

PUBLIER UN FLUX RSS

Initié par Netscape, le rss est un moyen simple d'ouvrir le contenu de votre site autrement que par l'utilisation de XML WebServices.

Ce tutoriel a trois objectifs :

- Vous fournir gratuitement les sources du générateur de flux rss
- Vous montrer comment il fonctionne
- Découvrir surtout la simplicité par laquelle on génère un flux XML avec .net

N'hésitez pas à télécharger les sources.

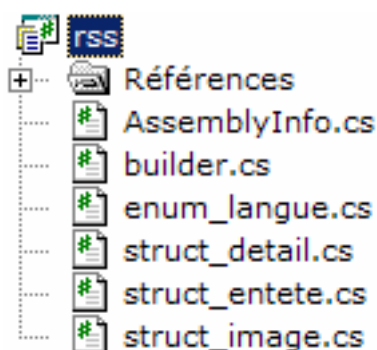
1. Le constructeur de flux RSS

Le projet rss dispose de 3 structures qui permettent de construire proprement les données qu'attendent le flux rss.

enum_langue est une énumération de codes langues afin d'éviter l'insertion de données un peut trop exotiques.

struct_entete alimente les données de base du flux rss comme l'url du site, l'éditeur et éventuellement l'image d'accompagnement structurée dans **struct_image**.

struct_detail s'articule autour des publications à exposer.



```

XmlTextWriter v_XmlTextWriter;

public builder(System.IO.TextWriter p_TextWriter)
{
    v_XmlTextWriter = new XmlTextWriter(p_TextWriter);
    v_XmlTextWriter.Formatting = Formatting.Indented;
}

```

La méthode builder va nous permettre de construire le flux rss qui sera exposé dans le Page.Response.Output de la page ASPX.

```

public void ConstruitRss(struct_entete p_datos)
{
    v_XmlTextWriter.WriteStartDocument();
    v_XmlTextWriter.WriteStartElement("rss");
    v_XmlTextWriter.WriteAttributeString("version", "2.0");
    v_XmlTextWriter.WriteStartElement("channel");
    v_XmlTextWriter.WriteElementString("title", p_datos.Titre);
    v_XmlTextWriter.WriteElementString("link", p_datos.Lien);
    v_XmlTextWriter.WriteElementString("description", p_datos.Description);
    v_XmlTextWriter.WriteElementString("language", Convert.ToString(p_datos.Langue));

    if(p_datos.Editeur.Length > 0)
        v_XmlTextWriter.WriteElementString("managingEditor", p_datos.Editeur);

    if (p_datos.Copyright.Length > 0)
        v_XmlTextWriter.WriteElementString("copyright", p_datos.Copyright);

    v_XmlTextWriter.WriteElementString("generator", p_datos.Generateur);

    if(p_datos.Webmaster.Length > 0)
        v_XmlTextWriter.WriteElementString("webMaster", p_datos.Webmaster);

    v_XmlTextWriter.WriteElementString("lastBuildDate", DateTime.Now.ToString("r"));
    v_XmlTextWriter.WriteElementString("ttl", p_datos.Ttl.ToString());

    // Gestion de l'image
    if (p_datos.Image.ImageUrl.Length > 0)
    {
        // Je dois générer l'élément image
        v_XmlTextWriter.WriteStartElement("image");

        if (p_datos.Image.Titre.Length > 0)
            v_XmlTextWriter.WriteElementString("title", p_datos.Image.Titre);

        v_XmlTextWriter.WriteElementString("url", p_datos.Image.ImageUrl);

        if (p_datos.Image.Lien.Length > 0)
            v_XmlTextWriter.WriteElementString("link", p_datos.Image.Lien);

        if (Convert.ToInt32(p_datos.Image.Largeur) > 0)
            v_XmlTextWriter.WriteElementString("width", p_datos.Image.Largeur.ToString());

        if (Convert.ToInt32(p_datos.Image.Hauteur) > 0)
            v_XmlTextWriter.WriteElementString("height", p_datos.Image.Hauteur.ToString());

        if (p_datos.Image.Description.Length > 0)
            v_XmlTextWriter.WriteElementString("link", p_datos.Image.Description);

        v_XmlTextWriter.WriteEndElement();
    }
}

```

Ma méthode ConstruitRss fabrique le corps du flux XML.

WriteStartDocument() me permet de démarrer mon document XML.

WriteStartElement({MonElement}) construit un nœud sur lequel je peux ajouter un attribut avec WriteAttributString({MonAttribut},{SaValeur}).

```
/// <summary>
/// Ajoute une publication dans le Channel
/// </summary>
/// <param name="p_datas">Passe la structure du détail</param>
public void AjoutePublication(struct_detail p_datas)
{
    // Création de l'élément
    v_XmlTextWriter.WriteStartElement("item");

    v_XmlTextWriter.WriteElementString("title",p_datas.Titre);
    v_XmlTextWriter.WriteElementString("link",p_datas.Url);

    if (p_datas.Auteur.Length > 0)
        v_XmlTextWriter.WriteElementString("author",p_datas.Auteur);

    v_XmlTextWriter.WriteElementString("pubDate",p_datas.DatePublication.ToString());

    if (p_datas.Categorie.Length > 0)
        v_XmlTextWriter.WriteElementString("category",p_datas.Categorie);

    if (p_datas.Sujet.Length > 0)
        v_XmlTextWriter.WriteElementString("subject",p_datas.Sujet);

    v_XmlTextWriter.WriteElementString("description",p_datas.Description);
    // Cloture de l'élément
    v_XmlTextWriter.WriteEndElement();
}
```

La méthode `AjoutePublication` permet l'insertion d'un item rss. Les données sont passées dans la variable `p_datas`. On utilise toujours `WriteStartElement()` et `WriteElementString()` pour ajouter un nœud fils.

