

Workshop SharePoint 2007 Développement de WebPart

Edgar Maucourant (<http://blog.nftinside.com>)



Sommaire

Introduction -----	p3
Première partie : Création d'une WebPart simple -----	p4
Etape 1 : Création d'un nouveau projet WebPart-----	p4
Etape 2 : Création des contrôles-----	p7
Etape 3 : Afficher le tout !-----	p8
Etape 4 : Déployer la WebPart-----	p9
Etape 5 : Activer la Feature-----	p10
Etape 6 : Ajouter la WebPart à la page d'accueil-----	p11
Deuxième partie : Afficher vos données avec SPGridView -----	p13
Etape 7 : Ajout d'un DataSet typé-----	p13
Etape 8 : Instanciation et peuplement du DataSet -----	p14
Etape 9 : Ajout du control SPGridView-----	p15
Etape 10 : Création des colonnes et ajout du SPGridView aux Controls-----	p16
Etape 11 : Afficher le SPGridView»-----	p16
Etape 12 : Déploiement-----	p17
Troisième partie : Ajoutons de la pagination ! -----	p18
Etape 13 : Ajout de la pagination -----	p18
Etape 14 : Ajout du gestionnaire d'évènement-----	p18
Conclusion -----	p19

Introduction

Bienvenu(e) dans ce Workshop d'une heure consacré à SharePoint 2007, et plus spécifiquement au développement d'une WebPart qui utilisera le contrôle SPGridView pour afficher vos données sur la page d'accueil

Nombre d'entre vous ont sûrement déjà entendu parler de SharePoint mais il est bon de rappeler quelques points essentiels de ce merveilleux produit. SharePoint 2007 se distingue en 3 produits :

WSS (Windows SharePoint Services) la version gratuite contenant toutes les briques élémentaires de collaboration (liste, bibliothèque, flux de travail, Webpart, etc...).

MOSS Standard (Microsoft Office SharePoint Server) incluant un meilleur moteur de recherche, la gestion du contenu (ECM), des Workflow amélioré, etc...

Et enfin, MOSS Entreprise ajoutant la partie Excel Services, la Business Intelligence et le Business Data Catalog (un sorte de moulinette super évoluée☺)

Avec ces trois produits Microsoft est en passe de s'imposer sur le marché des logiciels de gestion de contenu Internet et Intranet, notamment grâce à l'utilisation de technologies maintes fois éprouvées et bien connues des informaticiens (Active Directory, Windows Server, SQL Server et .Net pour le développement).

Important : Préambule aux manipulations

SharePoint repose sur le même moteur que tous les sites ASP.NET. Dans tous ces sites les assemblées constituant les pages sont compilés au moment de leur utilisation ce qui peut ralentir fortement la génération des pages au premier visionnage (dans le cas de SharePoint cela peut aller jusqu'à plus de 30 secondes...).

Il est donc important que vous accédiez dès maintenant à la page d'accueil du site afin de laisser SharePoint compiler la page tranquillement pendant que vous commencerez ce Workshop.

Pour accéder au site, double-cliquez sur le lien présent sur le bureau

Important : Mot de passe

La session doit être ouverte avec le compte **Administrateur sans mot de passe**.

Première partie : Création d'une WebPart simple

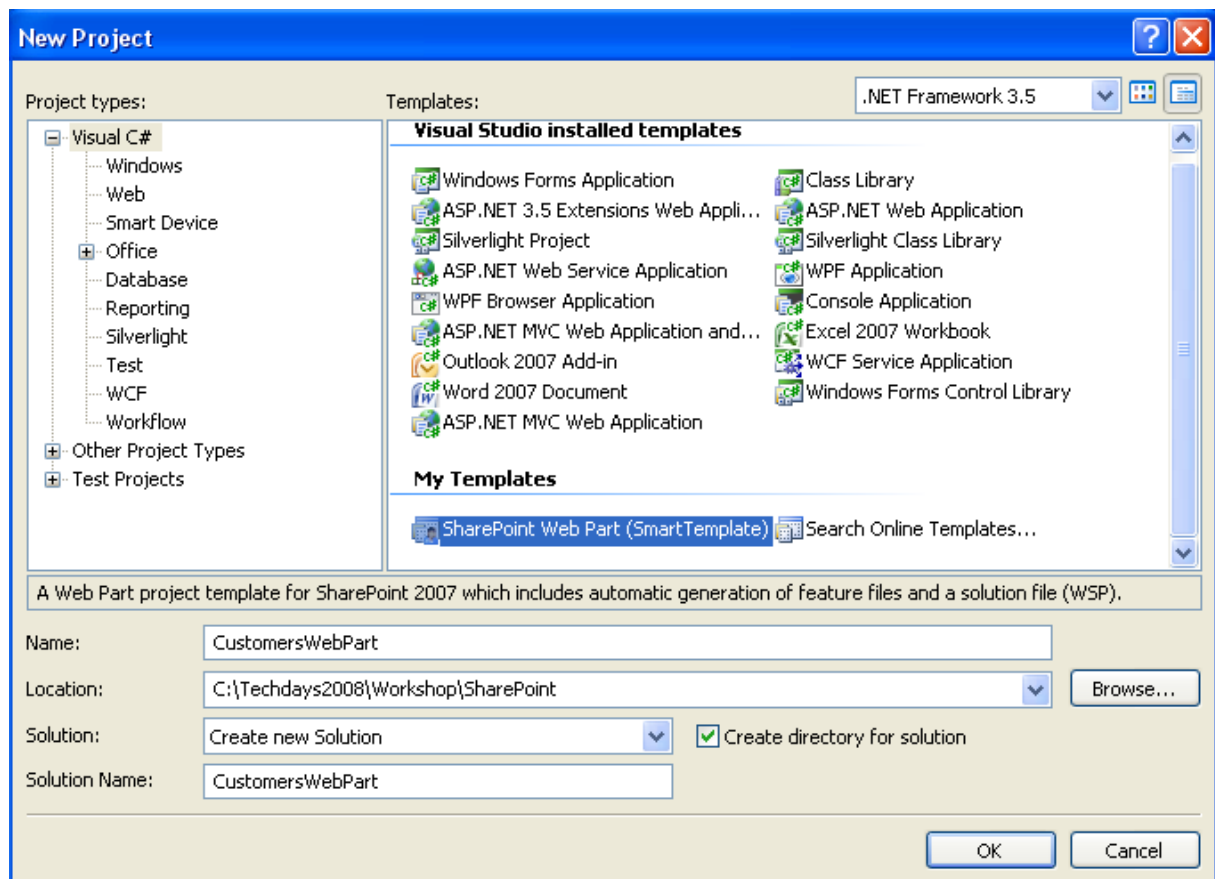
Lors cette première partie vous verrez comment créer et déployer une WebPart simple permettant de saisir un titre dynamiquement pour cette WebPart. Ceci vous permettra de vous familiariser avec les concepts de base du développement de WebParts.

