THAM Contenido de la Ayuda para Programadores

Esta tabla de contenidos nos lista todas las fichas y páginas que tienen código asociado en la aplicación THAM. también contiene la biblioteca, THAMLIB.LSL, que se usa en la aplicación. Haga Clic sobre un tema para visualizar el código que tiene asociado.

NAUFRAGI

Métodos

mouseEnter
mouseExit
mouseRightUp
keyPhysical
menuAction

Procedimientos

MenúPrincipal

Variable, Constantes, y ventanas Uses

Ventana Var
Ventana Const
Ventana Uses

Métodos personalizados y procedimientos de THAMLIB.LSL

LimpiarValoresPágina
CrearMenú
MensajeTHAM
EstablecerValoresPágina
NombreDePágina
HaLlamadoARetornar
EsLlamadaVerdadero
EsLlamadaFalso

mouseEnter

NAUFRAGI.mouseEnter

Es una rutina de mensajes a nivel de ficha que muestra nombres de objetos (excluyendo los objetos de tipo Texto y Mapa de bits) en la barra de estado dependiendo de la posición del cursor del ratón . Emplea el método de la biblioteca THAMLIB, **THAMLIB.MensajeTHAM.**

```
method mouseEnter(var eventInfo MouseEvent)

  ; El sistema de mensajes a nivel de ficha se encarga de evaluar cuándo el
ratón
  ; entra en el ámbito de un objeto. Busca el mensaje apropiado en ayuda.db
y
  ; lo muestra empleando el método MensajeTham de la biblioteca ThamLib

  ; evita las llamadas sucesivas al método
  if eventInfo.isPreFilter() then
    eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
    NombreInterfaz = interfaz.Name
    ; excluye los objetos de texto y bitmap.
    if interfaz.name <> self.name AND  ; elimina el mensaje si se trata de
la propia ficha
       (interfaz.class = "Field" OR interfaz.class = "Multirecord" OR
interfaz.class = "Button") then
       thamlib.MensajeTham(NombreFicha.name,
thamlib.NombreDePágina(interfaz), NombreInterfaz)
    endif
  endif
endMethod
```

mouseExit

NAUFRAGI.mouseExit

Borra los mensajes de **mouseEnter** de la barra de estado cuando el cursor del ratón abandona la zona donde se encuentra un objeto.

```
method mouseExit(var eventInfo MouseEvent)
   ; evita sucesivas llamadas a MouseExits
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
      if interfaz.class <> "Text" and interfaz.class <> "Bitmap" then
         message("")
      endif
   endif
endMethod
```

mouseRightUp

NAUFRAGI.mouseRightUp

LLama al método **crearMenú** de la biblioteca THAMLIB cuando se suelta el botón derecho del ratón. Esto genera el menú Ayuda del código. Si se hace clic sobre un objeto Texto o sobre un objeto de Mapa de bits, aparecerá el menú Ayuda de los objetos contenidos debido a que estos objetos no tienen código asociado.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; ejecuta el método thamlib.crearMenú() encargado de mostrar el menú
   ; de ayuda a nivel de objeto.

   ; evita sucesivas llamadas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      ; si se trata de texto o bitmaps, y no es CuadroBandera,
      ; busca el contenedor correspondiente
      if interfaz.name <> "diveFlagBox" and interfaz.name <> "diveFlagBox1"
then
           while
           interfaz.class = "Text" or interfaz.class = "Bitmap"
           interfaz.attach(interfaz.ContainerName)
           endWhile
      endif
      thamlib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endMethod
```

keyPhysical

NAUFRAGI.keyPhysical

Captura las pulsaciones específicas de tecla y ejecuta según proceda **pushButton** o **moveTo** call. Por ejemplo, *Alt+A* llama a **BotónAceptar.pushButton, y** *Alt+G* llama **BotónLimpiarConsulta.moveTo.** También se detecta *F1* para llamar a **helpShowContext.**

```
method keyPhysical(var eventInfo KeyEvent)
  Var
    laTecla String
  endVar
  laTecla = eventInfo.vChar()  ; dime qué tecla han pulsado
  if eventInfo.isPreFilter() then
    if eventInfo.isAltKeyDown() then
       switch
         case laTecla = "A" or laTecla = "a" : disableDefault
    BotónAceptar.pushbutton()
         case laTecla = "C" or laTecla = "c" : disableDefault

    BotónCancelar.pushbutton()
         case laTecla = "D" or laTecla = "d" : disableDefault

    intacto.moveto()
         case laTecla = "E" or laTecla = "e" : disableDefault

    BotónLimpiarConsulta.pushbutton()
         case laTecla = "G" or laTecla = "g" : disableDefault

    pasajeros.moveto()
         case laTecla = "I" or laTecla = "i" : disableDefault

    tesoro.moveto()
         case laTecla = "N" or laTecla = "n" : disableDefault

    naufragio.moveto()
         otherwise                                                    :
doDefault
       endswitch
     else
       switch
         case laTecla = "VK_F1" : disableDefault

  helpShowContext(ficheroAyuda, 20005)
         otherwise                          : doDefault
       endSwitch
    endif
  endif
endMethod
```

menuAction

NAUFRAGI.menuAction

Comprueba la cadena devuelta por una selección de **NAUFRAGI.customMenu** y llama al método apropiado. Por ejemplo, eligiendo el elemento Cancelar se llama a **BotónCancelar.pushButton.** Eligiendo Siguiente llama a **active.action(DataNextRecord).** Un ampersand (&) en el elemento del menú hace que la siguiente tecla sea tomada como tecla rápida. Por ejemplo, &Cancelar llama a **BotónCancelar.pushButton** cuando se pulsa *Alt+C*.

```
method menuAction(var eventInfo MenuEvent)
  Var
    OpciónMenú String
    FichaAcercaDe Form
  endVar

  if eventInfo.isPrefilter() then
    doDefault
  else
    ; ¿Qué opción del menú ha sido seleccionada?
    OpciónMenú = eventInfo.menuChoice()
    switch
      case OpciónMenú = "&Aceptar"           : BotónAceptar.pushbutton()
      case OpciónMenú = "&Cancelar"          : BotónCancelar.pushbutton()
      case OpciónMenú = "&Eliminar"          :
BotónLimpiarConsulta.pushButton()
      case OpciónMenú = "&Primero\tCtrl+F11" :
miMando.BotónPrimero.pushButton()
      case OpciónMenú = "&Ultimo\tCtrl+F12"  :
miMando.BotónUltimo.pushButton()
      case OpciónMenú = "&Siguiente\tF12"    :
miMando.BotónSiguiente.pushButton()
      case OpciónMenú = "&Anterior\tF11"     :
miMando.BotóAnterior.pushButton()
      case OpciónMenú = "&Indice"                  :
helpShowContext(FicheroAyuda, 20005)
      case OpciónMenú = "&Uso de la Ayuda"   : helpOnHelp()
      case OpciónMenú = "Acerca de &THAM"    : thamlib.EsLlamadaVerdadero()
if FichaAcercaDe.open("SobTham.fsl") then

  ; abre un cuadro de diálogo Modal
  FichaAcercaDe.Wait()

    ;espera y lo cierra
  FichaAcercaDe.Close()

  thamlib.EsLlamadaFalso()
                                                                  endif
      ; bloque empleado para cerrar la ficha desde el menú del sistema
      case eventInfo.ID() = MenuControlClose :
                                             eventInfo.setErrorCode(1)
                              miMando.hide()
                                             formReturn(True)
    endswitch
  endif
```

```
endMethod
```

MenúPrincipal

NAUFRAGI.MenúPrincipal

Es un procedimiento personalizado asociado a la ficha. Llamado por **NAUFRAGI.PáginaNaufragios.open** para visualizar la opción Naufragios del menú.

```
proc MenúPrincipal()

  Var
    MenúPrincipal     Menu
    MenúEmergente1,
    MenúEmergente2,
    MenúEmergente3    popUpMenu
  endVar

  ; este procedimiento personalizado muestra el menú principal de naufragios
  MenúEmergente1.addText("&Aceptar")
  MenúEmergente1.addText("&Cancelar")
  MenúEmergente1.addText("&Eliminar")
  MenúPrincipal.addPopUp("&Selecciones", MenúEmergente1)

  MenúEmergente2.addtext("&Primero\tCtrl+F11")
  MenúEmergente2.addtext("&Ultimo\tCtrl+F12")
  MenúEmergente2.addtext("&Siguiente\tF12")
  MenúEmergente2.addtext("&Anterior\tF11")
  MenúPrincipal.addPopUp("&Registro", MenúEmergente2)

  MenúEmergente3.addText("&Contenido")
  MenúEmergente3.addText("&Uso de la Ayuda")
  MenúEmergente3.addText("Acerca de &THAM")
  MenúPrincipal.addPopUp("&Ayuda", MenúEmergente3)

  MenúPrincipal.show()
endproc
```

NAUFRAGI Ventana Var

Las Variables declaradas en una ventana de variables son accesibles para todos los objetos de la ficha.

```
Var
   NombreInterfaz String
   NombreFicha,
   miMando        Form
   PuedeCerrar,
   FueLlamada     Logical
   interfaz       UIObject
   thamLib        Library
endVar
```

NAUFRAGI Ventana Const

Las constantes declaradas en una ventana de variables son accesibles para todos los objetos de la ficha.

```
Const
  FicheroAyuda = ":TRABAJO:usuario.hlp"
endConst
```

NAUFRAGI Ventana Uses

En la ventana *Uses* se declaran rutinas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
  CrearMenú(var NombreFicha Form, NombrePágina String, interfaz UIObject)
  MensajeTHAM(NombreFicha String, NombrePágina String, NúmeroDeObjeto
String)
  EstablecerValoresPágina(NombreFicha String, NombrePágina String) Logical

  NombreDePágina(interfaz UIObject) String
  HaLLamadoaRetornar() Logical
  EsLlamadaVerdadero()
  EsLlamadaFalso()
endUses
```

NAUFRAGI.PáginaNaufragios

Métodos

open
close
setFocus
LimpiarDatosPágina

ventana Uses

Uses

open

NAUFRAGI.PáginaNaufragios.open

Esta ficha debe ser llamada por VACACIÓN.FSL; ésta no puede ejecutarse directamente. Abre la biblioteca THAMLIB.LSL si se encuentra en el directorio WORK. Abre una Barra Rápida personalizado (MANDOVID.FSL) y la sitúa en la esquina superior izquierda de la pantalla. Llama al procedimiento **MenúPrincipal** para visualizar el menú de selección Naufragios.

```
method open(var eventInfo Event)
  ; Este método no abrirá la ficha naufragi.fsl a no ser que
  ; haya sido llamdada por VACACIÓN.fsl. La Variable EsLlamada de thamlib
  ; debe ser verdadera para abrir la fcha. El método EsLlamadaVerdadero(),
  ; invocada por el método pusbutton de VACACIÓN, se encarga de esta tarea.

  delayScreenUpdates(Yes)
  ; abre la biblioteca (asegúrese de que se encuentra en el directorio de
trabajo)
  if not thamlib.Open("thamlib", GlobalToDesktop) then
    msgStop("Fallo", "No pude abrir THAMLIB.")
    formReturn(false)
  else
    FueLlamada = thamlib.HaLlamadoaRetornar()
    if FueLlamada = False then
      msgStop("¡Alto!", "No puede abrir esta ficha directamente. Debe ser
abierta por VACACIÓN.fsl")
      close()
    else
      ; muestra el menú personalizado
      MenúPrincipal()
      NombreFicha.attach()
    endif
  endif
  delayScreenUpdates(No)
endMethod
```

close

NAUFRAGI.PáginaNaufragios.close

Cierra la Barra Rápida MANDOVID.FSL.

```
method close(var eventInfo Event)
   ; el mando está abierto
   if thamlib.EstáMandoAbierto() then
       ; cierra el mando
     miMando.close()
     thamlib.MandoEstáCerrado()
    endif
endMethod
```

setFocus

NAUFRAGI.PáginaNaufragios.setFocus

Visualiza la Barra Rápida MANDOVID.FSL, abriendola primero si fuese necesario.

```
method setFocus(var eventInfo Event)
  Var
    x,
    y,
    w,
    h LongInt
  endVar
  ; el mando ya está abierto
  if thamlib.EstáMandoAbierto() then
    if miMando.isAssigned() then
      ; mando abierto -- lo asociamos a esta ficha
      miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    else
      ; asociamos el mando
      miMando.attach(thamlib.NombreMando())
      miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    endif
  else
    ; mando cerrado -- lo abrimos
    miMando.Open("MandoVid.fsl")
    miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    thamlib.MandoEstáAbierto()
  endif
endMethod
```

