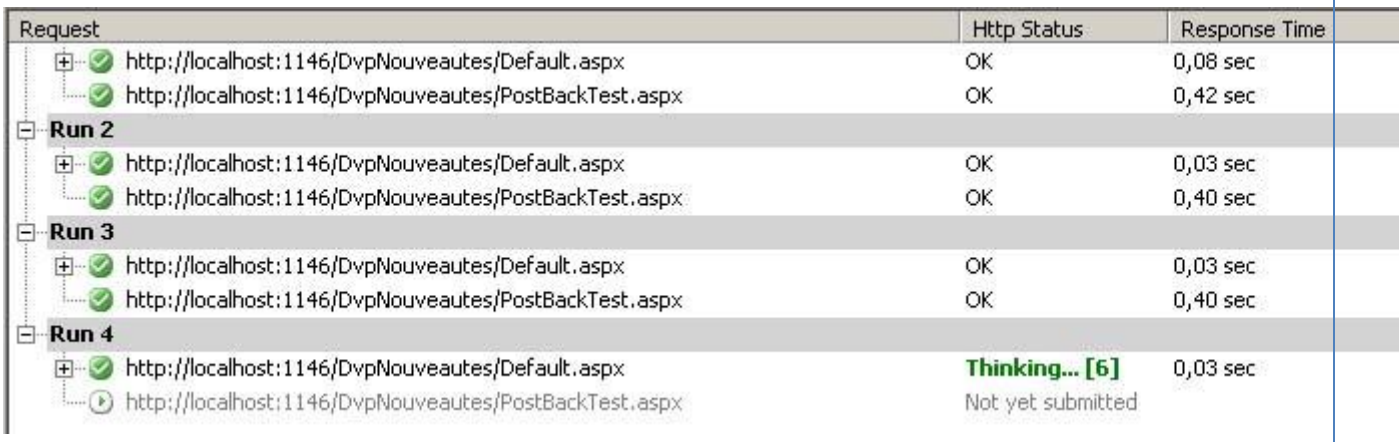
... d'enregistrer les différents tests à effectuer ...



... d'exécuter un certain nombre de fois, dans certaines conditions ...

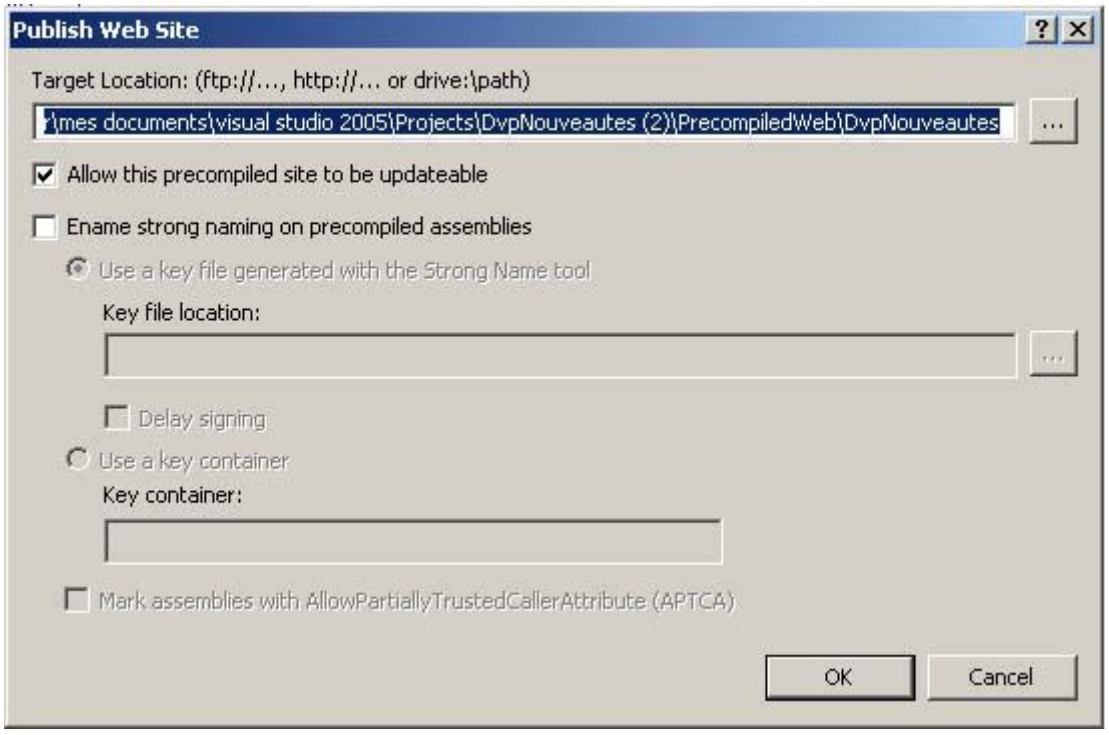


... et d'en lire les résultats.

