La Aplicación Agenda (AGENDA.FSL)

La ficha AGENDA.FSL es un libro de direcciones de uso directo. Para obtener la información sobre un cliente debe utilizar uno de los siguientes métodos :

- Seleccione una letra de la lista que aparece en el lado derecho de la ficha; en cuyo caso aparecerán en la lista de Nombres de la parte inferior de la ficha todos aquellos clientes cuyo nombre comience por dicha letra.

  Por ejemplo, seleccionando la letra F obtendremos aquellos clientes cuyo nombre comienza por F.

- Pulse el botón Búsqueda por Número de Cliente y aparecerá un cuadro de diálogo. Introduzca el número del cliente en el cuadro de diálogo Buscar valor.

  Compruebe que Caracteres comodín .. está seleccionado y que tiene seleccionado el campo Nº de Cliente en el cuadro de selección de campos, y pulse Aceptar. La Información del cliente especificado aparecerá en la parte alta de la ficha.

  Por ejemplo, introduciendo 1221 en el cuadro de diálogo Buscar valor se visualizará la información de Fondo Submarino

Además puede emplear las herramientas estándar de Paradox para desplazarse y editar la tabla.

Referencia

Si desea obtener más información sobre las técnicas de programación empleadas para crear esta aplicación, consulte la Ayuda de programación de la Agenda

Ayuda de programación de la Agenda

La ficha AGENDA.FSL es un libro de direcciones de uso directo. Compuesta por un pequeño número de métodos, esta aplicación proporciona un acceso a los datos rápido y eficaz; mostrando una técnica para visualizar el resultado de una consulta dentro de una ficha.

Si quiere desarrollar una aplicación parecida, debe conocer cómo se crean las tablas y los formatos, y cómo se relacionan las fichas y los marcos de tabla a las tablas, según se explica en la *Guía del Usuario de Paradox para Windows.* Este sistema de ayuda contiene una breve descripción de la estructura básica de las tablas de esta aplicación y de los objectos de diseño dentro de la ficha AGENDA.FSL

Referencia

Seleccione uno de los siguientes objetos si desea obtener una descripción del código asociado a él:

marco de tabla LETRAS

*objeto de registro encontrado*TF

botón BuscarClienteNúmero

Estructura de las Tablas

Las tablas siguientes forman parte de la estructura básica de la aplicación Agenda:

- CLIENTES.DB contiene la información de los clientes que se utiliza en la Agenda.
- XXX.DB es una tabla vacía que cuenta con un sólo campo: Temp (A1). Se trata de una tabla temporal empleada para desasociar *encontradosTF* de BUSQUEDA.DB durante la consulta.
- BUSQUEDA.DB tiene un campo: Nombre (A30). Se encarga de mostrar los nombres de los clientes que comiencen por la letra seleccionada en su marco de tabla.
- ALFABETO.DB tiene un campo: Letra (A1), contiene 27 registros (uno por cada letra del alfabeto). Almacena las letras en el marco de tabla.

Vea también

Objetos de diseño

Objetos de Diseño

Los siguientes objetos de diseño se colocan después de asociar la ficha a las Tablas:

- Un objeto multi-registro asociado a la tabla *Clientes*; que se compone de 1 registro y 1 campo. Por defecto su nombre es CLIENTES
- Un marco de tabla asociado a la tabla *Letras*. Por defecto su nombre es ALFABETO.
- Un marco de tabla asociado a la tabla BUSQUEDA, cuyo nombre se ha cambiado a *encontradosTF*.
- Un botón llamado *buscarClienteNúmero.*

Vea También

Estructura de las Tablas

Métodos de la Ficha

El código asociado a la ficha Agenda realiza dos tareas: comprueba que se está ejecutando la aplicación desde el directorio de trabajo, e invoca al sistema de ayuda de la aplicación cuando se pulsa F1.

Seleccione uno de los siguientes métodos si desea obtener una descripción del código asociado a la ficha:

método **open**

método **keyPhysical**

Método open de la ficha

El código asociado al método estándar **open** se asegura de que se está ejecutando la aplicación desde el directorio de trabajo. La llamada a **isFile** hace la comprobación: cuando se proporciona un nombre de archivo sin ruta de acceso o sin alias, Paradox mira ,por defecto,en el directorio de trabajo.

```
method open(var eventInfo Event)
  if eventInfo.isPreFilter() then
     ;este código se ejecuta para todos los objetos de la ficha
  else
     if not eventInfo.isPreFilter() then
           if not isFile("agenda.fsl") then
                 msgInfo("¡Error de arranque!", "Los archivos de
                             ejemplo de ObjectPal deben encontrarse
                             en el directorio de trabajo.")
                 close()
           endIf
     endIf
  endIf
endmethod
```

