Register | Sign In | CodePlex Home
Search all CodePlex projects Search

FoxyPreviewer

Donate

PREREQUISITES:

Visual FoxPro 9 SP2

If you use Top-Level forms, the installation of the latest cumulative HotFix for VFP9SP2 KB 968409 is required, to fix a bug from SP2 that makes the toolbar in a Top-Level form look disabled. The easiest way to update your VFP9SP2 with the latest hotfixes is to install Woody's runtime installer, that can be found here:
Woody's VFP9 SP2 Runtime Installer with all hotfixes (Rev 7423)
Mirror at Foxpert.com NEW

FAQS

LINKS

Turkish documentation / Türkçe Dökümantasyon

Spanish documentation / Documentacion en español

Guía de Instalacíon de FoxyPreviewer en español NEW

USAGE:

To use it with your own reports has become even more simple than it was originally. Now you can use FoxyPReviewer in two ways:

» **1 - WITHOUT CHANGING ANY CODE IN YOUR APP** (Simplified and **recommended** mode)

See how simple it is to change completely the look and give some super powers to your reports:

```
DO FOXYPREVIEWER.APP
REPORT FORM YourReport PREVIEW
```

That's all !
That means that all you need to do is to "DO FoxyPreviewer.App" at the beginning of your executable, and ALL your reports that use the "PREVIEW" clause will be shown using the simplified but super powerful interface of FoxyPreviewer!

NEW **IMPORTANT:**
Make sure to store the file FoxyPreviewer.App in a folder that has READ/WRITE permissions, because it will create and store some helper files that it needs to use at the same folder.

» **2 - THE ORIGINAL MODE** (Complete mode) **NOT RECOMMENDED**

With just four lines of code you'll be able to use FoxyPreviewer at full power:

```
LOCAL loReport AS "PreviewHelper" OF "FoxyPreviewer.App"
loReport = CREATEOBJECT("PreviewHelper")
loReport.AddReport(_Samples + "\Solution\Reports\colors.frx", "FOR Client = 'VFPx' ") && FRX File, Clauses
loReport.RunReport()
```

The main file is **FOXYPREVIEWER.APP** that contains the **PreviewHelper** class that is the class that you need to instantiate to have the new toolbar options. It uses the ReportListener's ExtensionHandler class to make all the changes in the original toolbar.

COMPARISON BETWEEN THE 2 MODES

Feature	Default VFP 9 Preview	FoxyPreviewer Simplified	FoxyPreviewer Complete
Search in Preview	x	Yes	Yes
Printer prompt dialog button	x	Yes	Yes
Preview miniatures of pages	x	Yes	Yes
Save as Image file	x	Yes	Yes
Save as PDF	x	Yes	Yes
Save as RTF	x	Yes	Yes
Save as XLS / XML	x	Yes	Yes

System Requirements

For Use

- VFP9 SP2 with latest hotfix
- VFP9 SP2 Report*.App files

Ads by Lake Quincy Media

Save as HTML	x	Yes	Yes
Save as TXT	x	x	Yes
Copies spinner	x	Yes	Yes
Printers combo	x	Yes	Yes
Justified texts	x	Yes	Yes
Big buttons	x	Yes	Yes
Send emails	x	Yes	Yes
Change settings interactively	x	Yes	Yes
Localized language	x	Yes	Yes
Enhanced Progress bar	x	Yes	Yes
Lines of code needed to run a report	1	1	4
Adress book to send emails	x	Yes	Yes
Change buttons images automatically	x	Yes	Yes
Expand numeric fields to fit	x	Yes	Yes

CUSTOMIZING SETTING PROPERTIES

The properties below are available both in Complete and in Simplified mode

For The "Simplified" mode, immediately after you "DO FOXYPREVIEWER.APP" all the properties below will be available in the "_Screen.oFoxyPreviewer" object.
For the "Complete" mode, these properties are available directly in the "PreviewHelper" object.

Of course, you may choose what features will be available in the report preview toolbar, customize your documents, and even how the dialogs will appear. There are some few and obvious properties that you may set in order to fit your needs. Below are the methods and properties available:

Methods

(all obligatory)

- AddReport(tcFRXFile, tcClauses, tcAlias)
  - That's the place where you determine what report and which clauses will be used
  - **tcFRXFile** - character, the FRX to be run
  - **tcClauses** - character, the report clauses
  - **tcAlias** - character, the name of the table or cursor to be selected before running the report

  eg.: .AddReport(_Samples + "\Solution\Reports\wrapping.frx", "NODIALOG FOR title = 'S' RANGE 3,3")

- RunReport()

  Runs the chosen report, and shows the preview window and FoxyPreviewer toolbar

Properties

(all optional)

You don't need to configure them at all. You can let your users determine all their preferences using the preferences button.