Note : De base Visual Studio 2008 ne définit pas de type de projet « Développement de WebPart SharePoint ». Nous pourrions très bien utiliser un projet de type « bibliothèque de classes » mais ce serait long et fastidieux. Nous nous appuyerons au contraire sur un projet Open Source disponible sur CodePlex et appelé : SmartTemplates. Cette application ajoute un nouveau type de projet à Visual Studio : «SharePoint WebPart (SmartTemplate) »

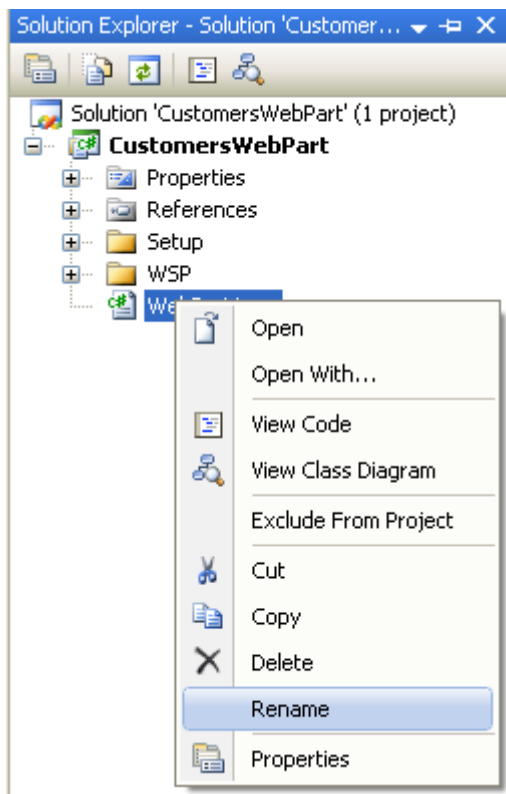
Etape 1 : Création d'un nouveau projet WebPart

Ouvrez Visual Studio 2008 et créez un nouveau projet, dans la fenêtre de sélection de projet cliquez que Visual C# dans le menu de gauche, puis dans la fenêtre de droite en bas sélectionnez « SharePoint WebPart (SmartTemplates) »

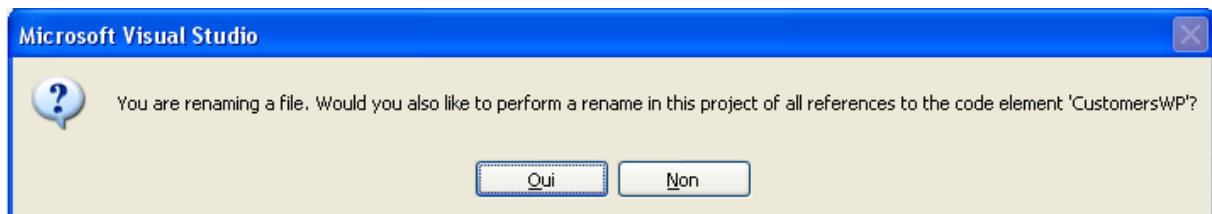
Nommez votre projet '**CustomersWebPart**' et placez le dans le répertoire '**C:\Techdays2008\Workshop\SharePoint**' comme dans l'image ci-dessous



La première des étapes à réaliser maintenant est de renommer votre fichier **'WebPart1'** en **'CustomersWP'**



Dans la fenêtre qui s'ouvre demandant de renommer toutes les occurrences de **'WebPart1'** en **'CustomersWP'** cliquez sur **'Oui'**



Note : Le déploiement de WebParts dans SharePoint peut se faire de différentes manières, l'une de celle-ci consiste à utiliser un fichier de **'Feature'** permettant un déploiement propre de nouvelles fonctionnalités dans SharePoint. Ce fichier décrit sous forme XML les informations permettant de déployer notre fonctionnalité (**'Feature'** en anglais). Dans le cas présent la fonctionnalité est une WebPart.

L'explication de l'intérêt des Features ne serait être couvert dans un Workshop si court, l'auteur invite donc les utilisateurs intéressés à rechercher sur Internet de plus amples informations.

Le déploiement de Features est bien souvent couplé à l'utilisation d'une Solution. Une solution consiste en un fichier **'wsp'** (au format CAB) empaquetant l'intégralité des fichiers et ressources nécessaires au déploiement de la Feature. L'utilisation d'une solution permet de déployer (encore une fois) proprement et efficacement vos fichiers sur l'ensemble des serveurs de votre ferme.

Le déploiement de WebPart dans ce type de projet étant basé sur l'utilisation de fichiers «Features» et de Solutions (cf. note page précédent) nous allons devoir renommer aussi les occurrences de 'WebPart1' dans ces fichiers.

