Génération de documents Word avec OpenXML et .NET 3

Après avoir vu quelques uns des concepts de base du standard Open XML et notamment de OPC - Open Packaging Convention - dans un précédent article, attaquons nous aujourd'hui à la génération de document de type traitement de texte avec le langage WordProcessingML du standard ouvert Open XML. Pour ce faire, nous utiliserons les API fourni avec le framework .NET 3 et notamment l'espace de nom System.IO.Packaging mis à notre disposition.

Il y a au moins deux façons de générer des documents WordProcessingML : la première est de partir d'un document existant déjà formaté (et pourquoi pas enrichi de données métiers), la seconde est de créer un document 'from scratch', autrement dit à partir de rien. C'est cette dernière option que nous allons retenir, l'autre méthode sera utilisée dans de prochains articles plus complexes, dans lesquels vous n'aurez pas besoin de vous occuper de créer intégralement un document.

L'exhaustivité des spécifications de ce standard ouvert de l'ECMA facilite grandement l'exploitation des documents Open XML. Vous trouverez toutes les informations relatives au WordProcessingML dans la partie 4:2 du standard.

Introduction au document WordProcessingML

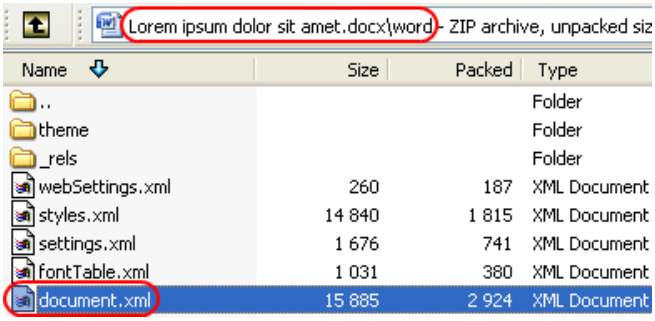
Un document WordProcessingML - un document Office Open XML orienté traitement de texte - est une compilation de deux types d'information :

- les propriétés : les styles, la définition de la numérotation, …
- les *stories* : le document principal, les entêtes, les pieds de page, les commentaires, …

Les propriétés représentent les différents éléments de présentation communs aux *stories* tel que la définition des styles ou de la numérotation des listes

Les *stories* – terme difficilement traductible en français – représentent les régions uniques dans lesquelles un utilisateur peut ajouter ou modifier le contenu. La *story* la plus importante, et la seule dont la présence est obligatoire, est celle du document principal qui décrit le contenu général du document. Comme pour les autres types de document OpenXML (SpreadsheetML pour les documents de tableur et PresentationML pour les documents de présentation), c'est vers la story principale que vous pointerez lorsque vous demanderez la relation de type *http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument* à votre paquet.

Dans la suite de cet article, nous ne traiterons que la story principale. On désignera aussi cette story sous l'appellation de partie principale, en rapport avec la structure physique de stockage OPC des documents Office Open XML. La partie principale d'un document WordProcessingML possède habituellement, bien que cela ne soit qu'un usage, l'URI */word/document.xml* comme le montre l'illustration suivante :



Le format de fichier Open XML utilise bien évidemment le XML pour décrire l'intégralité des documents. Concernant la story principale d'un document WordProcessingML, son contenu se trouve sous la balise *body* selon la structure suivante :

```
<w:document
xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
      <w:body>
            <w:p>
                  <w:r>
                        <w:t>Welcome to Office Open XML world !</w:t>
                  </w:r>
            </w:p>
      </w:body>
</w:document>
```

Listing 1

L'exemple du listing 1 décrit un document possédant un seul et unique paragraphe. Voici le détail de cette structure :

- document : c'est l'élément racine du document principal.
- body : cet élément peut contenir plusieurs paragraphes.
- p : représente un paragraphe, et peut contenir un ou plusieurs *run* et des propriétés dans une balise fille *pPr* communes à tous les runs de ce paragraphe.
- r : représente un run, c'est-à-dire un ensemble contigu de caractères ayant les mêmes propriétés. Tout comme pour les paragraphes, chaque run peut posséder un élément fils *rPr* définissant les propriétés spécifiques de ce run. Nous reviendrons juste après sur ce concept fondamental de Open XML.
- t : le contenu textuel du run parent. Le contenu est principalement du texte non formaté. Le formatage de ce texte est hérité des propriétés du run et des propriétés du paragraphe. Cet élément contient aussi souvent l'attribut *xmlspace="preserve"* afin de bien spécifier la prise en compte des espaces dans le texte.