General

- cTitle - character, the preview window title
- nShowToolBar - numeric, determines the visibility of the report toolbar when the preview is run. (1 = Visible (default), 2 = Invisible, 3 = Use resource)
- nWindowState - numeric, defines the previewform.WindowState 0 = Normal, 2 = Maximized
- nDockType - logical or numeric (0-4). If False, the dock will follow the resource file used. Or numeric, to force the toolbar docking.
- nMaxMiniatureDisplay - numeric, the quantity of reports to be shown in the miniatures form.
- cFormIcon - character, the file name of the icon to be used in the preview and other helper forms
- lDirectPrint - logical, Flag that directs the output directly to the printer, without preview (default = False)
- nThermType - numeric, the thermometer (progress bar) type used (1 = VFP9 Default, 2 = Enhanced windows compatible)
- lUseListener - logical, determines if ReportListeners will be used during printing. The report previsualization will always use listeners. There are some situations or incompatibilities between some printers and report listeners so you may try setting this property to .F. to get an "old way" printing. So, when .F., REPORTBEHAVIOR will be set to 80. Useful for dot-matrix printers. (Available only in complete mode)
- lQuietMode - logical, determines the QuietMode property for the listeners used. The progressbar for the report generation and email sending is determined by this property. && Default = .F.
- cDestFile - character, determines the filename of the output file to be generated when "RunReport()" is called. If you set this property the report will not be previewed in the complete mode. UPDATED This property is also populated with the last file name that the user chose for saving the output of the current report session.
- nSearchPages - numeric, determines the quantity of pages that will be scanned during Searches in previews. Default = -1 (all pages)
- nButtonSize - numeric, determines the size of the buttons in the preview toolbar (1 = 16x16 pixels (default), 2 = 32x32 pixels)
- cLanguage - character, the default language used in all dialogs and tooltips
- cPrinterName - character, the name of the default printer to be used in the complete mode
- lOpenViewer - logical, determines if the saved report will be automatically opened after generated, using the default app
- cSuccessor - character, the name of the report successor class. (For advanced users only)
- lExpandFields - logical, makes the report not to show the "*" if numeric values overflow. Since it makes the report running a little bit slower, by default it will continue with the original behavior. You have to set it manualy if you need it.
- cPrintJobName - character, the name of the current PrintJob used. That's the name that will apper in the printer spooler.

Toolbar

- lSendToEmail - logical, determines if the "send to email" button will be shown (not available yet)
- lSaveToFile - logical, determines if the "saveto file" button will be shown
- lShowCopies - logical, shows the copies spinner
- lShowMiniatures - logical, shows the miniatures page
- lShowSetup - logical, shows user preferences button in toolbar
- lShowPrinters - logical, determines if the available printers combo will be shown
- lShowSearch - logical, determines if the Search buttons will be visible
- lShowClose - logical, determines if the Close button will be visible
- nCopies - numeric, the default quantity of copies to be printed
- lPrintVisible - logical, shows the print button in the toolbar
- nZoomLevel - numeric, the initial zoom level of the preview window. Possible values are: 1-10%, 2-25%, 3-50%, 4-75%, 5-100% default, 6-150%, 7-200%, 8-300%, 9-500%, 10-whole page
- lPrinterPref - logical, shows the "Change printer preferences" button in the toolbar
- nPrinterPropType - numeric, the type of printer preferences dialog (1 = PRINTER PROMPT Printer dialog, 2 = Current printer properties)
- nCanvasCount - numeric, the initial nr of pages rendered on the preview form. Valid values are 1 (default), 2, or 4.

Toolbar Appearance

- cImgPrint - character, the FullPath of the image file to replace the image of the "Print" button in the preview toolbar.
- cImgPrintPref - character, the FullPath of the image file to replace the image of the "Printer Prompt" button in the preview toolbar.
- cImgSave - character, the FullPath of the image file to replace the image of the "Save" button in the preview toolbar.
- cImgClose - character, the FullPath of the image file to replace the image of the "Close" button in the preview toolbar.
- cImgClose2 - character, the FullPath of the image file to replace the image of the "Close" button in the preview toolbar. This 2nd image will be switched in the MouseOver event of the "Close"button.
- cImgEmail - character, the FullPath of the image file to replace the image of the "Email" button in the preview toolbar.
- cImgSetup - character, the FullPath of the image file to replace the image of the "Setup" button in the preview toolbar.
- cImgMiniatures - character, the FullPath of the image file to replace the image of the "Miniatures" button in the preview toolbar.
- cImgSearch - character, the FullPath of the image file to replace the image of the "Search" button in the preview toolbar.
- cImgSearchAgain - character, the FullPath of the image file to replace the image of the "Search Again" button in the preview toolbar.
- cImgSearchBack - character, the FullPath of the image file to replace the image of the "Search Back" button in the preview toolbar.

