Artículo
Bernardo Linarez · 9 jul, 2019 Lectura de 24 min

Análisis de archivos docx mediante XSLT

¡Hola a tod@s!

La tarea de administrar documentos de Office (documentos docx, tablas xlsx y presentaciones pptx), es bastante complicada. En este artículo se ofrece una manera para analizar, crear y editar documentos usando únicamente XSLT y ZIP.

¿Por qué? docx es el formato más popular para documentos, por lo que la habilidad para generar y analizar este formato siempre puede ser de utilidad.

Resolver el problema con un formato de una biblioteca prefabricada puede ser problemático por varias razones, entre las cuales tenemos:

- puede que la biblioteca no exista
- no necesita otra caja negra en su proyecto
- restricciones en la biblioteca: plataformas, funciones, etc.
- licencias
- velocidad de procesamiento

En este artículo, solamente utilizaré las herramientas básicas para trabajar con los documentos de tipo docx.

La estructura docx

¿Qué es un documento de tipo docx? Un archivo docx es un archivo zip, el cual físicamente contiene 2 tipos de archivos:

- archivos xml con extensiones o
- archivos multimedia (imágenes, etc.)

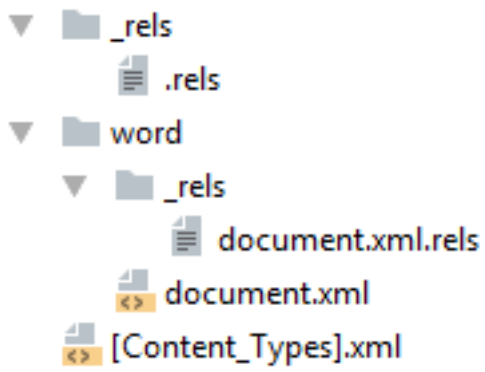
Y lógicamente tiene 3 tipos de elementos:

- Los tipos de contenido: una lista con algunos de los archivos multimedia (por ejemplo, png) que se utilizaron en el documento y en las secciones del documento (por ejemplo, el documento, el encabezado de la página).
- Secciones: cada una de las secciones por separado del documento. En los documentos de tipo docx esto se identifica como , lo cual incluye a los documentos xml y a los archivos multimedia.
- Las relaciones que identifican a las diferentes secciones del documento para vincularlas (por ejemplo, la conexión que hay entre las secciones del documento y el encabezado de la página), las partes externas también se definen aquí (por ejemplo, los hipervínculos).

Esto se describe a detalle en el **ECMA-376: Formatos de archivos en Open Office XML**, la parte principal es el **documento PDF** que consta de 5,000 páginas, además existen 2,000 páginas con contenido adicional.

Tamaño mínimo de los archivos docx

El archivo docx más sencillo después de descomprimirlo se ve de la siguiente manera:



Echemos un **vistazo** en su interior.

[ContentTypes].xml

Este archivo se encuentra en la raíz del documento y hace una lista de todos los tipos de extensiones MIME que están presentes en el documento:

```
<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/xml"/>
  <Override PartName="/word/document.xml"
      ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml"/>
</Types>
```

rels/.rels

Es la lista principal de las secciones del documento. En este caso, solamente existe un vínculo definido, matching rId1 identifier y word/document.xml file, y es el cuerpo principal del documento.

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

word/document.xml

Es el contenido principal del documento.

```
<w:document xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
   <w:body>
      <w:p w:rsidR="005F670F" w:rsidRDefault="005F79F5">
         <w:r>
            <w:t>Test</w:t>
         </w:r>
         <w:bookmarkStart w:id="0" w:name="GoBack"/>
         <w:bookmarkEnd w:id="0"/>
      </w:p>
      <w:sectPr w:rsidR="005F670F">
         <w:pgSz w:w="12240" w:h="15840"/>
         <w:pgMar w:top="1440" w:right="1440" w:bottom="1440" w:left="1440"
                  w:header="720" w:footer="720" w:gutter="0"/>
         <w:cols w:space="720"/>
         <w:docGrid w:linePitch="360"/>
      </w:sectPr>
   </w:body>
</w:document>
```

Aquí se observan las partes:

- <w:document> - es el documento en sí mismo
- <w:body> - es el cuerpo del documento
- <w:p> - indica un párrafo
- <w:r> - es la ejecución (de un fragmento) del texto
- <w:t> - es el texto en sí mismo
- <w:sectPr> - es una descripción de la página