Contrairement à des formats tel que le HTML, le format Open XML n'accepte pas l'intégration de propriétés de formatage directement à l'intérieur du contenu du texte. C'est pourquoi lorsque vous voudrez décrire le texte "Office **Open** XML" dans un document Open XML, il vous faudra créer au moins trois runs dans le paragraphe : un pour 'Office', un autre pour 'Open' avec la propriété 'gras' activé, et un dernier pour 'XML'.

Document WordProcessingML minimal

La création d'un document WordProcessing ML n'a rien de très compliqué, les spécifications (1:11.2) définissent un ensemble minimal pour le contenu d'un document de ce type. Logiquement, il y a trois parties distinctes qui doivent retenir notre attention : la partie de type de contenu, la partie de relations du paquet et la partie principale du document.

La partie de type de contenu

Cette partie dont l'URI est */[Content_Types].xml* doit au moins décrire le type de contenu des deux autres parties, celui des relations du paquet et du contenu principal) :

```
<Types xmlns="http://schemas.openxmlformats.org/package/2006/content-
types">
      <Default Extension="rels"
ContentType="application/vnd.openxmlformats-package.relationships+xml"/>
      <Default Extension="xml" ContentType="application/vnd.openxmlformats-
officedocument.wordprocessingml.document.main+xml"/>
</Types>
```

Listing 2

La partie de relations du paquet

Cette partie possédant l'URI */_rels/.rels* décrit l'ensemble des relations entre le paquet, c'est-à-dire votre document bureautique, et les différentes parties qui composent le contenu de celui-ci. Etant donné que seule la story principale du document est indispensable, le fichier de relations ne comporte qu'une seule et unique relation vers la partie principale :

```
<Relationships
xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
      <Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/o
fficeDocument" Target="document.xml"/>
</Relationships>
```

Listing 3

Remarque : bien que la partie principale d'un document WordProcessingML possède habituellement l'URI */word/document.xml*, nous créerons cette partie à la racine du paquet pour illustrer la souplesse du mécanisme de relations.

La partie principale

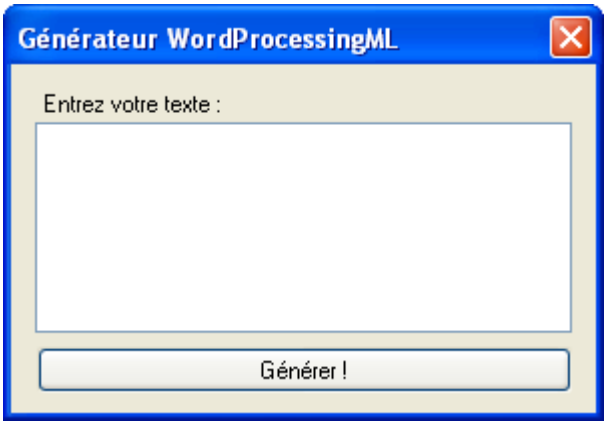
Cette partie décrit le contenu du document, et la structure minimale à observer est relativement simple (un paragraphe vide) :

```
<w:document
xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
      <w:body>
            <w:p/>
      </w:body>
</w:document>
```

Listing 4

Génération avec .NET

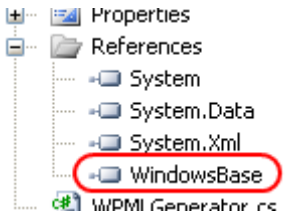
Maintenant que nous avons pris connaissance de la structure minimale à réaliser pour créer un document, nous allons nous adonner au développement d'un petit programme qui vous permettra de générer des documents WordProcessingML (lisible par Word 2007 et Word 2000/XP/2003 avec le pack de compatibilité à l'heure où cet article est écrit, et par OpenOffice, WordPerfect, … quand ces suites supporteront pleinement ce format). Ce programme dont vous trouverez les sources en annexe va vous permettre d'entrer du texte et de générer simplement un document Open XML :



Pour ce faire, nous allons utiliser l'API livrée avec le framework .NET 3 et particulièrement l'espace de noms *System.IO.Packaging*. Le seul prérequis nécessaire pour développer un programme avec cette API est donc la présence du framework .NET 3 sur votre machine.