LimpiarDatosPágina

NAUFRAGI.PáginaNaufragios.LimpiarDatosPágina

Borra todos los objetos campo de la página ajustando sus valores a una cadena vacía ("").

```
method LimpiarDatosPágina(NombrePágina String) Logical
   ; Este método personalizado limpia todos los objetos de la página
   ; que ha sido seleccionada.

   Var
      ObjetosPágina     TCursor
      ObjetosCampo      UIObject
   endVar

   ; lista todos los campos seleccionables de la página
   thamlib.ListaObjetosDePágina(NombreFicha.name, NombrePágina)
   ObjetosPágina.open("listaObj.db")
   ; limpia los objetos seleccionados
   scan ObjetosPágina :
      ObjetosCampo.attach(ObjetosPágina."Vía de Acceso")
      if ObjetosCampo  <> "" then
         ObjetosCampo = ""
      endIf
   endScan
   ; limpia los valores ficha.página en valconsu.db
   thamlib.LimpiarValoresPágina(NombreFicha.Name, NombrePágina)
   RETURN TRUE
endMethod
```

La Ventana Uses de NAUFRAGI.PáginaNaufragios

En la ventana *Uses* se declaran rutinas estándar empleadas por esta ficha. Las rutinas pueden almacenarse en una DLL, en una biblioteca ObjectPAL, o en otra ficha. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que todos los métodos están escritos en ObjectPAL,en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   ListaObjetosDePágina(NombreFicha String, NombrePágina String) Logical
   LimpiarValoresPágina(NombreFicha String, NombrePágina String)
   EstablecerNombreFichaLlamar(const NombreFichaLlamador String, xPos
LongInt, yPos LongInt)
   EstáMandoAbierto() Logical
   MandoEstáAbierto()
   MandoEstáCerrado()
   NombreMando() String
endUses
```

newValue

NAUFRAGI.PáginaNaufragios.Nombre_Barco.newValue

Reajusta el valor del objeto campo *Nombre_Barco* cuando se selecciona un barco nuevo en la lista desplegable Naufragio.

```
method newValue(var eventInfo Event)
   ; el campo naufragio necesita actualizarse para contener
   ; el nuevo valor de Nombre Barco
   if self <> naufragio then
     naufragio = self
   endif
endMethod
```

NAUFRAGI.PáginaNaufragios.Naufragio

Métodos

changeValue
newValue

changeValue

NAUFRAGI.PáginaNaufragios.Naufragio.changeValue

Comprueba si el valor de este objeto campo coincide con el valor de *Nombre_Barco*. Si no, resincroniza los dos campos.

```
method changeValue(var eventInfo ValueEvent)
  var
    NuevoValor String
    barcoTc TCursor
  endVar
  ; ¿Cúal es el valor nuevo?
  NuevoValor = eventInfo.newValue()
  ; si el nuevo valor difiere del incluido en el campo Nombrebarco
  ; y no está blanco, sincronizamos los campos
  if NuevoValor <> NombreBarco and NuevoValor <> ""   then
    ; sincronizamos los campos
    BarcoTc.attach(NombreBarco)
    BarcoTc.Locate("Nombre del barco", NuevoValor)
    NombreBarco.moveToRecord(BarcoTc)
  endif
endMethod
```

newValue

NAUFRAGI.PáginaNaufragios.Naufragio.newValue

Envía el nuevo valor de este campo objeto para accionar su método estándar **changeValue**.

```
method newValue(var eventInfo Event)
   ; Acepta el campo si lo ha variado el usuario
   ; para provocar un changeValue
   if eventInfo.reason() = EditValue then
      self.action(EditCommitField)
   endif
endMethod
```

NAUFRAGI.PáginaNaufragios.ListaNaufragios

MÉTODOS

open
pushButton

open

NAUFRAGI.PáginaNaufragios.ListaNaufragios.open

Rellena esta lista con entradas de los campos Nombre del Barco de NAUFRAGI.DB.

```
method open(var eventInfo Event)
   self.dataSource = "[Naufragi.Nombre del Barco]"
endMethod
```

pushButton

NAUFRAGI.PáginaNaufragios.ListaNaufragios.pushButton

Envía el valor seleccionado de *ListaNaufragios* a *Naufragio*.

```
method pushButton(var eventInfo Event)
   Naufragio.action(EditCommitField)
endMethod
```

pushButton

NAUFRAGI.PáginaNaufragios.BotónAceptar.pushButton

Emplea **THAMLIB.setPageValues** para enviar los criterios seleccionados en Naufragio y regresa a VACACIÓN.FSL. También oculta la Barra Rápida personalizada.

```
method pushButton(var eventInfo Event)
   eventInfo.getTarget(interfaz)
   ; asigna los valores de la página a valconsu.db
   thamlib.EstablecerValoresPágina(NombreFicha.name,
thamlib.NombreDePágina(interfaz))
   ; oculta el mando
   miMando.hide()
   formReturn(True)
endMethod
```

pushButton

NAUFRAGI.PáginaNaufragios.BotónAyuda.pushButton

Muestra la Ayuda utilizando **helpShowContext**. El valor 20005 es un identificador de contexto para un elemento del archivo de ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method pushButton(var eventInfo Event)
  helpShowContext(FicheroAyuda, 20005)
endMethod
```

pushButton

NAUFRAGI.PáginaNaufragios.BotónCancelar.pushButton

Oculta la Barra Rápida personalizada y regersa a VACACIÓN.FSL sin efectuar ninguna otra operación.

```
method pushButton(var eventInfo Event)
   ; oculta el mando
   miMando.hide()
   ; retorna a Vacación sin realizar procesos
   formReturn(True)
endmethod
```

pushButton

NAUFRAGI.PáginaNaufragios.BotónLimpiarConsulta.pushButton

Borra los valores de Naufragio selecionados actualmente empleando **THAMLIB.LimpiarDatosPágina.** Ésta se ejecuta cuando se selecciona el botón Limpiar Consulta.

```
method pushButton(var eventInfo Event)
   ; ¿Qué objeto es?
   eventInfo.getTarget(interfaz)
   ; limpia los objetos de tipo campo y los valores de valconsu.db
   LimpiarDatosPágina(thamlib.NombreDePágina(interfaz))
endmethod
```

VIDAMAR

Métodos

mouseEnter
mouseExit
mouseRightUp
keyPhysical
menuAction

Procedimientos

MenúUsuario

Variables , Constantes, y ventanas Uses

ventana de Variables
ventana de Constantes
ventana Uses

Métodos y procedimiemtos de THAMLIB.LSL

LimpiarValoresPágina
crearMenú
MensajeTHAM
EstablecerValoresPágina
NombreDePágina
HaLlamadoARetornar
EsLlamadaVerdadero
EsLlamadaFalso

mouseEnter

VIDAMAR.mouseEnter

Es una rutina de mensajes al nivel de ficha que muestra nombres de objetos (excluyendo las clases Texto y Mapa de bits) en la barra de estado dependiendo de la posición del cursor del ratón. Emplea el método de la biblioteca THAMLIB, **THAMLIB.MensajeTHAM.**

```
method mouseEnter(var eventInfo MouseEvent)

{
  El sistema de mensajes a nivel de ficha examina la entrada del ratón
(mouseEnter) en el ámbito
  de un objeto y selecciona el mensaje apropiado de la tabla menayuda.db
bajo el control del
  método mensajeTham de la biblioteca
}

  ; evita llamadas repetidas a mouseEnter
  if eventInfo.isPreFilter() then
    eventInfo.getTarget(interfaz)  ; ¿Qúe objeto es?
    NombreInterfaz = interfaz.Name
    ; excluimos los objetos de texto y bitmaps
    if interfaz.name <> self.name AND  ; elimina el mensaje si es la ficha
    (interfaz.class = "Field" OR interfaz.class = "Multirecord" OR
interfaz.class = "Button") then
       thamlib.mensajeTham(NombreFicha.name,
thamlib.NombreDePágina(interfaz), NombreInterfaz)
    endif
  endif
endmethod
```

mouseExit

VIDAMAR.mouseExit

Borra el mensaje de la barra de estado de **mouseEnter** cuando el cursor abandona la zona de un objeto.

```
method mouseExit(var eventInfo MouseEvent)
   ; evita sucesivas llamadas a mouseExit
   if eventInfo.isPreFilter() then
     eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
     if interfaz.class <> "Text" and interfaz.class <> "Bitmap" then
       message("")
     endif
   endif
endmethod
```

mouseRightUp

VIDAMAR.mouseRightUp

LLama al método **CrearMenú** de la biblioteca THAMLIB cuando se suelta el botón derecho del ratón. Esto crea el menú de Ayuda del código. Si se hace clic sobre un objeto texto o sobre un Mapa de bits, aparecera el menú de ayuda para los objetos que contiene debido a que estos objetos no tienen código asociado.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Ejecuta el método thamlib.CrearMenúCalls() encargado de visualizar
   ; el menú de ayuda de un objeto.

   ; evita llamadas repetidas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
       if interfaz.name <> "CuadroDeBandera" and interfaz.name <>
"CuadroDeBandera1" then
         while
            interfaz.class = "Text" or interfaz.class = "Bitmap"
            interfaz.attach(interfaz.ContainerName)
            endWhile
       endif
      thamlib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endmethod
```

keyPhysical

VIDAMAR.keyPhysical

Captura pulsaciones de tecla específicas y después efectúa según preceda una llamada a **pushButton** o **moveTo** (o llama al sistema de Ayuda, si se pulsa *F1*). Por ejemplo, *Alt+A* llama a **BotónAceptar.pushButton**.

```
method keyPhysical(var eventInfo KeyEvent)
  var
    laTecla String
  endvar
  laTecla = eventInfo.vChar()  ; dime que tecla ha sido pulsada
  if eventInfo.isPreFilter() then
    if eventInfo.isAltKeyDown() then
      switch
        case laTecla = "A" or laTecla = "a" : disableDefault

    BotónAceptar.pushbutton()
        case laTecla = "C" or laTecla = "c" : disableDefault

BotónCancelar.pushbutton()
        case laTecla = "E" or laTecla = "e" : disableDefault

BotónLimpiarConsulta.pushbutton()
        case laTecla = "M" or laTecla = "m" : disableDefault

    CampoNombreComun.moveto()
        case laTecla = "S" or laTecla = "s" : disableDefault

    CampoNombreLatín.moveto()
        otherwise                                                    :
doDefault
        endswitch
     else
      switch
        case laTecla = "VK_F1"                          : disableDefault


    helpShowContext(ficheroAyuda, 20002)
        otherwise : doDefault
      endSwitch
    endif
  endif
endmethod
```

menuAction

VIDAMAR.menuAction

Comprueba la cadena devuelta por una opción del menú y llama al método apropiado. Por ejemplo, eligiendo el elemento Cancelar llama a **BotónCancelar.pushButton.** Eligiendo Siguiente llama a **miMado.BotónIrSiguiente.pushButton()**. Un ampersand (&) en la cadena que define el elemento del menú hace que la letra que va a continuación de éste sea una tecla activa. Por ejemplo, &Cancelar llama a **BotónCancelar.pushButton** cuando se pulsa *Alt+C*.

