Making Quick Reports Compatible with Excel

By Yvan Ayaay, Technical Support Engineer, 4D Inc.

TN 06-08

Introduction

クイックレポートは、4D に組み込まれているレポートを作成するためのツールです。クイックレポートで出力できるレポートタイプには HTML ファイルも含まれています。HTML レポートの出力に使用されるテンプレートを応用すれば XML タイプのレポートが作成できます。XML タイプであれば、XML をサポートする様々なアプリケーションでインポートすることができ、例えば、クイックレポートで作成したレポートを Excel(Excel 2003)で開くことができます。このテクニカルレポートでは、そのようなレポートの作成方法を紹介しています。

Abstract

クイックレポートで選択することのできる出力形式は、プリンタ、テキストファイル、HTML ファイル、4D View、4D Chart のいずれかです。出力先に HTML ファイルを指定した場合、HTML を作成するためのテンプレートが使用されます。テンプレートは、本来のレポートに似たレイアウト、または独自のレイアウトを実現するために使用され、データはタグによって処理されて組み込まれます。それらのタグに少し手を加えれば、HTML だけでなく XML ファイルを作成することもできます。Microsoft Excel(Excel 2003)は XML をサポートしているため、特定のスキーマに従って作成された XML を解析して開いたり、独自の XML フォーマットである XML スプレッドシートファイルを開くことができます。

このテクニカルノートでは、最初にクイックレポートで HTML ファイルを作成する方法について論じ、HTML テンプレートや使用されるタグについて簡単に説明します。次に、それらのタグとクイックレポートコマンドを利用して XML ファイルを出力する方法を紹介します。出力されるのは Microsoft Excel(Excel 2003)で開くことのできる 2 種類の XML ファイルです。一方は標準の XML ファイル、他方は Excel 独自の XML フォマットである XML スプレッドシートフォーマット(SpreadsheetML)ファイルです。

Quick Report Output: HTML File

クイックレポートは、クイックレポートエディタまたはクイックレポートコマンドを使用して作成します。選択することのできる出力形式は、プリンタ、テキストファイル、HTML ファイル、4D View、4D Chart のいずれかです。出力先に HTML ファイルを指定した場合、HTML を作成するためのテンプレートが使用されます。デフォルトの HTML テンプレートも用意されていますが、カスタマイズされた HTML レイアウトで出力するためにカスタムテンプレートを

使用することもできます。

次のコードでは、コマンドでクイックレポートを構築し、HTML ファイル形式で出力します：

```
`Method: GenerateHTMLFile
C_TEXT(MyTemplate;MyValTex;$path)
C_LONGINT(QRArea)
$path:="myQR.html"
QRArea:=QR New offscreen area ` Create a QR offscreen area
QR SET REPORT TABLE(QRArea;Table(->[CustomerInfo])) `Set Customer table as the current table for QR area
QR INSERT COLUMN(QRArea;1;->[CustomerInfo]FName) `Insert FName field as 1st column
QR INSERT COLUMN(QRArea;2;->[CustomerInfo]LName) `Insert Lname as 2nd column
QR INSERT COLUMN(QRArea;3;->[CustomerInfo]Company) `Insert Company as 3rd column
QR INSERT COLUMN(QRArea;4;->[CustomerInfo]Years) `Insert Years as 4th column
QR INSERT COLUMN(QRArea;5;->[CustomerInfo]Status) `Insert Status as 5th column
ARRAY REAL($aColumns;1)
$aColumns{1}:=1 `Array for column to order.
ARRAY REAL($aOrder;1)
$aOrder{1}:=-1 ` Array for sort order. Negative 1 for descending and positive 1 for ascnding.
QR SET SORTS(QRArea;$aColumns;$aOrder) ` Set the sort order based on column
QR SET DESTINATION(QRArea;qr HTML file ;$path) ` Set destination to HTML FILE
ALL RECORDS([Customer])
QR RUN(QRArea) `Executes Quick Report.
QR DELETE OFFSCREEN AREA(QRArea) ` Delete QR offscreen area
```