Output types allowed in the "Save as.." button from the toolbar

- lSaveAsImage - logical, includes the "save as Image" option in menu
- lSaveAsHTML - logical, includes the "save as HTML" option in menu
- lSaveAsRTF - logical, includes the "save as RTF" option in menu
- lSaveAsXLS - logical, includes the "save as XLS" option in menu
- lSaveAsPDF - logical, includes the "save as PDF" option in menu
- lSaveAsTXT - logical, includes the "save as TXT" option in menu (available only in Complete mode)
- cOutputPath - character, destination path used to save the outputs

Generic Email settings

- nEmailMode - numeric, the email type (1 = MAPI, 2 = CDOSYS HTML, 3 = CDOSYS TEXT, 4 = Custom procedure)
- lEmailAuto - logical, Automatically generates the report output file
- cEmailType - character, the file type to be used in Emails (PDF, RTF, HTML or XLS)
- cEmailPRG - character, the name of a PRG that will fire your custom email. In this PRG, you need to receive one parameter, tcFiIle, that is the temporary output file that you'll send by email. A complete sample, "MYSENDMAIL.PRG" is available, showing you how you can send your emails. To use it, you need to set the value of this property, for instance: .cEmilPrg = "MySendMail.Prg"
- cSaveDefName - character, the default name of the save file. Available in the SAVE AS dialog OR automatically used if lEmailAuto
- lAutoSendMail - logical, to send an email automatically when processing the report (available only in Complete mode)

CDOSYS EMAIL PROPERTIES

- cSMTPServer - character, the SMTP server address
- nSMTPPort - numeric, the SMTP port (usually 25)
- lSMTPUseSSL - logical, determines if the SMTP server requires SSL (security connection)
- cSMTPUserName - character, your SMTP user name
- cSMTPPassword - character, your SMTP password
- cEmailTo - character, the destination email. You may use the comma "," separator to use more than one address
- cEmailCC - character, the destination COPY email. You may use the comma "," separator to use more than one address
- cEmailBCC - character, the destination BLIND COPY email. You may use the comma "," separator to use more than one address
- cEmailReplyTo - character, the email to be used for replies
- cEmailSubject - character, the email subject
- cEmailBody - character, the email text body
- cEmailFrom - character, the email sender information. You may add some custom captions, eg: 'FoxyPreviewer team<foxyteam@hotmail.com>'
- cEmailBodyFile - character, the HTML file to be used as email body
- lReadReceipt - logical, determines if the message will ask for a read receipt
- lPriority - logical, determines if the priority level will be high
- cEncryptProcedure - character, The programmer can apply his own Scrambling method on the password string (for advanced users)

- cDecryptProcedure - character, The programmer can apply his own Scrambling method on the password string (for advanced users)
- cCryptKey - character, the crypt key used to encrypt the SMTP password stored in the settings table (for advanced users)
- cAttachments - character, the fullpath of the files to be attached to the email message. Use a comma "," as a delimiter between files (for advanced users)
- cAdressTable - character, the name of the alias or FullPath of a table that contains an adress book with emails to be used when sending email messages using the provided form. This table MUST contain a Field of Character type, named "email". Apart from that, it can contain any other data that you want to be visible in a search grid, helping the users to choose the destination. For a better comprehension, please refer to the samples provided and the FAQS. (for advanced users)
- cAdressSearch - character, optional, the name of the field to be used for making searches. (for advanced users)