```
method menuAction(var eventInfo MenuEvent)
  var
    OpciónMenú String
    Acercade Form
  endvar
  if eventInfo.isPrefilter() then
    doDefault
  else
    ;  ¿Qué opción ha sido seleccionada?
  OpciónMenú = eventInfo.menuChoice()
    switch
      case OpciónMenú = "&Aceptar"             : BotónAceptar.pushbutton()
      case OpciónMenú = "&Cancelar"           : BotónCancelar.pushbutton()
      case OpciónMenú = "&Eliminar"           :
BotónLimpiarConsulta.moveto()
      case OpciónMenú = "&Primero\tCtrl+F11"  :
miMando.BotónIrPrimero.pushButton()
      case OpciónMenú = "&Ultimo\tCtrl+F12"   :
miMando.BotónIrUltimo.pushButton()
      case OpciónMenú = "&Siguiente\tF12"     :
miMando.BotónIrSiguiente.pushButton()
      case OpciónMenú = "&Previous\tF11"      :
miMando.BotónIrAnterior.pushButton()
      case OpciónMenú = "&Indice"                   :
helpShowContext(ficheroAyuda, 20002)
      case OpciónMenú = "&Uso de la Ayuda"    : helpOnHelp()
      case OpciónMenú = "&Acerca de THAM"     :
thamLib.EsLlamadaVerdadero()
                                                                    if
Acercade.open("SobTham.fsl") then

  ; abrimos un cuadro de diálogo modal

  Acercade.Wait()

    ; esperamos y lo cerramos

    Acercade.Close()

    thamLib.EsLlamadaFalso()
                                                          endif
      ; bloque empleado para cerrar la ficha co el menú del sistema
      case eventInfo.ID() = MenuControlClose  : eventInfo.setErrorCode(1)
                                                miMando.hide()
```

```
      FormReturn(True)
      endswitch
   endif
endmethod
```

MenúUsuario

VIDAMAR.MenúUsuario

Éste es llamado por **VIDAMAR.PáginaVidaMar.open** para construir y visualizar el menú de selección de *VIDAMAR*.

```
proc MenúUsuario()

  Var
    MenúPrincipal Menu
    MenúEmergente1, MenúEmergente2, MenúEmergente3 popUpMenu
  endVar

  ; este procedimiento muestra un menú personalizado
  MenúEmergente1.addText("&Aceptar")
  MenúEmergente1.addText("&Cancelar")
  MenúEmergente1.addText("&Eliminar")
  MenúPrincipal.addPopUp("&Selecciones", MenúEmergente1)

  MenúEmergente2.addtext("&Primero\tCtrl+F11")
  MenúEmergente2.addtext("&Ultimo\tCtrl+F12")
  MenúEmergente2.addtext("&Siguiente\tF12")
  MenúEmergente2.addtext("&Anterior\tF11")
  MenúPrincipal.addPopUp("&Registro", MenúEmergente2)

  MenúEmergente3.addText("&Contenido")
  MenúEmergente3.addText("&Uso de la Ayuda")
  MenúEmergente3.addText("Acerca de &THAM")
  MenúPrincipal.addPopUp("&Ayuda", MenúEmergente3)

  MenúPrincipal.show()
endproc
```

VIDAMAR Ventana Var

Las variables declaradas en la ventana de Variables de ficha son accesibles para todos los objetos contenidos en la ficha.

```
Var
   NombreInterfaz,
   Selección,
   NuevoValor                String
   Interfaz                  UIObject
   NombreFicha,
   miMando             Form
   FueLlamada                Logical
   Thamlib             Library
   pez                 Tcursor
endVar
```

VIDAMAR Ventana Const

Las constantes declaradas en la ventana de variables de ficha son accesibles para todos los objetos de la ficha.

```
Const
   FicheroAyuda = ":trabajo:usuario.hlp" ; vía de acceso al fichero de ayua
endConst
```

VIDAMAR Ventana Uses

La ventana *Uses* declara rutinas externas empleadas por esta ficha. Las rutinas se pueden almacenar en una DDL, en una biblioteca ObjectPAL, o en otra ficha. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, y no en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal

  LimpiarValoresPágina(NombreFicha String, NombrePágina String)
  CrearMenú(var NombreFicha Form, NombrePágina String, interfaz UIObject)
  MensajeTHAM(NombreFicha String, NombrePágina String, interfaz String)
  EstablecerValoresPágina(NombreFicha String, NombrePágina String) Logical

  NombreDePágina(interfaz UIObject) String
  HaLlamadoaRetornar() Logical
  EsLlamadaVerdadero()
  EsLlamadaFalso()

enduses
```

VIDAMAR.PáginaVidaMar

MÉTODOS

open
close
setFocus

ventana Uses

Ventana Uses

open

VIDAMAR.PáginaVidaMar.open

Esta ficha debe ser llamada desde VACACIÓN.FSL; no puede llamarse directamente. Abre la biblioteca si se encuentra en el directorio de TRABAJO. Abre una Barra Rápida personalizada (MANDOVID.FSL) y la sitúa en la esquina superior derecha de la pantalla. Esta llama al procedimiento **MenúUsuario** para visualizar el menú de selección *VIDAMAR*.

```
method open(var eventInfo Event)
   ; Este método no abrirá la ficha vidamar.fsl a no ser que
   ; haya sido llamada por VACACIÓN.fsl. Para poder ser abierta la ficha
vidamar,
   ; la variable EsLlamada debe ser true
   ; Se emplea thamlib.EsLlamadaVerdadero() para asignar el valor True
   ; invocada desde el método estándar pushbutton de VACACION.fsl

   delayScreenUpdates(Yes)

   ; abre la biblioteca (asegúrese de que está en el directorio de trabajo)

   if not thamlib.Open("thamlib", GlobalToDesktop) then
      msgStop("Fallo", "No pude abrir THAMLIB.")
      formReturn(false)
    else
      FueLlamada = thamLib.HaLlamadoaRetornar()
      if fueLlamada = False then
        msgStop("¡Alto!", "No puede abrir esta ficha directamente. Debe ser
abierta desde VACACION.fsl")
        close()
      else
        ; muestra el menú diseñado
        MenúUsuario()
        NombreFicha.attach()
      endif
   endif
   delayScreenUpdates(No)
endmethod
```

close

VIDAMAR.PáginaVidaMar.close

Cierra la Barra Rápida asociada MANDOVID.FSL.

```
method close(var eventInfo Event)
  ; el mando está abierto
  if thamlib.EstáMandoAbierto() then
     ; cierra el mando
    miMando.close()
    thamlib.MandoEstáCerrado()
  endif
endmethod
```

setFocus

VIDAMAR.PáginaVidaMar.setFocus

Muestra la Barra Rápida MANDOVID.FSL, abriéndola primero si fuese necesrio.

```
method setFocus(var eventInfo Event)
  Var
    x,
    y,
    w,
    h  LongInt
  endVar

  if thamLib.EstáMandoAbierto() then
    if miMando.isAssigned()  then
      ; Mando abierto-- lo asociamos a la ficha
      miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 1050)
    else
      ; asociamos el mando
      miMando.attach(thamlib.NombreMando())
      miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 1050)
    endif
  else
    ; el mando no está abierto -- lo abrimos
    miMando.Open("MandoVid.fsl")
    miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 1050)
    thamLib.MandoEstáAbierto()
  endif
endmethod
```

VIDAMAR.PáginaVidaMar Ventana Uses

En la ventana Uses se declaran rutinas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
  EstablecerNombreFichaLlamar(const NombreFichaLlamador String, xPos
LongInt, yPos LongInt)
  MandoEstáAbierto()
  MandoEstáCerrado()
  EstáMandoAbierto() Logical
   NombreMando() String
   ListaObjetosDePágina(NombreFicha String, NombrePágina String) Logical
endUses
```

VIDAMAR.PáginaVidaMar.CampoNombreComún

MÉTODOS

newValue
changeValue

newValue

VIDAMAR.PáginaVidaMar.CampoNombreComún.newValue

Envia el valor de Nombre Común de las especies seleccionadas al campo objeto *CampoNombreComún*. Este valor se emplea para buscar los lugares correspondientes.

```
method newValue(var eventInfo Event)
   ; De acuerdo, acepta el campo si lo ha variado el usuario
   if eventInfo.reason() = EditValue then
      self.action(EditCommitField)
   endif
endmethod
```

changeValue

VIDAMAR.PáginaVidaMar.CampoNombreComún.changeValue

Comprueba si el campo Nombre Comun de Especies ha cambiado. En ese caso, todos los valores de campo se actualizan.

```
method changeValue(var eventInfo ValueEvent)
   ; ¿Cúal es el nuevo valor?
   NuevoValor = eventInfo.NewValue()
   ; si el nuevo valor no es el mismo que el incluido en el campo nombre
común
   if NuevoValor <> registroVidaMar.registroPeces.Nombre_Común then
      Pez.attach(RegistroVidaMar.registroPeces)
      Pez.Locate("Nombre Común", NuevoValor)
      RegistroVidaMar.registroPeces.moveToRecord(pez)
      CampoNombreLatín = Nombre_Especie
   endif
endmethod
```

open

VIDAMAR.PáginaVidaMar.ListaNombreComún.open

Rellena la lista con las entradas del campo Nombre_Común de VIDAMAR.DB.

```
method open(var eventInfo Event)
   self.dataSource = "[VidaMar.Nombre Común]"
endmethod
```

VIDAMAR.PáginaVidaMar.CampoNombreLatín

Métodos

newValue
changeValue

newValue

VIDAMAR.PáginaVidaMar.CampoNombreLatín.newValue

Éste envía el valor del nombre en Latín de las especies seleccionadas del objeto campo *CampoNombreLatín*. Este valor se emplea después en **VACACIÓN.destqbe** para encontrar los lugares correspondientes.

```
method newValue(var eventInfo Event)
   ; De acuerdo, acepta el campo si lo ha variado el usuario
   if eventInfo.reason() = EditValue then
      self.action(EditCommitField)
   endif
endmethod
```

changeValue

VIDAMAR.PáginaVidaMar.CampoNombreLatín.changeValue

Comprueba si el campo Nombre_Especie de la especie seleccionada ha cambiado. Si fuese así, el campo Nombre Común se reajusta para que coincida y se muestra.

```
method changeValue(var eventInfo ValueEvent)
   ; ¿Cúal es el nuevo valor?
   NuevoValor = eventInfo.NewValue()
   ; si el nuevo valor no es el mismo que el incluido en el campo nombre
especie
   if NuevoValor <> Nombre_Especie then
      Pez.attach(RegistroVidaMar.registroPeces)
      Pez.Locate("Nombre Especie", NuevoValor)
      RegistroVidaMar.registroPeces.moveToRecord(Pez)
      CampoNombreComún = Pez."Nombre Común"
   endif
endmethod
```

open

VIDAMAR.PáginaVidaMar.ListaNombreLatín.open

Rellena la lista con las entradas del campo Nombre Especie de VIDAMAR.DB.

```
method open(var eventInfo Event)
   self.dataSource = "[vidamar.Nombre Especie]"
endmethod
```

newValue

VIDAMAR.PáginaVidaMar.ImagenPez.newValue

Si el campo seleccionado Nombre_Común cambia, se reajustan los objeto campo *CampoNombreComún* y *CampoNombreLatín* y se muestra el grafico correspondiente de VIDAMAR.DB.