結果として出力されるのは、次のような HTML ファイルです：

FName	LName	Company	Years	Status
リチャード	ニクソン	XYZコーポレーション	10	有効
ジョン	アダムズ	CDE株式会社	5	有効
ジム	カーター	ABC	12	無効

上記のコードの場合、HTML の出力にはデフォルト HTML テンプレートが使用されています。QR SET HTML TEMPLATE コマンドを使用すれば、異なるテンプレートあるいはカスタムテンプレートを選択して HTML レポートを出力することができます。テンプレートは、テキスト変数に代入したものを QR SET HTML TEMPLATE コマンドに渡しますが、このとき目的のデータ処理を実行させるために特別なタグを埋め込んでおきます。これらのタグを利用すれば XML 形式のクイックレポートを作成することもできます。

HTML Template Tags

このテクニカルノートでも使用している HTML テンプレートタグの一部を簡単に紹介します：

<!--#4DQRheader--> ... <!--/#4DQRheader--> レポートのタイトル行を定義するために使用されるタグです。

<!--#4DQRrow--> ... <!--/#4DQRrow-->行毎に繰り返される内容を定義するために使用されるタグです。
<!--#4DQRcol--> ... <!--/#4DQRcol-->行内における列毎に繰り返される内容を定義するために使用されるタグです。列の並び順はクイックレポートの順番に対応します。列の番号を定義することによって特定の列からのデータを参照することができます。例えば、2番目の列からのデータを挿入する場合は<!--#4DQRcol;2--> ... <!--/#4DQRcol;2-->と記述します。
<!--#4DQRdata-->セルのデータに置換されるタグです。

こららのタグは、HTMLテンプレートに埋め込んでおき、特定のフォーマットに整形されたHTMLファイルを出力するために使用します。分かりやすくするため、次のような単純なテーブルを出力する場合について考慮してみましょう。

CustomerID	*Name*	*City*
111	John Smith	Seattle
112	Ben Gordon	San Jose

CustomerID、Name、Cityはレポートのカラムタイトルです。これらの値を取得するためには次のようにタグを使用します：

```
<HTML>
<!--#4DQRHeader-->
    <!--#4DQRCol-->
        <!--#4DQRData-->
    <!--/#4DQRCol-->
<!--/#4DQRHeader-->
</HTML>
```

上記のテンプレートは、次のようなHTMLを出力します：

```
CustomerID
Name
City
```

行のデータを取得するためには次のようにタグを使用します：

```
<HTML>
<!--#4DQRRow-->
    <!--#4DQRCol-->
        <!--#4DQRData-->
    <!--/#4DQRCol-->
<!--/#4DQRRow-->
<HTML>
```

上記のテンプレートは、次のようなHTMLを出力します：

112
Ben Gordon
San Jose
111
John Smith
Seattle

こうしたHTMLテンプレートタグは、XML形式のレポートを出力するために使用することができます。そのようにして作成したレポートは、Microsoft Excel(Excel 2003)などのアプリケーションでは容易にインポートすることができます。

Generating an XML Quick Report Output

Microsoft Excel(Excel 2003で動作テストを実施)は、2種類のXMLをサポートしており、標準的なXMLファイルを解析してスプレッドシートを作成、またはExcel独自のXMLフォーマットで保存されたXMLスプレッドシートを開くことができます。ここでは、まず標準的なXMLファイルを作成し。それをMicrosoft Excelで解析する方法、次いでMS Excelで開くことのできるXMLスプレッドシートのテンプレートを作成する方法について説明します。

Generating a Basic XML file

Microsoft Excel(2003)は、スキーマに従ってXMLドキュメントを解析し、データをインポートすることができます。XML(Extensible Markup Language)はデータ交換のための標準言語です。(XMLについてはテクニカルノート03-48で詳しく取り上げられています。)XMLはMicrosoft Excelを始めとする多数のアプリケーションでサポートされており、この形式でクイックレポートを出力すれば、容易にデータをインポートすることができます。
(注記：Microsoft Excel 2003はカスタムXMLをサポートしています。XMLサポートについてはExcelのバージョンを確認してください。)

