Filtrer les modèles de site SharePoint

Filtrer l'affichage des modèles de site SharePoint

Je vous propose de découvrir à l'aide de cet article quels mécanismes peuvent vous permettent de filtrer les Sites Templates disponibles au sein de vos sites SharePoint, et quel intérêts présentent chacun d'entre eux.

Introduction

Les définitions de site SharePoint permettent de spécifier tout ce qui doit être présent dans un site à sa création:

- Quelles fonctionnalités doivent être activées ?
- Quelles listes doivent être présentes ?
- A quoi ressemblera la page d'accueil ?

Bref, les modèles de sites SharePoint permettent aux utilisateurs de créer très simplement des sites en adéquation avec les règles de gouvernance définies.

Vous retrouverez comment **Créer un Site Template pour WSS V3** sur un des précédents articles de **Fabrice Romelard**.

Vous pourrez compléter cette gestion des templates de sites par la découverte du feature stapling à travers le **blog de Gaëtan Bouveret** (VF) et ce très bon article paru sur le **SharePoint Magazine** (VO).

Présentation

Les gestionnaires de sites disposent donc de plusieurs modèles disponibles pour créer des sites. C'est bien, sauf que les demandes de modèles sont souvent accompagnées de conditions telles que :

- "*il faut que le modèle ne serve qu'à créer des collections de sites*"
- ou bien "*les modèles utilisables doivent changer en fonction du site sur lequel se trouve l'utilisateur*"
- ou encore "*la liste des modèles utilisables doit varier en fonction des droits utilisateurs*"
- ...

Bref, tout un ensemble de demandes qu'on pourrait penser non gérées dans SharePoint, liées au thème de la gouvernance, ... et pour lesquelles il existe bel et bien des réponses techniques.

L'objectif de cet article est donc de comprendre quels mécanismes permettent de filtrer les modèles de sites disponibles, et quels intérêts présentent chacun d'entre eux.

Filtrage spécifié dans la définition de site

C'est avec cette approche qu'il est possible de limiter les modèles disponibles lors de la création de nouvelles collections de sites (sites racines), ou à contrario, de limiter ceux disponibles lors de la création de nouveaux sous-sites au sein d'une collection.

Cette approche sous entend que vous créez une nouvelle définition de site (ou que vous en modifiez une, mais gardez en tête que modifier des éléments standards est source de problèmes).

En d'autres termes, un nouveau dossier dans le répertoire SiteTemplates (comprenant au moins un fichier onet.xml), et un nouveau fichier WEBTEMPxxx.xml dans le dossier \12\TEMPLATE\\XML.

Les propriétés RootWebOnly et SubWebOnly

Ce fichier WEBTEMPxxx.xml comporte une liste de configurations exploitant deux propriétés nommées RootWebOnly et SubWebOnly. Fixées à True ou False, ces propriétés permettent de n'afficher le modèle de site que lors de la création d'une collection de sites ou, au contraire, uniquement lors de la création d'un nouveau sous-site. Utilisez donc l'une ou l'autre de ces propriétés pour déterminer la portée de la définition de site.

La propriété VisibilityFeatureDependency

Les configurations du fichier WEBTEMPxxx.xml peuvent également comporter la propriété VisibilityFeatureDependency. Cette propriété permet d'assujettir l'affichage d'un modèle à la présence d'une feature dans le site parent. Sous forme de string, sa valeur représente le GUID de la feature dont la présence est requise pour afficher le modèle. Idéal pour restreindre l'utilisation de vos modèles personnalisés à une application web ou une collection de sites spécifiques.

La propriété Hidden

Enfin, la propriété Hidden. Simple booléen indiquant si le modèle est visible ou non, quelque soit le cas. A quoi pourrait servir un modèle caché quelque soit le cas ? Et bien tout simplement à créer de nouveaux sites par code uniquement. En d'autres termes, vos utilisateurs ne pourront pas s'appuyer sur une configuration de modèle marquée Hidden, mais votre code si. Idéal dans le cadre de migration de sites dont les modèles ne sont conservés que pour la rétro-compatibilité par exemple, ou pour tout autre modèle à réserver.