```
method newValue(var eventInfo Event)
   if Nombre_Común <> CampoNombreComun then
      CampoNombreComún = Nombre_Común
      CampoNombreLatín = Nombre_Especie
   endif
endmethod
```

pushButton

VIDAMAR.PáginaVidaMar.BotónAceptar.pushButton

Emplea **THAMLIB.EstablecerValoresPágina** para enviar las especies de vida marina seleccionadas y regresa a VACACIÓN.FSL. También oculta la Barra Rápida personalizada.

```
method pushButton(var eventInfo Event)
   eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
   ; asigna los valores de campo a valconsu.db
   thamlib.EstablecerValoresPágina(NombreFicha.name,
thamlib.NombreDePágina(interfaz))
   ; ocultamos el mando
   miMando.hide()
   formReturn(True)
endmethod
```

pushButton

VIDAMAR.PáginaVidaMar.BotónAyuda.pushButton

Muestra la Ayuda llamando a **helpShowContext**. El valor 20002 es un identificador de contexto de un elemento del archivo de ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method pushButton(var eventInfo Event)
  helpShowContext(ficheroAyuda, 20002)
endmethod
```

pushButton

VIDAMAR.PáginaVidaMar.BotónCancelar.pushButton

Oculta la Barra Rápida personalizada y regresa a VACACIÓN.FSL sin efectuar ninguna otra operación.

```
method pushButton(var eventInfo Event)
   ; ocultamos el mando
   miMando.hide()
   ; vuleve a VACACION sin realizar procesos
   formReturn(True)
endmethod
```

pushButton

VIDAMAR.PáginaVidaMar.BotónLimpiarConsulta.pushButton

Borra los valores de vida marina seleccionados actualmente empleando **THAMLIB.LimpiarValoresPágina.** Ésta se ejecuta cuando se selecciona el botón Limpiar Consulta**.**

```
method pushButton(var eventInfo Event)
   ; ¿Qué objeto es?
   eventInfo.getTarget(interfaz)
   ; limpia los valores correspondientes a la ficha.página de valconsu.db
   thamLib.LimpiarValoresPágina(NombreFicha.Name,
thamlib.NombreDePágina(interfaz))
endmethod
```

SOBTHAM

Métodos

open
mouseRightUp
keyPhysical

Variables, Constantes, y Ventanas Uses

ventana de Variables
ventana de constantes
ventanas Uses

Métodos y precedimientos de la biblioteca THAMLIB.LSL

crearMenú
NombrePágina
HaLlamadoARetornar

open

SOBTHAM.open

La ficha SOBTHAM debe llamarse desde VACACIÓN.FSL; ésta no puede llamarse directamente. Abre la biblioteca SOBTHAM.FSL si la encuentra en el directorio de TRABAJO.

```
; Este método no permite que se abra esta ficha a no ser que
; haya sido llamada por VACACION.fsl. La variable EsLlamada incluida en la
biblioeca
; thamlib.lsl debe ser verdadera.
; EsLlamada recibe el valor True en el método thamlib.EsLlamadaVerdadero()
invocada
; desde el método the pushbutton de VACACION.

method open(var eventInfo Event)
   if not eventInfo.isPreFilter() then
      ; abre la biblioteca (asegúrese de que esta se encuentra en el
directorio de trabajo)
     if not thamLib.open("ThamLib", globalToDesktop) then
        msgStop("Fallo", "No pude abrir Thamlib.")
        close()
     else
        EsLlamada = thamLib.HaLlamadoaRetornar()
        if EsLlamada = False then
          msgStop("¡Alto!", "No puede abrirse esta ficha directamente. Debe
ser abierta desde VACACION.fsl.")
          close()
        else
          NombreFicha.attach()
        endif
     endif
   endif
endmethod
```

mouseRightUp

SOBTHAM.mouseRightUp

Llama al método **CrearMenú** de la biblioteca THAMLIB cuando se suelta el botón derecho del ratón. Esto crea el menú de Ayuda del código. Si se hace clic sobre un objeto texto o sobre un Mapa de bits, aparecerá el menú de ayuda para los objetos que contiene debido a que estos objetos no tienen código asociado. Esto crea el menú de Ayuda.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Llama al método thamlib.CrearMenú() para mostrar el menú de ayuda
   ; a nivel de objeto.

   ; evita llamadas sucesivas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      ; si no es el bitmap CuadroDeBandera,
      ; busca el nombre del contenedor mientras se trate de un texto o u
bitmap
      if interfaz.name <> "CuadroDeBandera" then
         while interfaz.class = "Text" or interfaz.class = "Bitmap"
            interfaz.attach(interfaz.ContainerName)
            endWhile
      endif
      thamLib.crearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endMethod
```

keyPhysical

SOBTHAM.keyPhysical

Captura *F1* para llamar a **helpShowContext.** El valor 20000 es un identificador de contexto para un elemento del archivo de ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method keyPhysical(var eventInfo KeyEvent)
   ; si se pulsa F1, muestra la ayuda de VACACION
   if eventInfo.isFirstTime() then
      if eventInfo.vChar() = "VK_F1" then
         disableDefault
         helpShowContext(FicheroAyuda, 20000)
      endif
   endif
endmethod
```

SOBTHAM Ventana Var

Las Variables declaradas en la ventana de Variables son accesibles para todos los objetos contenidos en la ficha.

```
Var
   NombreFichaLlamador Form
   interfaz                         UIObject
   NombreFicha                      Form
   thamLib                          Library
   FueLlamado                       Logical
endVar
```

SOBTHAM Ventana Const

Las constantes declaradas en la ventana de Constantes son accesibles para todos los objetos de la ficha.

```
Const
   FicheroAyuda = ":TRABAJO:usuario.hlp"
endConst
```

SOBTHAM Ventana Uses

En la ventana *Uses* se declaran rutinas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   crearMenú(var NombreFicha Form, NombrePágina String, Interfaz UIObject)
   NombreDePágina(interfaz UIObject) String
   HaLlamadoaRetornar() Logical
endUses
```

pushButton

SOBTHAM.FichaSobre.BotónAceptar.pushButton

Devuelve el control a VACACIÓN.

```
method pushButton(var eventInfo Event)
   formReturn()
endmethod
```

VACACIÓN

Métodos

close
keyPhysical
mouseEnter
mouseExit
mouseRightUp
destqbe

Variables, Constantes, y Ventanas Uses

Ventana Var
Ventana Const
Ventana Uses

Métodos y procedimientos de la biblioteca THAMLIB.LSL

LimpiarValoresPágina
crearMenú
MensajeTHAM
NombreDePágina
EsLlamadaVerdadero
EsLlamadaFalso

close

VACACIÓN.close

Restaura la Barra Rápida de Paradox y elimina el menú personalizado.

```
method close(var eventInfo Event)
   if eventInfo.isPrefilter() then
      doDefault
   else
      ; cierra la ficha vidamar
      if fichaVidaMar.isAssigned() then
         fichaVidaMar.close()
      endif
      ; cierra la ficha lugares
      if fichaLugares.isAssigned() then
           fichaLugares.close()
      endif
      ; cierra la ficha naufragi
      if fichaNaufragios.isAssigned() then
           fichaNaufragios.close()
      endif
      ; cierra la ficha pedtham
      if fichaReservas.isAssigned() then
           thamlib.puedeCerrarFicha()
           fichaReservas.close()
      endif
      ; reestablece la barra rápida
      showSpeedBar()
      ; borra el menú personalizado y reestablece el menú de Paradox
      removeMenu()
      ; cierra la ayuda
      helpQuit(ficheroAyuda)
      ; cierra la ficha principal
   endif
endmethod
```

keyPhysical

VACACIÓN.keyPhysical

Captura pulsaciones de tecla y las convierte en llamadas a métodos, si procede.

```
method keyPhysical(var eventInfo KeyEvent)
  ; se emplea el método estándar keyphysical en la ficha para capturar las
teclas rápidas
  ; Capturará la pulsación de la tela  F1, que llama a la ayuda.

  Var
    laTecla String
    laPágina String
  endVar
  if eventInfo.isPreFilter() then
    laPágina = thamlib.nombreDePágina(active)       ; dime el nombre de la
página activa
    laTecla = eventInfo.vChar()                                    ; dime que
tecla se ha pulsado
    switch
      case laPágina = "PáginaPrincipal" :
        if eventInfo.isAltKeyDown() then
         switch
               case laTecla = "E" or laTecla = "e" :     disableDefault

        PáginaPrincipal.BotónLimpiarConsulta.pushButton()
                    case laTecla = "V" or laTecla = "v" :  disableDefault

        PáginaPrincipal.BotónVidaMarina.pushButton()
               case laTecla = "P" or laTecla = "p" :  disableDefault

        PáginaPrincipal.BotónProcesar.pushButton()
               case laTecla = "L" or laTecla = "l" :  disableDefault

        PáginaPrincipal.BotónLugares.pushButton()
                    case laTecla = "N" or laTecla = "n" :  disableDefault

        PáginaPrincipal.BotónNaufragio.pushButton()
               case laTecla = "S" or laTecla = "s" :  disableDefault

        PáginaPrincipal.BotónSalir.pushButton()
               otherwise
  :  doDefault
               endswitch
         else
         switch
               case laTecla = "VK_F1" :      disableDefault

  helpShowContext(ficheroAyuda, 20000)
               otherwise                        :   doDefault
         endSwitch
        endif
       case laPágina = "páginaResultados" :
         if eventInfo.isAltKeyDown() then
```

```
           switch
                 case laTecla = "R" or laTecla = "r" :  disableDefault

           páginaResultados.BotónReservar.pushButton()
                 case laTecla = "P" or laTecla = "p" :  disableDefault

           páginaResultados.BotónPáginaPrincipal.pushButton()
                 case laTecla = "S" or laTecla = "s" :  disableDefault

           páginaResultados.BotónSalir.pushButton()
                 otherwise
   :  doDefault
                 endswitch
            else
             switch
                 case laTecla = "VK_F1" :  disableDefault

helpShowContext(ficheroAyuda, 20001)
                 otherwise                        :  doDefault
            endSwitch
           endif
      endswitch
   else
     doDefault
   endif
endmethod
```

mouseEnter

VACACIÓN.mouseEnter

Es una rutina de mensajes a nivel de ficha que muestra los nombres de objetos en la barra de estado dependiendo de la posición del cursor del ratón. Emplea el método de la biblioteca THAMLIB , **THAMLib.MensajeTHAM.**

```
; El sistema de mensajes a nivel de ficha examina la entrada del ratón
(mouseEnter) en el ámbito
; de un objeto y selecciona el mensaje apropiado de la tabla menayuda.db
; bajo el control del método mensajeTham de la biblioteca

method mouseEnter(var eventInfo MouseEvent)
   ; previene que se repitan sucesivas llamadas al método estándar MouseEnter
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(Interfaz)  ; ¿Qué objeto es?
      nombreInterfaz = Interfaz.Name
      ; excluimos los objetos de texto y bitmaps
      if Interfaz.name <> self.name AND  ; elimina el mensaje si es la ficha
         (Interfaz.class = "Field" OR Interfaz.class = "Multirecord" OR
          Interfaz.class = "Button") then
         thamlib.mensajeTham(NombreFicha.name,
thamlib.NombreDePágina(Interfaz), nombreInterfaz)
      endif
   endif
endmethod
```

mouseExit

VACACIÓN.mouseExit

Borra los mensajes de **mouseEnter** de la barra de estado cuando el cursor del ratón abandona la zona donde se encuentra un objeto.

```
method mouseExit(var eventInfo MouseEvent)
  ; El sistema de mensajes a nivel de ficha examina cada la salida del ratón
(MouseExit)
  ; del ámbito de un objeto y limpia el mensaje de la línea de estado.

