

Sharepoint 2007 : Les Application Pages

par Ludovic Lefort ([Site web](#)) ([Blog](#))

Présentation, création et déploiement d'Application Pages dans Sharepoint 2007.

- 0 - Introduction
- 1 - Présentation
 - 1.1 - Qu'est ce qu'une Application Page ?
 - 1.2 - Application Pages contre Site Pages
- 2 - Création d'une Application Page
 - 2.1 - La page ASPX
 - 2.2 - Le code behind
- 3 - Déploiement via une solution
 - 3.1 - Création de la feature
 - 3.2 - Encapsulation dans une solution
 - 3.3 - Déploiement de la solution
 - 3.4 - Activation de la feature
 - 3.5 - Résultat
- 4 - Conclusion
- 5 - Référence
- 6 - Liens
- 7 - Sources du projet

0 - Introduction

Dans cet article je vais définir ce que sont les Application Pages, comment les créer et surtout comment les déployer via une solution Sharepoint.

1 - Présentation

1.1 - Qu'est ce qu'une Application Page ?

Les Applications Pages sont les pages de configuration partagées par tous les sites hébergés par un même serveur web.

Un bon exemple est la page Settings.aspx que l'on obtient via le menu "Site Actions / Site Settings".

L'avantage des Application Pages par rapport aux Site Pages est qu'elles sont uniques et non customisable, elles peuvent donc être chargées une fois pour toutes par le serveur web.

1.2 - Application Pages contre Site Pages

| | Application Pages | Site Pages |
|---------------|---|---|
| Rôles | Permettent l'administration de Sharepoint. Elles sont communes à tous les sites hébergés sur un même serveur Web. | Ce sont des pages de contenus spécifiques à un seul site. |
| Emplacement | Dans le répertoire 12\Template\Layouts de Sharepoint. Le répertoire _layouts de chaque site pointe sur ce même répertoire. | Peut se trouver sur le disque si elle fait partie d'un "Site Definition". Les Site Pages peuvent également être chargées depuis la "Content Database". |
| Customisation | Non | Oui |
| Master Page | Application.master | Master.master |

2 - Création d'une Application Page

Pour cet exemple nous allons créer une page qui affichera le nom du site en cours, rien de bien méchant en soi.

Pour cela créons un nouveau projet de type "Class Library" dans Visual Studio.Net

2.1 - La page ASPX

SiteInformation.aspx

```
<%@ Assembly Name="Microsoft.SharePoint, Version=12.0.0.0, Culture=neutral,
PublicKeyToken=71e9bce111e9429c" %>
<%@ Assembly Name="SiteInformation, Version=1.0.0.0, Culture=neutral,
PublicKeyToken=46b0b4a5d479ed08" %>
<%@ Page Language="C#" MasterPageFile="~/_layouts/application.master"
Inherits="SiteInformation.SiteInformation" EnableViewState="false" EnableViewStateMac="false" %>

<asp:Content ID="Main" ContentPlaceHolderID="PlaceHolderMain" runat="server">
    Site name : <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
</asp:Content>
```

Détaillons maintenant ce code :

