Artículo
Rizmaan Marikar · 21 abr, 2022 Lectura de 13 min

Cómo generar documentos EXCEL con Caché ObjectScript

Hay varias maneras de generar ficheros Excel usando tecnología InterSystems: por ejemplo utilizando informes generados con InterSystems Reports, o los antiguos informes ZEN, o incluso haciendo uso de librerías Java de terceros. Las posibilidades son casi infinitas.

Pero, ¿qué pasa si quieres crear una sencilla hoja de cálculo sólo con ObjectScript? (sin aplicaciones de terceros)

En mi caso, necesito generar informes que contengan muchos datos sin procesar (a los financieros les encantan), pero mi antiguo informe ZEN fallaba y me da lo que me gusta llamar un "archivo con cero bytes". Básicamente, Java se queda sin memoria y provoca una sobrecarga en el servidor de informes.

Esto se puede hacer usando Office Open XML (OOXML). El formato Office Open XML está compuesto por un número de archivos XML dentro de un paquete ZIP. Así que, básicamente, necesitamos generar estos archivos XML y comprimirlos renombrandolos a .xslx. Así de fácil.

Los archivos siguen un sencillo conjunto de convenciones llamadas *Open Packaging Conventions* (OPC). Hay que declar los tipos de contenido de las partes, así como indicar a la aplicación que lo consumirá donde debería empezar.

Parar crear una sencilla hoja de cálculo, necesitamos un mínimo de 5 ficheros:

- workbook.xml
- worksheet.xml
- [ContentTypes].xml
- styles.xml
- rels
    - .rels
    - workbook.xml.rels

workbook.xml
El *workbook* es el contenedor de diferentes *worksheets*. El *workbook* es donde puedes referenciar estilos, tablas de cadenas de texto compartidas, y otras piezas de información cuyo ámbito es la totalidad de la hoja de cálculo.

```
ClassMethod GenerateWorkbookXML(){
    set status =$$$OK
    set xmlfile = tempDirectoryPath_"workbook.xml"
    try{
    set stream = ##class(%Stream.FileCharacter).%New()
    set sc=stream.LinkToFile(xmlfile)
    do stream.WriteLine("<?xml version='1.0' encoding='UTF-8' standalone='yes'?>")
    do stream.WriteLine("<workbook xmlns='http://schemas.openxmlformats.org/spreadshe
etml/2006/main' xmlns:r='http://schemas.openxmlformats.org/officeDocument/2006/relati
onships'>")
    do
stream.WriteLi
ne("<sheets> <sheet name='"_workSheetName_"' sheetId='1' r:id='rId1'/>")
    do stream.WriteLine("</sheets> </workbook>")
```

```
        do stream.%Save()
    }catch{
        set status=$$$NO
    }
    kill stream
    return status
}
```

rels/workbook.xml.rels

Sólo necesitamos crear una relación que tenga un id de rId1 de manera que coincida con la referencia desde la parte de workbook.xml

```
ClassMethod CreateRelsXML(){
set status =$$$OK

    set isunix=$zcvt($p($zv," ",3,$l($p($zv," ("),")),"U")["UNIX"
    if isunix {
        set ext="/"
    }else{
        set ext="\"
    }
    set xmlfile = fileDirectory_"_rels"_ext_"workbook.xml.rels"
    set stream = ##class(%Stream.FileCharacter).%New()
    set sc=stream.LinkToFile(xmlfile)
    do stream.WriteLine("<?xml version='1.0' encoding='UTF-8' standalone='yes'?>")
    do stream.WriteLine("<Relationships xmlns='http://schemas.openxmlformats.org/pack
age/2006/relationships'>")
      do stream.WriteLine("<Relationship Id='rId1'
Type='http://schemas.openxmlformats.org/officeDocument/2006/relationships/worksheet'
Target='worksheet.xml'/>")
      do stream.WriteLine("<Relationship Id='rId2' Type='http://schemas.openxmlformat
s.org/officeDocument/2006/relationships/styles' Target='styles.xml' />")
    do stream.WriteLine("</Relationships>")
    try{
        do stream.%Save()
    }catch{
        set status=$$$NO
    }
    kill stream
    set xmlfile = fileDirectory_"_rels"_ext_".rels"
    set stream = ##class(%Stream.FileCharacter).%New()
    set sc=stream.LinkToFile(xmlfile)

    do stream.WriteLine("<?xml version='1.0' encoding='UTF-8' standalone='yes'?>")
    do stream.WriteLine("<Relationships xmlns='http://schemas.openxmlformats.org/pack
age/2006/relationships'>")
      do stream.WriteLine("<Relationship Id='rId1' Type='http://schemas.openxmlformat
s.org/officeDocument/2006/relationships/officeDocument' Target='workbook.xml'/>")
    do stream.WriteLine("</Relationships>")
    try{
        do stream.%Save()
    }catch{
        set status=$$$NO
    }
    kill stream
    return status
}
```

