DESARROLLO BI-PUBLISHER LIBRA CLIENTES

Índice

1. Casos prácticos

En este apartado explicaremos el código necesario para realizar ciertas funcionalidades avanzadas que nos proporciona la herramienta RTF Template Builder.

1.1 Recomendaciones diseño del informe

1. Se recomienda agregar los campos repetición siguiendo la estructura del árbol, de modo que se arrastraría el grupo de la cabecera, y dentro sus hijos de forma iterativa. Tras esto ya podremos comenzar con nuestra maquetación, aplicando las tablas de Word.
2. Se recomienda así mismo el uso de los "**TextField**" que residen bajo la pestaña de "**Desarrollador**". Mediante esto podemos añadir una etiqueta en la cual se puede insertar nuestro código de modo que sea mucho más sencillo su lectura y posterior maquetación.
    a. En los literales asignados, se recomienda añadir espacios en blanco, lo cual hará nos separará el código y nos evitará añadir espacios dentro del informe el cual si se renderizarían.
    b. Se recomienda también distinguir mediante colores estas secciones de código, de modo que la apertura y cierre de un Grupo de Repetición tendrían el mismo color.

1.2 Insertar Atributo

Para visualizar un campo de un DataSet bastará con usar la opción campo y arrastrarlo dentro del Foreach que contiene la apertura de este grupo, en caso contrario podría darse el caso de:

- No se renderiza
- Se muestra el valor de un campo con mismo nombre del DataSet sobre el que se ha arrastrado

Por lo cual es muy importante mantener la coherencia dentro de la estructura del Modelo de datos.

Código generado:

```
<?ESTADO?>
```

1.2 Rutas XPATH y Contexto

Las rutas XPATH se refieren a las rutas de los atributos que se encuentran dentro del árbol del fichero XML de ejemplo generado y que podemos acceder a él mediante estas rutas.
El contexto se refiere a las distintas partes de un documento RTF o Word.

1.2.1 Rutas XPATH

En ocasiones necesitaremos acceder a atributos que no pertenecen al dataset actual en el que nos encontramos:

- Formatos condicionales: según parámetros o campos de otro dataset.
- Atributos de un DataSet padre.

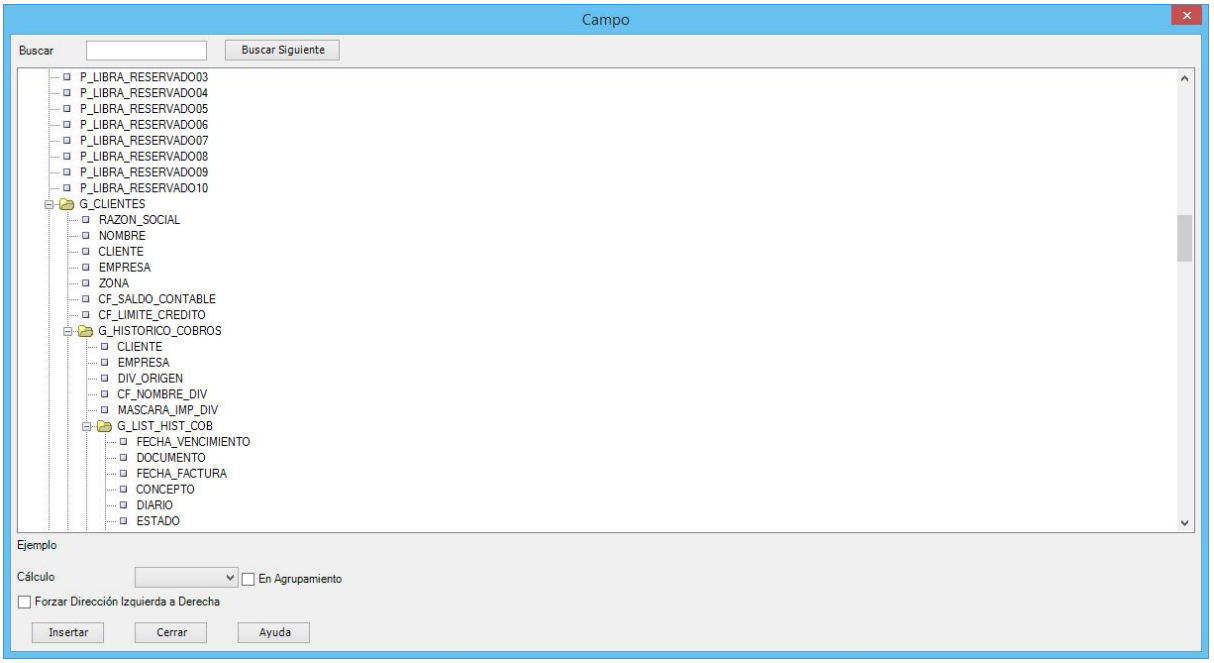
En este caso tendremos que indicar la ruta completa del XPATH del XML.