Exemples

```
<Template Name="GRIBS" ID="21121">
   <Configuration ID="0"
      Title="Template site racine"
      Hidden="FALSE"
      ImageUrl="/_layouts/images/stsprev.png"
      RootWebOnly="True"
      Description="Un modèle que vous ne pouvez voir que lorsque vous créez une nouvelle collection de sites."
      DisplayCategory="Grib's Templates" />
   <Configuration ID="1"
      Title="Template sous-site"
      Hidden="FALSE"
      ImageUrl="/_layouts/images/stsprev.png"
      SubWebOnly="True"
      Description="Un modèle que vous ne pouvez voir que lorsque vous créez un sous-site."
      DisplayCategory="Grib's Templates" />
   <Configuration ID="2"
      Title="Template avec dépendance de feature"
      Hidden="FALSE"
      ImageUrl="/_layouts/images/stsprev.png"
      VisibilityFeatureDependency="12A2485F-4F4B-6136-9142-FA4F1147C8B4"
      Description="Un modèle que vous ne pouvez voir que
lorsque le site parent a la feature 12A2485F-4F4B-6136-9142-FA4F1147C8B4 active."
      DisplayCategory="Grib's Templates" />
   <Configuration ID="3"
      Title="Template Masqué"
      Hidden="True"
      ImageUrl="/_layouts/images/stsprev.png"
      Description="Un site masqué dans tous les cas"
      DisplayCategory="Grib's Templates" />
</Template>
```

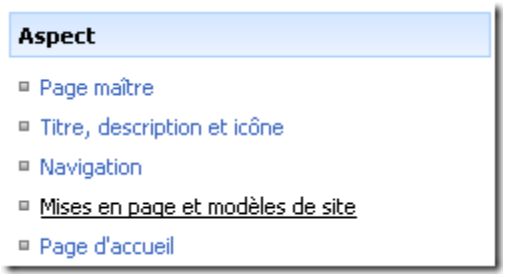
Filtrage réalisé au niveau des sites

Cette approche permet de différencier la liste des modèles disponibles selon les sites. Chaque site dispose d'une collection de modèles dits "autorisés".

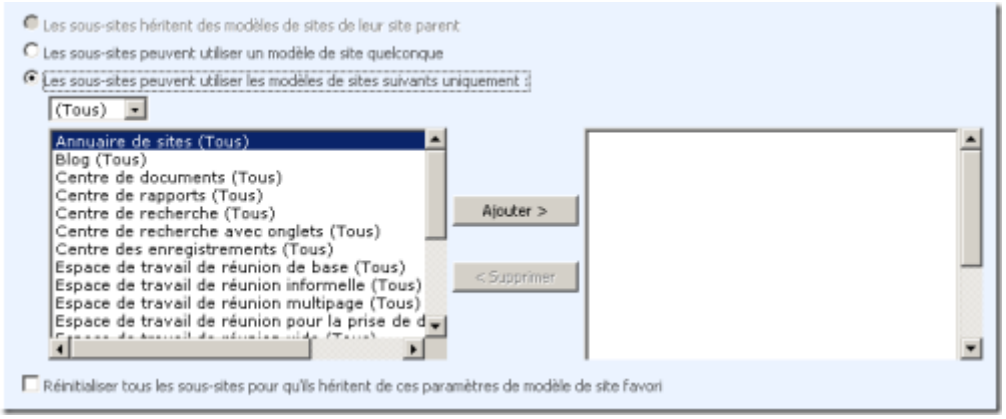
Il est donc possible de modifier cette collection pour chaque site et ainsi, d'obtenir une liste de modèles différente pour chaque site. Pratique pour proposer une liste restreinte de modèles de sites lorsqu'on laisse la possibilité aux utilisateurs de créer de nouveaux sous-sites.

Via l'interface web (MOSS uniquement)

Concrètement, cette opération peut être menée via l'interface Web, depuis les paramètres du sites :



Puis sélectionnant les modèles à activer



A noter au passage que ce lien **est exclusivement présent sous MOSS**.

Par code

Quoi qu'il en soit, c'est le modèle objet de WSS qui est utilisé ici... vous pouvez donc administrer la liste des modèles disponibles via l'API, qu'il s'agisse de templates WSS ou de templates MOSS.

La méthode SetAvailableWebTemplates de la classe SPWeb permet de définir cette collection. L'idée consiste donc à créer une collection (quitte à réutiliser la collection déjà existante à l'aide de la méthode GetAvailableWebTemplates), lui ajouter/retirer certains modèles et l'affecter au site.