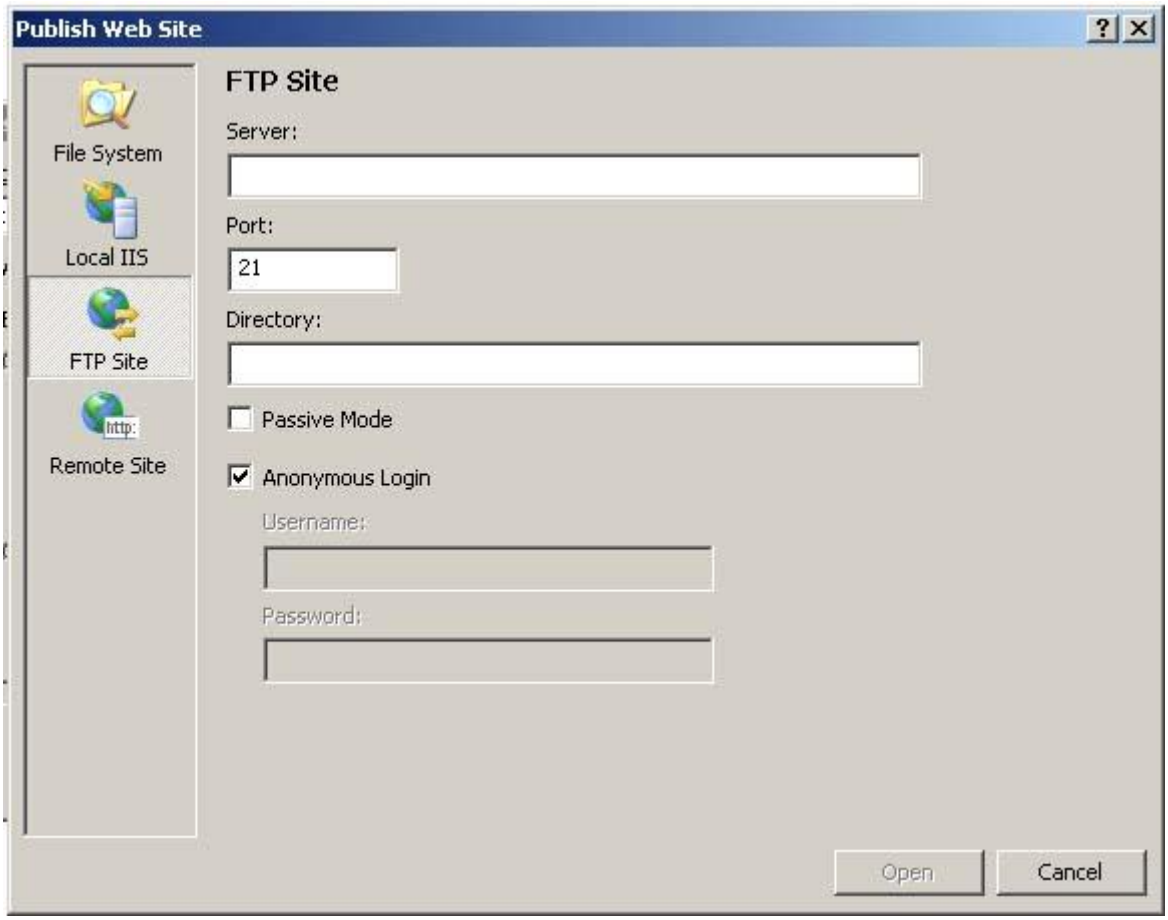


D. Publication de l'application

Visual Studio 2005 permet de publier une application directement par ftp, http ou dans le système de fichiers local.