Acceder al nodo actual	**.**
Acceder al nodo padre	**..**
Todos los elementos en el XML	**//**
Acceder a elementos descendientes	**/**

Ejemplo de acceso al parámetro P_LIBRA_EMPRESA

```
/DATA_DS/P_LIBRA_EMPRESA
```

Ejemplo de estructura de un XML de un Modelo de Datos y cómo acceder a cada uno de los campos mediante rutas XPATH.



Si estuviéramos dentro de G_HISTORICO_COBROS y quisiéramos acceder al atributo NOMBRE de G_CLIENTES tendremos que escribir su ruta completa que es.

/DATA_DS/G_CLIENTES/NOMBRE

1.2.2 Contexto:

section	La declaración afecta a toda la sección de Word, incluyendo la cabecera y el pie de página Por ejemplo si declaramos for-each@section esto crea una nueva región para cada row con el reinicio del numero de paginas y la cabecera y el pie de página.
column	La declaración afecta a toda la columna de una tabla, se suele utilizar para mostrar u ocultar columnas de tablas dependiendo de los datos.
cell	La declaración afecta a la celda de una tabla. Se utiliza con @column tablas pivotantes para crear número de columnas dinámicas.
block	La declaración afecta a múltiples fo:block (párrafos de Word). Se suele utilizar para parrafo o celda de una tabla.
inline	Se utiliza para declaraciones unica en el interior de por ejemplo celdas Es usado por ejemplo para variables.
incontext	

inblock	Se suele utilizar para llamadas a templates.
inlines	La declaración afecta a multiple secciones en el interior. Una sección inline es un texto que se usa el mismo formato como un grupo de palabras que se visualizan como Bold.
begin	La declaración permite el inicio de la declaración de funciones tipo XSL de estilos.
end	Fin de donde se pueden declarar las funciones de tipo XSL de estilos.

1.3. Grupos Repetición

Desde la pestaña BI Publisher tenemos la opción de grupo repetición. Tras esto nos creará dos Campos Texto (TextField), donde cada uno tendrá el siguiente código:

Apertura del grupo:

```
<?for-each:G_MI_GRUPO?>
```

Cierre del grupo

```
<?end for-each?>
```

Se recomienda cambiar los nombre por:

Apertura del grupo:	‘ FE_MI_GRUPO ’
Cierre del grupo	‘ EF_MI_GRUPO ’

Consejo: Añadir espacios antes y después del literal y cambiar el color del texto, facilitando la legibilidad del código

1.4 Formato Condicional

Es posible indicar que cierta sección se muestre o no según una condición dependiente de los datos del informe.