Cuando abra este documento en un editor de texto, verá un documento con una sola palabra .

word/rels/document.xml.rels

Contiene una lista de los vínculos que hay en la sección . El nombre del vínculo de un archivo se crea a partir del nombre de la sección del documento a la que pertenece, mediante la extensión rels. Una carpeta con el vínculo de un archivo se llama rels, y se coloca en el mismo nivel que la sección con la cual se relaciona. En word/document.xml no existe ningún vínculo, por lo que el archivo está vacío:

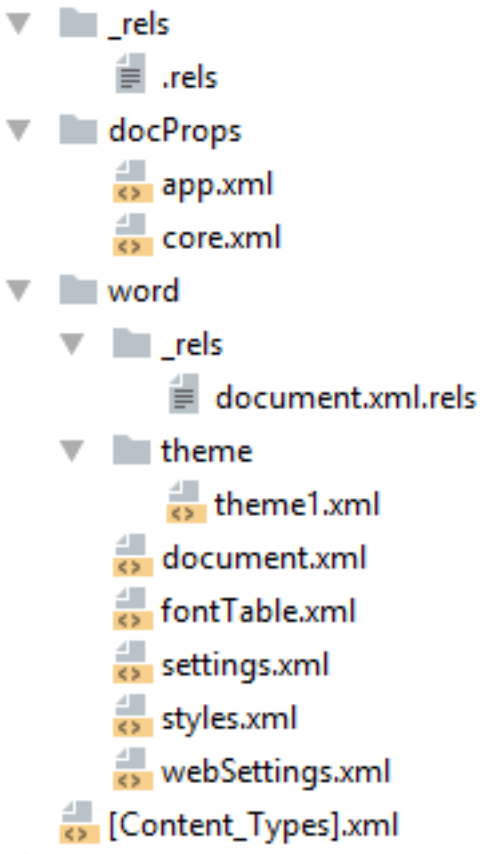
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

</Relationships>

Incluso si no existen vínculos, este archivo debe existir.

docx y Microsoft Word

El formato **docx** que se generó con Microsoft Word o con cualquier otro editor contiene **varios archivos adicionales**.



Estos archivos son:

- docProps/core.xml: son los metadatos básicos del documento de acuerdo con **Open Packaging Conventions** y Dublin Core **[1]**, **[2]**.
- docProps/app.xml: **contiene la información general sobre el documento**: el número de páginas, las palabras, los caracteres, la aplicación en la que se creó el documento, etc.
- word/settings.xml - **es la configuración del documento**.
- word/styles.xml - **los estilos** que se aplicaron al documento. Separa los datos de la visualización.
- word/webSettings.xml: **la configuración** para visualizar HTML y para convertir los documentos en HTML.
- word/fontTable.xml: **es una lista** de las fuentes en el documento.

- word/theme1.xml - **el tema** (consiste en esquemas de color, las fuentes y el formato).

Por lo general, entre más complejo sea un documento tendrá más secciones.

La ingeniería inversa de docx

La primera tarea es averiguar cómo podemos guardar cualquier fragmento del documento en xml, y después crear (o analizar) dichos documentos por nuestra cuenta. Para esto necesitamos:

- Un compresor Zip
- Una biblioteca para formatear archivos XML (Word arroja los archivos XML sin sangrías, todo en una sola línea)
- Una herramienta para ver las diferencias entre los archivos, yo utilicé git y de TortoiseGit

Herramientas

- Para Windows: **zip**, **unzip**, **libxml2**, **git**, **TortoiseGit**
- Para Linux: apt-get, install zip, unzip, libxml2, libxml2-utils git

Además, los **scripts** serán necesarios para el almacenamiento automático y dar formato en XML. Cómo utilizarlo en Windows:

```
- descomprime el archivo del documento en la carpeta dir y los formatos xml
- comprime el archivo del documento en la carpeta dir
```

La manera de utilizarlo en Linux es similar, pero se emplea ./unpack.sh en lugar de unpack, y pack se convierte en ./pack.

Cómo utilizarlo

Para buscar cambios:

1. Creé un archivo docx vacío en el editor.
2. Descomprímalo dentro de la nueva carpeta, para ello utilice unpack.
3. Asigne una nueva carpeta.
4. Agregue las cosas que necesite (hipervínculos, tablas, etc.) al documento desde el paso 1.
5. Descomprima el archivo modificado dentro de una carpeta que ya exista.
6. Analice las diferencias, elimine los cambios innecesarios (vínculos hacia permutaciones, el orden del namespace, etc.).
7. Comprima el documento en la carpeta y compruebe que puede abrirlo nuevamente.
8. Asigne un destino para la carpeta que modificó.