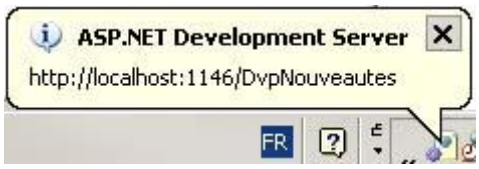
La publication par ftp, par exemple, est des plus classiques: il suffit d'indiquer le nom du serveur ainsi que le login et le mot de passe.



E. WebServer de test

Basé sur Casini, le serveur qui était téléchargeable gratuitement sur Internet et inclus directement avec WebMatrix, ce serveur de test permet d'exécuter les applications pour lesquelles aucun répertoire virtuel n'a été spécifié, ...

De plus, il n'est possible de se connecter au serveur qu'à partir de la machine qui l'héberge. Il n'est donc pas nécessaire de se soucier de problèmes de sécurité dans un premier temps vu que personne ne peut accéder à votre serveur.



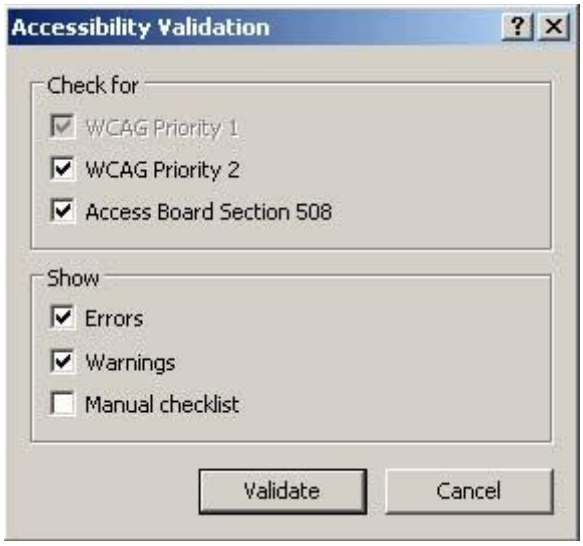
F. Accessibilité de la page web

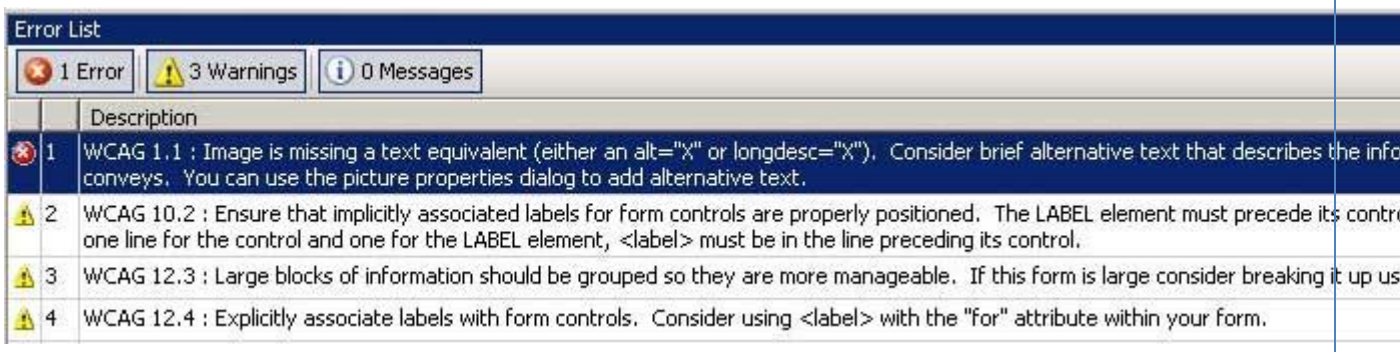
Visual Studio .NET 2005 possède d'autres fonctionnalités non présentes dans la précédente version.

On peut citer, entre autres, la vérification de l'accessibilité des pages web, c'est à dire que tous les attributs qui, selon le standard et donc de la DTD du W3C, soient bien remplis. Un exemple vaut mieux qu'un long discours: une image, dans un navigateur wap ne peut être affiché, c'est alors le texte qui est

censé l'être ou encore un attribut dont les guillemets ne sont pas fermés. Si ce texte n'est pas spécifié, le navigateur se trouvera face à quelque chose qu'il ne peut utiliser / interpréter.

C'est là que VS.NET 2005 intervient.