Método keyPhysical de la ficha

El código asociado al método estándar **keyPhysical** invoca a la ventana de ayuda de la aplicación para mostrar la ayuda relativa al contexto cada vez que se pulsa F1. La llamada a **vChar** se encarga de identificar cada tecla cuando se pulsa. La llamada a **DisableDefault** bloquea los métodos de Paradox en respuesta a F1.

```
method keyPhysical(var eventInfo KeyEvent)
  if eventInfo.isFirstTime() then
     if eventInfo.vChar() = "VK_F1" then
           disableDefault
           helpShowIndex(":TRABAJO:agenda.hlp")
     endIf
  endIf
endmethod
```

Los métodos del Marco de Tabla LETRAS

Como todos los marcos de Tabla, *LETRAS* es un objeto compuesto.Puede inspeccionar los objetos de un objeto compuesto como acostumbra con los demás, y puede asociarles código. Recuerde que el código asociado a los métodos estándar de un registro simple afecta a todos los registros del marco de tabla.

Seleccione uno de los siguientes métodos o propiedades si desea obtener una descripción del código asociado al marco de tabla *LETRAS*:

método **open**

método **setFocus**

propiedad **TableName**

Seleccione uno de los siguientes objetos si desea obtener una descripción del código asociado a él:

objeto de registro encontradoTF

botón *BuscarClienteNúmero*

El método open del Marco de Tabla

El siguiente código está asociado al método estándar **open** del objeto registro. Utiliza la propiedad RowNo para especificar colores alternativos para las distintas filas del marco de tabla. La propiedad RowNo devuelve el número de fila del marco de tabla, no el de la tabla subyacente.

El registro superior de la tabla es la fila 1, el siguiente es la 2 y así para todos los registros visualizados en el marco de tabla. Como cada objeto de registro del marco de tabla se abre, se ejecuta este código. Puesto que *Self* se refiere al objeto que se encuentra ejecutando el código, la siguiente sentencia asigna un valor distinto a la variable *colNum* de cada registro:

```
colNum = Self.RowNo
```

Este es el código del método estándar **open** :

```
method open(var eventInfo Event)
  var
     colNum LongInt
  endVar

  colNum = self.rowNo
  if colNum.mod(2) = 0 then
     self.color = Gray
  else
     self.color = White
  endIf
endmethod
```

El método setFocus del Marco de Tabla

El siguiente código se encuentra asociado al método estándar **setFocus** dentro del objeto de tipo Field *Letras*. Este código se asocia a **setFocus** en vez de a **canArrive** porque **setFocus** determina el resaltado dentro de un objeto campo, facilitando la visión de la letra elegida. Este código (y esta aplicación) muestra uno de los caminos para visualizar los resultados de una consulta dentro de una ficha. Acuda a la *Guía del Usuario de Paradox para Windows* si desea obtener más información sobre la relación entre fichas y consultas..

```
method setFocus(var eventInfo Event)
   var
      cadena string
      consulta query
   endvar

   doDefault
   delayScreenUpdates(Yes)
   encontradoTF.visible = No              ; oculta el marco de tabla

   encontradoTF.TableName = "xxx.db"      ; lo asociamos a XXX.DB
                                          ; mientras escribamos en
                                          ; Busqueda.DB
   DMRemoveTable("BUSQUEDA.db")           ; desasociamos Busqueda.DB
                                          ; para poder escribir
                                          ; el resultado de la
                                          ; consulta

   cadena = self.value + ".."             ; asignamos un valor a la
                                          ; variable empleada como
                                          ; tilde en la consulta

   consulta = query                       ; contruimos las
                                          ; sentencias
                                          ; de la consulta

        clientes.db | N° de Cliente | Nombre      |
                | Check       | Check ~ss |

        endQuery

   executeQBE(consulta, "Busqueda.db")       ; ejecuta la
                                             ; consulta,
                                             ; escribe los
                                             ; resultados
                                             ; en Busqueda.DB

   encontradoTF.TableName = "Busqueda.db"    ; añade BUSQUEDA.DB
                                             ; al modelo de
                                             ; datos, lo asocia a
                                             ; encontradoTF
   encontradoTF.visible = Yes                ; muestra el marco de
                                             ; tabla encontradoTF
                                             ; una vez
                                             ; actualizado
   delayScreenUpdates(No)
endmethod
```

La propiedad TableName del Marco de Tabla

Utilice la propiedad TableName dos veces: primero para asociar *encontradoTF* a XXX.DB, lo que desasocia BUSQUEDA.DB para ejecutar la consulta y almacenar los resultados, y después para reasociar *encontradoTF* a BUSQUEDA.DB y ver los resultados. Cuando se emplea la propiedad TableName para asociar un marco de tabla a una tabla, se añade la tabla al modelo de datos de la ficha. Se llama al procedimiento **DMRemoveTable** del tipo Form para desasociar BUSQUEDA.DB y para borrarlo del modelo de datos de la ficha mientras se ejecuta la consulta y se escriben sus resultados. Si no llamamos al procedimiento **DMRemoveTable**, BUSQUEDA.DB podría asociarse y abrirse en la ficha, y podría fallar al intentar escribir el resultado de la consulta.