PDF properties

- lPDFasImage - logical, the PDF document will generated as an image document && Default = .F.
- nPDFPageMode - integer, determines the Page mode for the PDF document. && Default = 0, 0 = Normal view, 1 = Show the thumbnails pane
- lPDFEmbedFonts - logical, determines if the PDF engine will embed the true type fonts in the PDF document
- lPDFEncryptDocument - logical, determines if the PDF engine will encrypt the PDF document, allowing you to set other restrictions to the documents, using the properties below
- lPDFCanPrint - logical, determines if the 'encrypted' PDF document will allow printing. Works only if lEncryptDocument = TRUE (see above)
- lPDFCanEdit - logical, determines if the 'encrypted' PDF document will allow editing. Works only if lEncryptDocument = TRUE (see above)
- lPDFCanCopy - logical, determines if the 'encrypted' PDF document will allow copying. Works only if lEncryptDocument = TRUE (see above)
- lPDFCanAddNotes - logical, determines if the 'encrypted' PDF document will allow adding notes. Works only if lEncryptDocument = TRUE (see above)
- cPDFMasterPassword - logical, determines the master password of the 'encrypted' PDF document. Must be different from the 'User Password' property below. Works only if lEncryptDocument = TRUE (see above)
- cPDFUserPassword - logical, determines the user password of the 'encrypted' PDF document. Must be different from the 'Master Password' property above. Works only if lEncryptDocument = TRUE (see above)
- lPDFShowErrors - logical, determines if error messages during the PDF generation will be raised to the user. Please set this property to TRUE if you are facing some PDF issues, missing fields, etc. This will help us find the source of the problem.
- cPdfAuthor - character, the author of the document
- cPdfTitle - character, the title of the document
- cPdfSubject - character, the subject of the document
- cPdfKeyWords - character, the keywords that you want to include in the PDF document
- cPdfCreator - character, default: "PDFx / FoxyPreviewer"
- cPDFSymbolFontsList - character, a Fonts list that can't be converted in PDF. Usually, bar codes and symbol fonts. Delimited with commas, eg. "Webdings,Biro". Internally, FoxyPreviewer has already a list of more than 30 fonts that will always be converted to images. Please try first to generate your PDF without adding the font to the list. If it does not render correctly, just add it here!
- cPDFDefaultFont - character, the name of the default font name to be used as default in the PDF document

Excel properties

- cExcelDefaultExtension - character, fill it with "XML" if you want this extension to be used in Excel outputs. Ideal for OpenOffice Calc users

Error related

- lSilent - logical, Stay silent regarding errors AND write any messagebox to the cErrors properties
- cErrors - character, brings error messages when lSilent = .F.

Returned information (Read only properties)

- nVersion - numeric, returns the simplified FoxyPreviewer version information
- cVersion - character, returns the detailed FoxyPreviewer version information
- lPrinted - logical, to be checked after the report is run. Tells you if the user printed the current report
- lSaved - logical, to be checked after the report is run. Tells you if the user saved the current report to a file
- lEmailed - logical, to be checked after the report is run. Tells you if the user emailed the current report
- nPageTotal - numeric, the quantity of pages of the current report

UPDATED AUTOMATING WITH INTELLISENSE

- Simplified mode

Immediately after you Initialize FoxyPreviewer with "DO FOXYPREVIEWER.APP", it will insert an object to the "_Screen", that will allow you to change every configuration. This object will be available immediately, and you can consult it every time, using Intellisense, like below. Notice that the captions of the properties are shown correctly as well:

» Complete mode

I've also created a small PRG that updates the FOXCODE table in order to add an Intellisense script to use it. Please run the file INSTALLFOXCODE.PRG to update the Intellisense.

To test it, open any empty PRG or Method, and just type:

FOXYPREVIEW and a complete script with all available options will be shown. Obviously, you may delete or comment all the properties that you don't want to use.

TRANSLATING TO NON ENGLISH LANGUAGES

As listed above, the property **cLanguage** allows you to change the language that all dialogs, forms, titlebars and tooltips appear.
The default language is ENGLISH, and right now we have 17 available: English, Portuguese, Spanish, French, German, Greek, Turkish, Czech, Persian, Arabic, Italian, Indonesian, Polish, Swahili NEW, Russian NEW, Chinese NEW and Dutch NEW

To set the language, you may pass the english language name or the local language name, for example, to change the language to French:

```
loPreview.cLanguage = "FRENCH"
```

or

```
loPreview.cLanguage = "FRANÇAIS"
```

NEW And in the Simplified mode:

```
_Screen.oFoxyPreviewer.cLanguage = "SPANISH"
```

To change the default language you have 2 options:

1. Open the FoxyPreviewer project, browse the table FoxyPreviewer_DefaultSettings, find the record that contains the "cLanguage" property, replace it with your desired language, and recompile the project
2. Run once time a FoxyPreviewer report. It will create the settings table - FoxyPreviewer*Settings.dbf, and save it in the same folder that you stored FoxyPreviewer.App. Browse the table FoxyPreviewer*Settings.dbf, find the record that contains the "cLanguage" property, replace it with your desired language. Then make sure to distribute this file, and save it in the same folder of FoxyPreviewer.App .

NEW Some of the languages are not updated. Feel free to send to vfpimaging at hotmail dot com your updated translations, or even introduce a new language ! I'll be happy to update the localizations table in the future updates.

DISTRIBUTION

FoxyPreviewer consists of a set of separate utilities, that were adapted to work together.
These files include classes, prgs, images, dlls and header files.