  ; previene que se repitan sucesivas llamadas al método estándar mouseExit.
  if eventInfo.isPreFilter() then
    eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
    ; limpia el mensaje mostrado por el método mouseEnter de un objeto
    if interfaz.class <> "Text" and interfaz.class <> "Bitmap" then
      message("")
    endif
  endif
endmethod
```

mouseRightUp

VACACIÓN.mouseRightUp

Llama al método de la biblioteca THAMLIB **CrearMenú** cuando se suelta el botón derecho del ratón. Esto crea el menú de Ayuda de Código.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Ejecuta el método crearMenú de la biblioteca Thamlib que muestra un menú
   ; de ayuda a nivel de objeto

   ; previene que se repitan llamadas sucesivas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      ; si se trata de un texto o un bitmap, busca el nombre del contenedor
      ; a no ser que se trate del objeto CuadroDeBandera o CuadroDeBandera1,
      if ((interfaz.name <> "CuadroDeBandera") and (interfaz.name <>
"CuadroDeBandera1")) then
         while interfaz.class = "Text" or interfaz.class = "Bitmap"
            interfaz.attach(interfaz.ContainerName)
            endWhile
      endif
      thamlib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endmethod
```

DestinoConsulta

VACACIÓN.DestinoConsulta

Éste busca VALCONSU.DB y asigna valores de ésta en una consulta (**DestinoConsulta**) hecha en DESTINOS.DB. Los valores en VALCONSU.DB son ajustados cuando el usuario efectúa la selección en cualquiera de los siguientes archivos: LUGARES.FSL, NAUFRAGI.FSL, y VIDAMAR.FSL. Los destinos que correspondan se escriben en DESTIPOS.DB. Devolviendo False si VALCONSU.DB no puede ser abierta. **DestinoConsulta** se acciona eligiendo Process en **PáginaPrincipal.MenúPáginaPrincipal** o eligiendo el botón Process.

```
method DestinoConsulta() Logical

  var
    i SmallInt
    destinoConsulta         Query
    ValorConsulta                 TCursor
    NombreDestino,
    TemperaturaMedia,
    Alojamiento,
    VidaNocturna,
    NombreLugar,
    Característica,
    CategoríaBarco,
    InterésBarco,
    CondiciónBarco,
    NivelVACACION,
    NombreBarco,
    NombreComún,
    NombreEspecie                 String
  endvar

  ; El propósito de este método consiste en interrogar a la tabla
ValConsu.db y
  ; asignar los valores a una consulta que buscará los mismos destinos en
lugares.db.
  ; Los valores de ValConsu.db están determinados por las
opciones/selecciones que
  ; el usuario realice en la ficha Lugares y/o Naufragi y/o vidamar

  if not ValorConsulta.Open("ValConsu.db") then
    msgStop("Problema", "No pude abrir Valconsu.db")
    return false
  endif

  ; Examina ValConsu.db registro a registro y asigna los valores
  ; en las variables incluidas en DestinoConsulta.


  Scan ValorConsulta :
    switch
      case ValorConsulta."Nombre Campo" = "Alojamiento"
  : Alojamiento                    = ValorConsulta."Valor Consulta"
      case ValorConsulta."Nombre Campo" = "Caracteristica"           :
Característica              = ValorConsulta."Valor Consulta"
```

```
       case ValorConsulta."Nombre Campo" = "Vida Nocturna"       :
VidaNoctura                          = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Nombre Situacion"          :
NombreLugar                          = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Destino"
     : NombreDestino                 = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Categoria"                 :
CategoriaBarco               = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Interes"                   :
InterésBarco                         = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Estado"
  : CondiciónBarco                   = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Temperatura Media(C)"      :
TemperaturaMedia       = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Nombre del Barco"    :
NombreBarco                          = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Nombre Comun"              :
NombreComún                          = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Nombre Especie"      :
NombreEspecie                = ValorConsulta."Valor Consulta"
       case ValorConsulta."Nombre Campo" = "Nivel de VACACION"         :
NivelVACACION                        = ValorConsulta."Valor Consulta"
     endswitch
  endScan
  ; Define la consulta. Incluye en ella las variables asignadas desde
ValConsu.db

  Switch
    ; empleando el criterio de selección sobre naufragios y vida marina
    Case (not NombreBarco.isBlank() or
         not CategoríaBarco.isBlank() or
         not InterésBarco.isBlank() or
         not CondiciónBarco.isBlank())
     and (not NombreComún.isBlank()
         or not NombreEspecie.isBlank()) :

       message("primera opción")
       sleep(2000)
       DestinoConsulta = Query

          answer: :trabajo:destipos.db

          Destinos | N° Destino      | Destino              | Temperatura
Media(F) | Temperatura Media(C)    |
                        | Check _destino  | Check ~NombreDestino | Check
              | Check ~TemperaturaMedia |

          Destinos | Temperatura primavera(F) | Temperatura primavera(C) |
Temperatura verano(F) | Temperatura verano(C) |
                        | Check                          | Check
        | Check                 | Check                 |

          Destinos | Temperatura otoño(F) | Temperatura otoño(C) |
Temperatura invierno(F) | Temperatura invierno(C) |
```

```
                        | Check                    | Check                    |
Check                          | Check                             |

          Destinos | Alojamiento          |  Vida Nocturna       | Aguas |
Precio del Viaje |
                        | Check ~alojamiento | Check ~VidaNocturna | Check |
Check            |

           Lugares |  N° Situación                | N° Destino  | Nombre
Situación | Caracteristica  |
                        | _lugarBarco, _lugarVidaMar | _destino    |
~nombreDestino   | ~característica |

           Lugares | Nivel de VACACION |
                        | ~nivelVACACION    |

          Naufragi | Nombre del Barco | N° Situación | Categoría        |
Interés     |
                                | ~nombreBarco           | _lugarBarco     |
~categoríaBarco | ~interésBarco |

          Naufragi | Estado               |
                                | ~condiciónBarco |

          Vidaluga | N° Especie | N° Situación  |
                        | _especie   | _lugarVidaMar |

           Vidamar | N° Especie | Nombre Comun | Nombre Especie |
                        | _especie   | ~nombreComún | ~nombreEspecie |

       EndQuery

      ; empleando el criterio de selección sobre naufragios pero no sobre
vida marina

       Case (not NombreBarco.isBlank() or
           not CategoríaBarco.isBlank() or
           not InterésBarco.isBlank() or
           not CondiciónBarco.isBlank()) and
          (NombreComún.isBlank() and
           NombreEspecie.isBlank()) :

        message("opción 2")
        sleep(4000)
        DestinoConsulta = Query

          answer: :trabajo:destipos.db

          Destinos | N° Destino      | Destino                     |
Temperatura Media(F) | Temperatura Media(C)    |
                        | Check _destino | Check ~NombreDestino | Check
| Check ~temperaturaMedia |

          Destinos | Temperatura Primavera(F) | Temperatura Primavera(C) |
Temperatura Verano(F) | Temperatura Verano(C) |
```

```
                        | Check                                | Check
       | Check                  | Check                       |

          Destinos | Temperatura Otoño(F) | Temperatura Otoño(C) |
Temperatura Invierno(F) | Temperatura Invierno(C) |
                        | Check                    | Check                    |
Check                         | Check                           |

          Destinos | Alojamiento          | Vida Nocturna       | Aguas |
Precio del Viaje |
                        | Check ~alojamiento | Check ~vidaNocturna | Check |
Check            |

           Lugares |  N° Situación | N° Destino | Nombre Situación |
Caracteristica  |
                        | _lugarBarco      | _destino   | ~NombreLugar     |
~característica |

           Lugares | Nivel de VACACION |
                        | ~nivelVACACION    |

          Naufragi | Nombre del barco |  N° Situación | Categoría       |
Interés       |
                                | ~NombreBarco           | _lugarBarco
~categoríaBarco | ~interésBarco |

          Naufragi | Estado        |
                                | ~condiciónBarco |


       Endquery

     ; seleccionado el criterio vida marina pero no naufragios
     Case  nombreBarco.isBlank() and
          categoríaBarco.isBlank() and
          interésBarco.isBlank() and
          CondiciónBarco.isBlank() and
          not nombreComún.isBlank() and
          not nombreEspecie.isBlank() :

       DestinoConsulta = Query

          answer: :trabajo:destipos.db

           Destinos | N° Destino     | Destino              | Temperatura
Media(F) | Temperatura Media(C)    |
                        | Check _destino | Check ~nombreDestino | Check
| Check ~temperaturaMedia |

          Destinos | Temperatura primavera(F) | Temperatura primavera(C) |
Temperatura verano(F) | Temperatura verano(C) |
                        | Check                                | Check
       | Check                     | Check                          |
```

```
         Destinos | Temperatura otoño(F) | Temperatura otoño(C) |
Temperatura invierno(F) | Temperatura invierno(C) |
                        | Check                   | Check
   | Check                          | Check                          |

         Destinos | Alojamiento         | Vida nocturna        | Aguas |
Precio del Viaje |
                        | Check ~alojamiento | Check ~vidaNocturna  | Check |
Check           |

          Lugares | N° Situación  | N° Destino | Nombre Situación |
Caracteristica  |
                        | _lugarVidaMar | _destino   | ~nombreDestino   |
~característica |

          Lugares | Nivel de VACACION |
                        | ~nivelVACACION  |

         Vidaluga | N° Especie | N° Situación  |
                        | _especie   | _lugarVidaMar |

         Vidamar      | N° Especie | Nombre Comun | Nombre Especie |
                        | _especie   | ~nombreComún | ~nombreEspecie |

       Endquery

     ; no se seleccionó criterio ni vida marina ni naufragio
     Case  nombreBarco.isBlank() and
          categoríaBarco.isBlank() and
          interésBarco.isBlank() and
          condiciónBarco.isBlank() and
          nombreComún.isBlank() and
          nombreEspecie.isBlank() :

       DestinoConsulta = Query

          answer: :trabajo:destipos.db

          Destinos | N° Destino     | Destino                   | Temperatura
Media(F) | Temperatura Media(C)    |
                        | Check _destino | Check ~NombreDestino | Check
| Check ~TemperaturaMedia |

          Destinos | Temperatura primavera(F) | Temperatura primavera(C) |
Temperatura verano(F) | Temperatura verano(C) |
                        | Check                                | Check
       | Check                     | Check                          |

          Destinos | Temperatura otoño(F) | Temperatura otoño(C) |
Temperatura invierno(F) | Temperatura invierno(C) |
                        | Check                   | Check
   | Check                          | Check                          |

          Destinos | Alojamiento        | Vida Nocturna      | Aguas | Precio
del Viaje |
```

```
                        | Check ~alojamiento | Check ~VidaNoctura | Check |
Check                   |

           Lugares | N° Destino | Nombre Situación | Caracteristica  |
                        | _destino   | ~nombreDestino   | ~característica |

           Lugares | Nivel de VACACION |
                        | ~nivelVACACION    |

        endQuery
   endSwitch
   if not writeQBE(destinoConsulta, "VACACIONCon.qbe") then
     msgInfo("error","no pude escribir en VACACIONCon")
   endif
   ; Ejecuta la consulta y escribe los resultados en destipos.db.
   ; se emplea destipos.db en la ficha reserva.fsl para reservar.
   if not executeQBE(DestinoConsulta, ":trabajo:destipos.db") then
     msgInfo("Error","¡No he podido procesar la consulta!")
     return FALSE
   endif
   return TRUE
endmethod
```

VACACIÓN Ventana Var

Las variables declaradas en la ventana de Variables de la ficha son accesibles por todos los objetos que ésta contenga.

```
Var
   NombreInterfaz       String
   Interfaz                   UIObject
   NombreFicha,
   FichaLugares,
   FichaNaufragios,
   FichaVidaMar,
   FichaReservas,
   FichaAcercaDe        Form
   thamLib                    Library
   CierraMando                Logical
endVar
```

VACACIÓN Ventana Const

Las constantes declaradas en la ventana de Constantes de la ficha son accesibles por todos los objeto que ésta contenga.

```
Const
   FicheroAyuda = ":trabajo:usuario.hlp" ; vía de acceso al fichero de ayuda
de usuario debe estar definida (o el fichero debe estar en la vía de acceso)
endConst
```

VACACIÓN Ventana Uses

En la ventana *Uses* se declaran rutinas externas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera lÍnea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   LimpiarValoresPágina(NombreFicha String, NombrePágina String)
   CrearMenú(var NombreFicha Form, NombrePágina String, interfaz UIObject)
   MensajeTHAM(NombreFicha String, NombrePágina String, NúmeroDeObjeto
String)
   NombreDePágina(interfaz UIObject) String
   EsLlamadaVerdadero()
   EsLlamadaFalso()
    PuedeCerrarFicha()
endUses
```

VACACIÓN.PáginaPrincipal

Métodos

arrive
menuAction
open

Procedimientos

MenúPáginaPrincipal

arrive

VACACIÓN.PáginaPrincipal.arrive

Ajusta el titulo de *PáginaPrincipal* a "Página Principal" y muestra un menú desplegable personalizado llamando al procedimiento **PáginaPrincipal.MenúPáginaPrincipal**.