前述したHTMLタグを使用すれば、カスタムHTMLテンプレートを作成してXML形式に構造化されたクイックレポートを出力することができます。実行するコードは、HTMLレポートを作成する場合と変わりません。したがって、カスタマイズの作業は主にテンプレートを適切な形式で作成することにかかっています。

下記の単純なテーブルをXMLで出力する場合について考えてみましょう：

CustomerID	*Name*	*City*
111	John Smith	Seattle
112	Ben Gordon	San Jose

上記のデータを XML で表現すると次のようになります：

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Customer>
    <Customer>
        <CustomerID>
                112
        </CustomerID>
        <Name>
                Ben Gordon
        </Name>
        <City>
                San Jose
        </City>
    </Customer>
    <Customer>
        <CustomerID>
                111
        </CustomerID>
        <Name>
                John Smith
        </Name>
        <City>
                Seattle
        </City>
    </Customer>
</Customer>
```

このようなクイックレポートは次のような SampleQR メソッドで作成することができます：

```
`Method: SampleQR
`Description: This method illustrates how to generate a quick report in XML format.
C_TEXT(MyTemplate;MyValText;$path)
C_LONGINT(QRArea)
$path:="XMbasic.xml"
QRArea:=QR New offscreen area ` Create a QR offscreen area
QR SET REPORT TABLE(QRArea;Table(->[Customer])) `Set Customer table as the current table for QR area
QR INSERT COLUMN(QRArea;1;->[Customer]CustomerID) `Insert CustomerID as 1st column
QR INSERT COLUMN(QRArea;2;->[Customer]Name) `Insert Name as 2nd column
QR INSERT COLUMN(QRArea;3;->[Customer]City) ` Insert City as 3rd column
ARRAY REAL($aColumns;1)
$aColumns{1}:=1 `Array for column to order.
ARRAY REAL($aOrder;1)
$aOrder{1}:=-1 ` Array for sort order. Negative 1 for descending and positive 1 for ascending.
QR SET SORTS(QRArea;$aColumns;$aOrder) ` Set the sort order based on column and in descending order.
QR SET DESTINATION(QRArea;qr HTML file;$path) ` Set destination to HTML File
MyTemplate:= ConstructBasic_XMLTemplate (QRArea) ` Customize HTML template
QR SET HTML TEMPLATE(QRArea;MyTemplate)
ALL RECORDS([Customer])
```

QR RUN(QRArea) `Executes Quick Report.
QR DELETE OFFSCREEN AREA(QRArea) ` Delete QR offscreen area

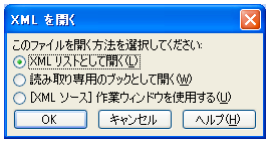
上記から分かるように、基本的には HTML ファイルを出力先に指定してクイックレポートコマンドを実行するだけです。カスタム HTML テンプレートは QR SET HTML TEMPLATE コマンドで設定します。テキスト変数 MyTemplate には XML ファイルを出力するためのカスタムテンプレートが代入されています。次の ConstructBasic_XMLTemplate メソッドは、クイックレポート用のカスタムテンプレートを構築して返すというものです。