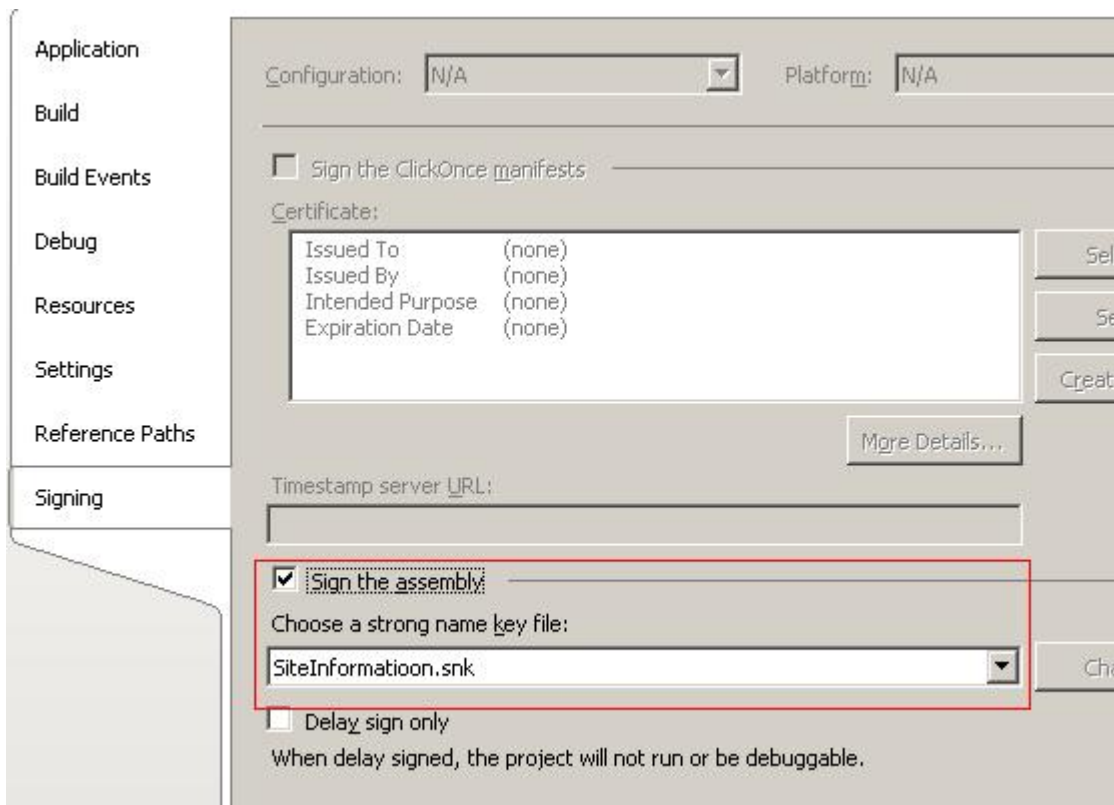
Pour commencer nous allons ajouter une référence vers l'API de Sharepoint.

Pour obtenir le "Strong Name" d'une dll, je vous conseil d'utiliser **Reflector for .Net**

```
<%@ Assembly Name="Microsoft.SharePoint, Version=12.0.0.0, Culture=neutral,
PublicKeyToken=71e9bce111e9429c" %>
```

Ajoutons maintenant une référence vers notre projet lui même afin de pouvoir faire le lien entre cette page et le code behind.

Afin de pouvoir déployer notre dll dans la gac nous devons la signer:



C'est uniquement après cette étape que l'on peut obtenir le **PublicKeyToken** de la librairie.

```
<%@ Assembly Name="SiteInformation, Version=1.0.0.0, Culture=neutral,
  PublicKeyToken=46b0b4a5d479ed08" %>
```

Pour pouvoir fonctionner dans Sharepoint notre page doit être liée avec la Master Page Application de Sharepoint

Il faut également la lier avec le code behind que l'on va écrire :

```
<%@ Page Language="C#" MasterPageFile="~/_layouts/application.master"
  Inherits="SiteInformation.SiteInformation" EnableViewState="false" EnableViewStateMac="false" %>
```

Ajoutons maintenant un label dans le Place Holder principal de la Master Page Sharepoint:

```
<asp:Content ID="Main" ContentPlaceHolderID="PlaceholderMain" runat="server">
  Site name : <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
</asp:Content>
```

2.2 - Le code behind

```
using System;
using System.Collections.Generic;
using System.Text;
```

```
using Microsoft.SharePoint;
using Microsoft.SharePoint.WebControls;
using System.Web;
using System.Web.UI.WebControls;

namespace SiteInformation
{
    public class SiteInformation : LayoutsPageBase
    {
        protected Label Labell=null;
        protected override void OnLoad(EventArgs e)
        {
            SPWeb web = this.Web;
            this.Labell.Text = web.Name;
        }
    }
}
```

Décrivons les parties importantes de ce code :

Ajoutons une référence vers les deux librairies **System.Web** et **Microsoft.Sharepoint** :

```
using Microsoft.SharePoint;
using System.Web;
```

Les Application Pages doivent hériter de la classe Sharepoint LayoutsPageBase, cela nous permettra de récupérer le contexte Sharepoint courant.

```
public class SiteInformation : LayoutsPageBase
```

Pour rappel nous sommes dans un projet de type "Class Library" et non dans un projet web, Cela signifie que Visual Studio est incapable de faire le lien entre notre fichier aspx et son code behind.

Si nous essayons d'atteindre notre label depuis le code behind, Visual Studio .net affichera des erreurs lors de la compilation.

Pour éviter cela il faut que le nom de la page soit le même que celui de la classe lié. Dans notre cas : **SiteInformation.aspx** et **SiteInformation.cs**.