[Content_Types].xml
Archivo estático (por el momento, aunque debería ser un archivo dinámico dependiendo del número de worksheets) que vincula workbook, worksheet y estilos. Cada archivo Office Open XML debe declarar los tipos de contenido usados en el paquete ZIP. Eso se hace con el archivo [Content_Types].xml.

```
ClassMethod GenerateConntentTypesXML(){
    set status =$$$OK
    set xmlfile = tempDirectoryPath_"[Content_Types].xml"
    set stream = ##class(%Stream.FileCharacter).%New()
    set sc=stream.LinkToFile(xmlfile)
    try{
        do stream.WriteLine("<?xml version='1.0' encoding='UTF-8' standalone='yes'?>"
)
        do stream.WriteLine("<Types xmlns='http://schemas.openxmlformats.org/package/
2006/content-types'>")
          do stream.WriteLine("<Default Extension='rels' ContentType='application/vnd
.openxmlformats-package.relationships+xml'/>")
        do stream.WriteLine("<Override PartName='/workbook.xml' ContentType='applicat
ion/vnd.openxmlformats-officedocument.spreadsheetml.sheet.main+xml'/>")
          do stream.WriteLine("<Override PartName='/worksheet.xml' ContentType='appli
cation/vnd.openxmlformats-officedocument.spreadsheetml.worksheet+xml'/>")
          do stream.WriteLine("<Override PartName='/styles.xml' ContentType='applicat
ion/vnd.openxmlformats-officedocument.spreadsheetml.styles+xml' />")
        do stream.WriteLine("</Types>")
        do stream.%Save()
    }catch{
        set status=$$$NO
    }
    kill stream
    return status
}
```

styles.xml
Todo lo necesario para el formateo se coloca aquí. Por el momento hay varios estilos estáticos (aunque debería ser dinámico dependiendo del workbook)

