FoxBarcode y **FoxBarcodeQR** son parte de **VFPx**, un proyecto de la Comunidad Mundial de Visual FoxPro para crear complementos de código abierto para Visual FoxPro 9.0.

Las páginas principales de ambos proyectos **VFPx** son:

- **FoxBarcode**: **https://github.com/VFPX/FoxBarcode**
- **FoxBarcodeQR**: **https://github.com/VFPX/FoxBarcodeQR**

En memoria de Guillermo Carrero *(05/02/1961 - 14/01/2013)*

FoxBarcodeQR

FoxBarcodeQR es una aplicación de software libre que ofrece una herramienta de código de barras para la comunidad de Visual FoxPro. Es un complemento de la clase **FoxBarcode** sólo para códigos **QR**.

FoxBarcodeQR utiliza las librerías **BarCodeLibrary.dll** (Darío Álvarez Aranda, México) , a partir de la versión 2.0 de **FoxBarcodeQR** utiliza **QRCodeLib.dll** (***www.validacfd.com***) agregando soporte para codificar mas de 255 caracteres y mayor control en la generación del código QR, y a partir de la versión 2.10 utiliza la API de Google (https://developers.google.com/chart/infographics/docs/qr_codes) para generar los códigos QR.

Puede escoger mediante distintos métodos cual librería utilizará.

Esta clase es una solución alternativa para todos los desarrolladores que solicitaron soporte para el **Código QR** con la clase **FoxBarcode**.

Características de las librerías externas

BarCodeLibrary.dll

- Genera códigos **QR** funcionales, pero sólo se puede establecer el tamaño y el tipo de la imagen generada.
- No tiene ajustes para el nivel de corrección de errores, colores y/o margenes.
- No soporta cadenas de mas de 255 caracteres.

La librería **BarCodeLibrary.dll** contiene solo 3 funciones:

- **LibraryVersion**: Retorna una cadena con la versión de la librería.
- **SetConfiguration**: Método para establecer el tamaño y el tipo de archivo de imagen a generar.

- **GenerateFile**: Método responsable de generar la imagen del código de barras QR.

QRCodeLib.dll (recomendada)
- Genera códigos **QR** funcionales con un mayor control de configuración.
- Soporta cadenas de mas de 255 caracteres.

La librería **BarCodeLib.dll** (versión 0.1b - ***www.validacfd.com***) contiene los métodos:
- **QRCodeLibVer:** Retorna una cadena con la versión de la librería.
- **FastQRCode:** Genera la imagen del código de barras QR con el texto a codificar.
- **FullQRCode:** Igual que el método anterior, pero con mayor control en la generación de la imagen del código QR.

Google API
- Genera códigos **QR** a través de una llamada POST a una URL, por lo que requiere conexión a internet
- Soporta cadenas de mas de 255 caracteres.

Métodos de FoxBarcodeQR

FoxBarcodeQR encapsula las funciones de las librerías **BarCodeLibrary.dll**, **QRCodeLib.dll** y la **API de Google**, en métodos propios de la clase para compatibilidad con las versiones anteriores y poder seleccionar la librería a utilizar .

Los métodos de la clase **FoxBarcodeQR** son:

QRBarcodeImage() que utiliza la librería **BarCodeLibrary.dll** y que recibe los siguientes parámetros:
- tcText: Texto para codificar
- tcFile: Nombre del archivo de imagen que desea generar. Si no se especifica ninguno, se genera un nombre de archivo aleatorio en la carpeta de archivos temporales de Windows.
- tnSize: El tamaño de la imagen generada. Recibe un número entero entre 2 y 12
  - 2 = 66 x 66 (en píxeles)
  - 3 = 99 x 99
  - 4 = 132 x 132
  - 5 = 165 x 165
  - 6 = 198 x 198
  - 7 = 231 x 231
  - 8 = 264 x 264
  - 9 = 297 x 297
  - 10 = 330 x 330
  - 11 = 363 x 363
  - 12 = 396 x 396
- tnType: El tipo de archivo de imagen generado. Recibe un número entero entre 0 y 2.
  - 0 = BMP
  - 1 = JPG
  - 2 = PNG