De plus tout les contrôles de la page doivent être redéclaré dans le code avec exactement les même noms :

```
protected Label Labell=null;
```

Dernière étape : Récupérer le site en cours et afficher son nom sur notre page lors de son chargement :

```
SPWeb web = this.Web;
this.Labell.Text = web.Name;
```

3 - Déploiement via une solution

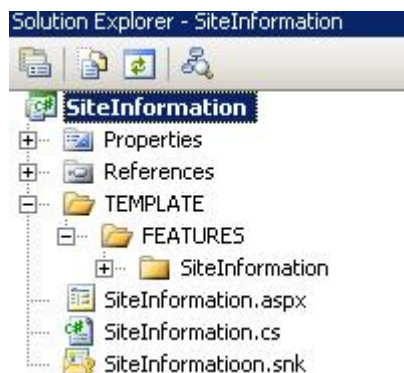
Comme pour tout développement Sharepoint le réel challenge c'est le déploiement ...

La solution la plus propre à mon sens est d'encapsuler le développement dans une feature et de la déployer à l'aide d'un fichier wsp.

Etant donné que j'envisage un article consacré uniquement au déploiement je ne vais pas trop entrer dans les détails.

3.1 - Création de la feature

La première chose à faire est de recréer la structure des dossiers de la feature tel que dans Sharepoint après le déploiement :



Nous allons donc créer les répertoires suivants : **TEMPLATE**, **FEATURES** et **SiteInformation**.

Une feature contient toujours un fichier **feature.xml** qui permet de la définir :

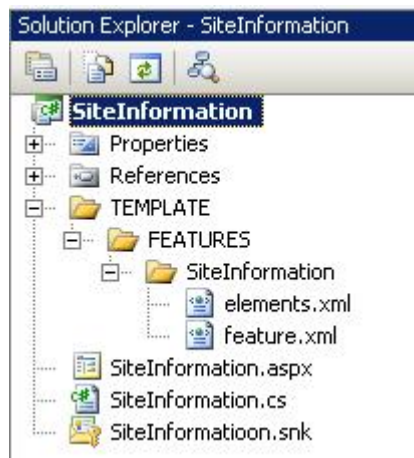
```
<?xml version="1.0" encoding="utf-8" ?>
<Feature xmlns="http://schemas.microsoft.com/sharepoint/"
  Id="EDFD660C-66BA-4c73-8928-DC84CE52D8AC"
  Title="Site Information"
  Description="Display information about the current site in the site settings page"
  Scope="Site"
  Hidden="false"
  ImageUrl="menuprofile.gif">
  <ElementManifests>
    <ElementManifest Location="elements.xml" />
  </ElementManifests>
</Feature>
```

Un second fichier **elements.xml** va définir les actions à entreprendre par la feature. Dans notre cas nous allons juste ajouter un lien dans la page Settings.aspx pointant vers la page créée précédemment :

```
<?xml version="1.0" encoding="utf-8" ?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <CustomAction
    Id="ConfigurationPages"
```

```
GroupId="SiteAdministration"  
Location="Microsoft.SharePoint.SiteSettings"  
Sequence="2000"  
Title="Display informations about this site"  
Description="Use this link to view informations regarding this site"  
>  
<UrlAction Url="_layouts/dvp/siteinformation.aspx"/>  
</CustomAction>  
</Elements>
```

Voici la nouvelle structure de notre projet :



3.2 - Encapsulation dans une solution

Créons maintenant une solution pour faciliter le déploiement de notre feature.

Une solution est un fichier .wsp qui contient:

- Tous les éléments à déployer : feature, pages, dll, images, ...
- Un fichier manifest qui sera le script d'installation
- Un fichier ddf pour définir l'emplacement de chaque élément

Pour créer le fichier manifest et ddf deux solutions s'offrent à nous : le faire à main dans notepad ou utiliser un très bon générateur gratuit : **WSPBuilder**

Pour ma part je préfère la deuxième solution.