```
`Method: ConstructBasic_XMLTemplate
`Description: This method constructs an HTML template that will generate an XML structured output.
` It takes the Quick Report area reference as parameter and returns the customized template
` as text.
C_TEXT($0;$tempText)
C_LONGINT($1;$QRarea;curPos)
C_LONGINT($numCols;$tableNum;$i;$hide;$rep;$size;$numFields;$fieldNum)
C_TEXT($ColName;$obj;$format;$title;$newLine;$fieldobj)
$QRarea:=$1 `Quick Report area reference
$numCols:=QR Count columns($QRarea) ` Count columns in QR area.
$tableNum:=QR Get report table($QRarea) `Get table number of table used in QR area.
$newLine:=Char(13)+Char(10)
$tempText:="" ` template text variable
$tempText:=$tempText+"<?xml version=¥"1.0¥" encoding=¥"<!--#4DQRCharSet-->¥"?>"
`use table name as the root element
$tempText:=$tempText+$newLine+"<"+Replace string(Table name($tableNum);" ";"_")+">"
$tempText:=$tempText+Char(13)+" <!--#4DQRRow-->"
`use table name as tag for each row.
$tempText:=$tempText+Char(13)+" < "+Replace string(Table name($tableNum);" ";"_")+">"
`Based on the number of columns, insert column names as tags with data for the column in between. This
will be repeated per row.
For ($i;1;$numCols)
    `Get information about the column in QR.
    QR GET INFO COLUMN($QRarea;$i;$title;$obj;$hide;$size;$rep;$format)
    $ColName:=$title
    ` retrieve column at a specific column number
    $tempText:=$tempText+Char(13)+" <!--#4DQRCol;"+String($i)+"-->"
    ` insert column name as tag per column
    $tempText:=$tempText+Char(13)+" <"+Replace string($ColName;" ";"_")+">"
    ` insert data in the column tag
    $tempText:=$tempText+Char(13)+" <!--#4DQRData-->"
    $tempText:=$tempText+Char(13)+" </"+Replace string($ColName;" ";"_")+">"
    $tempText:=$tempText+Char(13)+" <!--/#4DQRCol;"+String($i)+"-->"
End for
` insert matching close tags.
$tempText:=$tempText+Char(13)+" </"+Replace string(Table name($tableNum);" ";"_")+">"
$tempText:=$tempText+Char(13)+" <!--/#4DQRRow-->"
$tempText:=$tempText+$newLine+"</"+Replace string(Table name($tableNum);" ";"_")+">"
$0:=$tempText
```

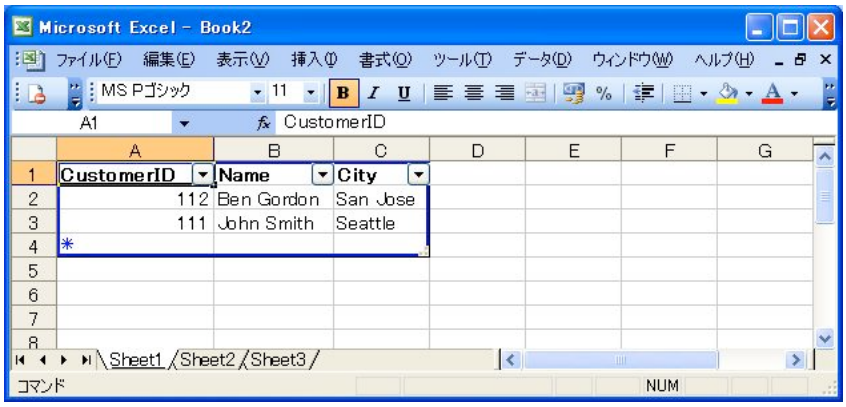
Customersテーブルを XMLフォーマットで表現するために、各列の名前は要素、各行はその子要素、データはその要素値として使用されています。

ConstructBasic_XMLTemplateメソッドでは、最初にXML encodingをテキスト変数に挿入し、次いでルート要素タグとしてテーブル名を使用しています。列は行数分だけ繰り返されるため、<!--#4DQRrow--> ... <!--/#4DQRrow-->タグで囲みます。タグとして挿入される列の名前はクイックレポートエリアから取得します。この作業は列の数だけループで繰り返します。列のタグの中には<4DQRcol;n--> ... <!--/#4DQRcol;n-->と<!--#4DQRdata->を組み込み、対応する列のデータが挿入されるようにします。タグは適切に閉じなければなりません。

SampleQRメソッドを実行するとXMLファイルが作成されます。このXMLファイルをExcelで開くと、次のようなダイアログが表示されます。標準的なXMLを開こうとすると、通常はこのダイアログが表示されます。ただしXMLスプレッドの場合はダイアログが表示されません。



「XMLリストとして開く」を選択すると、次のようにデータがインポートされます：



[Generating an XML Spreadsheet]

Microsoft Excel XP以降、XMLの新機能がExcelに加わり、SpreadsheetMLとも呼ばれるXMLスプレッドシートの使用が導入されました。このフォーマットはExcel独自のものです。

このフォーマットの特徴は、前述のような解析のステップを踏むことなく、通常の.xlsファイルを開く場合と同じようにスプレッドシートドキュメントを開くことができるという点です。

この仕組みを理解するために単純なCustomerテーブルのデータを再び使用することにします。

CustomerID	*Name*	*City*
111	John Smith	Seattle
112	Ben Gordon	San Jose

上記のテーブルデータをXMLスプレッドシートにすると次のようになります。