El Objeto de registro *encontradoTF*

El marco de Tabla *encontradoTF* es un objeto compuesto que contiene una cabecera, un objeto registro y algunos objetos de tipo campo. El objeto de registro muestra los nombres y números de los clientes encontrados durante la consulta. Puede seleccionar un nombre o un número para recibir mayor información.

Seleccione el siguiente método si desea obtener una descripción del código asociado al objeto registro *encontradoTF:*

método **canArrive**

Seleccione uno de los siguientes objetos si desea obtener una descripción del código asociado a él:

marco de tabla LETRAS

botón BuscarClienteNúmero

El Método canArrive de *encontradoTF*

El código asociado al método estándar **canArrive** del objeto registro bloquea y mueve a un registro en blanco del final de la tabla. El código asociado al objeto registro estándar **canArrive** impide el desplazamiento del registro vacío situado al final de la tabla. **canArrive** pide permiso para mover el foco a un objeto. Este código utiliza la propiedad BlankRecord para comprobar si el registro que se intenta mover es el registro en blanco (vacío) del final de la tabla.

```
method canArrive(var eventInfo MoveEvent)
  var
     ObjetoDestino UIObject
     ValorDestino AnyType
  endvar

  if self.blankRecord then
     eventInfo.setErrorCode(CanNotArrive)
  else
     eventInfo.getDestination(ObjetoDestino)
     CLIENTE.locate(ObjetoDestino.name, ObjetoDestino.Value)
  endIf
endmethod
```

Como la constante UserMove se utiliza con **eventInfo.reason**, el código de usuario de este método se ejecuta cuando **canArrive** es accionado por el usuario sobre la ficha, pero no cuando es accionado por Paradox o una sentencia de ObjectPAL. Si se permite el movimiento, la siguiente sentencia emplea el método **getDestination** de tipo Event para asignar a *ObjetoDestino* el UIObject que era el destino del movimento.

```
eventInfo.getDestination(ObjetoDestino)
```

Cuando, en la siguiente sentencia, *ObjetoDestino.name* devuelve el nombre del objeto de destino, *ObjetoDestino.value* devuelve el valor del objeto (nombre y valor son propiedades de UIObject), y se pasa al método **locate**, que busca el objeto multirregistro *CLIENTES*.

```
CLIENTES.locate(ObjetoDestino.name, ObjetoDestino.Value)
```

El Botón *BuscarClienteNúmero*

El método **pushButton** de este botón muestra el método estándar cuadro de diálogo Buscar valor de Paradox (descrito en la *Guía del Usuario de Paradox para Windows*). El cuadro de diálogo Buscar valor espera a que se introduzca un número y entonces busca en la tabla. Si la búsqueda tiene éxito, muestra los resultados, si falla visualiza un mensaje. Como el cuadro de diálogo Buscar valor es un método estándar de Paradox, no se puede controlar con ObjectPAL. Es responsabilidad del usuario utilizarlo correctamente.

```
method pushButton(var eventInfo Event)

;---------------------------------------------------------------------
; Este método intenta encontrar el cliente cuyo número se ha
; introducido, utilizando el cuadro de diálogo estándar Buscar valor.
;---------------------------------------------------------------------

  var
     ValorRetorno Logical
  endvar

  CLIENTE.N°_de_Cliente.moveTo()                ; mueve al campo N°
                                                ; de Cliente
  ValorRetorno = CLIENTE.action(DataSearch)     ; llama al cuadro y
                                                ; ejecuta la búsqueda
  If not ValorRetorno then                      ; Si falla la
                                                ; búsqueda
     beep()                                     ; pita y muestra un
                                                ; mensaje
     msgInfo("¡Huy!", "O bien seleccionó Cancelar o no pude encontrar el
  número introducido.")
  endIf

endmethod
```

Puede utilizarse el método **action** de tipo UIObject con constantes propias de ObjectPAL como DataSearch para acceder a muchas utilidades de Paradox. Si desea obtener más información sobre el empleo de **action,** acuda a la *Guía de Referencia ObjectPAL.*

Seleccione uno de los siguientes objetos si desea obtener una descripción del código asociado a él:

marco de tabla LETRAS

objeto de registro *encontradoTF*

Métodos del Botón de Ayuda

El código asociado al método estándar **pushButton** del botón llamado *botónAyuda* invoca la aplicación de la ficha Agenda.

```
method pushButton(var eventInfo Event)
  message("cargando la AYUDA; un momento, por favor...")
     helpShowIndex(":TRABAJO:agenda.hlp")
  message("")
endmethod
```