<p>Renommez tout d'abord le nom du fichier 'CustomersWebPart.WebPart1.webpart' définissant les propriétés de notre WebPart en 'CustomersWebPart.CustomersWP.Webpart'. Ce fichier se trouve dans l'arborescence présentée ci-contre.</p> <p>Note : Le but ici dans les prochaines étapes est de remplacer dans tous les fichiers, les occurrences de 'WebPart1' en 'CustomersWP'</p>	
<p>Ouvrez ensuite ce fichier et remplacez les occurrences de 'WebPart' par 'CustomersWP' Changez aussi le titre de votre WebPart en 'Mes clients' et sa description en « Cette Webpart affiche mes clients » comme présenté ci-contre (il manque un s à CustomersWP...)</p>	<pre><?xml version="1.0" encoding="utf-8"?> <webParts> <webPart xmlns="http://schemas.microsoft.com/WebPart/v3"> <metadata> <type name="CustomersWebPart.CustomerWP, CustomersWebPart" /> <importErrorMessage>Cannot import this web part.</importErrorMessage> </metadata> <data> <properties> <property name="Title" type="string">Mes clients</property> <property name="Description" type="string">Cette Webpart affiche mes clients</property> </properties> </data> </webPart> </webParts></pre>
<p>Enfin ouvrez le fichier 'webpartsmanifest.xml' situé juste en dessous du fichiers précédent et remplacez les occurrences de 'WebPart1'</p>	<pre><?xml version="1.0" encoding="utf-8" ?> <Elements xmlns="http://schemas.microsoft.com/sharepoint"> <Module Name="WebParts" List="13" Url="catalogs/wp"> <File Url="CustomersWebPart.CustomerWP.webpart" Type="WebPart" /> <Property Name="Group" Value="CustomersWebPart"></Property> </Module> </Elements></pre>

Nous en avons terminé avec les mises à jour de fichiers. Notre solution Visual Studio est à présent prête pour le déploiement de la WebPart

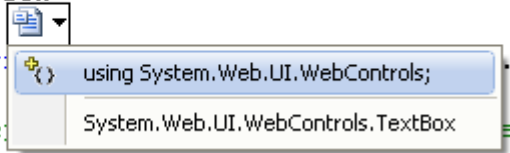
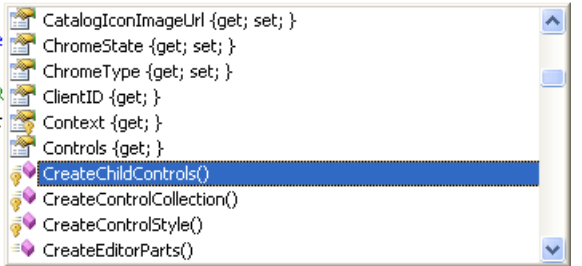

Etape 2 : Création des contrôles

Vous pouvez dès à présent ouvrir le fichier **'CustomerWP.cs'** qui contient notre classe WebPart. Supprimez les commentaires contenus dans le fichier afin d'y voir plus clair.

La première chose que vous remarquerez sûrement est que notre classe **'CustomerWP'** dérive de la classe **'Webpart'**. Dans SharePoint 2007 les WebParts sont des objets basés sur la même classe que les WebPart ASP.NET 2.0. Ainsi les développeurs ASP.NET connaissant le développement de WebParts or SharePoint n'auront que très peu de modifications à prendre en compte.

Nous allons à présent ajouter un contrôle de type **'TextBox'** et un de type **'Button'** à notre classe. Le premier nous servira à saisir notre texte, le second à modifier le titre de la WebPart avec ce texte. Ces contrôles devront être ajoutés à la collection **'Controls'** de la WebPart.

Le meilleur bloc de code pour initialiser ces contrôles et les ajouter à la collection est la méthode **'createChildControl'** qui est héritée de la class WebPart. Cette méthode est appelée à chaque affichage de la WebPart.

<p>Déclarez un contrôle de type 'TextBox' appelé 'customerName' (utilisez le refactoring de Visual Studio pour importer le namespace nécessaire)</p>	<pre>public class CustomerWP : System.Web.UI.WebControl { protected TextBox protected override { //TODO: Re writer.Write("I'm alive!!"); } }</pre> 
<p>Déclarez un deuxième contrôle de type 'Button' nommé 'btn'</p> <p>Redéfinissez la méthode 'createChildControl' en tapant 'protected override' puis en choisissant la méthode dans la fenêtre intellisense</p>	<pre>protected TextBox customerName; protected Button btn; public override protected ove { //TODO: R writer.Wr } }</pre> 
<p>Dans cette méthode, initialisez le champ TextBox et le Bouton. Placez le text 'OK' sur le bouton et ajoutez lui un gestionnaire d'évènement pour 'Click' (utilisez le raccourci 'Tab'+ 'Tab' voir ci-contre)</p>	<pre>btn = new Button(); btn.Text = "OK"; btn.Click += new EventHandler(btn_Click); (Press TAB to insert)</pre> 

<p>A présent, ajoutez les deux contrôles à la collection 'Controls' de la WebPart</p> <p>Votre code devrait ressembler à celui-ci-</p>	<pre>public class CustomerWP : System.Web.UI.WebControls.WebParts.WebPart { protected TextBox customerName; protected Button btn; protected override void CreateChildControls() { customerName = new TextBox(); this.Controls.Add(customerName); btn = new Button(); btn.Text = "OK"; btn.Click += new EventHandler(btn_Click); this.Controls.Add(btn); } void btn_Click(object sender, EventArgs e) { throw new NotImplementedException(); } }</pre>
<p>La dernière action à réaliser et de coder la logique du bouton. Ajoutez le code ci-contre dans le gestionnaire d'évènement 'Click' du bouton</p>	<pre>void btn_Click(object sender, EventArgs e) { this.Title = this.customerName.Text; }</pre>

Etape 3 : Afficher le tout !