```
<?xml version="1.0"?>
<?mso-application progid="Excel.Sheet"?>
<Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns:x="urn:schemas-microsoft-com:office:excel"
xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
xmlns:html="http://www.w3.org/TR/REC-html40">
<DocumentProperties xmlns="urn:schemas-microsoft-com:office:office">
<Author>IVAN</Author>
<LastAuthor>Yvan Ayaay</LastAuthor>
<Created>2006-02-20T06:13:35Z</Created>
<Company>NONE</Company>
<Version>11.5606</Version>
</DocumentProperties>
<OfficeDocumentSettings xmlns="urn:schemas-microsoft-com:office:office">
<DownloadComponents/>
<LocationOfComponents HRef="file:///D:¥F__¥"/>
</OfficeDocumentSettings>
<ExcelWorkbook xmlns="urn:schemas-microsoft-com:office:excel">
<WindowHeight>9210</WindowHeight>
<WindowWidth>15195</WindowWidth>
<WindowTopX>0</WindowTopX>
<WindowTopY>30</WindowTopY>
<ProtectStructure>False</ProtectStructure>
<ProtectWindows>False</ProtectWindows>
</ExcelWorkbook>
<Styles>
<Style ss:ID="Default" ss:Name="Normal">
<Alignment ss:Vertical="Bottom"/>
<Borders/>
<Font/>
<Interior/>
<NumberFormat/>
<Protection/>
</Style>
</Styles>
<Worksheet ss:Name="Customer">
<Table ss:ExpandedColumnCount="3" ss:ExpandedRowCount="3" x:FullColumns="1"
x:FullRows="1">
<Column ss:AutoFitWidth="0" ss:Width="56.25"/>
```

```
<Row>
<Cell><Data ss:Type="String">CustomerID</Data></Cell>
<Cell><Data ss:Type="String">Name</Data></Cell>
<Cell><Data ss:Type="String">City</Data></Cell>
</Row>
<Row>
<Cell><Data ss:Type="Number">112</Data></Cell>
<Cell><Data ss:Type="String">Ben Gordon</Data></Cell>
<Cell><Data ss:Type="String">San Jose</Data></Cell>
</Row>
<Row>
<Cell><Data ss:Type="Number">111</Data></Cell>
<Cell><Data ss:Type="String">John Smith</Data></Cell>
<Cell><Data ss:Type="String">Seattle</Data></Cell>
</Row>
</Table>
<WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
<Selected/>
<ProtectObjects>False</ProtectObjects>
<ProtectScenarios>False</ProtectScenarios>
</WorksheetOptions>
</Worksheet>
</Workbook>
```

このように、最初にXMLの処理命令(Processing Information)およびドキュメントのメタデータが宣言されています。実際のデータはWorksheet要素の中に組み込まれています。したがって、この形式でデータをダイナミック出力する場合、Worksheet要素の中にHTMLタグを挿入することになります。

XMLスプレッドシートをクイックレポートで出力する場合、前述のSampleQRメソッドはそのまま使用できますが、テンプレートの構築にはConstructBasic_XMLTemplateの代わりに後述のConstructXMLSpreadSheetメソッドをコールするようにします：

```
`Method: SampleQR
..
QR SET DESTINATION(QRArea;qr HTML file ;$path) ` Set destination to HTML File
MyTemplate:= ConstructXMLSpreadSheet (QRArea) ` Customize HTML template
QR SET HTML TEMPLATE(QRArea;MyTemplate)
..
```

ConstructXMLSpreadSheetメソッドは、XMLスプレッドシート形式の出力に必要なテンプレートを構築して返すというものです：