In order to simplify the distribution and to avoid adding many individual files to your project, all the related files have been compiled into a single APP file - FoxyPreviewer.App

The usage is simple:

1 - **Simplified mode:**
Just store the **FoxyPreviewer.App** file somewhere at the LOCAL disk, in a place that can be found by your executable. Make sure to make "SET PATH" include that folder.
DON'T embed this file in your project. If you want make sure to mark this file as "**Excluded**"

2 - **Enhanced mode:**
Store the file FoxyPreviewer.App in the same folder that you have your EXE or the Report*.App files.
Add to your main project just one file: FOXYPREVIEWERCALLER.PRG

The usage is EXACTLY the same as if you were using the PRG - the only difference is that instead of starting the FoxyPreviewer class directly, you'll access the Helper class that I created that you'll include in your EXE.
This was necessary in order to allow FoxyPreviewer to access the FRX files from your EXE. Being an external library, FoxyPreviewer.App can't access directly the FRX that are in your EXE. That's the reason that I'm asking you to include the helper file, FoxyPreviewerCaller.Prg in your EXE. That way, the helper class is inside your EXE, and it can access your FRX, and pass them to the Previewer class.

Behind the scenes, this helper class stores a copy of the FRX classes from your EXE in a TEMP file, that is used by FoxyPreviewer. This helper

class does the needed cleanups, deleting the temporary files. All the properties available in the main class are available in the helper as well, so you'll use it the same way:

```
SET PROCEDURE TO FoxyPreviewerCaller.prg ADDITIVE

LOCAL loReport as "FoxyPreviewerCaller" OF "FoxyPreviewerCaller.Prg"
loReport = CREATEOBJECT("FoxyPreviewerCaller")

WITH loReport as ReportHelper
        .AddReport("Sample.frx", "RANGE 1,1 NODIALOG") && ' The RANGE clause is optional !
        .RunReport()

        DO CASE
        CASE .lPrinted
                MESSAGEBOX("Report was printed !",64, "Report status")
        CASE .lSaved
                MESSAGEBOX("Report was saved as file:" + CHR(13) + .cDestFile,;
                        64, ;
                        "Report status")
        OTHERWISE
                MESSAGEBOX("Report Preview was closed without saving or printing",48, "Report status")
        ENDCASE
ENDWITH
```

UPDATED
If you don't want to use a separate APP file, you can have FoxyPreviewer included in your project normally.

Make sure to include all classes, VCX/VCT, images, the header files (*.H), the dialog forms, the file FoxyPreviewer.Prg in your Project, and compile. That's all. ALl these files are in the "Source" folder of FoxyPreviewer.
Then, instead of calling FoxyPreviewer.App, you'll call Foxypreviewer.**PRG**

BUT:
THIS IS NOT RECOMMENDED !
That's because FoxyPreviewer is still under development, and changes, fixes, enhancements may be applied very soon. When that happens, you'll need to update all the sources in your project. So, the recommended is to use just the APP, and distribute it separately. That way, you wont need to update your EXE when a new Version of FoxyPreviewer is released.

USING THE ADRESS BOOK TO SEND EMAILS

The adress book form contains a grid that will bring all the fields provided by your cursor or table, so please make sure to send only the fields that you want to appear. The grid brings some facilities:
- Column reorder by double-clicking at the column header (Thanks to Paul Mrozovsky, with his RCS Grid reorder class)
- The search field can be changed by right-clicking at the column header
- Selecting all the adresses, or inverting the selections, by clicking at the 1st column header

Just create a cursor or table, containing an obligatory field called "email", and set a property in the FoxyPreviewer object telling the name of the Alias / Cursor or path to access that table.

The command below creates a table using the Customers table that comes with VFP, adding the "email" field, that originated the abopve adress book:

```
*' Creating a table with the adress book
SELECT CAST(LOWER(GETWORDNUM(Contact, 1, " "))+"@vfp4.com" AS C(30)) As email,* FROM (_samples + '\data\customer') ;
        WHERE .T. INTO TABLE c:\Test2 READWRITE
```

Next step is to tell FoxyPreviewer the name of your table!