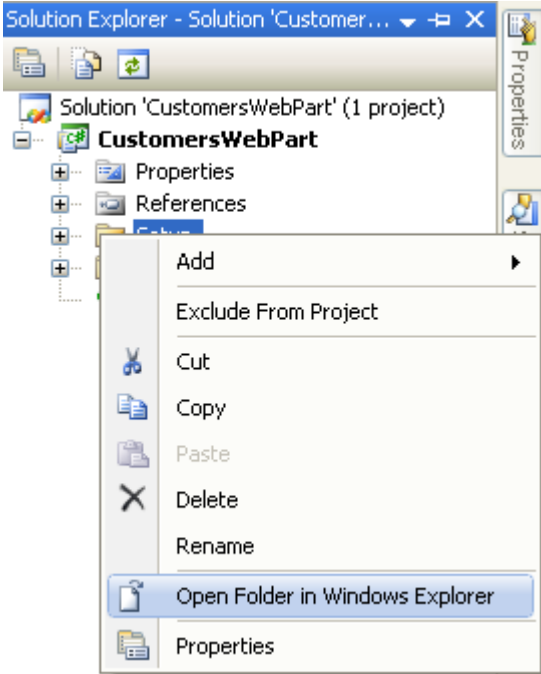

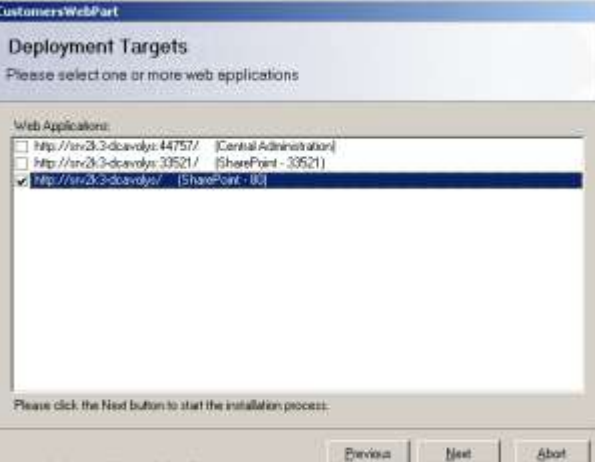
Nous venons de définir nos deux contrôles et de les ajouter à la collection de contrôles de la WebPart, mais ce n'est pas tout. A présent nous devons définir ce qui sera affiché à l'écran, c'est à dire quel sera le rendu de notre WebPart. Pour cela nous allons utiliser la méthode **'RenderContents'** qui est appelée au moment d'afficher la WebPart. Cette méthode est héritée, mais est déjà redéfinie par défaut dans notre type de projet.

<p>Supprimez le code contenu dans la méthode 'RenderContents' et remplacez le par celui-ci-contre. Ce code utilise l'objet 'writer' type 'HtmlTextWriter' passé en argument pour écrire du code HTML de mise en forme. Il appelle aussi la méthode 'RenderControl' sur chaque contrôle pour générer le code HTML du contrôle.</p>	<pre>protected override void RenderContents(System.I { writer.Write("<table border=\\"0\\">"); writer.Write("<tr>"); writer.Write("<td>"); customerName.RenderControl(writer); writer.Write("</td><td>"); btn.RenderControl(writer); writer.Write("</td></tr></table>"); }</pre>
---	--

Notre WebPart est à présent terminé nous allons maintenant voir comment nous pouvons la déployer sur notre serveur de développement

Etape 4 : Déployer la WebPart

Pour déployer la WebPart, nous allons utiliser une Feature et une solution, mais fort heureusement pour nous le projet SmartTemplate va nous faciliter la tâche en créant pour nous le fichier de solution et en nous permettant de déployer notre WebPart à l'aide d'une application WinForm !

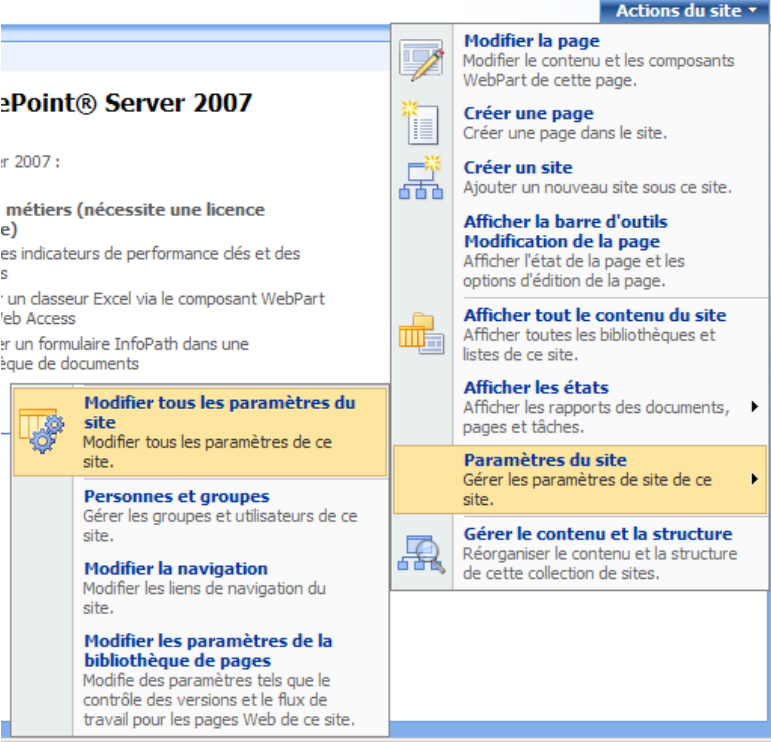

<p>Première action, générer notre solution. Pour cela rien de plus simple : appuyez sur 'F6' ☺</p> <p>Ensuite une fois la solution générée, ouvrez le répertoire 'Setup' en faisant un clic droit sur le répertoire 'Setup' dans le 'solution explorer' puis en cliquant sur 'Open Folder in Windows Explorer'</p> <p>Ce dossier contient plusieurs fichiers dont un s'appelle 'setup.exe' et représente l'application Winform que nous allons utiliser</p>	
<p>Une fois lancée cette application va d'abord vérifier que toutes les conditions nécessaires au bon déroulement du déploiement soient réunies. Si c'est le cas vous pourrez passer à l'étape suivante. Sinon vous devrez corriger le problème avant de pouvoir poursuivre.</p>	
<p>Une fois les conditions d'utilisation acceptées, l'application va vous demander sur quelle(s) application(s) Web elle doit déployer la solution. La première correspond à l'administration centrale, la seconde au fournisseur de services partagés. La troisième correspond au site intranet de la société. Décochez les deux premières lignes et gardez la dernière '(SharePoint – 80)'</p>	

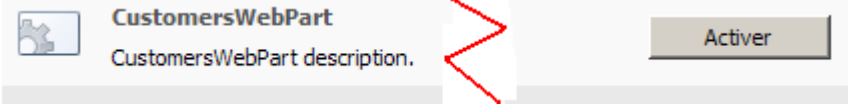
L'application va alors procéder au déploiement de la solution sur la Ferme de serveur (ici nous n'en avons qu'un). Une fois la procédure terminée vous pouvez fermer la fenêtre.