Ejemplo 1. Texto en negritas

Como en el primer ejemplo buscaremos una etiqueta que haga que el texto aparezca en negritas.

1. Creé un documento bold.docx en negritas con el texto Test en letra normal (no en negritas).
2. Descomprimirlo: descomprima bold.dox con la letra en negritas.
3. **Asigne un destino para el resultado**.
4. Haga que el texto Test aparezca en negritas.
5. Descomprimirlo: descomprima el archivo bold.docx con la letra en negritas.
6. Al principio, las diferencias eran las siguientes:

Path	Extension	Status	Lines added	Lines removed
Modified Files				
bold/docProps/app.xml	.xml	Modified	1	1
bold/docProps/core.xml	.xml	Modified	2	2
bold/word/document.xml	.xml	Modified	5	3
bold/word/settings.xml	.xml	Modified	2	0

Analicemos esto detalladamente:

docProps/app.xml

@@ -1,9 +1,9 @@

- <TotalTime>0</TotalTime>

+ <TotalTime>1</TotalTime>

No es necesario cambiar el tiempo.

docProps/core.xml

@@ -4,9 +4,9 @@

- <cp:revision>1</cp:revision>

+ <cp:revision>2</cp:revision>

<dcterms:created xsi:type="dcterms:W3CDTF">2017-02-07T19:37:00Z</dcterms:created>

- <dcterms:modified xsi:type="dcterms:W3CDTF">2017-02-07T19:37:00Z</dcterms:modified>

+ <dcterms:modified xsi:type="dcterms:W3CDTF">2017-02-08T10:01:00Z</dcterms:modified>

La versión del documento y la fecha de las modificaciones no son relevantes.

word/document.xml

@@ -1,24 +1,26 @@

<w:body>

- <w:p w:rsidR="0076695C" w:rsidRPr="00290C70" w:rsidRDefault="00290C70">

```
+    <w:p w:rsidR="0076695C" w:rsidRPr="00F752CF" w:rsidRDefault="00290C70">
       <w:pPr>
         <w:rPr>
+          <w:b/>
           <w:lang w:val="en-US"/>
         </w:rPr>
       </w:pPr>
-      <w:r>
+      <w:r w:rsidRPr="00F752CF">
         <w:rPr>
+          <w:b/>
           <w:lang w:val="en-US"/>
         </w:rPr>
         <w:t>Test</w:t>
       </w:r>
       <w:bookmarkStart w:id="0" w:name="GoBack"/>
       <w:bookmarkEnd w:id="0"/>
     </w:p>
-    <w:sectPr w:rsidR="0076695C" w:rsidRPR="00290C70">
+    <w:sectPr w:rsidR="0076695C" w:rsidRPR="00F752CF">
```

Los cambios en w:rsidR no son necesarios, esta información es útil para Microsoft Word. Aquí está el cambio más importante:

```
         <w:rPr>
+          <w:b/>
```

en el párrafo con Test. Aparentemente el elemento <w:b/> hace que el texto se marque en negritas. Guardemos este cambio y vamos a revertir todos los demás.

word/settings.xml

```
@@ -1,8 +1,9 @@
+  <w:proofState w:spelling="clean"/>
```

@@ -17,10 +18,11 @@

+ <w:rsid w:val="00F752CF"/>

Esto no contiene nada que tenga relación con el texto en negritas. Vamos a revertirlo.

7. Comprima una carpeta que tenga algún cambio relevante (añada <w:b/>), compruebe que puede abrir el **documento** y que este se muestra conforme a lo previsto. 8 **Asigne un destino para el archivo con el cambio**.