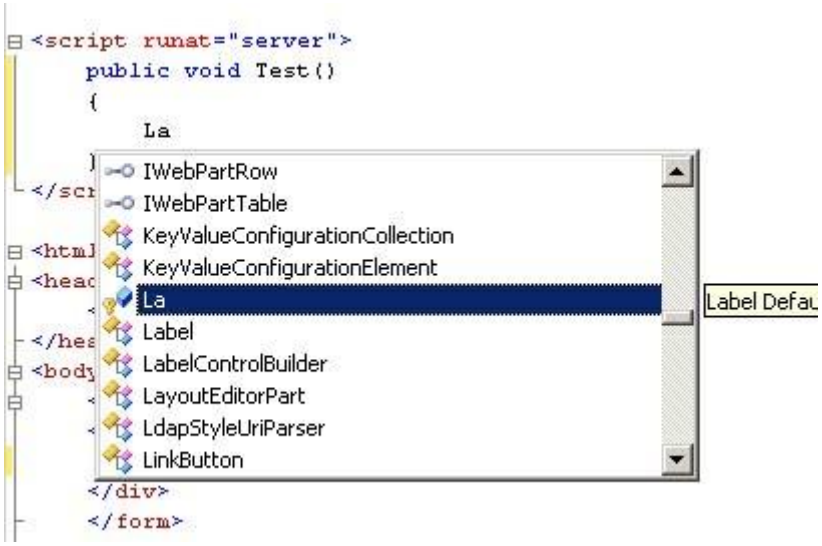




G. Intellisense présent dans même dans un page sans code-behind

Tout est dans le titre...

L'illustration suivante le prouve, il est désormais possible d'utiliser l'intellisense même pour les adeptes du "tout-en-un":



III. asp.NET 2.0 en général

A. asp.NET 2.0 et les standards web

asp.NET permet enfin de faire des applications qui génèrent des pages html valides!

Les adeptes des standards web ne pourront plus lui reprocher ceci. Enfin les designers web et les développeurs vont trouver un terrain d'entente.

B. Sql Caching

Nous avons souvent des pages dont le contenu ne varient que de temps à autre. asp.NET 2.0 amène la mise en cache d'une page et ce, de manière relativement simple. Effectivement, il suffit d'ajouter une directive au niveau de la page.

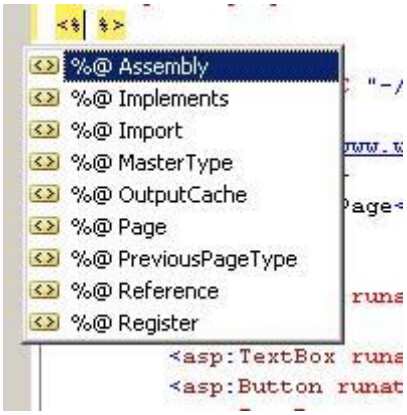
Directive pour mettre en cache

```
<%@ OutputCache Duration=600 VaryByParam=state SqlDependency="database:table" %>
```

Dès lors, pour que le contenu de la page soit chargé de manière dynamique en effectuant un questionnement de la source de données, il faut que la durée de mise en cache soit dépassée ou encore que les données soient modifiées.

On se trouve donc face à une amélioration des performances vu qu'il n'est plus nécessaire d'effectuer beaucoup de retours serveur. De plus, le gain se trouve également en bande passante vu qu'il n'est pas nécessaires de faire transiter des informations entre le serveur web et le serveur de base de données.

Les directives, que nous avions parfois du mal à retenir par coeur, se retrouvent également dans l'IntelliSense de VS.NET 2005 comme on peut le voir sur l'image suivante:



C. Nouveaux composants pour asp.NET 2.0

Il existe toute une série de nouveaux contrôles dans asp.NET 2.0 (annoncé: plus d'une cinquantaine en version finale).

La plupart de ceux-ci ressemblent aux composants que nous passions des journées à développer pour permettre de réaliser des sites attrayants et simples d'utilisation.

On trouve également des composants que l'on retrouvait dans la version précédente d'asp.NET, cependant ceux-ci sont plus complets, mieux configurables et repondent plus rapidement aux exigences des développeurs et designers.

On retrouvera par exemple un composant GridView qui est plus complet que le traditionnel DataGrid. Malheureusement, il y a tellement de choses à dire sur le sujet qu'il est préférable de rester très abstrait sur le sujet.

Vous trouvez très certainement des informations sur ces composants prochainement sur developpez.com.