Bien que la solution soit déployée, notre Feature n'est pas activée. En effet pour permettre une plus grande flexibilité et une plus grande sécurité, SharePoint donne la possibilité aux administrateurs SharePoint de pouvoir activer ou désactiver les Features. Ces Features peuvent être déployées au niveau de la ferme, d'une application Web, d'une collection de sites ou d'un site en particulier. Elles pourront alors être activées aux besoins par l'administrateur de l'élément.

Etape 5 : Activer la Feature

Dans notre cas la Feature a été déployée au niveau de la collection de site. Pour l'activer nous allons devoir nous rendre sur la galerie de fonctionnalités pour la collection de site.

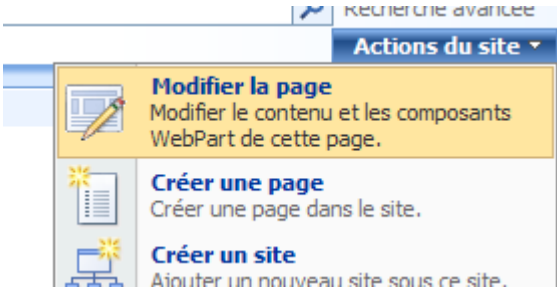
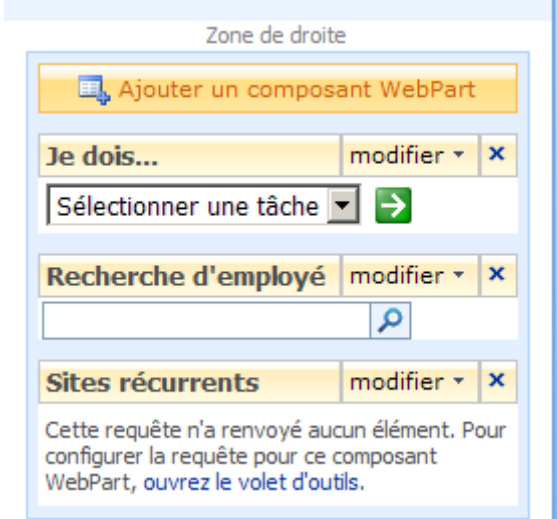
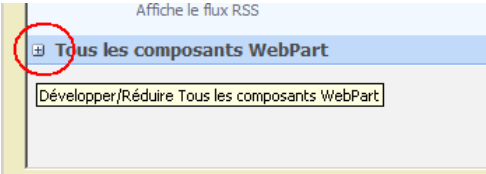
<p>Rendez vous sur la page d'accueil du site intranet 'http://srv2K3-dcavolys'. Sur la page d'accueil cliquez sur 'Action du site' en haut à droite de la fenêtre. Puis dans le menu dynamique sélectionnez le sous-menu 'Paramètres du site', et enfin cliquez sur 'Modifier tous les paramètres du site'</p>	 <p>The screenshot shows the 'Actions du site' menu in SharePoint 2007. The menu items are: 'Modifier la page', 'Créer une page', 'Créer un site', 'Afficher la barre d'outils Modification de la page', 'Afficher tout le contenu du site', 'Afficher les états', 'Paramètres du site' (highlighted), and 'Gérer le contenu et la structure'. The 'Paramètres du site' option is highlighted in yellow.</p>
<p>Dans la page de paramètres du site dans la colonne de droite clique sur 'Fonctionnalités de la collection de sites'</p>	 <p>The screenshot shows the 'Administration de la collection de sites' page. The page title is 'Administration de la collection de sites'. Below the title is a list of features with expandable icons:</p> <ul style="list-style-type: none"> ▣ Paramètres de recherche ▣ Zones de recherche ▣ Mots clés de recherche ▣ Corbeille ▣ Paramètres de l'annuaire de sites ▣ Rapports sur l'utilisation de la collection de sites ▣ <u>Fonctionnalités de la collection de sites</u> ▣ Hiérarchie des sites ▣ Connexion au site portail

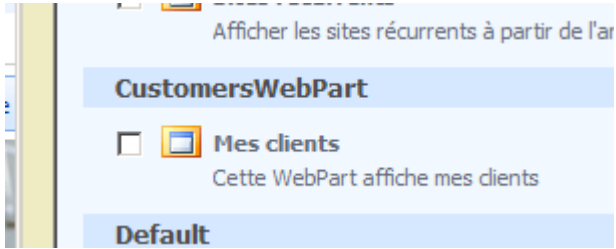
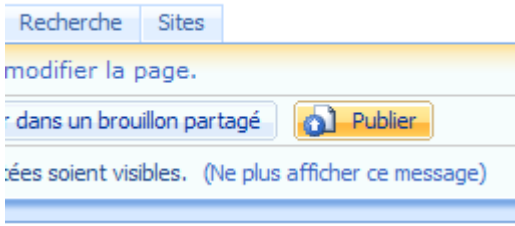
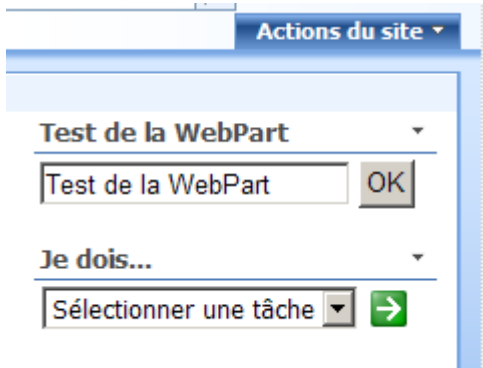
<p>Dans la page des Features trouvez la fonctionnalité 'CustomersWebPart' (en principe en haut de la liste) et cliquez sur le bouton 'Activer'</p>	
--	--

Une fois la Feature activée vous pouvez revenir sur la page d'accueil. Il reste à présent une dernière étape pour voir le fruit de votre travail. Nous devons ajouter la WebPart sur la page, et pour cela nous allons devoir passer en mode édition.