Para ello habrá que modificar o insertar el siguiente código dentro de un “Campo Texto”.

```
<?if:CONDICION?>.....<?end if?>
```

Ejemplos:
Mostrar atributo ‘ESTADO’ sólo si su código es ‘ESPA’.

```
<? if:CODIGO = ‘ESPA’ ?> <?ESTADO?> <?end if?>
```

Si insertamos un condicional en un campo, en el caso en que su condición no se cumple, resultará en un salto de línea. Para evitar esto disponemos de la propiedad "**@inlines**"

Código:

```
<?if@inlines:CONDICION?>.....<?end if?>
```

De forma semejante también existen los siguientes modificadores: **@cell, @column, @row;** celda, columna o fila respectivamente que podemos añadir a nuestra condición.

```
<?if@inlines:/DATA_DS/CONTROL_AGENTE!='N'?><?AGENTE?><?end if?>
```

Formato condicional para columnas de las tablas de manera que ocupe la última columna el resto del espacio sobrante:

Columna Tabla	<?if@column:/ITEMS/@type="PRIVATE"?><?end if?>
Campo Tabla	<?if@column:/G_HC/@type="PRIVATE"?><?CF_DOMICILIACION_MANUAL?><?end if?>

1.4.1 Condiciones IF ELSE

Existen varias alternativas para generar condiciones con cláusula else.

Opción 1:

```
<?xdofx:if element_condition then result1 else result2 end if?>
```

Ejemplo:

```
<?xdofx:if INVOICE_AMOUNT > 5000 then 'Higher'
else
if INVOICE_AMOUNT <3000 then 'Lower'
else
'Equal'
end if?>
```

Opción 2:

```
<?xdoxslt:ifelse(condition,true,false)?>
```

Ejemplo:

```
<?xdoxslt:ifelse(20=21,'yes 20 and 21 are equal','No 20 and 21 are not equal')?>
```

1.5 Choose When:

```
<?choose:?>
   <?when: CAMPO1 <= CAMPO2)?>
        <?CAMPO1?>
   <?end when?>
   <?otherwise:?>
        <?CAMPO2?>
   <?end otherwise?>
<?end choose?>
```

1.6 Operaciones con Strings:

<?string(NUMBER)?>	<?string('NUMBER_ID')?>	
<?string-length(STRING)?>	<?string-length('Test this string')?>	16
<?substring(STRING,START,LENGTH)?>	<?substring('This is a test', 6, 4)?>	'is a'
<?concat(STRING,STRING[,STRING...]?>	<?concat('This ', 'is a', ' test.')?>	This is a test.

1.7 Redondear hacia arriba un número:

<?ceiling(123.12)?>	124

1.8 Funciones extendidas XDO, XDOXSLT, XDOFX y XSL

Bi Publisher dispone de un conjunto de **funciones extendidas** de **SQL y XSL** para usar en los RTF templates.

La sintaxis para estas funciones extendidas es la siguiente:

```
<?xdofx:expression?>
```

Para funciones extendidas de tipo SQL usamos la siguiente sintaxis:

```
<?xdoxslt:expresion?>
```

Más información en el manual de referencia de estas funciones:
http://docs.oracle.com/cd/E12844_01/doc/bip.1013/e12187/T421739T481158.htm#4535440

1.8.1 Operaciones con funciones XDOFX

<?xdofx:2+3?>	SUMA
<?xdofx:2-3?>	RESTA
<?xdofx:2*3?>	MULTIPLICACIÓN
<?xdofx:2 div 3?>	DIVISIÓN
<?xdofx:2**3?>	EXPONENCIAL
<?xdofx:3||2?>	CONCATENACIÓN
<?xdofx:round (2.777, 2)?> returns 2.78	ROUND
<?xdofx:decode('abc','xxx','Result01','abc,'Result02','Default')?>	DECODE

1.8.2 Operaciones con funciones XDOXSLT:

<?xdoxslt:abs(NUMBER)?>	VALOR ABSOLUTO
<?xdoxslt:replace(STRING, OLD_CHAR, NEW_CHAR)?>	REEMPLAZAR
<?xdoxslt:round(NUMBER [, DECIMALS])?>	REDONDEAR
<?xdoxslt:rtrim(STRING)?>	ELIMINA LOS ESPACIOS
<?xdoxslt:truncate(NUMBER [, DEC_INT])?>	TRUNCAR EL NUMERO

1.8.3 Operaciones con funciones XSL:

Con este tipo de funciones podemos modificar los estilos de nuestro RTF como color del texto color de fondo etc.

