Manual del Programador para WinFx

1. Introducción

1.1 ¿Qué es WinFx?

WinFx es una librería diseñada para programadores de Visual FoxPro (VFP) que integra funcionalidades avanzadas mediante la interoperabilidad con librerías .NET a través de COM. Esta herramienta simplifica el acceso a funciones como registro de eventos, manipulación de imágenes y ejecución de código asíncrono, todo desde el entorno familiar de Visual FoxPro.

WinFx está compuesta por varias librerías encapsuladas en un único archivo .app que facilita su despliegue y utilización, permitiendo una integración rápida y eficiente con aplicaciones existentes.

1.2 Características Principales

- **Interoperabilidad con .NET**: Integra fácilmente funcionalidades de librerías .NET dentro de aplicaciones Visual FoxPro.
- **Despliegue Simplificado**: WinFx.app centraliza todas las dependencias y funciones, reduciendo la complejidad de instalación.
- **Multifuncionalidad**: Incluye herramientas para logging, manejo de imágenes y ejecución de código asíncrono, esenciales en el desarrollo moderno.
- **Registro Automático de Dependencias**: Facilita el proceso de registrar componentes necesarios para la interoperabilidad.

1.3 Requisitos

- **Lenguaje de programación**: Visual FoxPro 9.0 o 10.0.
- **Entorno**: Compatible con sistemas Windows, idealmente Windows 7 o superior.
- **Versión de NET**: 4 o superior.
- **Dependencias**: Librerías .NET integradas que se registran automáticamente durante la primera ejecución.

2. Instalación

2.1 Instalación Básica

Para instalar WinFx, sigue los siguientes pasos:

1. **Copiar WinFx.app**: Copia el archivo WinFx.app en una carpeta accesible desde tu aplicación principal de Visual FoxPro.
2. **Registro de Dependencias (Primera Ejecución)**:
    - Ejecuta Visual FoxPro como administrador.
    - En la ventana de comandos, ejecuta:

```
DO WinFx.app WITH "FULL"
```

Esto extraerá y registrará las dependencias necesarias en el sistema.

3. **Uso en Ejecuciones Futuras**:
    - Para las siguientes ejecuciones, simplemente llama a la librería sin parámetros:

```
DO WinFx.app
```

Esto creará las instancias necesarias dentro del objeto **_screen.winFx.**

2.2 Opciones de Extracción

WinFx ofrece diferentes modos de extracción y registro de dependencias:

- **Extracción Completa**: Extrae y registra todas las dependencias.

```
DO WinFx.app WITH "FULL"
```

- **Extracción y Registro de Ficheros Específicos**: Extrae y registra un fichero específico.

```
DO WinFx.app WITH "fichero.ext"
```

- **Solo Extracción**: Extrae todos los ficheros sin crear instancias.

```
DO WinFx.app WITH "UNPACK"
```

3. Uso Básico

3.1 Inicialización de WinFx

Después de instalar y registrar las dependencias, inicializa WinFx ejecutando:

```
DO WinFx.app
```

Esto creará el objeto winFx dentro de _screen, que contiene todas las librerías disponibles.

3.2 Uso de Librerías Contenidas

3.2.1 Logger

La librería Logger permite la creación de logs y la escritura en el registro de eventos de Windows.

- **Escribir en un Fichero Log**:

```
_screen.WinFx.Logger.SetLogFile("MyLog.log")
_screen.WinFx.Logger.LogInfo("Información")
_screen.WinFx.Logger.LogError("Mensaje de error")
_screen.WinFx.Logger.LogWarning("Mensaje de advertencia")
```

- **Escribir en el Registro de Eventos de Windows**:

```
_screen.WinFx.Logger.bLogInWindowsEvent = .T.
_screen.WinFx.Logger.LogInfo("Información")
_screen.WinFx.Logger.LogError("Mensaje de error")
_screen.WinFx.Logger.LogWarning("Mensaje de advertencia")
```

3.2.2 DevImages

La librería DevImages facilita el uso de imágenes del assembly DevExpress.Images en controles de Visual FoxPro.

- **Obtener y Utilizar una Imagen**:

```
_screen.Picture =
_screen.WinFx.DevImages.GetImage("grayscaleimages/actions/apply_32x32.png")
```

Esto devuelve la ruta temporal de la imagen para que pueda ser usada en controles como ImageBox, Container, CommandButton, etc.

3.2.3 Async

La librería Async permite ejecutar código asíncrono desde Visual FoxPro, mejorando la capacidad de respuesta de las aplicaciones.

- **Ejecutar Código Asíncrono**:

```
_screen.WinFx.Async.RunAsync("codigo_vfp_a_ejecutar", "callback_expr")
```

- **Ejemplo**:

```
_screen.WinFx.Async.RunAsync(lcCode, "MessageBox('Proceso Finalizado')")
```

Aquí, lcCode contiene el código que se ejecutará en un hilo separado, y la expresión MessageBox('Proceso Finalizado') se ejecutará al finalizar la tarea.

4. Documentación de la API

4.1 Clases Principales y Métodos

4.1.1 Logger

- **SetLogFile(tcFilePath)**: Define la ruta del fichero log.
- **LogInfo(tcMessage)**: Registra un mensaje de información.
- **LogError(tcMessage)**: Registra un mensaje de error.
- **LogWarning(tcMessage)**: Registra un mensaje de advertencia.

4.1.2 DevImages

- **GetImage(tcRelativePath)**: Devuelve la ruta temporal de una imagen dada una ruta relativa en el assembly DevExpress.Images.

4.1.3 Async

- **RunAsync(tcScript, tcCallbackExpr)**: Ejecuta código Visual FoxPro de manera asíncrona. El parámetro tcCallbackExpr es opcional y se ejecuta al finalizar el script.

5. Casos de Uso Avanzados

5.1 Integración con Aplicaciones Visual FoxPro Existentes

WinFx puede ser fácilmente integrado en aplicaciones Visual FoxPro existentes para mejorar sus capacidades sin necesidad de modificar significativamente el código existente.

5.2 Optimización del Rendimiento

La ejecución asíncrona mediante la librería Async permite que las aplicaciones mantengan su capacidad de respuesta, incluso cuando se ejecutan tareas que normalmente bloquearían el hilo principal.

6. Solución de Problemas (FAQ)

6.1 Errores Comunes

- **Error al Registrar Dependencias**: Asegúrate de ejecutar Visual FoxPro como administrador la primera vez que se utiliza WinFx.app con el parámetro "FULL".

6.2 Preguntas Frecuentes

- **¿Qué hago si una dependencia no se registra correctamente?**: Intenta ejecutar WinFx.app con el parámetro "UNPACK" y luego registra manualmente el fichero necesario usando el parámetro "fichero.ext".