A partir de la versión 2.0 de **FoxBarcodeQR**, los nuevos métodos añadidos utilizan la librería **QRCodeLib.dll** v.01b:

FullQRCodeImage() que recibe los mismos parámetros que QRBarcodeImage()
- tcText: Texto para codificar

- tcFile: Nombre del archivo de imagen que desea generar. Si no se especifica ninguno, se genera un nombre de archivo aleatorio en la carpeta de archivos temporales de Windows.
- tnSize: El ancho y alto en pixeles de la imagen generada
- tnType: (solo por compatibilidad) La librería solo genera archivo de imagen tipo 0 = BMP

Para configurar las otras opciones se utilizan las siguientes propiedades:

- lAutoConfigurate: .T. para seleccionar una versión de código QR más grande si la cantidad de datos lo requiere.
- lAutoFit: Trabaja conjuntamente con lAutoConfigurate
- nBackColor: Color del fondo del código QR
- nBarColor: Color de las barras del código QR
- nCorrectionLevel: Nivel de corrección de errores :
  - 0 = Nivel L (7 %)
  - 1 = Nivel M (15 %)
  - 2 = Nivel Q (25%)
  - 3= Nivel H (30%)
- nEncoding: Algoritmo de codificación:
  - 0 = Alfabético: Codifica caracteres alfanuméricos (digitos 0-9; mayúsculas A-Z; otros nueve caracteres: **Espacio $ % * + – . / :**)
  - 1 = Byte = 1: Codifica valores binarios (8-bit data)
  - 2 = Numérico: Codifica unicamente valores numéricos (digitos 0-9)
  - 3 = Kanji: Codifica caracteres Kanji. Los caracteres Kanji en Código QR pueden tener valores 8140-9FFC y E040-EBBF
  - 4 = Auto: Selección automática del algoritmo de codificación. (Recomendado)
- nMarginPixels: Margen en pixeles
- nModuleWidth: Tamaño de los módulos en pixeles
- nHeight: Alto de la imagen en pixeles
- nWidth: Ancho de la imagen en pixeles

FastQRCodeImage() es igual que el método FullQRCodeImage() y solo se pasa el texto a codificar. El resto se configura automáticamente, sin ningún control del usuario.

- tcText: Texto para codificar
- tcFile: Nombre del archivo de imagen que desea generar. Si no se especifica ninguno, se genera un nombre de archivo aleatorio en la carpeta de archivos temporales de Windows.

A partir de la versión 2.10 de **FoxBarcodeQR** se agregó un nuevo método que utiliza la **API de Google**:

GooQRCodeImage() recibe los mismos parámetros que los métodos anteriores para uniformar la clase:

- tcText: Texto para codificar
- tcFile: Nombre del archivo de imagen que desea generar. Si no se especifica ninguno, se genera un nombre de archivo aleatorio en la carpeta de archivos temporales de Windows.
- tnSize: El ancho y alto en pixeles de la imagen generada
- tnType: (solo por compatibilidad) La API solo genera archivo de imagen tipo 2 = PNG

Esta API nos permite ajustar algunas otras propiedades como:

- nCorrectionLevel: Nivel de corrección de errores :
  - 0 = Nivel L (7 %)
  - 1 = Nivel M (15 %)
  - 2 = Nivel Q (25%)
  - 3= Nivel H (30%)
- nMarginPixels: Margen en columnas

Todos estos métodos retornan la ruta y el nombre del archivo de la imagen generada con el código QR.