```
method arrive(var eventInfo MoveEvent)
   ; define el título de página
   setTitle("Página Principal")
   ; muestra la barra rápida personalizada para esta página
   MenúPáginaPrincipal()
endmethod
```

menuAction

VACACIÓN.PáginaPrincipal.menuAction

Comprueba la cadena devuelta por una opción del menú *PáginaPrincipal* y llama al método apropiado. Por ejemplo, eligiendo el elemento del menú Vida Marina llama a **BotónVidaMarina.pushButton.** Eligiendo Resultados se situará en la *PáginaResultados.*

```
method menuAction(var eventInfo MenuEvent)
   var
      opciónMenú String
   endvar

   ; ¿Qué opción del menú ha seleccionado?
   opciónMenú = eventInfo.menuChoice()
   switch
      case opciónMenú = "&Lugares"              : BotónLugares.pushbutton()
      case opciónMenú = "&Vida Marina"     : BotónVidaMarina.pushbutton()
      case opciónMenú = "&Naufragios"           : BotónNaufragio.pushbutton()
      case opciónMenú = "&Procesar"        : BotónProcesar.pushbutton()
      case opciónMenú = "&Eliminar"        : BotónLimpiarConsulta.pushbutton()
      case opciónMenú = "&Salir"                : BotónSalir.pushbutton()
      case opciónMenú = "&Resultados"           : PáginaResultados.moveto()
      case opciónMenú = "&Indice"               :
helpShowContext(ficheroAyuda, 10000)
      case opciónMenú = "&Uso de la Ayuda" : helpOnHelp()
      case opciónMenú = "&Acerca de THAM"  : thamlib.esLlamadaVerdadero()
                                                                  if
FichaAcercaDe.open("SobTHAM.fsl") then
                                                                  ; abre
un cuadro de diálogo modal

   FichaAcercaDe.Wait()
                                                                  ;
espera y la cierra

   FichaAcercaDe.Close()

   thamlib.esLlamadaFalso()
                                                              endif
   endswitch
endmethod
```

open

VACACIÓN.PáginaPrincipal.open

Abre la biblioteca THAMLIB, oculta la Barra Rápida de Paradox y borra la consulta.

```
method open(var eventInfo Event)
   ; no permite abrir el fichero si no está en el directorio de trabajo
   if not isFile("VACACION.fsl") then
      msgInfo("¡Error de arranque!", "Los ficheros de la aplicación THAM deben
encontrarse en el directorio de trabajo.")
      close()
   endif
   ; abre la biblioteca (asegúrese que está en el directorio de trabajo)
   if not thamlib.open("THAMlib", globalToDesktop) then
      msgStop("Fallo", "no pude abrir la biblioteca thamlib")
   endif
   ; asocia la ficha a la variable NombreFicha
   NombreFicha.attach()
   hideSpeedBar()
   maximize()
   BotónLimpiarConsulta.pushbutton()
endmethod
```

MenúPáginaPrincipal

VACACIÓN.PáginaPrincipal.MenúPáginaPrincipal

Es llamado por **PáginaPrincipal.arrive** para mostrar el menú *PáginaPrincipal*.

```
proc MenúPáginaPrincipal()
  Var
    menúPrincipal  Menu
    menúDesplegable1,
    menúDesplegable2 popUpMenu
  endVar

  ; este procedimiento personalizado muestra el menú de la página principal
  ; definimos el menú
  menúDesplegable1.addText("&Lugares")
  menúDesplegable1.addText("&Vida Marina")
  menúDesplegable1.addText("&Naufragios")
  menúDesplegable1.addSeparator()
  menúDesplegable1.addText("&Procesar")
  menúDesplegable1.addText("&Eliminar")
  menúDesplegable1.addText("&Salir")
  menúPrincipal.addPopUp("Selecci&onar", menúDesplegable1)
  menúPrincipal.addText("&Resultados")
  menúDesplegable2.addText("&Contenido")
  menúDesplegable2.addText("&Uso de la Ayuda")
  menúDesplegable2.addText("Acerca de &THAM")
  menúPrincipal.addPopUp("&Ayuda", menúDesplegable2)
  menúPrincipal.show()
endproc
```

pushButton

VACACIÓN.PáginaPrincipal.BotónProcesar.pushButton

Este es llamado eligiendo Process en *pageMenu*. LLama a **DestinoConsulta** tpara procesar la consulta. Visualizando cualquier coincidencia en *resultsPage.*

```
method pushButton(var eventInfo Event)
   ; asociamos MarcoCoincidencias a destinos.db temporalmente
   PáginaResultados.MarcoCoincidencias.TableName = "Destinos.db"

   ; borramos destipos.db si ya forma parte del modelo de datos.
   ; puede repetir la consulta
   if DMHasTable("destipos.db") then
      DMRemovetable("destipos.db")
   endif
   ; ejecutamos la consulta
   Try
      If Not DestinoConsulta() then
        Return
       Endif
   onFail  ; consulta fallida - abandonamos el método
      fail(1, "No pude abrir valconsu.db")
   Endtry
   ; consulta finalizada -- añadimos el resultado en destipos.db al
   ; modelo de datos y lo asociamos a MarcoCoincidencias
   PáginaResultados.MarcoCoincidencias.TableName = "destipos.db"
   ; borramos destinos.db del modelo de datos
   DMRemoveTable("Destinos.db")
   ; nos movemos a la página de resultados para mostrar los destinos elegidos
   PáginaResultados.moveTo()
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónAyuda.pushButton

Muestra la Ayuda llamando a **helpShowContext**. El valor 20000 es una identificación de contexto para un elemento del archivo de Ayuda.. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method pushButton(var eventInfo Event)
   ; proporciona la ayuda de esta página
   helpShowContext(ficheroAyuda, 20000)
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónSalir.pushButton

Éste visualiza un cuadro de diálogo para salir de THAM. Llama a **close** para cerrar la ficha si se elije SI.

```
method pushButton(var eventInfo Event)
   ; pregunta al usuario se realmente quiera abandonar la aplicación
   selección = msgQuestion("Salir de T*H*A*M", "¿Seguro que quiere abandonar
la aplicación?")
   ; si el usuario confirma la opción, cerramos la ficha
   if selección= "Yes" then
      close()
   endif
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónLimpiarConsulta.pushButton

Borra la consulta actual en VALCONSUL.DB utilizando una consulta CHANGETO para cambiar el valor de la consulta para cambiar el campo Valor_Consulta en cada registro a ser borrado.

```
method pushButton(var eventInfo Event)
  var
    LimpioConsulta    Query
  endVar
  ; construimos la cadena
  LimpioConsulta=Query

    valconsu.db | Valor Consulta |
                | changeto blank |

  EndQuery
  executeQBE(LimpioConsulta)
  ; vacía Destipos.db
  if isTable("Destipos.db") then
    empty("Destipos.db")
  endif
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónVidaMarina.pushButton

Abre VIDAMAR.FSL, lo maximiza, y sitúa el foco en él.

```
method pushButton(var eventInfo Event)
   ; abre VidaMar.fsl
   ; una vez abierta, la muestra y la pone en estado de espera
   if isAssigned(FichaVidaMar) then
      FichaVidaMar.bringtoTop()
      FichaVidaMar.wait()  ; la pone en estado de espera
      ; retorno del usuario -- ocultamos VidaMar
      FichaVidaMar.hide()
      maximize()
      PáginaPrincipal.moveTo()
   else
      ; VidaMar no está aierta -- la abrimos
      thamLib.EsLlamadaVerdadero()
      if FichaVidaMar.Open("VidaMar", WinStyleMaximize) then
         FichaVidaMar.Wait() ; la pone en estado de espera
         ; retorno del usuario -- ocultamos VidaMar
            FichaVidaMar.hide()
         maximize()
         PáginaPrincipal.moveTo()
      else
         ; Error al abrir la ficha
         msgInfo("Estado", "Lo siento, la fichas de VidaMar no está
disponible")
            thamlib.EsLlamadaFalso()
      endif
   endif
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónLugares.pushButton

Abre LUGARES.FSL, lo maximiza, y sitúa el foco en él.

```
method pushButton(var eventInfo Event)
   ; abre lugares.fsl
   ; una vez abierta, la muestra y la pone en estado de espera
   if isAssigned(FichaLugares) then
      FichaLugares.bringtoTop()
      FichaLugares.wait()  ; la pone en estado de espera
      ; retorno del usuario -- ocultamos Lugares
      FichaLugares.hide()
      maximize()
      PáginaPrincipal.moveTo()
   else
      ; Lugares no está aierta -- la abrimos
      thamLib.EsLlamadaVerdadero()
      if FichaLugares.Open("Lugares", WinStyleMaximize) then
         FichaLugares.Wait() ; la pone en estado de espera
         ; retorno del usuario -- ocultamos Lugares
            FichaLugares.hide()
         maximize()
         PáginaPrincipal.moveTo()
      else
         ; Error al abrir la ficha
         msgInfo("Estado", "Lo siento, la fichas de Lugares no está
disponible")
            thamlib.EsLlamadaFalso()
      endif
   endif
endmethod
```

pushButton

VACACIÓN.PáginaPrincipal.BotónNaufragio.pushButton

Abre NAUFRAGI.FSL, lo maximiza, y sitúa el foco en él.