D. Master pages / Content pages

Souvent présenté comme "la" révolution d'asp.NET, le méchanisme des master pages / content pages permet enfin de réaliser facilement, et de manière intégrée à VS.NET 2005, ce que nous devions manipulés de long moments avant d'avoir le résultat escompté.

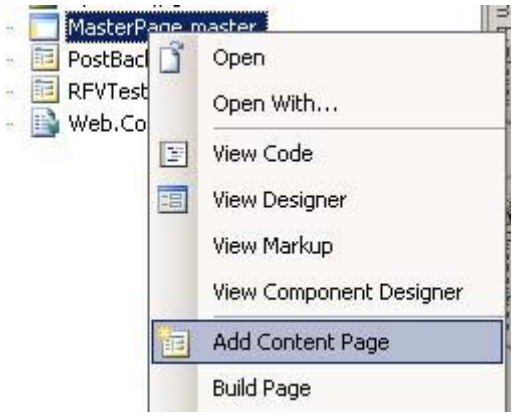
Ce principe permet de créer une structure contenante et de créer le contenu à part afin de l'ajouter. Cette phrase ne vous paraît pas claire? A moi non plus, je vous rassure... Concrètement, vous pouvez créer un template de votre page avec par exemple le logo de votre société, le composant de Login, votre menu, ... et ensuite de créer le contenu d'une seule page. Grâce à cela, il suffit de spécifier que l'on utilise telle ou telle masterpage...

```
<@ Master="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="MasterPage" >

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title Page</title>
</head>
<body>
```

```
    <form id="form1" runat="server">
    <div>
        <table width="100%">
            <tr>
                <td>
                    Ceci est ma masterpage que j'utilise pour l'exemple sur
developpez.com<br />
                    Je="http://www.developpez.com">Lien vers dvp</a>
                </td>
            </tr>
            <tr>
                <td>
                    <asp:contentplaceholder id="ContentPlaceHolder1"
runat="server">
                    </asp:contentplaceholder>
                </td>
            </tr>
            <tr>
                <td>
                    Réalisé par Ditch (www.developpez.com)
                </td>
            </tr>
        </table>
     </div>
    </form>
</body>
</html>
```

ce qui donne visuellement (dans VS.NET 2005):

Il ne reste dès lors qu'à ajouter, dans l'explorateur de projets, une Content Page qui lui est associée:



et d'y ajouter un contenu...

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Default2.aspx.cs" Inherits="Default2"
Title="Untitled Page" %>
<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
Le contenu de la page que j'ajoute
</asp:Content>
```

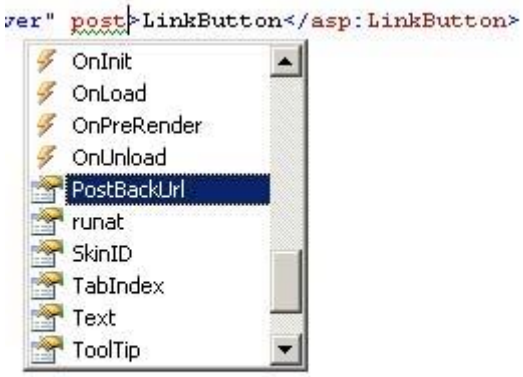
Le passage en mode design permet de voir le résultat:

E. Amélioration de la technique de passage d'informations d'une page à l'autre

Comme je l'explique dans le **tutorial expliquant le passage de variable en asp.NET 1.x**, il était nécessaire d'utiliser le passage d'arguments dans l'url ou les variables de session pour passer des valeurs d'une page à l'autre. Ce n'est plus le cas désormais.

Effectivement, il est possible d'effectuer des postbacks d'une page à l'autre et donc de passer des éléments d'une page à l'autre.

Pour cela, il suffit de définir le "PostBackUrl" dans les propriétés d'un Button ou d'un LinkButton par exemple.



Dans la page cible, pour récupérer la valeur, une série de propriétés supplémentaires existent:

PostBackTest.aspx.cs:Page_Load