N'oublions pas le `WriteEndElement()` pour fermer le nœud.

```
/// <summary>
/// Cloture les elements et ferme le document
/// </summary>
public void FinDuDocument()
{
    // Fin du Channel
    v_XmlTextWriter.WriteEndElement();

    // Fin du rss
    v_XmlTextWriter.WriteEndElement();

    // Fin du document
    v_XmlTextWriter.WriteEndDocument();

    v_XmlTextWriter.Flush();
    v_XmlTextWriter.Close();
}
```

Enfin, la méthode `FinDuDocument()` clôt les différents nœuds et le document XML.

2. La génération du flux rss

Le moment est venu de générer le flux rss.

Afin que votre flux soit accessible depuis l'url <http://MonSite/rss>, créer une page index.aspx qui devra être défini comme page par défaut du répertoire dans les paramètres IIS.

Pour la démonstration, nous avons créé une page rss.aspx. Commencez par supprimer toutes les données HTML de la page générée par VisualStudio.net.

```
// Gestion du contexte de retour pour tenir compte de la richesse des caractères Français
Context.Response.ContentEncoding = System.Text.Encoding.GetEncoding("ISO-8859-1");

// Reponse XML
Page.Response.ContentType = "text/xml";

// Alimentation des données de l'entete du fil rss
struct_entete v_entete = new struct_entete();
v_entete.Titre = "Dotnet-tech.com : Le site sur les technologies .Net";
v_entete.Copyright = "Dotnet-tech.com";
v_entete.Description = "Articles et tutoriels autour des technologies .Net";
v_entete.Lien = "http://www.dotnet-tech.com/";
v_entete.Editeur = "elise.dupont@europe.com";
v_entete.Langue = enum_langue.fr;
v_entete.Generateur = "Elise mano";

// Image
v_entete.Image = new struct_image();
v_entete.Image.Titre = "Dotnet-tech.com";
v_entete.Image.ImageUrl = "http://www.dotnet-tech.com/images/long_banner.gif";
v_entete.Image.Lien = "http://www.dotnet-tech.com/";
v_entete.Image.Hauteur = 70;
v_entete.Image.Largeur = 600;
v_entete.Image.Description = "Le site sur les technologies .Net";

// Envoi des information
builder v_rss = new builder(Page.Response.Output);
v_rss.ConstruitRss(v_entete);
```

Comme souvent, c'est dans le Page_Load que tout ce passe.

Notez le codage du rss en ISO-8859-1 afin de tenir compte des caractères étendus utilisés dans la langue française.

On spécifie un retour de la requête sous forme text/xml.

J'alimente ensuite mes données dans ma structure d'entête. Je préfère utiliser une structure plutôt que de passer n arguments à ma méthode. Si par malheur vous aviez oublié de passer un argument, vous seriez obligé de modifier tous les appels. La structure s'avère plus souple dans le cadre de la maintenance de votre code source.

On crée alors une nouvelle instance de l'objet builder qui sera exposée en réponse http et on appelle la méthode ConstruitRss avec les données de l'entête.

```
// Ajout des publications
struct_detail v_detail = new struct_detail();

v_detail.Auteur = "Auteur : Elise Dupont <elise.dupont@europe.com>";
v_detail.DatePublication = DateTime.Now.AddDays(-1);
v_detail.Titre = "Architecture : Introduction aux Applications Blocks";
v_detail.Url = "http://www.dotnet-tech.com/tutoriels/application-blocks/";
v_detail.Description = @"<div align=""center""> <img src=""http://www.dotnet-te
v_detail.Categorie = "";
v_detail.Sujet = "";
v_rss.AjoutePublication(v_detail);

v_detail = new struct_detail();
v_detail.Auteur = "Auteur : Frédéric Mélantois";
v_detail.DatePublication = DateTime.Now.AddDays(-2);
v_detail.Titre = "ASP.Net : Construisez vos premiers contrôles serveur ASP.Net";
v_detail.Url = "http://www.dotnet-tech.com/tutoriels/controle_serveur/";
v_detail.Description = @"<div align=""center""> <img src=""http://www.dotnet-te
v_detail.Categorie = "";
v_detail.Sujet = "";
v_rss.AjoutePublication(v_detail);

// Termine mon document
v_rss.FinDuDocument();
```

Viens maintenant l'ajout des items ou publications. J'alimente toujours une structure, celle du détail et d'invoque la méthode AjoutePublication().

Ne surtout pas oublier la méthode FinDuDocument() qui va clôturer le flux rss.

Et c'est tout ! Votre flux rss est maintenant publié en un temps record.

3. En Savoir plus

RSS sur XML.FR

<http://xmlfr.org/actualites/tech/000816-0001>

RSS Reader, freeware développé sur le framework .net

<http://www.rssreader.com>

L'auteur : Laurent GEFFROY

<http://www.laurentgeffroy.com>

lgeffroy-at-club-internet.fr