Un petit exemple de ce que ça donne côté code :

```
/// <summary>
/// Exemple filtrant les modèles disponibles pour le site racine,
///pour ne garder que les modèles de la catégorie "Collaboration"
/// </summary>
/// <param name="site"></param>
public void FilterTemplates(SPSite site)
{
    SPWebTemplateCollection templates = null;
    Collection<SPWebTemplate> collection = new Collection<SPWebTemplate>();
    using (SPWeb rootWeb = site.RootWeb)
    {
        templates = rootWeb.GetAvailableWebTemplates((uint)1036);
        foreach (SPWebTemplate template in templates)
        {
            if (template.FilterCategories.Equals("Collaboration", StringComparison.InvariantCultureIgnoreCase))
            {
                collection.Add(template);
            }
        }
        rootWeb.SetAvailableWebTemplates(collection, (uint)1036);
        rootWeb.Update();
    }
}
```

Bien sûr, ce genre de code peut très bien être contenu dans une feature receiver pour proposer une liste de modèles sur mesure à l'installation d'une feature.

Filtrage dynamique (création du site par code)

Nous avons donc vu comment filtrer les modèles en fonction d'un site et de la nature du site à créer (site racine, sous site). Mais le jeu de modèles proposés sera toujours le même quelque soit l'utilisateur qui s'apprête à créer un site. Il n'est pas vraiment possible de faire autrement, mais il est possible de rendre les modèles dynamiques.

Certains modèles présents dans MOSS n'existent pas en tant que tel. Le portail collaboratif, par exemple, n'a aucun fichier Onet.xml associé.

Il s'appuie en réalité sur le mécanisme de provisionning. Ce mécanisme permet l'exécution de code au moment de la création du site, ce qui permet donc de modifier le modèle définitif à la volée.

Valable à la fois pour WSS et pour MOSS, les seules définitions de site à l'utiliser en standard sont des modèles de site MOSS. Par exemple, le modèle de portail collaboratif s'en sert pour créer non pas un site, mais 6 sites (le site racine et 5 sous sites).

Le provisonning dans le fichier WEBTEMP

Notez les 3 attributs ProvisionAssembly, ProvisionClass et ProvisionData de la configuration du site portail collaboratif.

Ces 3 éléments permettent l'exécution de code en lieu et place du mécanisme de création de site standard

```
<Template Name="GRIBSAMPLE" ID="21000">
 <Configuration ID="0" Title="Site générique" Type="0" Hidden="FALSE"
  ImageUrl="/_layouts/1033/images/template_corp_intranet.png"
  Description="Site généré dynamiquement en fonction du profil."
  ProvisionAssembly="Gribs.SharePoint, Version=1.0.0.0, Culture=neutral, PublicKeyToken=4ae93c9121eb7296"
  ProvisionClass="Gribs.SharePoint.MyProvisionClass "
  ProvisionData=""
  SubWebOnly="TRUE"
  DisplayCategory="Gribs"/>
</Template>
```

Le provisonning dans le code

Si vous avez déjà développé des Receivers pour SharePoint, vous ne serez pas dépaysé par le code ci dessous.

Une classe qui hérite de SPWebProvisioningProvider et une méthode surchargée (ici la méthode Provision). L'exemple ci dessous applique un modèle différent selon si c'est l'administrateur de la collection de sites qui créé un nouveau site, ou si c'est un utilisateur avec moins de droits.

Ce mécanisme peut donc permettre de proposer un modèle "générique" qui sélectionnera le modèle final selon n'importe quels critères de votre choix.

```
namespace Gribs.SharePoint
{
   public class MyProvisionClass : Microsoft.SharePoint.SPWebProvisioningProvider
   {
      public override void Provision(SPWebProvisioningProperties props)
      {
         SPWeb web = props.Web;
         if (web.CurrentUser.IsSiteAdmin)
         {
            web.ApplyWebTemplate("STS#1");
         }
         else
         {
            //apply another web template
         }
      }
   }
}
```

Conclusion

Les solutions techniques sont donc nombreuses et offrent de très larges possibilités. Elément clé avant d'exploiter ce genre de solutions : la politique de gouvernance.

En effet, il reste indispensable de préciser au mieux la stratégie qui devra s'appliquer aux espaces collaboratifs SharePoint.

Voici quelques liens utiles si cet article vous a intéressé :

- **MSDN - SPWebTemplate.ProvisionClass Property (Microsoft.SharePoint)**
- **SharePoint Deployment : Feature, Feature, Solution .... mais aussi WebProvisioningProvider**
- **MSDN - SPWeb.SetAvailableWebTemplates Method (Microsoft.SharePoint)**
- **MSDN - Site Definitions and Configurations**
- **MSDN - Configuration Element (Site)**
- **MSDN - WebTemp.xml**
- **Créer un Site Template pour WSS V3**
- **Office Online**
- **Club SPS MOSS FRANCE(FR)**

En vous souhaitant de bons projets de développement.

Sebastien PICAMELOT (alias gribouillon)
Consultant / Formateur SharePoint - **Winwise**