Ejemplo 2. Pie de página

Pasemos a un ejemplo más complejo: cómo añadir un pie de página en un documento. **Primera asignación**. Añada el texto ‘ 123 ’ en el pie de página y descomprima el documento. La primera diferencia es la siguiente:

Path	Extension	Status	Lines added	Lines removed
Modified Files				
footer/[Content_Types].xml	.xml	Modified	3	0
footer/docProps/app.xml	.xml	Modified	1	1
footer/docProps/core.xml	.xml	Modified	2	2
footer/word/_rels/document.xml.rels	.rels	Modified	5	2
footer/word/document.xml	.xml	Modified	1	0
footer/word/settings.xml	.xml	Modified	11	0
footer/word/styles.xml	.xml	Modified	44	0
Not Versioned Files				
footer/word/endnotes.xml	.xml	Unknown		
footer/word/footer1.xml	.xml	Unknown		
footer/word/footnotes.xml	.xml	Unknown		

Revierta inmediatamente los cambios en docProps/app.xml y en docProps/core.xml, de la misma forma como se hizo en el primer ejemplo.

[Content_Types].xml

@@ -4,10 +4,13 @@

<Default Extension="xml" ContentType="application/xml"/>

<Override PartName="/word/document.xml" ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml"/>

+ <Override PartName="/word/footnotes.xml" ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.footnotes+xml"/>

+ <Override PartName="/word/endnotes.xml" ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.endnotes+xml"/>

+ <Override PartName="/word/footer1.xml" ContentType="application/vnd.openxmlformats-

officedocument.wordprocessingml.footer+xml"/>

El pie de página claramente se parece a lo que necesitamos. Entonces, ¿qué debemos hacer con las notas al pie de página y las notas finales? ¿Son necesarios para añadir un pie de página, o solamente son un producto secundario que se creó al mismo tiempo? Discernir la respuesta no siempre es fácil, aquí están los enfoques más importantes:

- Analice los cambios: ¿están conectados entre sí?
- Haga experimentos
- Por último y si todavía no entiende lo que está pasando, esto es lo que debe hacer:

word/rels/document.xml.rels

La primera diferencia es la siguiente:

@@ -1,8 +1,11 @@

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

+ <Relationship Id="rId5" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/theme" Target="theme/theme1.xml"/>

<Relationship Id="rId3" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/webS..." Target="webSettings.xml"/>

+ <Relationship Id="rId4" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/font..." Target="fontTable.xml"/>

<Relationship Id="rId2" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/sett..." Target="settings.xml"/>

<Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/styles" Target="styles.xml"/>

- <Relationship Id="rId5" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/theme" Target="theme/theme1.xml"/>

- <Relationship Id="rId4" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/font..." Target="fontTable.xml"/>

+ <Relationship Id="rId6" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/footer" Target="footer1.xml"/>

+ <Relationship Id="rId7" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/endn..." Target="endnotes.xml"/>

+ <Relationship Id="rId8" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/foot..." Target="footnotes.xml"/>

</Relationships>

Como podemos ver, algunos de los cambios se deben al hecho de que Word cambió el orden de los vínculos, restauremos el orden y hagamos que las diferencias sean más pequeñas:

```
@@ -3,6 +3,9 @@

+  <Relationship Id="rId6" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/footer" Target="footer1.xml"/>

+  <Relationship Id="rId7" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/endn..." Target="endnotes.xml"/>

+  <Relationship Id="rId8" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/foot..." Target="footnotes.xml"/>
```

ya aparecen de nuevo los pies de página, las notas al pie de página y las notas finales. Todos ellos están conectados con el documento principal, vamos a echarle un vistazo:

word/document.xml

```
@@ -15,10 +15,11 @@

      </w:r>

      <w:bookmarkStart w:id="0" w:name="GoBack"/>

      <w:bookmarkEnd w:id="0"/>

    </w:p>

    <w:sectPr w:rsidR="0076695C" w:rsidRPr="00290C70">

+     <w:footerReference w:type="default" r:id="rId6"/>

      <w:pgSz w:w="11906" w:h="16838"/>

      <w:pgMar w:top="1134" w:right="850" w:bottom="1134" w:left="1701" w:header="708" w:footer="708" w:gutter="0"/>

      <w:cols w:space="708"/>

      <w:docGrid w:linePitch="360"/>

    </w:sectPr>
```

Para variar, aquí solamente se realizan los cambios necesarios, mediante un vínculo explícito al pie de página en **sectPr**. No hay vínculos para las notas al pie de página ni para las notas finales en el documento, por lo que asumiremos que estos no son necesarios.

word/settings.xml

```
@@ -1,19 +1,30 @@

 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>

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