Ejemplos

En el siguiente ejemplo, se crean dos imágenes de código QR, la primera con el método **QRBarcodeImage()** y la segunda con el método **FullQRCodeImage()**:

```
SET PROCEDURE TO LOCFILE("FoxBarcodeQR.prg") ADITIVE
*--- Crear un objeto FoxBarcodeQR
LOCAL loFbc, lcQRImage
loFbc = CREATEOBJECT("FoxBarcodeQR")

*-- Utilizando la librería BarCodeLibrary.dll
lcQRImage1 = loFbc.QRBarcodeImage("https://comunidadvfp.blogspot.com",,6,0)

*-- Utilizando la librería QRCodeLib.dll (www.validacfd.com)
loFbc.nBackColor = RGB(0,255,255) && Yelow
loFbc.nBarColor = RGB(0,0,128) && Blue
loFbc.nCorrectionLevel = 2 && Q 25%
lcQRImage2 = loFbc.FullQRCodeImage("https://comunidadvfp.blogspot.com",,198,0)
```

Con BarCodeLibrary.dll

Con QRCodeLib.dll con mas opciones para configurar

A partir de ésta nueva versión 2.0 de **FoxBarcodeQR** se pueden codificar cadenas de caracteres mayores a 255 caracteres con la librería **QRCodeLib.dll** y con la **API de Google**.

Ejemplo:

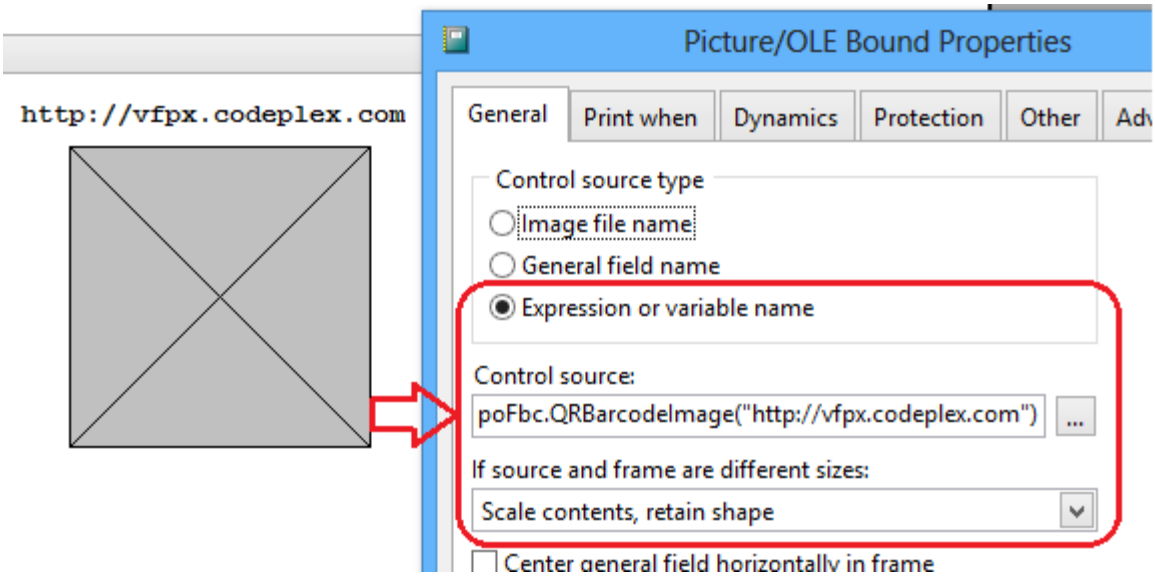
```
SET PROCEDURE TO LOCFILE("FoxBarcodeQR.prg") ADITIVE
```

```
*--- Crear un objeto FoxBarcodeQR
LOCAL loFbc, lcQRImage
loFbc = CREATEOBJECT("FoxBarcodeQR")

lcString = "+ .0010. -"
DO WHILE LEN(lcString) < 500
lnI = LEN(lcString) + 10
lcString = lcString + "+ ." + TRANSFORM(lnI, "@L 9999") + ". -"
ENDDO
*-- Utilizando la librería QRCodeLib.dll (www.validacfd.com)
lcQRImage = loFbc.FullQRCodeImage(lcString,,330)

*-- Utilizando la API de Google
lcQRImage2 = loFbc.GooQRCodeImage(lcString,,330)
```