Excel Styles	ID	Style	Excel Format
	1	default	Text
	2	#;[Red]-#	Number
	3	#.##;[Red]-#.##	Number
	4	yyyy/mm/dd	Date
	5	hh:mm	Date
	6	Header and Center Aligned	Text
	7	Header 2 Left Aligned	Text
	8	Good(Green Highlight)	General
	9	Bad(Red Highlight)	General
	10	Neutral(Orange Highlight)	General
	11	yyyy/mm/dd hh:mm	Date

```
ClassMethod CreateStylesXML(){
    set status =$$$OK
    set xmlfile = tempDirectoryPath_"styles.xml"
    try{
        set stream = ##class(%Stream.FileCharacter).%New()
```

```
        set sc=stream.LinkToFile(xmlfile)
         do stream.WriteLine("<?xml version=""1.0"" encoding=""UTF-8"" standalone=""y
es""?>")
        do stream.WriteLine("<styleSheet xmlns=""http://schemas.openxmlformats.org/sp
readsheetml/2006/main"" xmlns:mc=""http://schemas.openxmlformats.org/markup-compatibi
lity/2006"" mc:Ignorable=""x14ac x16r2 xr"" xmlns:x14ac=""http://schemas.microsoft.co
m/office/spreadsheetml/2009/9/ac"" xmlns:x16r2=""http://schemas.microsoft.com/office/
spreadsheetml/2015/02/main"" xmlns:xr=""http://schemas.microsoft.com/office/spreadshe
etml/2014/revision"">")
        do stream.WriteLine("<numFmts count=""4"">")
        do stream.WriteLine("<numFmt numFmtId=""166"" formatCode=""#,##0;[Red]\-#,##0
""/>")
        do stream.WriteLine("<numFmt numFmtId=""168"" formatCode=""#,##0.00;[Red]\-#,
##0.00""/>")
        do stream.WriteLine("<numFmt numFmtId=""169"" formatCode=""dd\/mm\/yyyy;@""/>
")
        do stream.WriteLine("<numFmt numFmtId=""170"" formatCode=""dd/mm/yyyy\ hh:mm"
"/></numFmts>")
        do stream.WriteLine("<fonts count=""5"" x14ac:knownFonts=""1"">")
        do stream.WriteLine("<font><sz val=""10""/><color theme=""1""/><name val=""Ca
libri""/><family val=""2""/><scheme val=""minor""/></font>")
        do stream.WriteLine("<font><sz val=""10""/><color rgb=""FF006100""/><name val
=""Calibri""/><family val=""2""/><scheme val=""minor""/></font>")
        do stream.WriteLine("<font><sz val=""10""/><color rgb=""FF9C0006""/><name val
=""Calibri""/><family val=""2""/><scheme val=""minor""/></font>")
        do stream.WriteLine("<font><sz val=""10""/><color rgb=""FF9C5700""/><name val
=""Calibri""/><family val=""2""/><scheme val=""minor""/></font>")
        do stream.WriteLine("<font><b/><sz val=""10""/><color theme=""1""/><name val=
""Calibri""/><family val=""2""/><scheme val=""minor""/></font></fonts>")
        do stream.WriteLine("<fills count=""5"">")
        do stream.WriteLine("<fill><patternFill patternType=""none""/></fill>")
        do stream.WriteLine("<fill><patternFill patternType=""gray125""/></fill>")
        do stream.WriteLine("<fill><patternFill patternType=""solid""><fgColor rgb=""
FFC6EFCE""/></patternFill></fill>")
        do stream.WriteLine("<fill><patternFill patternType=""solid""><fgColor rgb=""
FFFFC7CE""/></patternFill></fill>")
        do stream.WriteLine("<fill><patternFill patternType=""solid""><fgColor rgb=""
FFFFEB9C""/></patternFill></fill></fills>")
        do stream.WriteLine("<borders count=""1""><border><left/><right/><top/><botto
m/><diagonal/></border></borders>")
        do stream.WriteLine("<cellStyleXfs count=""4"">")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""0"" fillId=""0"" borderId=""
0""/>")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""1"" fillId=""2"" borderId=""
0"" applyNumberFormat=""0"" applyBorder=""0"" applyAlignment=""0"" applyProtection=""
0""/>")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""2"" fillId=""3"" borderId=""
0"" applyNumberFormat=""0"" applyBorder=""0"" applyAlignment=""0"" applyProtection=""
0""/>")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""3"" fillId=""4"" borderId=""
0"" applyNumberFormat=""0"" applyBorder=""0"" applyAlignment=""0"" applyProtection=""
0""/></cellStyleXfs>")
        do stream.WriteLine("<cellXfs count=""12""><xf numFmtId=""0"" fontId=""0"" fi
llId=""0"" borderId=""0"" xfId=""0""/>")
        do stream.WriteLine("<xf numFmtId=""49"" fontId=""0"" fillId=""0"" borderId="
"0"" xfId=""0"" quotePrefix=""1"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""166"" fontId=""0"" fillId=""0"" borderId=
""0"" xfId=""0"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""168"" fontId=""0"" fillId=""0"" borderId=
```

```
""0"" xfId=""0"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""169"" fontId=""0"" fillId=""0"" borderId=
""0"" xfId=""0"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""20"" fontId=""0"" fillId=""0"" borderId="
"0"" xfId=""0"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""49"" fontId=""4"" fillId=""0"" borderId="
"0"" xfId=""0"" applyNumberFormat=""1"" applyFont=""1""/>")
        do stream.WriteLine("<xf numFmtId=""49"" fontId=""4"" fillId=""0"" borderId="
"0"" xfId=""0"" applyNumberFormat=""1"" applyFont=""1"" applyAlignment=""1""><alignme
nt horizontal=""center""/>")
        do stream.WriteLine("</xf>")
        do stream.WriteLine("<xf numFmtId=""49"" fontId=""1"" fillId=""2"" borderId="
"0"" xfId=""1"" applyNumberFormat=""1""/>")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""2"" fillId=""3"" borderId=""
0"" xfId=""2""/>")
        do stream.WriteLine("<xf numFmtId=""0"" fontId=""3"" fillId=""4"" borderId=""
0"" xfId=""3""/>")
        do stream.WriteLine("<xf numFmtId=""170"" fontId=""0"" fillId=""0"" borderId=
""0"" xfId=""0"" applyNumberFormat=""1""/></cellXfs>")
        do stream.WriteLine("<cellStyles count=""4""><cellStyle name=""Bad"" xfId=""2
"" builtinId=""27""/>")
        do stream.WriteLine("<cellStyle name=""Good"" xfId=""1"" builtinId=""26""/><c
ellStyle name=""Neutral"" xfId=""3"" builtinId=""28""/>")
        do stream.WriteLine("<cellStyle name=""Normal"" xfId=""0"" builtinId=""0""/><
/cellStyles><dxfs count=""0""/>")
        do stream.WriteLine("<tableStyles count=""0"" defaultTableStyle=""TableStyleM
edium2"" defaultPivotStyle=""PivotStyleLight16""/>    ")
        do stream.WriteLine("<extLst><ext uri=""{EB79DEF2-80B8-43e5-95BD-54CBDDF9020C
}"" xmlns:x14=""http://schemas.microsoft.com/office/spreadsheetml/2009/9/main"">")
        do stream.WriteLine("<x14:slicerStyles defaultSlicerStyle=""SlicerStyleLight1
""/></ext><ext uri=""{9260A510-F301-46a8-8635-F512D64BE5F5}"" xmlns:x15=""http://sche
mas.microsoft.com/office/spreadsheetml/2010/11/main"">")
        do stream.WriteLine("<x15:timelineStyles defaultTimelineStyle=""TimeSlicerSty
leLight1""/></ext></extLst>")
        do stream.WriteLine("</styleSheet>")
        do stream.%Save()
    }catch{
        set status=$$$NO
    }
    kill stream
    return status
}
```

