Crystal Reports and Visual Basic

Reports can be made quickly and easy for use with VB with Crystal Reports. But there are some problems you can encounter. For example if you use ODBC the name of the used DSN connection is saved within the report. Also when your users want to change the layout of the report they need a direct connection to the database. You can use views or a special useraccount to protect the structure of your database.

An other way is to make use of DataDefinition files. They are just ASCII files which contains the fields and their properties that are to be used in the report. Using such a file means that there is no direct connection with the database. In Visual Basic the query is executed and the result – together with de DataDefinition file – will produce the report.

To make it work you have to take three steps:

1 making the DataDefinition file
2 making the report
3 merge them in VB and show the result

The Biblio.mdb will be used in this example.

Step 1: *making the DataDefinition file*

The report needs to be a view to be of all present authors with their titles and year of publication. The fields Author.Author, Titles.Tiltle and Titles.Year Published must be shown on the report. Due the keys the table Title Authors is used. The SQL to get the result is:

```
SELECT Authors.Author, Titles.Title, Titles.[Year Published]
FROM Titles INNER JOIN (Authors INNER JOIN [Title Author] ON Authors.Au_ID = [Title
Author].Au_ID) ON Titles.ISBN = [Title Author].ISBN;
```

Making the DataDefinition file can be done in Crystal Reports but also in any ordinary editor like notepad. The format is fixed:

Fieldname
Fieldtype
Fieldsize
Description

All on one line separated by a TAB.

```
Author  String  255     name
Title   String  255     Title
YearPublished  String  4       year
```

Don't forget to save the file with the name: AuthorTitles.ttx Crystal Reports needs that to be able to define a report on this file.

Step 2: *making the report*

Start Crystal Report; choose for **New**, **Standard** report. On the tab **Data** select **Active Data**. On the next screen the choice is **Data definition**. Select with **Browse** the file you have made in step 1. Click on **Finish**. On the tab **Fields** select **Add all** and then **Preview Report**.
The report will be shown. As values the given descriptions will be shown. With **Design** you can change what you want on the report. Just save the report (standard extension .**rpt**). For quick merging in VB use the same name for the report as for the DataDefinition file: AuthorTitles.rpt

Step 3: *merge them in VB and show the result*

Open a new project in VB (name = prjDDCR). Place on the default form (name = frmDDCR) a commandbutton (name = cmdShowReport; caption = Show report). Don't forget to add the reference

for **DAO** (Microsoft DAO Object Library) and for **Crystal Reports** (Crystal Reports Engine Object Library).

Add the next code to the general section of the form. (It's just an example so all values are hard-coded.)

```
Option Explicit

Private db              As DAO.Database
Private CDOSet          As Object
Private RepApp          As Object
Private CrystRep        As CRPEAuto.Report
Private RepDb           As CRPEAuto.Database
Private RepTables       As CRPEAuto.DatabaseTables
Private RepTable        As CRPEAuto.DatabaseTable
Private LabelRows()     As Variant
```

On the Form_Load event add the next code. Don't forget the code in the Form_Unload event! Change the path to the right location of your local Biblio.mdb

```
Private Sub Form_Load()
    Set db = OpenDatabase(App.Path & "\biblio.mdb")
End Sub

Private Sub Form_Unload(Cancel As Integer)
    db.Close
    End
End Sub
```

Under the cmdShowReport_Click event is the code which merge the result of the executed SQL with the DataDefinition file and the actual report.

```
Private Sub cmdShowReport_Click()
    Dim strSQL          As String
    Dim rs              As DAO.Recordset
    Dim intFN           As Integer
    Dim strFN           As String
    Dim strLine         As String
    Dim intX            As Integer
    Dim intC            As Integer
    Dim intLabelCount   As Integer
    '
    Set RepApp = CreateObject("Crystal.CRPE.Application")
    Set CrystRep = RepApp.OpenReport(App.Path & "\AuthorTitles.rpt")
    Set CDOSet = CreateObject("CrystalDataObject.CrystalComObject")
    '
    intLabelCount = 0
    intFN = FreeFile
    strFN = App.Path & "\AuthorTitles.ttx"
    Open strFN For Input As intFN
    Do While Not EOF(intFN)
        Line Input #intFN, strLine
        If Len(strLine) <> 0 And Right(strLine, 2) <> "%%" Then
            CDOSet.AddField Split(strLine, vbTab)(0), vbString
            intLabelCount = intLabelCount + 1
        End If
    Loop
    '
    strSQL = "SELECT Authors.Author, Titles.Title, Titles.[Year Published]" + _
        " FROM Titles INNER JOIN (Authors INNER JOIN [Title Author] ON " + _
        " Authors.Au_ID = [Title Author].Au_ID) ON Titles.ISBN = [Title Author].ISBN;"
    Set rs = db.OpenRecordset(strSQL)
    With rs
        If Not (.EOF And .BOF) Then
            .MoveLast
            ReDim LabelRows(.RecordCount - 1, intLabelCount - 1)
            .MoveFirst
            For intX = LBound(LabelRows) To UBound(LabelRows)
                For intC = 0 To .Fields.Count - 1
                    LabelRows(intX, intC) = CStr("" & .Fields(intC).Value)
                Next 'intC
                .MoveNext
            Next 'intX
            '
```

```
            CDOSet.AddRows LabelRows
            Set RepDb = CrystRep.Database
            Set RepTables = RepDb.Tables
            Set RepTable = RepTables(1)
            Call RepTable.SetPrivateData(3, CDOSet)
            CrystRep.Preview "AuthorTitles.rpt"
        Else
            '
        End If
    End With
    '
    Set rs = Nothing
    Set RepApp = Nothing
    Set CrystRep = Nothing
    Set CDOSet = Nothing
End Sub
```

Compile the project. Just hit the commandbutton. Crystal Reports will open a separate window to show the result of the executed query in the report. You can open as much reports as you wish.

The advantage of using a DataDefinition file is that no difficult solutions has to be made on the database side. You don't have to make views available to the users. You don't have to make a special user with only access to those tables you want. Adding a new report does not mean to be forced to compile a new executable. If you save the SQL and the DataDefinition files in the database you only have to update the relevant tables. You can save the reports also in the database but it's better to have them located outside the database. Showing all available reports form a certain location is just easy to do. Merging the three files together is just a simple extension on the example project.