<xsl:attribute name="background-color" xdofo:ctx="block">COLOUR</xsl:attribute>	Color fondo celda	<xsl:attribute name="background-color" xdofo:ctx="block">red</xsl:attribute>
<xsl:attribute name="font-weight" xdofo:ctx="block"> bold </xsl:attribute>	Texto como BOLD	<xsl:attribute name="font-weight" xdofo:ctx="block"> bold </xsl:attribute>

1.8.4 Operaciones con funciones FO:

```
<fo:block white-space-collapse="false" padding-bottom="3pt" linefeed-treatment="preserve"></fo:block>
<fo:block padding-bottom="10px"></fo:block>
```

http://w3schools.sinsixx.com/xslfo/xslfo_tables.asp.htm

1.9 Máscaras en RTF Templates

En este apartado se explicará cómo realizar el formateo de campos numéricos y fecha haciendo uso de las máscaras proporcionadas por libra.

1.9.1 Máscaras numéricas

Para aplicar una máscara a un campo numérico bastará con usar esta función **(siempre y cuando el importe no sea NULL)**:

```
<?xdoxslt:format_number([ATRIBUTO_NUMERICO], [DECIMALES], $_XDOLOCALE)?>
```

Donde:
ATRIBUTO_NUMERICO: Se corresponde con el campo del DataSet al que le queremos aplicar la máscara.

DECIMALES: Se corresponde con el campo que almacena el número de decimales significativos de representación. Si se aplica un estándar, habrá que añadir el XPATH completo.

Por ejemplo:

```
<?xdoxslt:format_number(PRECIO_PRES, /DATA_DS/G_BIP/DECIMALES_IMPORTES, $_XDOLOCALE)?>
```

Otro forma de formato:

```
<?format-number: xdoxslt:round(PRECIO_PRES,/DATA_DS/G_BIP/DECIMALES_IMPORTES);/DATA_DS/G_BIP/MASCARA_IMPORTES?>
```

1.9.1.1 Campos query y de expresión (CE_)

El código a aplicar será el siguiente:

```
<?format-number: xdoxslt:round([ATRIBUTO_NUMERICO],[DECIMALES]);[MASCARA]?>
```

Donde:

ATRIBUTO_NUMERICO: Se corresponde con el campo del DataSet al que le queremos aplicar la máscara.

MASCARA: Se corresponde con el campo que almacena la máscara a aplicar. Si se aplica una máscara estandar, habrá que añadir el XPATH completo.

DECIMALES: Se corresponde con el campo que almacena el número de decimales significativos de representación. Si se aplica un estandar, habrá que añadir el XPATH completo.

Por ejemplo:

```
<?format-number:
xdoxslt:round(PRECIO_PRES,/DATA_DS/G_BIP/DECIMALES_IMPORTES);/DATA_DS/G_BIP/MASCARA_IMPORTES?>
```

~~Si queremos añadir una máscara numérica constante, por ejemplo, para números de cuentas bancarias tendremos que hacer lo siguiente:~~

```
<?format-number:ATRIBUTO;'0000'?>
```

~~Con los '0000' indicamos la longitud del número y en caso de no cubrirlo completamente lo rellenará con 0. Por ejemplo el número 23 aplicando esta máscara obtendremos '0023'.~~

1.9.1.2 Campos de función agregada (CS_)

Para añadir máscaras a los campos de función agregada (CS_) tendremos que hacer una copia mediante un atributo de tipo CE o campo Expresión ya que Bi Publisher nos entrega dichos valores con símbolo decimal "," en lugar de "." (*)

*Solucionado en la versión Bi Publisher 12c.

1.9.1.3 Variables calculadas en el RTF (acumulados)

Cuando usamos campos calculados en el RTF mediante:

```
<?xdoxslt:set_variable($_XDOCTX,'saldo',0)?>
<?xdoxslt:get_variable($_XDOCTX,'saldo')?>
```

En el proceso de acumulación se puede dar el caso que el resultado fuese cero.

En este caso, debido a errores en la precisión de números decimales en Java (recordar que Bi Publisher hace uso de Java), en lugar de un 0 podemos encontrarnos con un número muy pequeño el cual el proceso de formateo no es capaz de determinar.