WSPBuilder est simple à utiliser il suffit de créer un répertoire du même nom que notre solution contenant **wspbuilder.exe**, **wspbuilder.exe.config**, **CabLib.dll** ainsi que trois répertoires :

- \12 qui représente le répertoire **C:\Program Files\Common Files\Microsoft Shared\web server extensions\12** sur votre serveur Sharepoint
- \80 qui représente la racine de votre Web Application dans IIS
- \GAC qui doit contenir toutes les librairies à enregistrée dans la GAC

Voici la structure à obtenir :

.SiteInformation

.SiteInformation\WSPBuidler.exe

.SiteInformation\WSPBuidler.exe.config

.SiteInformation\CabLib.dll

.SiteInformation\12\TEMPLATE\FEATURES\SiteInformation\feature.xml

.SiteInformation\12\TEMPLATE\FEATURES\SiteInformation\elements.xml

.SiteInformation\12\TEMPLATE\LAYOUTS\dvp\SiteInformation.aspx

.SiteInformation\GAC\SiteInformation.dll

Exécutons maintenant WSPBuilder.exe et là ... magie ... SiteInformation.wsp prêts à être déployé !

3.3 - Déploiement de la solution

Nous allons déployer notre solution sur le serveur Sharepoint.

Pour cela ouvrir un command prompt et depuis le répertoire \bin de Sharepoint faites :

stsadm -o addsolution -filename c:\SiteInformation.wsp en considérant que le fichier SiteInformation.wsp se trouve à la racine de votre lecteur C.

Allons maintenant dans la **Central Administration de Sharepoint** et cliquons sur **Solution Management** dans l'onglet **Operations**

Global Configuration

- Timer job status
- Timer job definitions
- Master site directory settings
- Site directory links scan
- Alternate access mappings
- Quiesce Farm
- Manage farm features
- **Solution management**

Une nouvelle page s'ouvre avec la liste des solutions installées sur notre serveur, Ouvrez **SiteInformation.wsp**

Cliquez maintenant sur **Deploy Solution**

Central Administration > Operations > Solution Management > Solution Properties

Solution Properties

[Deploy Solution](#) | [Retract Solution](#) | [Back to Solutions](#)

| | |
|---------------------------------------|---------------------|
| Name: | siteinformation.wsp |
| Type: | Core Solution |
| Contains Web Application Resource: | Yes |
| Contains Global Assembly: | Yes |
| Contains Code Access Security Policy: | No |







3.4 - Activation de la feature

Allons maintenant dans les **site settings** de notre Site Collection et cliquons sur le menu **Site Collection Features**

Site Collection Administration

- ▣ Search settings
- ▣ Search scopes
- ▣ Search keywords
- ▣ Recycle bin
- ▣ Site directory settings
- ▣ Site collection usage reports
- ▣ **Site collection features**
- ▣ Site hierarchy
- ▣ Portal site connection
- ▣ Site collection audit settings
- ▣ Audit log reports
- ▣ Site collection policies

La liste des features installées sur notre Site Collection apparait, Cliquons sur le bouton **Activate** de notre nouvelle feature :

-  This feature uploads all web parts required for Search Center
-  **Office SharePoint Server Standard Site Collection features**
Features such as user profiles and search, included in the Office SharePoint Server Standard License
-  **Reporting**
Creates reports about information in Windows SharePoint Services.
-  **Routing Workflows**
Workflows that send a document for feedback or approval.
-  **Site Information**
Display information about the current site in the site settings page
-  **Three-state workflow**
Use this workflow to track items in a list.

L'intérêt évident de l'utilisation de feature réside dans la facilité de les activer/désactiver.

3.5 - Résultat

Maintenant que la feature est activée retournons dans les Site Settings et constatons l'apparition de notre lien :

Site Administration

- ▣ Regional settings
- ▣ Site libraries and lists
- ▣ Site usage reports
- ▣ User alerts
- ▣ RSS
- ▣ Search visibility
- ▣ Sites and workspaces
- ▣ Site features
- ▣ Delete this site
- ▣ Related Links scope settings
- ▣ **Display informations about this site**

Cliquons sur ce lien pour afficher notre page.aspx.

4 - Conclusion

Les Applications Pages fournissent une solution simple pour développer des pages d'administration communes à tout un serveur Web.

5 - Référence

Livre : Inside Microsoft Windows Sharepoint Services 3.0 par Ted Pattison et Daniel Larson

6 - Liens

Retrouvez le site du projet complet sur codeplex : <http://www.codeplex.com/spsiteinformation>

Mon site : <http://lefortludovic.developpez.com>

Mon blog : <http://blogs.ezos.com/blog/le>

7 - Sources du projet

Cliquez [ici](#) pour télécharger les sources de ce projet.

Je tiens à remercier [Dolphy25](#) et [stephane eyskens](#) pour leurs corrections orthographiques.