worksheet.xml

Aquí es donde se colocan nuestros datos. La primera fila tendrá los títulos de las columnas. Las siguientes filas sólo tendrán datos.
Aquí definiremos los anchos de columna para cada columna; si no, las columnas por defecto se configurarán con ajuste automático.

Worksheet de muestra en xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<worksheet xmlns="https://schemas.openxmlformats.org/spreadsheetml/2006/main" xmlns:r
="https://schemas.openxmlformats.org/officeDocument/2006/relationships">
<sheetData>
<row>
    <c t="inlineStr">
        <is>
```

```
            <t>Name</t>
        </is>
    </c>
    <c t="inlineStr">
        <is>
            <t>Amount</t>
        </is>
    </c>
</row>
<row>
    <c t="inlineStr">
        <is>
            <t>Jhon Smith</t>
        </is>
    </c>
    <c>
        <v>1000.74</v>
    </c>
</row>
<row>
    <c t="inlineStr">
        <is>
            <t>Tracy A</t>
        </is>
    </c>
    <c>
        <v>6001.74</v>
    </c>
</row>
</sheetData>
</worksheet>
```

Excel de muestra

A	B
Name	Amount
Jhon Smith	1000.74
Tracy A	6001.74

Las fórmulas dentro de la worksheet las podemos incluir utilizando una etiqueta <f>

```
<c >
<f>B2*0.08</f >
</c >
<c >
```

```
<f>B2+C2</f >
</c>
```

y finalmente lo empaquetamos y renombramos a .xlsx (usando unix zip)

```
set cmd ="cd "_fileDirectory_" && find . -type f | xargs zip .."_ext_xlsxFile
```

Cómo generar un documento Excel

Este código de muestra genera un documento Excel.

```
set file = "/temp/test.xlsx"
set excelObj = ##class(XLSX.writer).%New(file)
do excelObj.SetWorksheetName("test1")
set status = excelObj.BeginWorksheet()
set row = 0
set row = row+1
;----------- excelObj.Cells(rowNumber,columnNumber,style,content)

set status = excelObj.Cells(row,1,1,"Header1")
set row = row+1
set status = excelObj.Cells(row,1,2,"Content 1")
set status = excelObj.EndWorksheet()
W !,excelObj.fileName
```

Podéis encontrar el código en :

https://github.com/RizmaanMarikar/ObjectScriptExcelGenerator

#ObjectScript #Caché #InterSystems IRIS #InterSystems IRIS for Health

---

URL de fuente:https://es.community.intersystems.com/post/c%C3%B3mo-generar-documentos-excel-con-cach%C3%A9-objectscript