Configuration de l'environnement

Pour pouvoir utiliser l'espace de noms *System.IO.Packaging*, vous devez importer l'assembly *WindowBase.dll* dans votre projet. L'assembly se trouve par défaut dans le répertoire *c:\Program Files\Reference Assembly\Microsoft\Framework\v3.0*.



Création du package

La création d'un document débute par la création d'un paquet OPC qui contiendra l'ensemble des parties du document, dans notre cas, les parties nécessaires pour créer un document minimal : la partie de type de contenu, la partie de relations du paquet et la partie principale.

L'utilisation des API *System.IO.Packaging* simplifie grandement la manipulation de la structure OPC ; ainsi pour créer un paquet Open XML, seule la ligne suivante est nécessaire :

```
Package pkg = Package.Open("MonDoc.docx", System.IO.FileMode.Create);
```

Listing 5

L'appel à la méthode Open() permet d'ouvrir ou de créer un paquet, et il vous faudra utiliser les méthodes Flush() et Close() pour signifier au paquet d'enregistrer toutes les parties que vous aurez manipulées lorsque vous voudrez fermer/enregistrer votre paquet.

Création de la relation vers la partie principale

Avant de créer la partie principale, nous allons tout d'abord créer le lien vers celle-ci. A noter que l'ordre des manipulations création partie/création lien importe peu.

Cette tâche nécessite tout d'abord la création d'une URI, ici *document.xml* à la racine du paquet, puis d'une relation à l'aide de la méthode CreateRelationship() du paquet nouvellement créé. Les paramètres sont simples et correspondent aux attributs d'une relation : l'URI cible, la portée de la liaison – la partie se trouve dans le paquet : *TargetMode.Internal*, ou alors en dehors du paquet : *TargetMode.External* -, le type de la relation et optionnellement l'identifiant de la relation.

```
// Création de la relation vers la partie principale
Uri docUri = new Uri(@"/document.xml", UriKind.Relative);
pkg.CreateRelationship(docUri, TargetMode.Internal,
"http://schemas.openxmlformats.org/officeDocument/2006/relationships/office
Document", "rId1");
```

Listing 6

Remarque : nous n'avons pas besoin d'ajouter la relation au paquet avec une quelconque méthode, en effet l'appel à CreateRelationship() se charge d'ajouter automatiquement la relation ainsi créée dans le paquet. Dans le même esprit, nous ne créerons pas non plus la partie des relations du paquet, au même titre que la partie contenant les types de contenu : c'est toujours l'API en interne qui se charge de ces tâches élémentaires pour nous.

Création de la partie principale et de son contenu

La création de la partie principale est aussi aisée que l'ajout de la relation au paquet, une seule ligne suffit et suit la même logique :

```
// Création de la partie principale du document
PackagePart mainPart = pkg.CreatePart(docUri,
"application/vnd.openxmlformats-
officedocument.wordprocessingml.document.main+xml");
```

Listing 7

Maintenant que la partie est créée, nous allons lui ajouter du contenu XML tout en respectant la structure que nous avions évoquée précédemment. Pour pouvoir modifier le contenu d'une partie, vous devez tout d'abord récupérer le flux de celle-ci à l'aide de la méthode GetStream() :

```
using (Stream docStream = mainPart.GetStream())
{
      XmlDocument xmlDoc = new XmlDocument();

      // Création de la structure du story principal
      XmlElement documentElem = xmlDoc.CreateElement("w:document",
wNamespace);
      XmlElement bodyElem = xmlDoc.CreateElement("w:body", wNamespace);
      XmlElement paragraphElem = xmlDoc.CreateElement("w:p", wNamespace);
      XmlElement runElem = xmlDoc.CreateElement("w:r", wNamespace);
      XmlElement textElem = xmlDoc.CreateElement("w:t", wNamespace);
      XmlText textElemContent = xmlDoc.CreateTextNode(textContent);
```