» For users of the **SIMPLIFIED** mode, use the _Screen.oFoxyPreviewer object:

```
*' Setting the global properties
```

```
DO FoxyPrevieweer.App
_Screen.oFoxyPreviewer.cAdressTable  = "c:\Test2.dbf"
_Screen.oFoxyPreviewer.cAdressSearch = "Contact"                &&' Optional
_Screen.oFoxyPreviewer.cEmailTo      = "test@foxypreviewer.org"  &&' Optional
REPORT FORM YourReport PREVIEW
```

- For users of the **COMPLETE** mode:

```
LOCAL loReport as "PreviewHelper" OF ("FoxyPreviewer.App")
loReport = CREATEOBJECT("PreviewHelper")
WITH loReport as ReportHelper
        .cAdressTable  = "c:\Test2.dbf"
        .cAdressSearch = "Contact"
        .AddReport(_Samples + "\Solution\Reports\percent.frx")
        .RunReport()
ENDWITH
```

As the dialog image above said, you first need to configure your email SMTP. This can be done using the Settings dialog, at the Email Tab.
Make sure to select CDO-HTML and provide your SMTP server info.
Obviously, you can also set the according properties directly, or even edit the Settings table, providing the needed information.
Please refer to the documentation page of FoxyPreviewer for more detailed information about the properties needed for setting your email.
There are several pictures available to ease things for you.

ABOUT THE OUTPUT TYPES AVAILABLE

- PDF - The PDF Engine is run by the LibHaru library, originally created by Takeshi Kano. FoxyPreviewer, during the first run will install the file **libhpdf.dll** in the same folder of FoxyPreviewer.App . The PDFS generated can be opened by any PDF reader application. THe PDF engine allows lots of customizations. Please refer to the properties available in this document, or use the Settings form to change these definitions interactively.
- RTF - The RTF generated is very rich, and the result is really very close the the original report. It can be viewed using MS Word, MS Word Viewer (freeware), OpenOffice Writer and other tools.
- XLS - The Excel output is somehow tricky. In fact, the output generated is not a pure Excel (XLS) type. That's a rich XML type that "MS Excel" and "OpenOffice Calc" can open without any pain. Excel 2003 can open these files normally, but newer versions will probably show a dialog saying that the file is in a wrong format. Just ignore and tell it to try to open and the worksheet will be opened. OpenOffice Calc needs that these files use the XML extension.
- HTML - In the Simplified mode, FoxyPreviewer uses some very basic HTML Style commands to generate the output, with a very good result, and without any external library. In the complete mode we use an adapted version of the HTMLListener from VFP9 FFCs that uses an XSLT transformation to map the XML to HTML. During this process it uses the MSXML4 library. If that's the case for you, make sure to have the MSXML4 library installed in the user's computer.
- IMAGE Types - FoxyPreviewer allows you to save your reports as images. Just select the desired extension in the PUTFILE dialog, or using the cDestFile property directly, and it will generate the image according to that extension. For TIFFs, Multipage one single Multipage image file will be generated for the current report. For all the other types, that don't allow multipages, several images, one for each page of he report will be generated, with the suffix of the page number in the file name.
- TXT - This output type is only available in the complete mode

FAQS:

How can I generate a PDF (or any other available type) without opening the preview window ?

- Simplified Mode

The trick is t use the new "Object Type" 's available. Just change the Object Type value to get different outputs automatically, super fast !
New Object Type's available, after you Initialize FoxyPreviewer with a "DO FOXYPREVIEWER.APP" :

- 10 = Regular PDF
- 11 = PDF AS IMAGE, to be used when the report is somehow complicated, with many objects over the others. The rendered PDF document pages are images, that don't allow making searches, with a baz zoom effect
- 12 = RTF
- 13 = XLS
- 14 = HTML

```
DO LOCFILE("FoxyPreviewer.App")
*' Make PDF
REPORT FORM ;
        (_Samples + "\Solution\Reports\Wrapping.frx") ;
        OBJECT TYPE 10 ; && OBJTYPE 10 = PDF , 11 = PDF AS IMAGE , 12 = RTF , 13 = XLS , 14 = HTML
        TO FILE "c:\TestReport.Pdf" ; && Destination
        PREVIEW && Open the default PDF viewer
```

» Complete Mode

The trick is to use the property cDestFile. FoxyPreviewer will generate the output according to the file extension that you pass that property.

```
SET PROCEDURE TO LOCFILE("FoxyPreviewer.App") ADDITIVE
LOCAL loReport as "PreviewHelper" OF "FoxyPreviewer.App"
loReport = CREATEOBJECT("PreviewHelper")
WITH loReport as ReportHelper
        .AddReport(_Samples + "\Solution\Reports\percent.frx", "NODIALOG")
        .cDestFile = "c:\Teste1.pdf"  && Use to create an output without previewing
        .RunReport()
ENDWITH
loReport = NULL
RUN /N Explorer.Exe c:\Teste1.pdf
```

The Print Preview Toolbar is visible but all buttons are disabled (not responding) in my Top-Level form report. How to fix that ?

Unfortunately, there is a known bug in VFP9 SP2, that makes all toolbars inside Top-Level forms not to respond. Fortunately, MS is distributing a hotfix for SP2, that fixes this issue. Please download the latest cumulative hotfix - KB968409 from
http://http://code.msdn.microsoft.com/KB968409/Release/ProjectReleases.aspx?ReleaseId=2445

Please follow carefully all the instructions provided in the text file included in the download, and your toolbar will start working as advertised.