```
C_TEXT($0;$tempText)
C_LONGINT($1;$QRarea;curPos;$numRecords)
```

```
C_LONGINT($numCols;$numRows;$tableNum;$i;$hide;$rep;$size;$numFields;$fieldNum)
C_TEXT($ColName;$obj;$format;$title;$newLine;$fieldobj;mytype;$metadataTemplate)
C_POINTER($tablePtr)
$QRarea:=$1
$numCols:=QR Count columns($QRarea) ` Count columns in QR
$tableNum:=QR Get report table($QRarea) `Get table number of current table in QR
$numFields:=Count fields($tableNum) `Count number of fields in table
$tablePtr:=Table($tableNum)
$numRecords:=Records in selection($tablePtr->) `Get number of records in selection
ARRAY TEXT(asFields;Count fields($tableNum))
For ($vlField;1;Size of array(asFields)) ` get all fieldnames of the fields in the table and save in array
    asFields{$vlField}:=Field name($tableNum;$vlField)
End for
`insert text in template text variable starting with Worksheet element tag
$tempText:="<Worksheet ss:Name=¥""+Replace string(Table name($tableNum);" ";"_")+"¥">"
`declare number of columns for the worksheet
$tempText:=$tempText+Char(13)+" <Table ss:ExpandedColumnCount=¥""+String($numCols)+"¥"
ss:ExpandedRowCount=¥""+String($numRecords+1)+"¥" x:FullColumns=¥"1¥" x:FullRows=¥"1¥">"
`Insert the column titles
$tempText:=$tempText+Char(13)+"<!--#4DQRHeader--><Row><!--#4DQRCol--><Cell><Data
ss:Type=¥"String¥"><!--#4DQRData-"+"-></Data></Cell><!--/#4DQRCol--></Row><!--/#4DQRHeader-->"
$tempText:=$tempText+Char(13)+"<!--#4DQRRow--><Row>"
`Insert column data for each row.
For ($i;1;$numCols)
    QR GET INFO COLUMN($QRarea;$i;$title;$obj;$hide;$size;$rep;$format) ` get column
    information
    curPos:=Position( " ] " ; $obj)
    $fieldobj:=Substring($obj;curPos+1) `$fieldobj contains field name
    $fieldNum:=Find in array(asFields;$fieldobj) `find field number
    GET FIELD PROPERTIES($tableNum;$fieldNum;$fieldType) `determine the type of field
    mytype:= GetType ($f ieldType) `GetType function returns the type of data of the field type.
    $ColName:=$title ` title of the column
    $tempText:=$tempText+Char(13)+"<!--#4DQRCol;"+String($i)+"-->" ` content of column at certain
    column number will be retrieved
    $tempText:=$tempText+Char(13)+"<Cell><Data ss:Type=¥""+mytype+"¥">" ` declare data type
    $tempText:=$tempText+Char(13)+"<!--#4DQRData-->" ` insert data in column
    $tempText:=$tempText+Char(13)+"</Data></Cell> <!--/#4DQRCol;"+String($i)+"-->" `insert
    matching closing tags
End for
`insert the closing tags of the lower part of the XMLSpreadsheet
$tempText:=$tempText+Char(13)+"</Row><!--/#4DQRRow-->"
$tempText:=$tempText+Char(13)+"</Table>"
$tempText:=$tempText+Char(13)+"<WorksheetOptions xmlns=¥"urn:schemas-microsoftcom:
office:excel¥">"
$tempText:=$tempText+Char(13)+"<Selected/>"
$tempText:=$tempText+Char(13)+"<ProtectObjects>False</ProtectObjects>"
$tempText:=$tempText+Char(13)+"<ProtectScenarios>False</ProtectScenarios>"
$tempText:=$tempText+Char(13)+"</WorksheetOptions>"
$tempText:=$tempText+Char(13)+"</Worksheet>"
$tempText:=$tempText+Char(13)+"</Workbook>"
```

$metadataTemplate:= ***ReadTemp*** ("XMLtemp2.txt") `ReadTemp reads a file that contains a template for the upper part of the XML Spreadsheet.
$0:=$metadataTemplate+$tempText ` Concatenate the upper part and the lower part of the XML Spreadsheet to be returned as template.

ConstructXMLSpreadsheet メソッドでは、WorkSheet 要素の中がダイナミックに生成されるようにテンプレートを構築しています。metadata の部分(XML スプレッドシートの冒頭から WorkSheet 要素の手前まで)は外部ファイルに保存されており、生成されたテンプレートと最終的に合成されます。WorkSheet 要素タグに挟まれた内容は、次のようになっており、列のヘッダを含むレポートデータが組み込まれるようになっています。