```
  <w:zoom w:percent="100"/>
+ <w:proofState w:spelling="clean"/>
  <w:defaultTabStop w:val="708"/>
  <w:characterSpacingControl w:val="doNotCompress"/>
+ <w:footnotePr>
+   <w:footnote w:id="-1"/>
+   <w:footnote w:id="0"/>
+ </w:footnotePr>
+ <w:endnotePr>
+   <w:endnote w:id="-1"/>
+   <w:endnote w:id="0"/>
+ </w:endnotePr>
  <w:compat>
    <w:compatSetting w:name="compatibilityMode" w:uri="http://schemas.microsoft.com/office/word" w:val="15"/>
    <w:compatSetting w:name="overrideTableStyleFontSizeAndJustification"
w:uri="http://schemas.microsoft.com/office/word" w:val="1"/>
    <w:compatSetting w:name="enableOpenTypeFeatures" w:uri="http://schemas.microsoft.com/office/word"
w:val="1"/>
    <w:compatSetting w:name="doNotFlipMirrorIndents" w:uri="http://schemas.microsoft.com/office/word"
w:val="1"/>
    <w:compatSetting w:name="differentiateMultirowTableHeaders"
w:uri="http://schemas.microsoft.com/office/word" w:val="1"/>
  </w:compat>
  <w:rsids>
    <w:rsidRoot w:val="00290C70"/>
+   <w:rsid w:val="000A7B7B"/>
+   <w:rsid w:val="001B0DE6"/>
```

Las funciones Settings hacen una lista de los vínculos para las notas al pie de página y las notas finales, esto probablemente las añadirá en el documento. Tenga en cuenta que el pie de página no aparece aquí.

word/styles.xml

```
@@ -480,6 +480,50 @@
      <w:rFonts w:ascii="Times New Roman" w:hAnsi="Times New Roman"/>
      <w:b/>
      <w:sz w:val="28"/>
    </w:rPr>
  </w:style>
+  <w:style w:type="paragraph" w:styleId="a4">
+    <w:name w:val="header"/>
+    <w:basedOn w:val="a"/>
+    <w:link w:val="a5"/>
+    <w:uiPriority w:val="99"/>
+    <w:unhideWhenUsed/>
+    <w:rsid w:val="000A7B7B"/>
+    <w:pPr>
+      <w:tabs>
+        <w:tab w:val="center" w:pos="4677"/>
+        <w:tab w:val="right" w:pos="9355"/>
+      </w:tabs>
+      <w:spacing w:after="0" w:line="240" w:lineRule="auto"/>
+    </w:pPr>
+  </w:style>
+  <w:style w:type="character" w:customStyle="1" w:styleId="a5">
+    <w:name w:val="Верхний колонтитул Знак"/>
+    <w:basedOn w:val="a0"/>
+    <w:link w:val="a4"/>
+    <w:uiPriority w:val="99"/>
+    <w:rsid w:val="000A7B7B"/>
+  </w:style>
+  <w:style w:type="paragraph" w:styleId="a6">
```

```
+    <w:name w:val="footer"/>
+    <w:basedOn w:val="a"/>
+    <w:link w:val="a7"/>
+    <w:uiPriority w:val="99"/>
+    <w:unhideWhenUsed/>
+    <w:rsid w:val="000A7B7B"/>
+    <w:pPr>
+      <w:tabs>
+        <w:tab w:val="center" w:pos="4677"/>
+        <w:tab w:val="right" w:pos="9355"/>
+      </w:tabs>
+      <w:spacing w:after="0" w:line="240" w:lineRule="auto"/>
+    </w:pPr>
+  </w:style>
+  <w:style w:type="character" w:customStyle="1" w:styleId="a7">
+    <w:name w:val="Нижний колонтитул Знак"/>
+    <w:basedOn w:val="a0"/>
+    <w:link w:val="a6"/>
+    <w:uiPriority w:val="99"/>
+    <w:rsid w:val="000A7B7B"/>
+  </w:style>
 </w:styles>
```

Nos interesan los cambios de estilo, pero solamente si lo que buscamos es hacer cambios en el estilo. En este caso, el cambio puede revertirse.

word/footer1.xml

Eche un vistazo al pie de página en sí (algunos namespaces se omiten para mejorar la legibilidad, pero deben estar presentes en el documento):

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