```
method pushButton(var eventInfo Event)
   ; abre naufragi.fsl
   ; una vez abierta, la muestra y la pone en estado de espera
   if isAssigned(FichaNaufragios) then
      FichaNaufragios.bringtoTop()
      FichaNaufragios.wait()  ; la pone en estado de espera
      ; retorno del usuario -- ocultamos Naufragios
      FichaNaufragios.hide()
      maximize()
      PáginaPrincipal.moveTo()
   else
      ; Naufragios no está aierta -- la abrimos
      thamLib.EsLlamadaVerdadero()
      if FichaNaufragios.Open("Naufragios", WinStyleMaximize) then
         FichaNaufragios.Wait() ; la pone en estado de espera
         ; retorno del usuario -- ocultamos Naufragios
            FichaNaufragios.hide()
         maximize()
         PáginaPrincipal.moveTo()
      else
         ; Error al abrir la ficha
         msgInfo("Estado", "Lo siento, la fichas de naufragios no está
disponible")
            thamlib.EsLlamadaFalso()
      endif
   endif
endmethod
```

VACACIÓN.PáginaResultados

Métodos

arrive
menuAction

Procedimientos

MenúPáginaPrincipal

arrive

VACACIÓN.PáginaResultados.arrive

Ajusta el titulo de *PáginaResultados* a "Vacaciones" y muestra un menú personalizado emergente llamando al procedimiento **PáginaResultados.MenúPáginaPrincipal**

```
method arrive(var eventInfo MoveEvent)
   ; muestra el título y el menú de la página
   setTitle("Vacaciones")
   MenúPáginaResultados()
endmethod
```

menuAction

VACACIÓN.PáginaResultados.menuAction

Comprueba la cadena devuelta por una selección del menú *PáginaResultados* y llama al método apropiado. Por ejemplo, eligiendo el elemento del menú Reservar llama a **BotónReservar.pushButton.** Seleccionando Principal se situará en la *PáginaPrincipal.*

```
method menuAction(var eventInfo MenuEvent)
   var
      opciónMenú String
   endVar
   ; ¿Qué opción de meú ha sido seleccionada?
   opciónMenú = eventInfo.menuChoice()
   switch
      case opciónMenú = "&Reservar"                      :
BotónReservar.pushbutton()
      case opciónMenú = "&Salir"                         :
BotónSalir.pushbutton()
      case opciónMenú = "&Principal"                     :
BotónPáginaPrincipal.moveto()
      case opciónMenú = "&Primero\tCtrl+F11"             :
active.action(DataBegin)
      case opciónMenú = "&Ultimo\tCtrl+F12"              :
active.action(DataEnd)
      case opciónMenú = "&Siguiente\tF12"                :
active.action(DataNextRecord)
      case opciónMenú = "&Anterior\tF11"                 :
active.action(DataPriorRecord)
      case opciónMenú = "&Insertar\tIns"                    :
active.action(DataInsertRecord)
      case opciónMenú = "&Borrar\tCtrl+Del"           :
active.action(DataDeleteRecord)
      case opciónMenú = "C&ancelar Cambios\tAlt+Bksp" :
active.action(DataCancelRecord)
      case opciónMenú = "&Indice"                     :
helpShowContext(ficheroAyuda, 10000)
      case opciónMenú = "&Uso de la Ayuda"            : helpOnHelp()
      case opciónMenú = "&Acerca de THAM"             :
thamLib.esLlamadaVerdadero()

        if FichaAcercaDe.open("SobTHAM.fsl") then

         ; abre un cuadro de diálogo modal

         FichaAcercaDe.Wait()

           ; espera y lo cierra

         FichaAcercaDe.Close()

           thamLib.esLlamadaFalso()

      endif
```

```
   endswitch
endmethod
```

MenúPáginaResultados

VACACIÓN.PáginaResultados.MenúPáginaResultados

Éste llama a **PáginaResultados.arrive** para mostrar un menú desplegable personalizado.

```
proc MenúPáginaResultados()
  Var
    MenúResultados Menu
    MenúEmergente1,
    MenúEmergente2, MenúEmergente3 popUpMenu
  endVar

  ; este procedimiento personalizao muestra el menú de la página
PáginaResultados
  MenúEmergente1.addText("&Reservar")
  MenúEmergente1.addText("&Salir")
  MenúResultados.addPopUp("&Destinos", MenúEmergente1)
  MenúResultados.addText("&Principal")
  MenúEmergente2.addtext("&Primero\tCtrl+F11")
  MenúEmergente2.addtext("&Ultimo\tCtrl+F12")
  MenúEmergente2.addtext("&Siguiente\tF12")
  MenúEmergente2.addtext("&Anterior\tF11")
  MenúEmergente2.addSeparator()
  MenúEmergente2.addtext("&Insertar\tIns")
  MenúEmergente2.addtext("&Borrar\tCtrl+Del")
  MenúEmergente2.addtext("C&ancelar Cambios\tAlt+Bksp")
  MenúResultados.addPopUp("Rec&ord", MenúEmergente2)
  MenúEmergente3.addText("&Contenido")
  MenúEmergente3.addText("&Uso de la Ayuda")
  MenúEmergente3.addText("&Acerca de THAM")
  MenúResultados.addPopUp("&Ayuda", MenúEmergente3)
  MenúResultados.show()
endproc
```

pushButton

VACACIÓN.PáginaResultados.BotónAyuda.pushButton

Éste nos muestra la Ayuda llamando a **helpShowContext**. El valor 20001 es un identificador de contexto para un elemento del archivo de Ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method pushButton(var eventInfo Event)
   helpShowContext(ficheroAyuda, 20001)
endmethod
```

pushButton

VACACIÓN.PáginaResultados.BotónReservar.pushButton

Abre la ficha PEDIDOS.FSL, maximiza la ventana, y espera las entradas del usuario.

```
method pushButton(var eventInfo Event)
   ; abrimos reserva.fsl

   if isAssigned(FichaReservas)  then
      fichaReservas.bringtoTop()
      fichaReservas.wait()
      ; retorno del usuario, se oculta fichaReservas
      fichaReservas.hide()
      PáginaPrincipal.moveTo()
      maximize()
   else
      thamLib.esLlamadaVerdadero()
      if fichaReservas.Open("Reservas", WinStyleMaximize) then
         fichaReservas.Wait()  ; queda en estado de espera
         ; retorno del usuario, se oculta lugares
         fichaReservas.hide()
         maximize()
         PáginaPrincipal.moveTo()
      else
         msgInfo("Estado", "Lo siento, la ficha Lugares no está disponible")
         thamlib.esLlamadaFalso()
      endif
   endif
endmethod
```

pushButton

VACACIÓN.PáginaResultados.BotónPáginaPrincipal.pushButton

Sitúa al usuario en *PáginaPrincipal* llamando a **PáginaPrincipal.moveTo**.

```
method pushButton(var eventInfo Event)
   PáginaPrincipal.moveTo()
endmethod
```

pushButton

VACACIÓN.PáginaResultados.BotónSalir.pushButton

Muestra un cuadro de diálogo para salir de THAM. Llama a **close** para cerrar la ficha en el caso de que se seleccione Si.

```
method pushButton(var eventInfo Event)
   ; pregunta al usuario se realmente quiera abandonar la aplicación
   selección = msgQuestion("Salir de T*H*A*M", "¿Seguro que quiere abandonar
la aplicación?")
   ; si el usuario confirma la opción, cerramos la ficha
   if selección= "Yes" then
      close()
   endif
endmethod
```

LUGARES

Métodos

keyPhysical
menuAction
mouseEnter
mouseExit
mouseRightUp

Variables, Constantes, y Ventanas Uses

ventana de Variables
ventana de Constantes
ventana Uses

Métodos y procedimientos de la biblioteca **THAMLIB.LSL**

LimpiarValoresPágina
crearMenú
MensajeTHAM
EstablecerValoresPágina
NombreDePágina
HaLlamadoARetornar
EsLlamadaVerdadero
EsLlamadaFalso

keyPhysical

LUGARES.keyPhysical

Captura pulsaciones de tecla y las convierte en llamadas a métodos, si procede.

```
method keyPhysical(var eventInfo KeyEvent)
  ; este método captura la pulsación de las teclas rápidas a nivel de ficha
  ; Capturará F1 (Ayuda).
  Var
    laTecla String
    laPágina String
  endVar

  if eventInfo.isPreFilter() then
    laPágina = thamlib.NombreDePágina(active)
    laTecla = eventInfo.vChar()                              ; dime qué tecla
se pulsó
    switch
      case laPágina = "PáginaLugares" :
          if eventInfo.isAltKeyDown() then
          switch
                case laTecla = "A" or
                        laTecla = "a" : disableDefault

  PáginaLugares.BotónAceptar.pushButton()
                case laTecla = "C" or
                        laTecla = "c" : disableDefault

PáginaLugares.BotónCancelar.pushButton()
                case laTecla = "D" or
                        laTecla = "d" : disableDefault

PáginaLugares.BotónPáginaDestinos.pushButton()
                case laTecla = "E" or
                        laTecla = "e" : disableDefault

PáginaLugares.BotónLimpiarConsulta.pushButton()
                case laTecla = "C" or
                        laTecla = "c" : disableDefault

PáginaLugares.GrupoCaracterísticas.Arrecifes.moveTo()
                case laTecla = "M" or
                        laTecla = "m" : disableDefault

PáginaLugares.GrupoAlojamiento.Barato.moveTo()
                case laTecla = "V" or
                        laTecla = "v" : disableDefault

PáginaLugares.GrupoVidaNocturna.Tranquila.moveTo()
                case laTecla = "L" or
                        laTecla = "l" : disableDefault

  PáginaLugares.LugarPreferencia.moveTo()
```

```
                case laTecla = "B" or
                        laTecla = "b" : disableDefault

   PáginaLugares.GrupoNivelVACACION.NivelVACACION.moveTo()
                case laTecla = "T" or
                        laTecla = "t" : disableDefault

   PáginaLugares.GrupoTemperatura.SubTropical.moveTo()
                otherwise               : doDefault
                endswitch
          else
           switch
                case laTecla = "VK_F1" : disableDefault

helpShowContext(FicheroAyuda, 20004)
                otherwise                       : doDefault
           endSwitch
          endif
          case laPágina = "PáginaDestinos" :
           if eventInfo.isAltKeyDown() then
           switch
                case laTecla = "A" or
                        laTecla = "a" : disableDefault

   PáginaDestinos.BotónAceptar.pushButton()
                case laTecla = "C" or
                        laTecla = "c" : disableDefault

   PáginaDestinos.BotónCancelar.pushButton()
                case laTecla = "D" or
                        laTecla = "d" : disableDefault

   PáginaDestinos.destinos.RegistroDestinos.NombreDestino.moveTo()
                case laTecla = "E" or
                        laTecla = "e" : disableDefault

   PáginaDestinos.BotónLimpiarConsulta.pushButton()
                case laTecla = "I" or
                        laTecla = "i" : disableDefault

   PáginaDestinos.Lugares.RegistroLugares.NombreLugar.moveTo()
                case laTecla = "T" or
                        laTecla = "t" : disableDefault

   PáginaDestinos.BotónAñadirEscogidos.pushButton()
                case laTecla = "S" or
                        laTecla = "s" : disableDefault

   PáginaDestinos.BotónLugares.pushButton()
                case laTecla = "R" or
                        laTecla = "r" : disableDefault

   PáginaDestinos.BotónLimpiarDestinosEscogidos.pushButton()
                otherwise               : doDefault
                endswitch
```

```
          else
           switch
                 case laTecla = "VK_F1" : disableDefault

helpShowContext(FicheroAyuda, 20004)
                 otherwise                      : doDefault
           endSwitch
          endif
        endswitch
     else
        doDefault
     endif
endmethod
```

menuAction

LUGARES.menuAction

Evita que el usuario emplee el menú de control para cerrar la ficha.

```
method menuAction(var eventInfo MenuEvent)
   if eventInfo.isPreFilter() then
     doDefault
   else
     ; Obliga a cerrar desde el menú del sistema
     if eventInfo.ID() = MenuControlClose then
      eventInfo.setErrorCode(1)
       ; si el mando es visible, lo oculta
       if isAssigned(miMando) then
         miMando.hide()
       endif
      formReturn(True)
     endif
   endif
endmethod
```

mouseEnter

LUGARES.mouseEnter

Es una rutina de mensajes a nivel de ficha que muestra los nombres de objetos en la barra de estado dependiendo de la posición del cursor del ratón. Emplea el método de la biblioteca THAMLIB, **THAMLib.MensajeTHAM.**

```
method mouseEnter(var eventInfo MouseEvent)
   ; evitas las llamadas sucesivas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      NombreInterfaz = interfaz.Name
      ; excluye el texto y los bitmaps
      ; si se trata de la ficha, elimina el mensaje
      if interfaz.name <> self.name AND
         (interfaz.class = "Field" OR interfaz.class = "Multirecord" OR
interfaz.class = "Button") then
         thamlib.MensajeTHAM(NombreFicha.name,
thamlib.NombreDePágina(interfaz), NombreInterfaz)
      endif
   endif
endmethod
```

mouseExit

LUGARES.mouseExit

Borra los mensajes de **mouseEnter** de la barra de estado cuando el cursor del ratón abandona la zona en la que se encuentra un objeto.

```
method mouseExit(var eventInfo MouseEvent)
   ; evita la repetición de mouseExit
   if eventInfo.isPreFilter() then
     eventInfo.getTarget(interfaz)  ; ¿Qué Objeto es?
     if interfaz.class <> "Text" and interfaz.class <> "Bitmap" then
       message("")
     endif
   endif
endmethod
```

mouseRightUp

LUGARES.mouseRightUp

Llama al método **crearMenú** de la biblioteca THAMLIB cuando se suelta el botón derecho del ratón. Esto crea el menú de Ayuda del Código.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Llama al método thamlib.CrearMenú() encargado de mostrar
   ; el menú de ayuda a nivel de objeto.