```
<Worksheet ss:Name="Customer">
<Table ss:ExpandedColumnCount="3" ss:ExpandedRowCount="3" x:FullColumns="1"
x:FullRows="1">
<Column ss:AutoFitWidth="0" ss:Width="56.25"/>
<Row>
<Cell><Data ss:Type="String">CustomerID</Data></Cell>
<Cell><Data ss:Type="String">Name</Data></Cell>
<Cell><Data ss:Type="String">City</Data></Cell>
</Row>
<Row>
<Cell><Data ss:Type="Number">112</Data></Cell>
<Cell><Data ss:Type="String">Ben Gordon</Data></Cell>
<Cell><Data ss:Type="String">San Jose</Data></Cell>
</Row>
<Row>
<Cell><Data ss:Type="Number">111</Data></Cell>
<Cell><Data ss:Type="String">John Smith</Data></Cell>
<Cell><Data ss:Type="String">Seattle</Data></Cell>
</Row>
</Table>
```

このように、WorkSheet ではタイトル行を最初の行として定義します。これを出力するために ConstructXMLSpreadsheet メソッドでは Worksheet タグおよび Table タグをテキスト変数に挿入した後、ヘッダタグ<!--#4DQRheader--> ... <!--/#4DQRheader->を利用してカラムのタイトルを書き込んでいます。すべてのカラムタイトルが挿入されるように、このタグ自体が<!--#4DQRcol--> ... <!--/#4DQRcol-->タグで囲まれています。カラムのタイトルは Cell タグおよび Data タグに囲まれています。このように、最初の行ではカラムタイトルをセルのデータとして繰り返し挿入します。

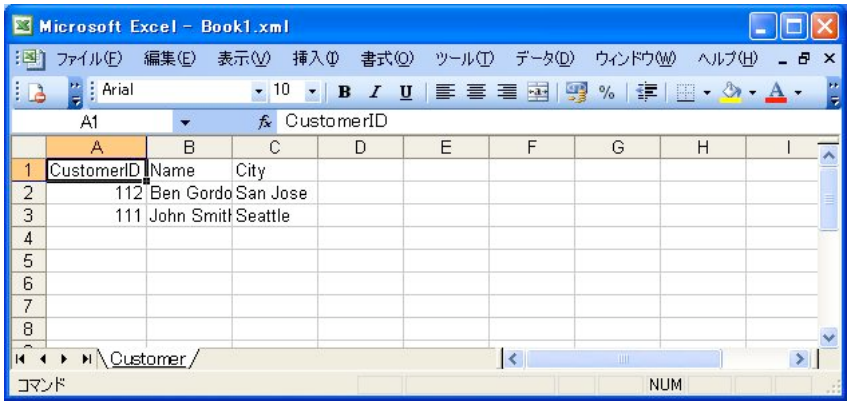
最初の行の後は、それぞれの行のデータがセルとして定義されます。データタイプは Data タグの中で宣言します。データがスプレッドシートで正しく表示されるように、データタイプはテンプレートの中でダイナミックに検出できなくてはなりません。ConstructXMLSpreadsheet

メソッドでは、列に対応するフィールド番号が調べられ、それに基づいてフィールドプロパティが特定されます。GetType メソッドは、フィールドタイプに応じ、Number あるいは String のどちらかを返す関数としてコールされています。これをクイックレポートの各カラムに対して実行できるように、カラムの数を取得した上でループを実行しています。