```
 <w:p w:rsidR="000A7B7B" w:rsidRDefault="000A7B7B">
   <w:pPr>
     <w:pStyle w:val="a6"/>
   </w:pPr>
   <w:r>
     <w:t>123</w:t>
   </w:r>
 </w:p>
</w:ftr>
```

Aquí está el texto del pie de página: '123'. Como en este ejemplo no registraremos los cambios de estilo, necesitamos eliminar el vínculo <w:pStyle w:val="a6"/>.

El análisis de todos los cambios da como resultado las siguientes suposiciones:

- Las notas al pie de página y las notas finales no son necesarias
- Necesitamos añadir un archivo word/footer1.xml
- En [Content_Types].xml necesitamos añadir un vínculo hacia el pie de página
- En word/_rels/document.xml.rels necesitamos añadir un vínculo hacia el pie de página
- En word/document.xml para colocar una etiqueta en <w:sectPr> necesitamos añadir <w:footerReference>

Estas suposiciones reducen las diferencias con respecto a este conjunto de cambios:

Path	Extension	Status	Lines added	Lines removed
Modified Files				
footer/[Content_Types].xml	.xml	Modified	1	0
footer/word/_rels/document.xml.rels	.rels	Modified	1	0
footer/word/document.xml	.xml	Modified	1	0
Not Versioned Files				
footer/word/footer1.xml	.xml	Unknown		

Entonces comprima **el documento** y ábralo. Si todo el procedimiento se realizó correctamente, se abrirá el documento y habrá un pie de página con el texto ‘ 123 ’ . Y aquí puede observar la **asignación** final del documento.

Así, el proceso para detectar los cambios se reduce para permitir la determinación de un conjunto de cambios mínimo, lo cual es suficiente para lograr el resultado deseado.

Práctica

Después de que encontremos el cambio que es necesario, lo más lógico es continuar a la siguiente etapa, que podría ser cualquiera de las siguientes:

- Crear un docx
- Analizar un docx
- Convertir un docx

Para ello, necesitamos a **XSLT** y a **XPath**.

Escribiremos una conversión bastante simple, vamos a reemplazar o añadir el pie de página en un documento.

Algoritmo

El algoritmo se ve de la siguiente manera:

1. Desempaquete el documento
2. Añada el que será nuestro pie de página
3. Añada un vínculo para [Content_Types].xml y word/_rels/document.xml.rels
4. En word/document.xml para cada etiqueta <w:sectPr> añada o reemplace la etiqueta <w:footerReference> con una referencia para nuestro pie de página
5. Comprima el documento.

Comencemos.

Descomprimir

En Caché ObjectScript es posible ejecutar los comandos del sistema operativo para la función **$zf(-1, oscommand)**. Llame a la función unzip para descomprimir el documento mediante **wrapper over $zf(-1)**:

```
/// Utilice %3 (unzip) para descomprimir el archivo %1 en la carpeta %2

Parameter UNZIP = "%3 %1 -d %2";


/// Descomprima el archivo fuente en la carpeta targetDir

ClassMethod executeUnzip(source, targetDir) As %Status

{

    set timeout = 100

    set cmd = $$$FormatText(..#UNZIP, source, targetDir, ..getUnzip())

    return ..execute(cmd, timeout)

}
```

Creación del archivo para el pie de página

La entrada recibe el texto para el pie de página, lo escribiremos en un archivo.xml:

```
<xml>TEST</xml>
```

En XSLT (archivo footer.xsl) crearemos un pie de página con texto desde la etiqueta xml (algunos namespaces se omitieron, aquí se muestra la **lista completa**):

```
<xsl:stylesheet
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns="http://schemas.openxmlformats.org/package/2006/relationships" version="1.0">
  <xsl:output method="xml" omit-xml-declaration="no" indent="yes" standalone="yes"/>
  <xsl:template match="/">

    <w:ftr xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
      <w:p>
        <w:r>
          <w:rPr>
            <w:lang w:val="en-US"/>
          </w:rPr>
          <w:t>
            <xsl:value-of select="//xml/text()"/>
          </w:t>
        </w:r>
      </w:p>
    </w:ftr>
  </xsl:template>
</xsl:stylesheet>
```

Llame al **convertidor XSLT**:

```
do ##class(%XML.XSLT.Transformer).TransformFile("in.xml", "footer.xsl", footer0.xml")
```

El resultado es el archivo del pie de página footer0.xml:

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