Se usará la siguiente sentencia para mostrar el número formateado:

```
<?choose:?>
        <?when: xdoxslt:get_variable($_XDOCTX,'saldo') <= number('0.0000000009')?>
                <?format-number:
xdoxslt:round(number('0.0000000000'),/DATA_DS/G_BIP/DECIMALES_IMPORTES);/DATA_DS/G_BIP/MASCARA_IMPORTES?>
```

```
        <?end when?>
        <?otherwise:?>
                <?format-number:
xdoxslt:round(xdoxslt:get_variable($_XDOCTX,'saldo'),/DATA_DS/G_BIP/DECIMALES_IMPORTES);/DATA_
DS/G_BIP/MASCARA_IMPORTES?>
        <?end otherwise?>
<?end choose?>
```

1.9.1 Máscaras fecha

Cuando se muestra un campo de tipo Date será necesario realizar un formateo previo. El entorno proporciona dos máscaras:

/DATA_DS/G_BIP/MASCARA_FECHA y /DATA_DS/G_BIP/MASCARA_FECHA_HORA, que tal y como su nombre indica, nos dan el formato de una campo fecha y otro para una fecha que incluye también la parte correspondiente a la hora.

El código para usar es el siguiente:

```
<?format-date:[CAMPO_APLICAR_MASCARA];[MASCARA]?>
```

Ejemplo:

```
<?format-date:FECHA_FACTURA;/DATA_DS/G_BIP/MASCARA_FECHA?>
```

1.10 Insertar imágenes BLOB de Base de Datos

La plantilla ya dispone en la cabecera del logo de la entidad (centro_contable, empresa según los filtros enviados). En caso de que sea necesario insertar una nueva imagen que provenga de la BDD, la cual esté insertada como un BLOB, tendremos que usar el siguiente código.

Importante

Es recomendable el uso de un atributo que nos indique si hay que mostrar o no la imagen, por ejemplo si el campo está vacío.
En anteriores versiones el código que comprueba si un campo BLOB es NULL
<?if campo != ‘’?>
no funcionaba, lo cual nos obligaba a hacer uso de este campo. Ahora dicha alternativa es opcional.

Opcional
Así mismo se podría entregar dos campos los cuales contengan el width y height de la imagen en px.

Insertar imágenes BLOB de la Base de Datos:

```
<fo:instream-foreign-object content type="image/jpg">
<xsl:value-of select="IMAGE_ELEMENT"/>
</fo:instram-foreign-object>
<?if:/CAMPO_MOSTRAR_LOGO = 'S' ?>
 <xsl:variable name="logo_width"><?/WIDTH_LOGO?></xsl:variable>
 <xsl:variable name="logo_height"><?HEIGHT_LOGO?></xsl:variable>
 <fo:instream-foreign-object content-type=”image/jpg” width="{$logo_width}" height="{$logo_height}">
   <xsl:value-of select=”LOGO”/>
 </fo:instream-foreign-object>
<?end if?>
```

Nota

En caso de no tener un campo que indique si hay que mostrar o no el logo, se usará la siguiente alternativa:
<?if: LOGO != '' ?>
en lugar del texto resaltado en negrita del código anterior.

Ejemplo inserción de imágenes en RTF:

px	<fo:instream-foreign-object content type="image/jpg" height="300 px" width="4 px"> <xsl:value-of select="IMAGEN"/> </fo:instream-foreign-object>
cm	<fo:instream-foreign-object content type="image/jpg" height="3 cm" width="4 cm"> <xsl:value-of select="IMAGEN"/> </fo:instream-foreign-object>
%	<fo:instream-foreign-object content type="image/jpg" height="300%" width="300%"> <xsl:value-of select="IMAGEN"/> </fo:instream-foreign-object>

Dependiendo del XPATH de la estructura de nuestro XML tendremos que especificar dónde está nuestra imagen de la siguiente manera:
XPATH