The easiest way to update your VFP9SP2 with the latest hotfixes is to install Woody's runtime installer, that can be found here: Woody's VFP9 SP2 Runtime Installer with all hotfixes (Rev 7423)
Mirror at Foxpert.com

UPDATED **When I run my reports with FoxyPreviewer sometimes asterisks symbols** * * * * * * * * * * appear instead of the field. This was originally working, before using FoxyPreviewer

That happens because FoxyPreviewer uses the **SET REPORTBEHAVIOR 90** mode, that uses GDI+ to render the texts. Unfortunately there is a slight difference of the size of the strings between these modes. To fix it, just edit your report and enlarge that field !

NEW You can set the new property - **lExpandFields** to make the report engine show the field numeric value ignoring the field size.
Using "**lExpandFields**", FoxyPreviewer retrieves the value that overflowed and resends it to the report engine with an enlarged field width.

This is a known issue, and Lisa Slater Nicholls wrote a short blog post regarding it:
Why do report layouts in VFP9 need wider field/expression controls than in VFP8 and earlier?

And here's another interesting text from Lisa, that explains the reason for that:

> *With REPORTBEHAVIOR=90, the new report engine uses GDI+ to render output, and text string rendering requires more space than plain old GDI.*
>
> *The Report Desiginer uses GDI - not GDI+ - to render the report layout components, including all the text strings that you see. So if you visually right-align a label report element, the report designer records the leftmost co-ordinates of the element (the text start position) in the layout.*
>
> *The length of the string under GDI+ rendering will most likely be greater than what you would think, based on what you see in the Designer.*

The Search button is not appearing
I can't see the save as PDF / RTF / XLS options when clicking the save button

Even if you setup correctly, marking these features to be shown, it may happen that FoxyPreviewer could not manage to load all of its libraries. Make sure NOT TO USE the property cDefaultListener. Probably this will make everything to work as desired.
The cDefaultLIstener is for ADVANCED users only, people who really know how to use reportlisteners. And if you really need it, make sure to be using a subclass of FoxyListener, that is in PR_ReportListener.vcx in the Sources folder.

My VERY BIG report can't run with FoxyPreviewer

Unfortunately the VFP9 reporting system has a limit of pages / characters to be rendered. In most cases, reports with more than 3000 pages do not work, raising the error "Insufficient GDI+ resources". This is not a FoxyPreviewer issue. This limitation comes from VFP. Please reduce the scope of your report, or use SET REPORTBEHAVIOR 80 to run that report.

Reports characters are appearing very small when printed in a dot-matrix printer

This is a limitation of the VFP9 reporting system. If you are using FoxyPreviewer in the complete mode, you can setup the property lUseListener = .F. for that specific report. This will make FoxyPreviewer run the report in the ReportBehavior 80 mode.
If you are using FoxyPreviewer in the simplified mode, unfortunately there is nothing to do, except:
- Using FoxyPreviewer in the complete mode for this specific report

```
* ' Sample showing how to use the Complete mode for Dot-matrix printers
LOCAL loReport AS "PreviewHelper" OF "FoxyPreviewer.App"
loReport = CREATEOBJECT("PreviewHelper")
loReport.AddReport(_Samples + "\Solution\Reports\colors.frx") && FRX File, Clauses
loReport.lUseListener = .F.
loReport.RunReport()
```

or
- Run this report using SET REPORTBEHAVIOR 80

How can I disable FoxyPreviewer?

Sometimes for some specific reason you need to turn off FoxyPreviewer, and reset your report settings to their original status.
Just run this single line of code:

```
DO FOXYPREVIEWER.APP WITH "Release"
```

NEW Another simple option:

```
SET REPORTBEHAVIOR 80
```

I can't send emails

Please check carefully the "Settings options" form.
In some computers the "MAPI" option does not work, and you should use the "CDOSYS" instead. Check with your SMTP server provider for the correct settings.

UPDATED During FoxyPreviewer initialization, I get the error "Could not load the FOXYPREVIEWER report factory"

This happens because you are probably using an ancient version of **ReportOutput.App**

At the FoxyPreviewer downloads page you'll find the correct versions for you to download directly. Look for the file called **Report**.APP VFP9 SP2* at the "Other available downloads" section:

Another option, you can also get an updated version directly from VFPX, or even get the sources of it directly from MS and recompile it:

To recompile ReportOutput.App:
1 - Get the sources of them, the best place is MS Website directly
http://www.microsoft.com/downloads/en/details.aspx?FamilyId=320B6438-BE76-48C7-A68E-1792E2AA3BF2&displaylang=en
2 - Get the ZIP, unpack it
3 - Go to the ReportOutput folder. Open the ReportOutput project
4 - Recompile that project, following Leonid's suggestions,
5 - Use the new App instead of the original.
Please make sure to be using the latest version, by renaming all the old versions you have in your computer.