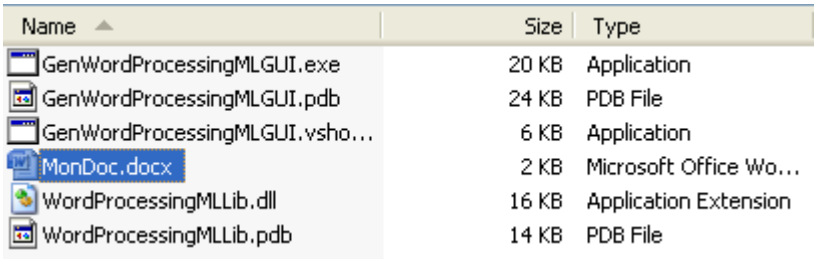
```
        xmlDoc.AppendChild(documentElem);
        documentElem.AppendChild(bodyElem);
        bodyElem.AppendChild(paragraphElem);
        paragraphElem.AppendChild(runElem);
        runElem.AppendChild(textElem);
        textElem.AppendChild(textElemContent);

        xmlDoc.Save(docStream);

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

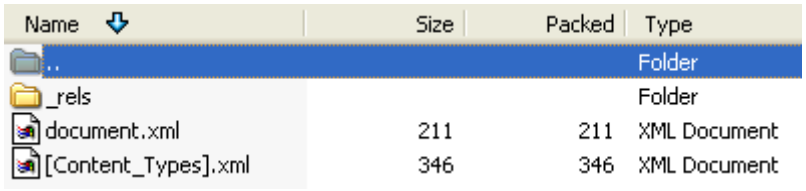
Listing 8

Après exécution du programme vous devriez constater la présence d'un fichier *MonDoc.docx* dans le répertoire de sortie du projet :

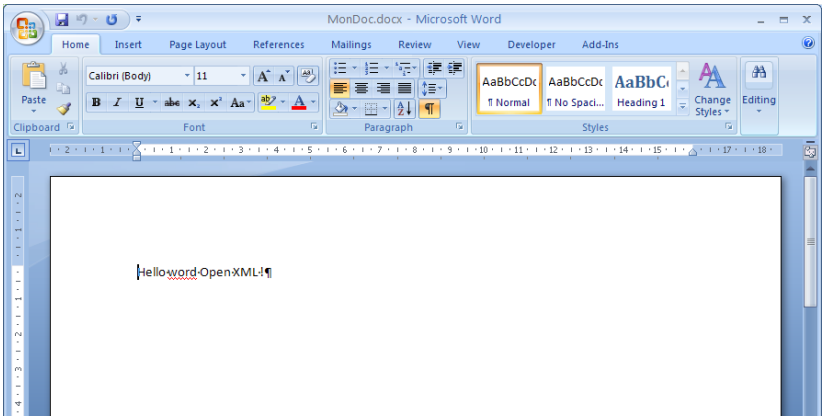


Celui-ci contient l'ensemble des parties minimales pour un document Office Open XML :

- La partie de type de contenu : [Content_Types].xml
- La partie des relations du paquet : _rels/.rels
- La partie principale contenant la story principale : document.xml



Voici ce que vous observerez en ouvrant le document généré dans Word 2007 (avec le texte 'Hello world Open XML !' saisie dans le champ de l'interface) :



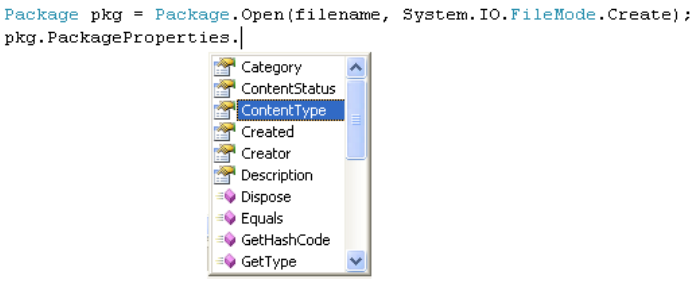
Vous pouvez constater que la génération d'un document basique – même si nous n'avons permis qu'un unique paragraphe homogène – est relativement simple grâce à l'utilisation des API de .NET 3. D'ailleurs, l'exemple que nous avons réalisé n'illustre que très peu des possibilités de cette API.

Néanmoins, vous l'avez sans doute remarqué, nous n'avons pas manipulé le document entièrement à l'aide de cette API. Le contenu même du document a fait l'objet d'une création d'un arbre XML et non d'une manipulation d'objets et d'appels de méthodes. Cela est dû au fait que l'espace de nom *System.IO.Packaging* n'est en réalité qu'une implémentation avancée des spécifications de l'Open Packaging Convention et non des spécifications complètes du format Office Open XML.

Ajouter une partie de propriétés

Après ce petit bémol sur les capacités de l'API de .NET 3 - choix qui néanmoins se justifie en de nombreux points - il serait dommage de ne pas aller un peu plus loin. C'est pourquoi nous allons voir comment changer les propriétés du document : l'auteur, la version, la date de création, …

En réalité rien de plus simple puisque c'est la propriété *PackageProperties* du paquet qui fera le travail à votre place (création de la partie, ajout de la relation et du type de contenu) :

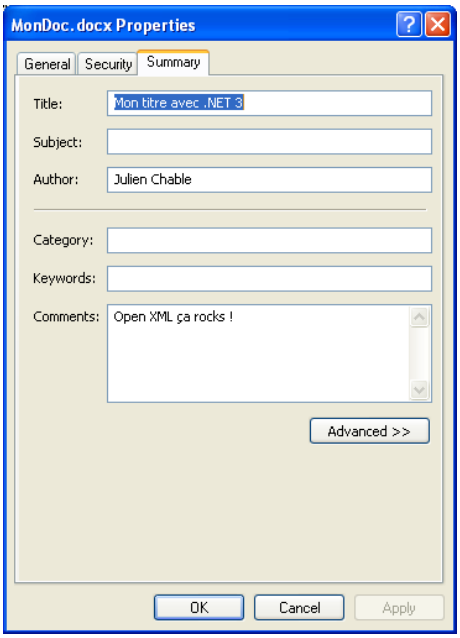


Ainsi, si vous souhaitez ajouter des propriétés ou même les modifier, vous le ferez de la façon suivante :