Etape 6 : Ajouter la WebPart à la page d'accueil

Pour pouvoir ajouter une WebPart sur une page nous devons passer en mode **'édition'** afin de faire apparaître les menus qui nous permettrons d'ajouter notre Webpart

<p>En utilisant le menu 'Actions du site' cliquez sur 'Modifier la page'</p>	
<p>Une fois dans l'édition de la page dans la zone de droite vous devriez obtenir un bouton orangé intitulé 'Ajouter un composant WebPart'. En cliquant dessus vous allez ouvrir une nouvelle fenêtre vous permettant de choisir la WebPart que vous souhaitez ajouter.</p>	
<p>Dans la nouvelle fenêtre cliquez sur le '+' à côté du titre 'Tous les composants WebPart' vous ferez ainsi apparaître l'ensemble des WebParts disponibles</p>	

<p>Dans la nouvelle liste choisissez la WebPart 'Mes Clients' et cliquez sur 'Ajouter'</p>	
<p>Après quelques secondes votre WebPart sera ajoutée à la zone de droite. Avant de pouvoir la tester nous devons quitter le mode 'Edition'. Pour cela cliquez sur le bouton 'Publish Page' au milieu la page</p>	
<p>Une fois la page publiée vous pouvez tester votre bouton. Le résultat devrait être équivalent à celui-ci-contre.</p>	

Nous avons à présent terminé cette partie. Dans la prochaine vous verrez comment afficher des données issues d'un DataSet Typé grâce à l'utilisation d'un contrôle SPGridView.

Deuxième partie : Afficher vos données avec SPGridView

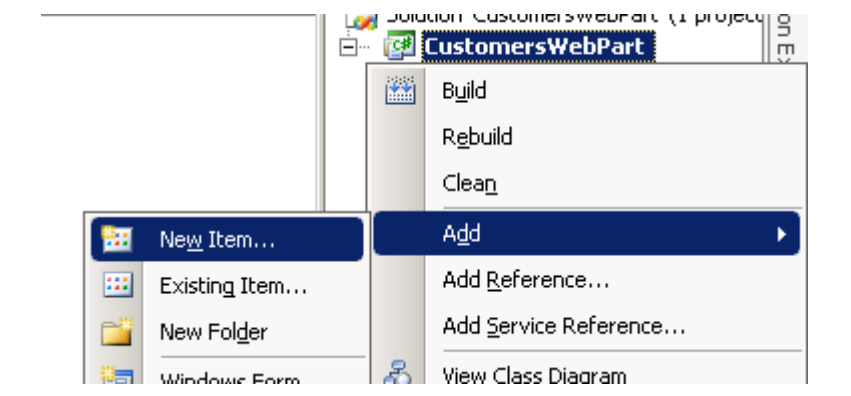
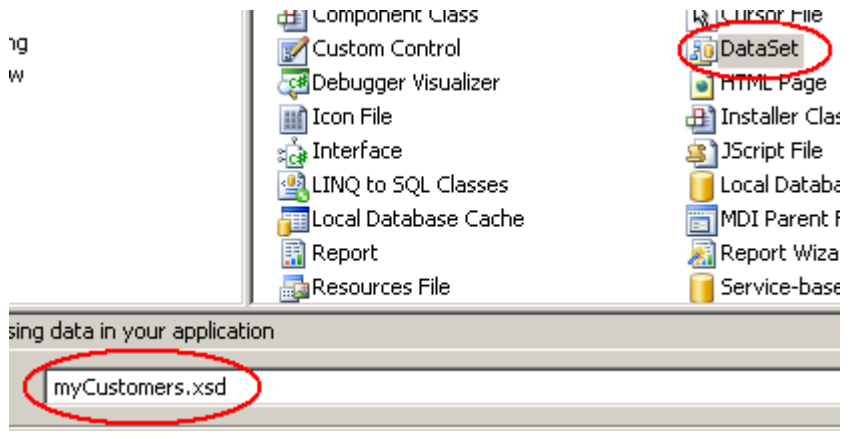
La plupart des développeurs ASP.NET (si ce n'est tous) connaissent bien le contrôle GridView qui leur permet d'afficher rapidement des données tabulaires à grands renforts de DataBinding. Le GridView est aussi disponible dans SharePoint, mais il y a mieux ! Il y a le SPGridView permettant d'afficher ses données à la manière des listes SharePoint. Ce contrôle apporte le tri, le regroupement, le paging, un menu contextuel etc... Bref le bonheur ! ☺

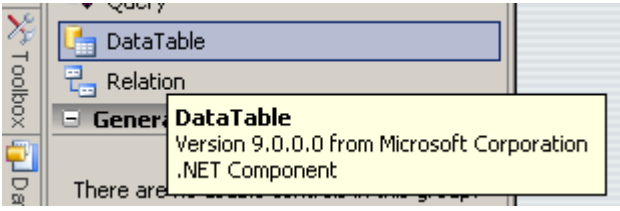

Une vision globale de ce contrôle dépasserait de loin le temps imparti pour ce Workshop, nous nous contenterons donc de mettre en place un SPGridView simple.

Le SPGridView tout comme le GridView repose sur sa propriété DataSource pour lui fournir les données qui seront affichées. Dans notre cas ces données seront issues d'un DataSet typé que nous allons créer et peupler.

Etape 7 : Ajout d'un DataSet typé

Bien que la plupart d'entre vous sachent déjà comment ajouter un DataSet typé à un projet je reprécise ci-dessous tout de même la démarche.