ACCEDER AL NODO ACTUAL	**.**
ACCEDER AL NODO PADRE	**..**
TODOS LOS ELEMENTOS EN EL XML	*//*
ACCEDER A ELEMENTOS DESCENDIENTES	*/*

Código de Ejemplo:

```
<?if: /DATA_DS/G_1/LOGO !='' ?>
  <fo:instream-foreign-object content-type="image/jpg">
```

```
    <xsl:value-of select="/DATA_DS/G_1/LOGO"/>
  </fo:instream-foreign-object>
<?end if?>
```

```
/IMAGE_ELEMENT
```

En caso de no recuperar el width y height del modelo de datos, se eliminarían las líneas:
<xsl:variable name="logo_width"><?/WIDTH_LOGO?></xsl:variable>
<xsl:variable name="logo_height"><?HEIGHT_LOGO?></xsl:variable>

Y se sustituirán width="{$logo_width}" height="{$logo_height}" por sus valores en px

Ejemplo:
<fo:instream-foreign-object content-type="image/jpg" width="150px" height="100px">

1.11 Definir parámetros, setear y obtener valores

Existen sentencias para dar de alta variables y parámetros dentro del mismo informe.

1.11.1 Uso de parámetros

```
<?param@begin: P_GROUPBY;string("CITY")?>
```

1.11.2 Uso de variables

Creación de una variable.

<?variable:variable_name?>	<?variable:x;10?>
Recuperar el valor	<?$x?>

Creación de una variable xdoslt.

```
<?xdoxslt:set_variable($_XDOCTX, 'variable name', value)?>
```

Recuperar el valor de una variable

```
<?xdoxslt:get_variable($_XDOCTX, 'variable name')?>
```

Combinando ambas sentencias podríamos crear por ejemplo una variable contador donde.

1. Se define una variable con valor a 0 antes de entrar en la agrupación donde queremos llevar la cuenta

```
<?xdoxslt:set_variable($_XDOCTX, 'vCont', 0)?>
```

2. Incremento de la variable contador

```
<?xdoxslt:set_variable($_XDOCTX, 'vCont', xdoxslt:get_variable($_XDOCTX, 'vCont' )+ 1)?>
```

Creación de variable con funciones tipo XSL:

```
<xsl:variable name="P_VAR_COLOR"><?P_COLOR?></xsl:variable>
```

Obtener valor:

```
<xsl:value-of select="$P_VAR_COLOR"/>
```

1.12 Suma y Sigue

Es bastante común el tener que mostrarle al usuario el total acarreado hasta la página de un campo numérico. Por ejemplo el importe parcial de una factura. En esta sección se explicará cómo conseguirlo.

Declaración de la variable contenedora del valor del suma_sigue. En el ejemplo se usa '**pt**'. Se podrán definir otros nombres si se van a usar varios totalizados.

1. Inicialización de la variable. (Antes del for-each que abre el DataSet que contiene la variable que queremos llevar su acarreo)

```
<?init-page-total:pt?>
```

2. Actualización de la variable. (dentro del for-each).

```
<?add-page-total:pt;'ATRIBUTO'?>
```

3. Reseteo de la variable. (Tras el cierre del for-each)

```
<?end-page-total:pt?>
```

Tras esto podremos renderizar el total acarreado hasta la página o a fin de página.

Se incluye el código para:

1. Suma acarreada en las páginas anteriores.
   Añadir antes de la apertura del body o bien se define en un template y se lleva al header con el <?call-template:SumaAnterior?>

```
<xsl:variable name="mask_ssa"><?CAMPO_MASCARA?></xsl:variable>
<xdofo:inline-total display-condition="exceptfirst" name="pt">
   Suma Anterior: <xdofo:show-brought-forward name="pt" format="{$mask_ssa}">
</xdofo:inline-total>
```

   Ejemplo:

```
<xsl:variable name="mask_ssa"><?/DATA_DS/G_BIP/MASCARA_IMPORTES?></xsl:variable>
<xdofo:inline-total display-condition="exceptfirst" name="pt">
   Suma Anterior: <xdofo:show-brought-forward name="pt" format="{$mask_ssa}">
</xdofo:inline-total>
```