```
Package pkg = Package.Open(filename, System.IO.FileMode.Open);
pkg.PackageProperties.Modified = DateTime.Now;
pkg.PackageProperties.Creator = createur;
pkg.PackageProperties.Description = description;
pkg.PackageProperties.Title = title;
pkg.Flush();
pkg.Close();
```

Listing 9

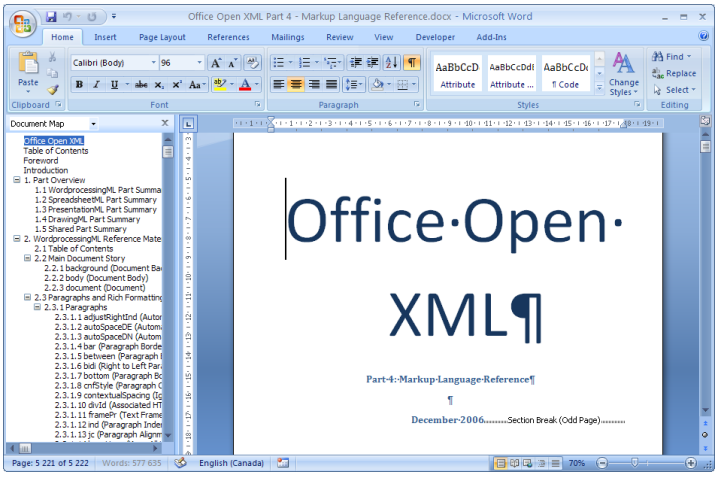
Après l'exécution du listing 9, vous pourrez observer le succès de l'opération avec la boîte des propriétés du document :



Exploiter les spécifications pour développer efficacement

Open XML étant un standard de l'ECMA (ECMA-376 : Office Open XML File Format), vous devrez tout d'abord récupérer les spécifications sur le site officiel (cf le lien dans les références en fin d'article). Vous constaterez que dans un souci de permettre aux développeurs d'exploiter rapidement et avec le plus de facilité possible les spécifications, celles-ci sont découpées en 5 parties bien distinctes.

En tant que développeur maintenant averti, nous allons utiliser la documentation de 'Référence des schémas' d'Open XML, - soit la partie 4 - qui a elle seule totalise prés de 5200 pages. Pour travailler efficacement, je vous encourage à prendre quelques secondes - voire minutes selon votre machine - pour ouvrir le document sous Word 2007 plutôt que d'utiliser sa version PDF, vous gagnerez ainsi du temps précieux à la longue.



Comme exemple d'utilisation de la spécification, nous allons tenter de mettre une couleur de fond à notre document.

Avant tout chose, il est important de chercher dans le sommaire – ou via la carte du document de Word 2007 – la partie réservée à WordprocessingML (§2). Logiquement vous savez que cette

propriété doit se trouver dans la story principal du document, et plus précisément sous la balise *document* : et c'est effectivement le cas !

Une fois situé dans le paragraphe de la spécification concernant l'élément *background* (§2.2.1) fils de l'élément *document* (§2.2.3), vous allez pouvoir trouver une description détaillé et un exemple général de l'utilisation de cette balise. Les autres informations à savoir, les balises parents et enfants, et les attributs donnent autant de possibilité de pouvoir naviguer directement par hyperliens vers les paragraphes des concernés.

Nous voulons spécifier un fond uni, et pour cela l'attribut *color* va nous être très utile. La spécification définie très précisément cet élément – et son format RRGGBB -, et la façon de l'utiliser avec de surcroît un exemple à la clef : une spécification exhaustive faites pour les développeurs !

Voici le code XML que nous allons rajouter à la suite de l'élément *document* :