<p>Depuis le '<i>Solution Explorer</i>' faites un clic droit sur le projet et sélectionnez '<i>Add->New Item</i>'</p>	
<p>Dans le sélecteur d'items choisissez '<i>DataSet</i>' et nommez votre item '<i>myCustomers.xsd</i>'</p>	

<p>Dans le designer de DataSet depuis la ToolBox glissez un nouvel élément 'DataTable'</p> <p>Renommez le 'Customers'</p>	
<p>En faisant un clic droit sur la DataTable ajoutez 3 nouvelles colonnes nommées : 'Firstname', 'Lastname', 'Companyname'</p> <p>Puis une quatrième nommée 'ID' dont vous changerez le type en 'Int32' en utilisant ses propriétés. Faite de cette colonne une clé primaire (<i>clic droit</i>)</p>	

Notre DataSet Typé est maintenant prêt. Il ne nous reste plus qu'à l'instancier et à le peupler en utilisant du code.

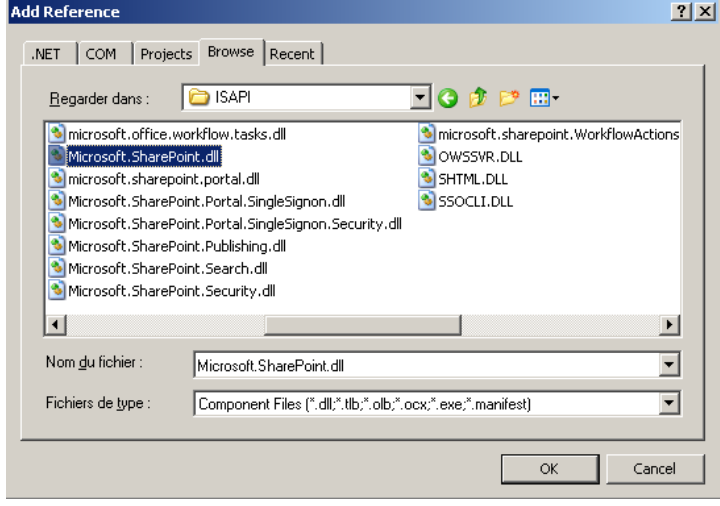
Etape 8 : Instanciation et peuplement du DataSet

<p>Dans le fichier 'CustomersWP.cs' juste en dessous des deux contrôles de la première partie, déclarez un nouveau DataSet typé de type 'myCustomers' nommé 'DS'</p>	<pre>protected TextBox customerName; protected Button btn; protected myCustomers DS;</pre>
<p>Dans la méthode 'createChildControl' après l'instanciation des contrôles TextBox et Button, ajoutez du code pour instancier votre DataSet</p>	<pre>this.Controls.Add(btn); DS = new myCustomers();</pre>
<p>A présent que notre DataSet est instancié nous allons le peupler en utilisant la méthode 'AddCustomersRow' de la DataTable 'Customers'</p>	<pre>DS.Customers.AddCustomersRow("Orlando", "Gee", "À Bike Store",1); DS.Customers.AddCustomersRow("Keith", "Harris", "Progressive Sports"</pre>

Note : Notre DataSet contiendra 20 enregistrements. Bien évidemment il est hors de question que vous les tapiez à la main. Vous trouverez dans **'C:\Techdays2008\Workshop\SharePoint\'** un fichier nommé **'FillDS.txt'** contenant le code pour créer les 20 enregistrements. Il vous suffit de faire un copier / coller. N'est ce pas qu'il est sympa l'auteur ☺

Etape 9 : Ajout du control SPGridView

Maintenant que nous avons des données à notre disposition il est temps de passer aux choses sérieuses et de voir comment travailler avec le SPGridView. Par défaut notre projet ne référence pas les DLL de SharePoint, ce sera donc notre première action.

<p>Depuis le menu 'Projet', cliquez sur 'Add Reference', puis dans la fenêtre de choix, cliquez sur l'onglet 'Browse' et naviguez jusqu'au répertoire 'C:\Program Files\Fichiers communs\Microsoft Shared\web server extensions\12\ISAPI' et sélectionnez la DLL 'Microsoft.SharePoint.dll'</p>	
<p>Importez l'espace de noms 'Microsoft.SharePoint.WebControls' dans lequel se situe la classe SPGridView</p>	<pre>using System.Web.UI.WebControls; using Microsoft.SharePoint.WebControls;</pre>
<p>En dessous des autres champs déclarez une variable 'gridView' de type 'SPGridView'</p>	<pre>protected myCustomers DS; protected SPGridView gridView;</pre>
<p>Dans la méthode 'createChildControl' en dessous du code pour peupler le DataSet ajoutez le code ci-contre.</p>	<pre>gridView = new SPGridView(); gridView.DataSource = DS.Customers.DefaultView; gridView.AutoGenerateColumns = false;</pre>

Note : Il est très important que la propriété **'AutoGenerateColumns'** du SPGridView soit toujours à **'false'** car ce contrôle ne supporte pas la génération automatique des colonnes (*au contraire du GridView*). Nous allons donc devoir créer chacune d'elles à la main et les ajouter à la collection des colonnes du SPGridView.

De plus il est important de noter que le SPGridView requiert des droits assez importants pour fonctionner. Etant donné la brièveté du Workshop, les règles de sécurité ont été oubliées et les assemblies ont tous les droits sur le serveur. Bien évidemment lors de vos essais personnels vous devrez passer par des fichiers de stratégies de sécurité adéquates.