```
    <w:p>
      <w:r>
        <w:rPr>
          <w:lang w:val="en-US"/>
        </w:rPr>
        <w:t>TEST</w:t>
      </w:r>
    </w:p>
</w:ftr>
```

Cómo añadir un vínculo para el pie de página en una lista de vínculos del documento principal

Por lo general, no existe un vínculo para el ID rId0. Sin embargo, puede utilizar XPath para obtener el ID que no existe. Añada un vínculo para el archivo footer0.xml mediante el ID rId0 en word/rels/document.xml.rels:

```
<xsl:stylesheet  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://schemas.openxmlformats.org/package/2006/relationships"  version="1.0">
    <xsl:output method="xml" omit-xml-declaration="yes" indent="no"  />
    <xsl:param name="new">
      <Relationship
        Id="rId0"
        Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/footer"
        Target="footer0.xml"/>
    </xsl:param>

    <xsl:template match="/*">
      <xsl:copy>
        <xsl:copy-of select="$new"/>
        <xsl:copy-of select="@* | node()"/>
      </xsl:copy>
    </xsl:template>
</xsl:stylesheet>
```

Cómo especificar los vínculos en el documento

Después, es necesario que en cada etiqueta <w:sectPr> se añada la etiqueta <w:footerReference> o se reemplace un vínculo dentro de ella con el vínculo hacia nuestro pie de página. **Al parecer**, cada etiqueta <w:sectPr> puede tener 3 etiquetas <w:footerReference> diferentes, una para la primera página, otra para las páginas pares y la última para todas las otras partes del texto:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

xmlns:r="http://schemas.openxmlformats.org/officeDocument/2006/relationships"

xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main"

version="1.0">

   <xsl:output method="xml" omit-xml-declaration="yes" indent="yes" />

   <xsl:template match="//@* ¦ //node()">

      <xsl:copy>

         <xsl:apply-templates select="@*"/>

         <xsl:apply-templates select="node()"/>

      </xsl:copy>

   </xsl:template>

   <xsl:template match="//w:sectPr">

      <xsl:element name="{name()}" namespace="{namespace-uri()}">

         <xsl:copy-of select="./namespace::*"/>

         <xsl:apply-templates select="@*"/>

         <xsl:copy-of select="./*[local-name() != 'footerReference']"/>

         <w:footerReference w:type="default" r:id="rId0"/>

         <w:footerReference w:type="first" r:id="rId0"/>

         <w:footerReference w:type="even" r:id="rId0"/>

      </xsl:element>

   </xsl:template>

</xsl:stylesheet>
```

Cómo añadir un pie de página en [Content_Types].xml

En [Content_Types].xml añada información para establecer que /word/footer0.xml es una aplicación: application/vnd.openxmlformats-officedocument.wordprocessingml.footer+xml:

```
<xsl:stylesheet  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://schemas.openxmlformats.org/package/2006/content-types"  version="1.0">

   <xsl:output method="xml" omit-xml-declaration="yes" indent="no"  />

   <xsl:param name="new">

      <Override

       PartName="/word/footer0.xml"

       ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.footer+xml"/>

   </xsl:param>


   <xsl:template match="/*">

      <xsl:copy>

         <xsl:copy-of select="@* | node()"/>

         <xsl:copy-of select="$new"/>

      </xsl:copy>

   </xsl:template>

</xsl:stylesheet>
```

Como resultado de todo lo anterior

El código completo está **disponible en GitHub**. Funciona de la siguiente manera:

```
do ##class(Converter.Footer).modifyFooter("in.docx", "out.docx", "TEST")
```

En donde:

- in.docx - es el documento original
- out.docx - es el documento final
- TEST - el texto que se añadió al pie de página

Conclusiones

Utilizando únicamente XSLT y ZIP, puede trabajar con éxito en los documentos docx, las tablas xlsx y las presentaciones pptx.

Preguntas abiertas

1. ¿Usted genera o analiza los archivos xlsx, docx? Si es así, ¿cómo lo hace?
2. Estoy buscando la versión 5 de los archivos con esquemas XSD en el ECMA-367 y algunos comentarios acerca de ellos. La quinta versión de XSD está disponible para su descarga en el sitio web de ECMA. Pero es difícil comprenderla si no hay algún comentario. La segunda versión del XSD

está disponible con comentarios.

Enlaces

- **ECMA-376**
- **descripción de los archivos de tipo docx**
- **Artículo detallado sobre docx**
- **Repositorio con scripts**
- **Repositorio con un editor para pies de página**

#Caché #Ensemble #InterSystems IRIS #InterSystems IRIS for Health

---