```
<w:background w:color="2C34FF"/>
```

Listing 10

Avant d'exécuter les sources associée à cet article, sachez que pour que cette propriété soit effective, vous devrez spécifier son activation dans la partie de configuration 'settings.xml' avec l'élément '*displayBackgroundShape*' (§2.15.1.25).

Conclusion

Cet article a été l'occasion d'une part de montrer un exemple basique, mais concret, de génération 'from scratch' d'un document WordProcessingML ; et d'une autre part de démontrer la facilité d'utilisation de l'API incluse dans .NET 3.

Evidemment, plus le formatage de votre document sera complexe et riche, et plus la structure XML à générer sera complexe. La manipulation des documents Open XML, que ce soit du WordProcessingML, du SpreadsheetML ou du PresentationML, est grandement simplifiée avec l'utilisation de .NET 3. Et même si l'on regrette de ne pas pouvoir manipuler les documents directement de façon orienté objet, .NET 3 et les spécifications du standard ouvert Open XML répondent admirablement aux besoins des développeurs et du marché.

Références

- Lien vers le téléchargement du framework .NET 3 : http://www.microsoft.com/downloads/details.aspx?FamilyId=19E21845-F5E3-4387-95FF-66788825C1AF&displaylang=en
- Lien vers le téléchargement du SDK Windows Vista et .NET 3 : http://www.microsoft.com/downloads/details.aspx?FamilyId=117ECFD3-98AD-4D67-87D2-E95A8407FA86&displaylang=en
- Lien vers le téléchargement des extensions pour Visual Studio 2005 : http://www.microsoft.com/downloads/details.aspx?FamilyID=935aabf9-d1d0-4fc9-b443-877d8ea6eab8&DisplayLang=en

- MSDN France section OpenXML : http://www.microsoft.com/france/msdn/office/openxml/default.mspx
- Site OpenXMLDeveloper : http://openxmldeveloper.org/
- Spécifications *Open Packaging Convention* (également inclut dans les spécifications de Open XML) : http://www.microsoft.com/whdc/xps/xpspkg.mspx
- Les spécifications Office Open XML File Format ECMA 376 : http://www.ecma-international.org/publications/standards/Ecma-376.htm

Grégory Renard
CTO Wygwam

Julien Chable
Développeur/Consultant