```
protected void Page_Load(object sender, EventArgs e)
{
    TextBox tb = (TextBox)PreviousPage.FindControl("TextBox2");
    Response.Write(tb.Text);
}
```

F. Amélioration des Validators

Auparavant, tous les composants devaient être validés pour autoriser le retour serveur. Il est maintenant possible de spécifier des groupes de validation.

Ainsi, il suffit que les composants de ce groupe soient validés pour autorisé le retour serveur. Ceci est très utile par exemple si vous avez plusieurs actions possibles dans une même page, par exemple un bouton pour créer un login et un autre pour s'authentifier. Pour créer le login, il faut un nom, c'est tout, alors que pour l'authentification, il faut un nom et un mot de passe.

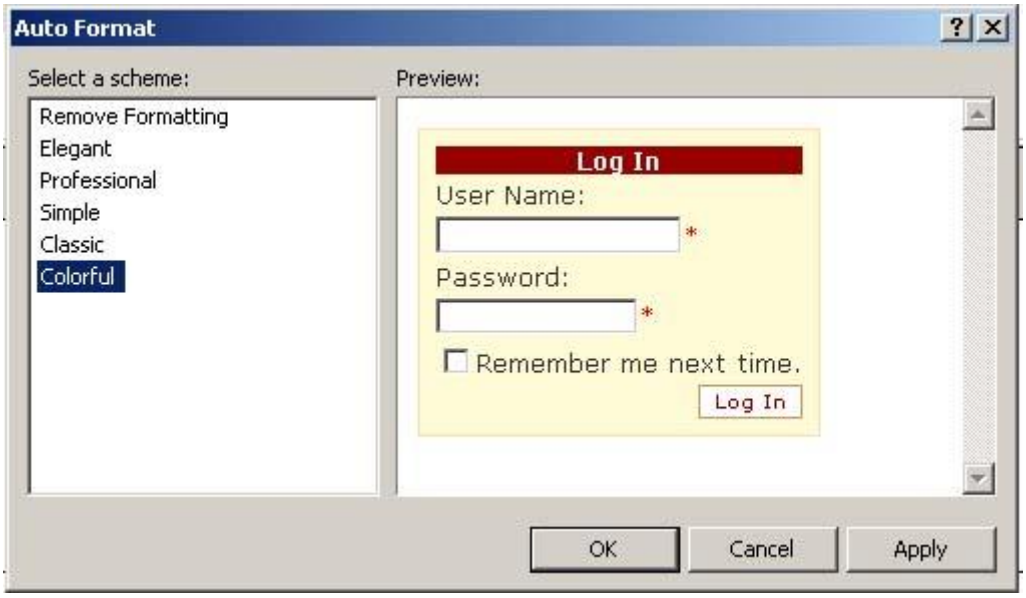
On peut ainsi définir un groupe "Create" et l'autre "Login". Ceci se fait très facilement: - on place un Validator (dans notre cas, un RequiredFieldValidator) en lui définissant un groupe

G. Utilisateurs et rôles: gestion sécurisée des accès

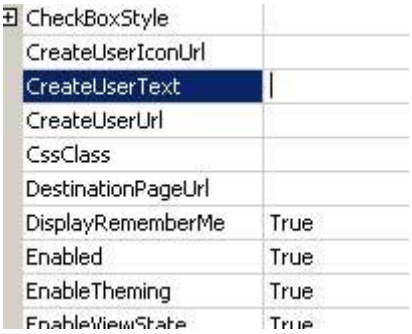
La plupart des sites nécessitent une authentification pour certaines parties dans celui-ci. Cela a été compris et implémenté dans la nouvelle version d'asp.NET.

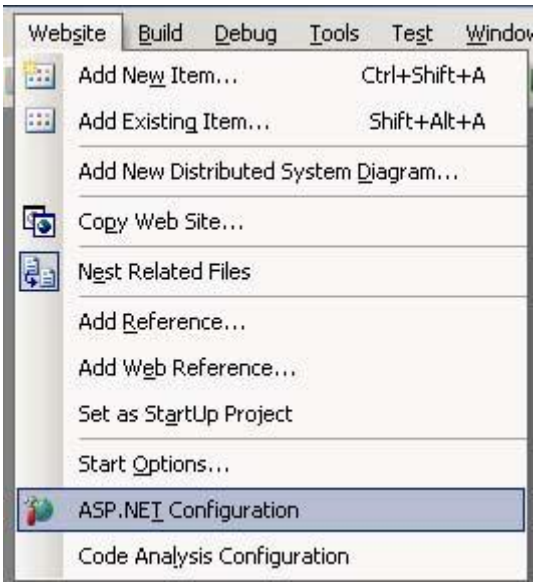
Il est ainsi très simple, à l'aide d'un assistant, de définir les informations que l'on souhaite connaître pour un utilisateur. Ces informations vont alors se trouver dans une base de données dont l'accès, la définition des tables, ... est entièrement géré par asp.NET.

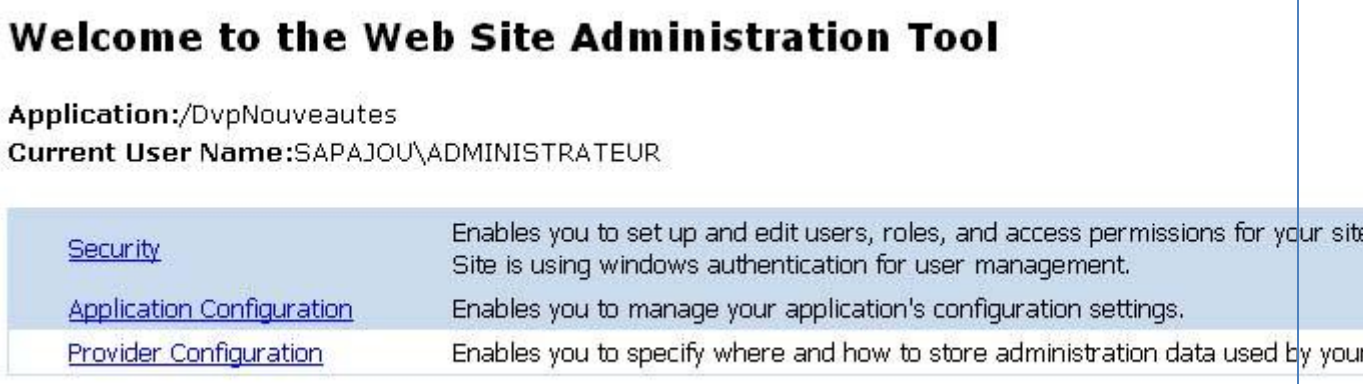