行に含まれるデータを取得するために<!--#4DQRrow--> ... <!--/#4DQRrow-->タグの中には For ループが組み込まれていて、行毎のデータが書き込まれるようになっています。データは<4DQRcol;n--> ... <!--/#4DQRcol;n >タグで囲まれた<!--#4DQRData-->タグによって取得しています。Column タグに挟まれた Cell および Data タグの中でこれらのタグを使用することにより、列毎、行毎のデータをセルに書き込むことができます。

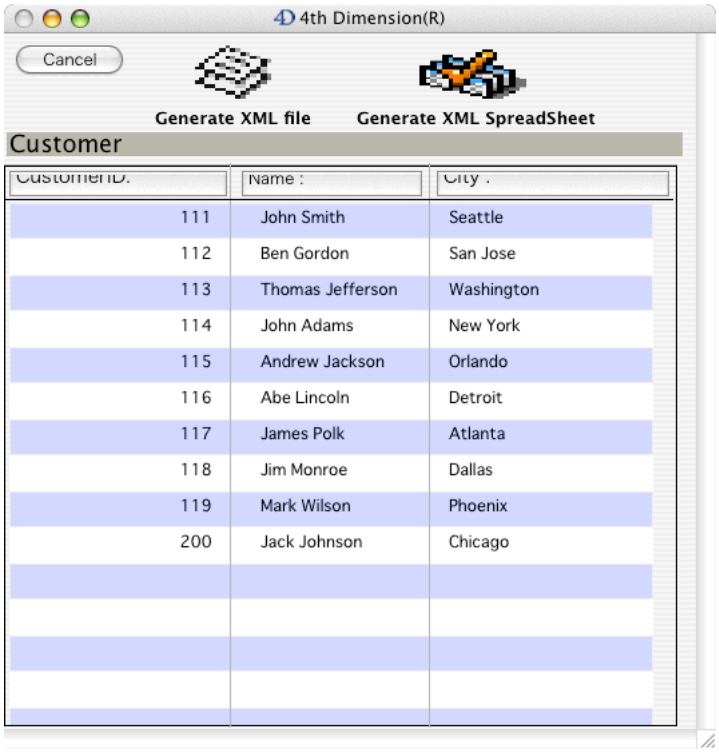
ConstructXMLSpreadsheet メソッドでは、最後に Worksheet タグを閉じています。この時点でテンプレートの後半が完成し、ファイルから読み込まれたテンプレート前半部分と合成された最終的なテキスト変数が結果として返されます。

作成されたテンプレートを使用して sampleQR メソッドを実行すると XML スプレッドシートが出力されます。このようにして作成された XML ファイルは、通常の.xls ファイルと同じように扱われ、前述のプロンプト画面は表示されません。(Microsoft Excel 2003 の場合。)



Sample Database

このテクニカルノートに収録されているサンプルデータベースには、Microsoft Excel で開くことのできる XML 形式のレポートを出力するためのメソッドが 2 種類含まれています。データベースを起動すると、次のようなダイアログが表示されます：



Generate XML file ボタンをクリックした場合は標準の XML ファイル、Generate XML SpreadSheet ボタンをクリックした場合は XML スプレッドシートファイルとしてリストのレコードが出力され、生成されたファイルの名前および保存場所が警告ダイアログで表示されます。レポートの作成にはクイックレポートコマンドを使用しています。

Conclusion

クイックレポートの出力先として HTML ファイルを指定すれば、Microsoft Excel(2003)で開くことのできる XML 形式のレポートを生成することができます。Excel 2003 は、標準の XML フォーマット、および独自の XML スプレッドシートフォーマットをサポートしています。Excel がサポートする XML 形式のレポートをクイックレポートで出力するためには、出力先を HTML ファイルに指定し、カスタム HTML テンプレートを使用する必要があります。カスタム HTML テンプレートは、ダイナミックな XML 出力に必要なテンプレートタグを組み込んで作成します。>