Can I use FoxyPreviewer without SP2 installed in my VFP9 ?

Yes !!!
For that, you'll just need to use the latest versions of ReportOutput.App and ReportPreview.App. Please read the previous item to get instructions on how you may get those files.

NEW Do I need to Release FoxyPreviewer after every report ?
No !!!
You need to initialize FoxyPreviewer ONLY ONE TIME, at your startup PRG. After that, all reports from your application will have the benefits of FoxyPreviewer without any code change. If for any reason you need to return to the original behavior, just call SET REPORTBEHAVIOR 80, to tell VFP to return to the old Reporting style. To get back to FoxyPreviewer style, just call SET REPORTBEHAVIOR 90. In fact, when you switch between different REPORTBEHAVIOR settings, FoxyPreviewer keeps loaded in memory, but is not used if you select "80". FoxyPreviewer occupies few resources, so there's no reason to release it all the time.
If you need a full release, then

```
DO FOXYPREVIEWER.APP WITH "Release"
```

This will restore the SET PROCEDURE, SET CLASSLIB, Extension handler, and PreviewContainer

NEW How can I provide user specific settings using FoxyPreviewer ?
There are several ways for you to do that.
At first run, FoxyPreviewer creates a Settings table that cotains information about the settings chosen by the user, when setting the **Preferences** window. This file is originally stored at the same folder of the file **FoxyPreviewer.App**. So, make save your FoxyPreviewer.app file in a folder that has Read/Write permissions.
You can also tell FoxyPreviewer the location where you want it to store the Settings file, or the folder where the settings file that you want is stored, by passing the directory as a parameter during the initialization:

```
DO FOXYPREVIEWER.APP WITH "c:\myApp\Users\John"
```

Apart from that, you can also determine which options will be available to each user in the Settings form. Apart from determining if he'll have the settings button available, you can also determine which PgFrame pages will be available, and even what controls will be enabled. Use the _Screen.oFoxyPreviewer.oSettingsDlg object. Use the Intellisense in development mode to see all the available properties or refer to the docs of FoxyPreviewer.

```
DO FoxyPreviewer.App

_Screen.oFoxyPreviewer.cLanguage  = "PORTUGUES"

*' After initializing FoxyPreviewer, here are some optional properties that you can set:
*' choosing what controls or tabs will be disabled in the settings form
*' This brings you an option to control what functions each user will have available
WITH _Screen.oFoxyPreviewer.oSettingsDlg
        .lEnableTabPdf        = .F.
        .lEnableLanguage      = .F.
        .lEnableChkSaveasTxt  = .F.
        .lEnableChkSaveasHtml = .F.
ENDWITH
```

Last edited Today at 2:15 AM by VfpImaging, version 117

Comments

Bairinis Today at 8:32 PM
Gracias ya resolví el problema, lo resolví con el siguiente comando SET REPORTBEHAVIOR 80

Bairinis Today at 4:51 PM
Felicitaciones, excelente aplicación, el zorro sigue vivo gracias a ustedes, agregue esta aplicación a mi proyecto, me funciono casi todo a la perfección, solo que cuando quiero imprimir un reporte sin mostrar la vista previa, pero quiero mostrar el cuadro de diálogo de configuración de la impresora antes de que comience la impresión lo hago con el siguiente comando: REPORT FORM (This.NameReport) TO PRINTER PROMPT NOCONSOLE me funciona bien, pero me muestra unos mensajes como de declaración de variables, como que SET TALK estuviera en ON y esos mensajes me des configuran lo que estoy mostrando en el formulario. Como hago evitar que se muestren esos mensajes, de ante mano gracias por la ayuda.

Koen Wed at 12:25 PM
Excellent piece of work: full documentation and nice application. Thanks for sharing!

vchigne1 Jul 20 at 8:38 AM

Excelent program! Kudos for the team, great job! My customers are very happy with the new capabilities of their reports

eselje Sep 17 2010 at 8:18 PM
If this is the first report that your client uses that takes advantage of listeners, be sure to deploy ReportOutput.app (from C:\Program Files (x86)\Microsoft Visual FoxPro 9) as well.

Sign in to add a comment

© 2006-2011 Microsoft | Get Help | Privacy Statement | Terms of Use | Code of Conduct | Advertise With Us | Version 2011.8.16.18101