De plus, asp.NET 2.0 à amené de nouveaux composants tels que le CreateUserWizard et le ChangePassword qui s'occupent d'effectuer les opérations classiques de gestion des utilisateurs, ce qui permet de gagner du temps lors du développement et ainsi se concentrer sur les choses plus "importantes".



Pour ajouter un lien vers une page d'enregistrement, il suffit de définir certaines propriétés du composant Login. Ceci montre bien que beaucoup de nouveaux éléments sont présents afin de faciliter la mise en place de fonctionnalités classiques.



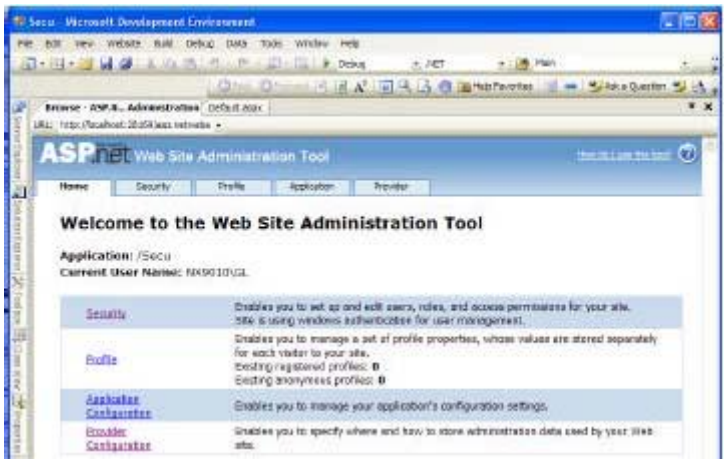


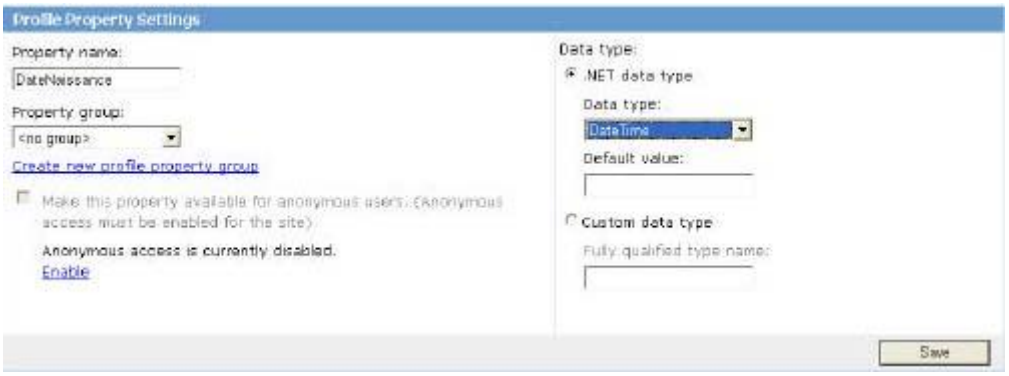


Tous ces composants se connectent à une base de données quelconque qui est entièrement géré par .NET 2.0. Les tables sont créées sans qu'aucune interaction de l'utilisateur soit nécessaire.

Suivant les applications, les informations sur les utilisateurs dont on a besoin diffèrent, les concepteurs l'ont compris et ceci est implémenté dans la version 2 d'asp.NET. La mise en place de propriétés supplémentaires pour un utilisateur se fait par l'intermédiaire d'une page de configuration, très simple à comprendre et à manipuler.

Par ailleurs, avec ces pages de configuration, nous trouvons des classes permettant de gérer très simplement les utilisateurs si l'on souhaite effectuer d' autres opérations que celles déjà implémentées





H. Internationalisation d'une application

Ceci est beaucoup plus simple que ce qui se trouvait en asp.NET 1.1, technique qui est expliquée ici. Il suffit d'ajouter