Con QRCodeLib.dll

Con la API de Google

Para incluir un código de barras QR en un informe, se debe insertar un objeto Image y establecer la propiedad "ControlSource" con una llamada al método **QRBarcodeImage()**, **FullQRCodeImage(), FastQRCodeImage()** o **GooQRCodeImage()**. Se recomienda ajustar "Escala de contenidos, mantener la forma" si el tamaño de la imagen difiere de la estructura.



Importante: Antes de ejecutar el informe y crear el objeto **FoxBarcodeQR**, se debe declarar la variable como **PRIVATE** de forma que ésta tenga alcance en el informe:

```
*--- Crear un objeto FoxBarcodeQR privado
PRIVATE poFbc
m.poFbc = CREATEOBJECT("FoxBarcodeQR")
...
REPORT FORM FoxBarcodeQR PREVIEW
```

Distribución

Los únicos archivos necesarios para ser distribuidos para que **FoxBarcodeQR** funcione correctamente son:

- FoxBarcodeQR.prg
- BarCodeLibrary.dll
- QRCodeLib.dll (www.validacfd.com)

Notas sobre la distribución e instalación del archivo BarCodeLibrary.dll y QRCodeLib.dll:

No se registran los archivos **BarCodeLibrary.dll y QRCodeLib.dll**. Debe estar ambos en la misma carpeta de la aplicación o en la carpeta del sistema de Windows.

BarCodeLibrary.dll y **QRCodeLib.dll** fueron probados y funcionan en Windows XP, 7, 8 y 10 (32 y 64 bits)

Agradecimientos

- A mi amigo **Guillermo** que me iluminó desde el cielo para realizar este complemento para nuestra clase **FoxBarcode**.
- A **www.validacfd.com** por su librería **QRCodeLib.dll**.

¿Qué hay de nuevo en FoxBarcodeQR?

v.2.10 - Liberación: 2021.02.27

- Uso de la API de Google (requiere conexión a internet)
  - Esta API soporta:
    - Codificación de más de 255 caracteres.
    - Nivel de recuperación de corrección de errores.
    - Solo tipo de imagen PNG.
  - Se corrigieron algunos errores menores

v.2.00 - Liberación: 2020.11.07

- Uso de la librería externa QRCodeLib.dll (www.validacfd.com) lo que permite:
  - Soporte para codificar mas de 255 caracteres.
  - Soporte para configurar el color del fondo y de las barras del Código QR.
  - Soporte para configurar la capacidad del nivel de corrección de error:
    - Nivel L = 7 % de las claves se pueden restaurar
    - Nivel M = 15 % de las claves se pueden restaurar
    - Nivel Q = 25 % de las claves se pueden restaurar
    - Nivel H = 30 % de las claves se pueden restaurar
  - Soporte para seleccionar el algoritmo de codificación:
    - Numérico: Solo valores numéricos (dígitos 0-9)
    - Alfabetico: Codifica caracteres alfanuméricos (digitos 0-9; mayusculass A-Z; otros nueve caracteres: espacio $ % * + – . / :)
    - Byte: Codifica datos binarios (8-bit data)
    - Kanji/kana: Codifica caracteres Kanji. Los caracteres Kanji en Código QR Code pueden ser valores 8140-9FFC y E040-EBBF
    - Auto: Seleccionar automáticamente el algoritmo de codificación
  - Soporta únicamente tipo de imagen BMP

v.1.17 - Liberación 2016.12.21

- Se han solucionado algunos errores menores.

v.1.00 - Liberación 2013.02.16

- Simbologías de código de barras: Código QR
- Tipos de imágenes: BMP, JPG y PNG
- Biblioteca DLL externa: BarCodeLibrary.dll