2. Suma acarreada al final de esta página.
   Añadir después del cierre del body o bien se define en un template y se lleva al footer con el <?call-template:SumaPosterior?>

```
<xsl:variable name="mask_ssp"><?CAMPO_MASCARA?></xsl:variable>
<xdofo:inline-total display-condition="exceptlast" name="pt">
        Total acarreado: <xdofo:show-carry-forward name="pt" format="{$mask_ssp}">
</xdofo:inline-total>
```

   Ejemplo

```
<xsl:variable name="mask_ssp"><?/DATA_DS/G_BIP/MASCARA_IMPORTES?></xsl:variable>
<xdofo:inline-total display-condition="exceptlast" name="pt">
        Total acarreado: <xdofo:show-carry-forward name="pt" format="{$mask_ssp}">
</xdofo:inline-total>
```

1.13. Papel Pautado (“Papel Pijama”)

En ocasiones queremos mostrar una tabla con el denominado papel pautado o papel pijama, el cual facilita la visibilidad del informe.

El entorno proporciona la posibilidad de elegir esta posibilidad cuando se lanza, por lo cual será conveniente añadir el código necesario, ya que después se parametrizará en libra para uso o no según nos convenga.

En el siguiente apartado se especifica los pasos a seguir para una tabla en la que su variable se denomina **vpp**.

Nota

declara una variable por cada tabla, de modo que cada una incremente y haga referencia a su tabla correspondiente.

Para insertar el papel pijama dentro de row de las tablas tenemos que hacer de la siguiente manera ya que para Word un row es cada línea de un table.

1. Creamos una variable fuera de cualquier agrupación a modo de variable global:

```
<?xdoxslt:set_variable($_XDOCTX, 'vpp',0)?>
```

2. Dentro del "for-each" en el que se renderiza la tabla que queremos aplicar el papel pijama será necesario incrementar el valor de esta variable por cada rowset.

```
<?for-each:G_LINEAS?>
<?xdoxslt:set_variable($_XDOCTX, 'vpp',xdoxslt:get_variable($_XDOCTX, 'vpp') + 1)?>
```

3. Justo después de donde empieza el for-each insertamos un campo de texto con el siguiente código que lo que hace es ver si es una línea par y en caso de ser así pintamos su background-color.

```
<?if@row:xdoxslt:get_variable($_XDOCTX, 'vpp') mod 2=0 and
/DATA_DS/G_BIP/MOSTRAR_PAPEL_PIJAMA='S'?>
  <xsl:attribute name="background-color" xdofo:ctx="incontext"><xsl:value-of
select="/DATA_DS/G_BIP/COLOR_PAPEL_PIJAMA"/></xsl:attribute>
<?end if?>
```

4. Por último reseteamos la variable, para en posteriores iteraciones se mantenga el mismo formato de pauta.

```
<?xdoxslt:set_variable($_XDOCTX, 'vpp',0)?>
```

1.14 Page Break

En caso de necesitar un salto de página en cualquier momento (por ejemplo por cada cabecera), se podrá añadir este código dentro de un campo texto.

```
<?split-by-page-break:?>
```

1.15 Mostrar cabecera de tablas por cada página

Para mostrar las cabeceras de las tablas por cada página tenemos que hacer lo siguiente:

1. Con el **cursor** nos ponemos **en una celda de la cabecera**
2. Pulsamos **botón derecho propiedades tabla** > **filas** >
3. Marcamos la opción **"Repetir como fila de encabezado en cada página"**.

1.16 Renderizar un numéro 'n' de filas de una tabla por página

En esta sección se explica cómo obtener renderizar un número dado de líneas para una tabla por página del informe.

Dentro del grupo de cualquier grupo de repetición, ponemos la siguiente sentencia:

```
<?if: position() mod NUM_MAX_FILAS=0 and position() != last()?>
<xsl:attribute name="break-before">page</xsl:attribute>
<?end if?>
```

Donde ‘NUM_MAX_FILAS’ es el número de filas que queremos mostrar de la tabla.