```
Culture="auto" UICulture="auto"
```

dans les directives de la page, ce qui donne:

```
<%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default.aspx.cs"
Inherits="_Default" Culture="auto" UICulture="auto" %>
```

Pour ajouter des fichiers resources, dans le menu Tools > Add Locale Resource... et le tour est joué. Rien de plus!

I. Thèmes / Skins

Des applications qui s'adaptent aux exigences des utilisateurs au niveau design, voilà le quotidien de plus en plus de développeurs asp.NET. Dans la version 2, asp.NET possède toute une série de propriétés permettant de gérer les thèmes.

Pour définir un thème, il suffit de spécifier:

```
<%@ Page Language="C#" ... Theme="TestSkin"%>
```

Ici, le thème a été appliqué au niveau de la page mais il est possible de le définir au niveau de l'application dans le web.config.

```
<configuration>
   <system.web>
        <pages theme="WeFly247" />
   </system.web>
</configuration>
```

ou encore par code

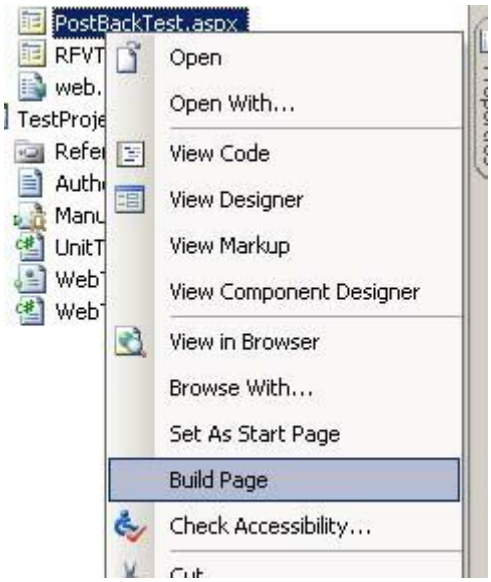
```
Page.Theme = "NomduTheme"(;)
```

De plus, de nouveau par la programmation, il est possible de spécifier un thème spécifique pour un utilisateur. Ce thème est généralement choisi dans ses options et rechargé lorsqu'il se loggue.

Les thèmes se définissent dans des fichiers .skin, fichiers texte dans lesquels ont défini les différentes présentations des éléments, qu'ils soient par défaut ou adapté à un skin.

J. Compilation d'une seule page

Le nouveau framework permet désormais de ne recompiler qu'une seule page comme on peut le voir sur l'illustration suivante:



Ceci permet donc d'effectuer des tests en situation réelle (il n'est pas toujours possible d'utiliser un serveur de test dans certains cas) et surtout de corriger des bugs sans pour autant redémarrer l'application et ainsi de supprimer les variables de sessions des différents utilisateurs. Ceci permet donc la mise à jour du site web même lors de grande affluence.