Etape 10 : Création des colonnes et ajout du SPGridView aux Controls

Du fait que le SPGridView ne gère pas la création automatique des colonnes nous allons devoir recourir à des objets de type **'BoundField'** pour créer les colonnes nous-mêmes

<p>Ajoutez le code suivant juste après le code instanciant le SPGridView. Ce code crée un objet de type 'BoundField' et spécifie que le champ utilisé pour peupler cette colonne sera le champ 'Firstname' de la table « Customers »</p>	<pre>BoundField colName = new BoundField(); colName.DataField = "Firstname"; colName.HeaderText = "Prénom"; gridView.Columns.Add(colName);</pre>
<p>Faites de même pour les 2 colonnes suivantes (cf. ci-contre)</p>	<pre>colName = new BoundField(); colName.DataField = "Lastname"; colName.HeaderText = "Nom"; gridView.Columns.Add(colName); colName = new BoundField(); colName.DataField = "Companyname"; colName.HeaderText = "Société"; gridView.Columns.Add(colName);</pre>
<p>Enfin ajoutez le contrôle SPGridView à la collection 'Controls' de la Webpart</p>	<pre>this.Controls.Add(gridView); gridView.DataBind();</pre>

Etape 11 : Afficher le SPGridView

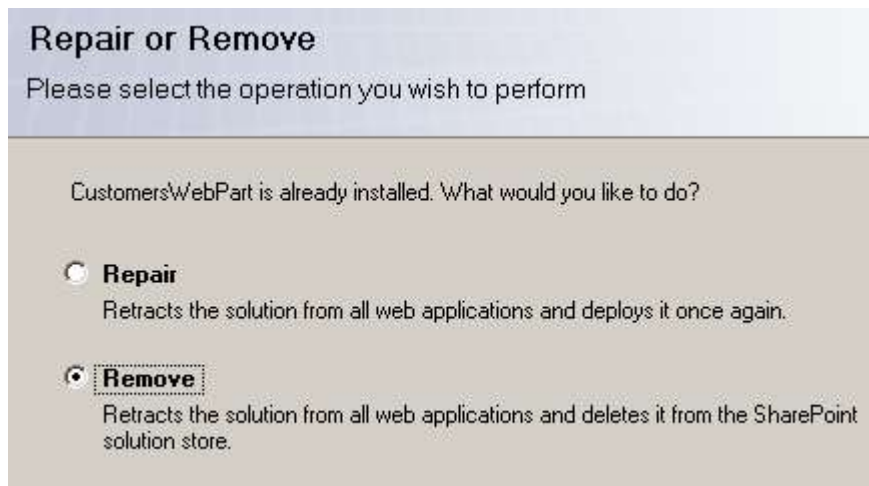
A présent notre contrôle est ajouté à la collection des contrôles de la WebPart, il nous suffit donc de l'afficher depuis la méthode **'RenderContents'** en appelant sa méthode **'RenderControl'**

<p>Ajoutez le code ci-contre dans la méthode 'RenderContents'. Notez que nous avons commenté l'affichage des contrôles précédents qui ne nous servent plus pour le moment.</p>	<pre>protected override void RenderContents(System { //writer.Write("<table border=\"0\">"); //writer.Write("<tr>"); //writer.Write("<td>"); //customerName.RenderControl(writer); //writer.Write("</td><td>"); //btn.RenderControl(writer); //writer.Write("</td></tr></table>"); gridView.RenderControl(writer); }</pre>
---	--

Etape 12 : Déploiement

Il est important ici de procéder par étapes. Etant donné que votre projet à déjà été déployé sur le serveur il convient d'abord de le supprimer du serveur avant de le redéployer et voici les étapes :

- 1) Supprimez la WebPart sur la page d'accueil
- 2) Désactivez la Feature au niveau de la collection de site (cf. Partie 1 pour activation)
- 3) Après avoir généré votre solution, exécutez 'setup.exe'. Sur la fenêtre ci-dessous choisissez '**Remove**'



- 4) Relancez 'setup.exe' et cette fois fait un déploiement complet (cf. Partie 1)
- 5) Ajoutez votre WebPart à la page d'accueil vous devriez obtenir ceci :

Actions du site

Mes clients

Prénom	Nom	Société
Orlando	Gee	A Bike Store
Keith	Harris	Progressive Sports
Donna	Carreras	Advanced Bike Components
Janet	Gates	Modular Cycle Systems
Lucy	Harrington	Metropolitan Sports Supply

Troisième partie : Ajoutons de la pagination !

Bien que déjà très sympathique cette WebPart de visualisation est très encombrante et le sera de plus en plus à mesure que la liste de nos clients grossira (très vite 😊). Or nous ne pouvons pas laisser une mise en page aussi chaotique sur notre site. Je vous propose d'ajouter pour cette troisième partie un peu de pagination à cette WebPart.

La pagination est gérée de manière très simple dans le SPGridView, une propriété à placer à true et un évènement à gérer et c'est tout ! (ou presque :p)

Etape 13 : Ajout de la pagination

Important : Il est très important que le code ci-dessous soit saisi entre l'ajout du SPGridView à la collection de contrôles et l'appel à DataBind sur celui-ci !

Ajoutez le code ci-contre à la méthode 'createChildControl' entre l'ajout du SPGridView à la collection de contrôles et l'appel à DataBind	<pre> this.Controls.Add(gridView); gridView.PageSize = 5; gridView.AllowPaging = true; gridView.PageIndexChanging += new GridViewPageEventHandler(gridView_PageIndexChanging); gridView.PagerTemplate = null; gridView.DataBind(); }</pre>
---	---

Le code ci-dessous autorise le Paging à hauteur de 5 éléments par page et définir un gestionnaire d'évènement en charge de mettre à jour le SPGridView lorsque l'on clique sur une des pages.

Etape 14 : Ajout du gestionnaire d'évènement

Dans le gestionnaire d'évènement créé automatiquement ajoutez le code suivant :	<pre>void gridView_PageIndexChanging(object sender, GridViewPageEventArgs e) { gridView.PageIndex = e.NewPageIndex; gridView.DataBind(); }</pre>
---	--

Et voilà votre WebPart supporte maintenant la pagination ! Vous pouvez la déployer en suivant les mêmes étapes que précédemment. Pour aller plus vite cette fois vous n'avez pas besoin de tout désinstaller, choisissez l'option **'Repair'** du setup.exe et la manip se fera toute seule !

Conclusion

Ce Workshop est à présent terminé, j'espère que vous avez pris plaisir à le suivre, au moins autant que j'ai eu plaisir à le préparer. Rendez-vous dans un prochain épisode 😊