   ; evita llamadas repetidas al método
   if eventInfo.isPreFilter() then
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      ; busca el cotenedor de los objetos text y bitmap
      ; siempre y cuando no sea CuadroBandera
      if interfaz.name <> "CuadroBandera1"  and interfaz.name <>
"CuadroBanderaPequeñas" then
         while
            interfaz.class = "Text" or interfaz.class = "Bitmap"
            interfaz.attach(interfaz.ContainerName)
            endWhile
      endif
      thamlib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endmethod
```

LUGARES Ventana Var

Las Variables declaradas en la ventana de Variables de la ficha, son accesibles para todos los objetos contenidos en esa ficha.

```
Var
   NombreInterfaz      String
  NombreFicha,
  FichaAcercaDe,
  miMando                   Form
  FueLlamada                Logical
  interfaz                  UIObject
  thamLib                   Library
endVar
```

LUGARES Ventana Const

Las constantes declaradas en la ventana de Contantes de la ficha son accesibles para todos los objetos contenidos en ésta.

```
Const
   FicheroAyuda = ":TRABAJO:usuario.hlp" ; vía de acceso al fichero de ayuda
endConst
```

LUGARES Ventana Uses

En la ventana *Uses* se declaran rutinas externas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   LimpiarValoresPágina(NombreFicha String, NombrePágina String)
   CrearMenú(var NombreFicha Form, NombrePágina String, Interfaz UIObject)
   MensajeTham(NombreFicha String, NombrePágina String, NúmeroDeObjeto
String)
   EstablecerValoresPágina(NombreFicha String, NombrePágina String) Logical

   NombreDePágina(interfaz UIObject) String
   HaLlamadoaRetornar() Logical
   EsLlamadaFalso()
   EsLlamadaVerdadero()
endUses
```

LUGARES.PáginaDestinos

Métodos

close
arrive
depart
menuAction
setFocus

Procedimientos

MenúPáginaResultados

Variables, Constantes, y ventanas Uses

ventana de Variables
ventana Uses

close

LUGARES.PáginaDestinos.close

Cierra la ficha y la Barra Rápida personalizada.

```
method close(var eventInfo Event)
   ; el mando está abierto
   if thamlib.EstáMandoAbierto() then
       ; cierra el mando
     miMando.close()
     thamlib.MandoEstáAbierto()
    endif
endmethod
```

arrive

LUGARES.PáginaDestinos.arrive

Ajusta el titulo de *PáginaDestinos* a "Destinos" y reestablece un menú personalizado llamando al procedimiento **PáginaDestinos.MenúPáginaResultados**.

```
method arrive(var eventInfo MoveEvent)
   ; cambia el título de la página
   setTitle("Destinos")
   ; Muestra el menú personalizado
   MenúPrincipal()
endMethod
```

depart

LUGARES.PáginaDestinos.depart

Oculta la Barra Rápida personalizada.

```
method depart(var eventInfo MoveEvent)
   ; ocultamos el mando
   miMando.Hide()
endMethod
```

menuAction

LUGARES.PáginaDestinos.menuAction

Comprueba la cadena devuelta por una opción del menú *PáginaDestinos* y llama al método apropiado. Por ejemplo, eligiendo el elemento del menú Lugares se llama a **BotónLugares.pushButton.**

```
method menuAction(var eventInfo MenuEvent)
   Var
     OpciónMenú String
   endVar
   ; ¿Qué opción de menú ha sido seleccionada?
   OpciónMenú = eventInfo.menuChoice()
   switch
     case OpciónMenú = "&Aceptar"                    :
BotónAceptar.pushbutton()
     case OpciónMenú = "&Cancelar"             : BotónCancelar.pushbutton()
     case OpciónMenú = "&Eliminar"             :
BotónLimpiarConsulta.pushbutton()
     case OpciónMenú = "&Lugares"              : BotónLugares.pushbutton()
     case OpciónMenú = "&Primero\tCtrl+F11"    :
miMando.BotónPrimero.pushButton()
     case OpciónMenú = "&Ultimo\tCtrl+F12"     :
miMando.BotóUltimo.pushButton()
     case OpciónMenú = "&Siguiente\tF12"       :
miMando.BotónSiguiente.pushButton()
     case OpciónMenú = "&Anterior\tF11"              :
miMando.BotónAnterior.pushButton()
     case OpciónMenú = "&Indice"                     :
helpShowContext(FicheroAyuda, 20003)
     case OpciónMenú = "&Uso de la Ayuda"      : helpOnHelp()
     case OpciónMenú = "&Acerca de THAM"       : thamlib.EsLlamadaVerdadero()
                                                                    if
FichaAcercaDe.open("SobTham.fsl") then

   ; abre un cuadro de diálogo modal

   FichaAcercaDe.Wait()

     ; espera y lo cierra

   FichaAcercaDe.Close()

   thamlib.EsLlamadaFalso()
                                                                 endif
   endSwitch
endMethod
```

setFocus

LUGARES.PáginaDestinos.setFocus

Muestra la Barra Rápida personalizada MANDOVID.FSL, abriéndola primero si fuese necesario.

```
method setFocus(var eventInfo Event)
  Var
    x,
    y,
    w,
    h  LongInt
  endVar
  ; el mando está abierto
  if thamlib.EstáMandoAbierto() then
    if miMando.isAssigned() then
      ; Mando abierto -- lo asociamos a esta ficha
       miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    else
      ; asociamos el mando
      miMando.attach(thamlib.NombreMando())
       miMando.bringToTop()
      miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    endif
  else
    ; el mando no está abierto, lo abrimos
    miMando.Open("Mandovid.fsl")
    miMando.EstablecerNombreFichaLlamar(getTitle(), 5500, 400)
    thamlib.MandoEstáAbierto()
  endif
endmethod
```

MenúPáginaResultados

LUGARES.PáginaDestinos.MenúPáginaResultados

Éste es llamado por **PáginaDestinos.arrive** para mostrar los menús de Selecciones.

```
proc MenúPrincipal()

   Var
      MenúPrincipal Menu
      MenúEmergente1,
      MenúEmergente2,
      MenúEmergente3 popUpMenu
   EndVar

   ; este procedimiento de usuario muestra el menú de la ficha Destinos
   ; definimos el menú
   MenúEmergente1.addText("&Aceptar")
   MenúEmergente1.addText("&Cancelar")
   MenúEmergente1.addText("&Eliminar")
   MenúPrincipal.addPopUp("&Selecciones", MenúEmergente1)
   MenúPrincipal.addText("&Lugares")
   MenúEmergente2.addtext("&Primero\tCtrl+F11")
   MenúEmergente2.addtext("&Ultimo\tCtrl+F12")
   MenúEmergente2.addtext("&Siguiente\tF12")
   MenúEmergente2.addtext("&Anterior\tF11")
   MenúPrincipal.addPopUp("&Registro", MenúEmergente2)
   MenúEmergente3.addText("&Contenido")
   MenúEmergente3.addText("&Uso de la Ayuda")
   MenúEmergente3.addText("Acerca de &THAM")
   MenúPrincipal.addPopUp("&Ayuda", MenúEmergente3)
   MenúPrincipal.show()
endproc
```

LUGARES.PáginaDestinos Ventana Var

Las variables declaradas en la ventana de Variables de la ficha son accesiblles para todos los objetos contenidos en ésta.

```
var
   NúmeroDestino                  Number
   DestinoTc,
   DestinoTemporalTc          TCursor
endvar
```

LUGARES.PáginaDestinos Ventana Uses

La ventana *Uses* declara rutinas externas empleadas por esta ficha. las rutinas se pueden almacenar en una DDL, en una biblioteca ObjectPAL, o en otra ficha. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, y no en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   EstablecerNombreFichaLlamar(const NombreFichaLlamador String, xPos
LongInt, yPos LongInt)
    NombreMando() String
   MandoEstáAbierto()
   MandoEstáCerrado()
    EstáMandoAbierto() Logical
endUses
```

pushButton

LUGARES.PáginaDestinos.BotónLimpiarDestinosEscogidos.pushButton

Borra las opciones de destino actuales y ajusta el contador de la lista a 0, lo cual borra la lista.

```
method pushButton(var eventInfo Event)
   ; vacía ListaDestinos
   ListaDestinos.DestinosEscogidos.list.count = 0
endmethod
```

pushButton

LUGARES.PáginaDestinos.BotónAñadirEscogidos.pushButton

A éste se le llama eligiendo el botón de añadir a la lista *(BotónAñadirEscogidos).* Añade los destinos seleccionados desde DESTINOS.DB a *ListaDestinos.* Esos destinos se añaden a VALCONSU.DB cuando se elige el botón Aceptar.

```
; Este método añade destinos de destinos.db a
; ListaDestinos. Los destinos de ListaDestinos se añaden
; a valconsu.db cuando el usuario pulsa BotónAceptar

method pushButton(var eventInfo Event)

  Var
    ValorDestinos String
    i,
    ContadorLista SmallInt
  endVar

  if active.name = "NombreDestino" then
    ContadorLista = ListaDestinos.DestinosEscogidos.list.count
    ValorDestinos = destinos.RegistroDestinos.NombreDestino.value
    ; se asegura de que el destino no está ya en ListaDestinos
    for i from 1 to ContadorLista
      ListaDestinos.DestinosEscogidos.list.selection = i
      if ListaDestinos.DestinosEscogidos.list.value = ValorDestinos then
        msgInfo("Ya está en la lista.", ValorDestinos)
        return
      endIf
    endFor
    ; añade el elemento a ListaDestinos
    ContadorLista = ContadorLista + 1
    ListaDestinos.DestinosEscogidos.list.count = ContadorLista
    ListaDestinos.DestinosEscogidos.list.selection = ContadorLista
    ListaDestinos.DestinosEscogidos.list.value = ValorDestinos
  endIf
endmethod
```

open

LUGARES.PáginaDestinos.DestinosEscogidos.open

Ajusta la propiedad list.count de el objeto lista a 0, lo cual vacía la lista.

```
method open(var eventInfo Event)
   self.list.count = 0
endmethod
```

pushButton

LUGARES.PáginaDestinos.BotónLugares.pushButton

Sitúa el foco en *PáginaLugares.*

```
method pushButton(var eventInfo Event)
   PáginaLugares.moveTo()
endMethod
```

pushButton

LUGARES.PáginaDestinos.BotónLimpiarConsulta.pushButton

Borra los valores actuales de consulta empleando **THAMLIB.LimpiarValoresPágina.**

```
method pushButton(var eventInfo Event)
   BotónLimpiarDestinosEscogidos.pushButton()
   eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
   thamLib.LimpiarValoresPágina(NombreFicha.name,
thamlib.NombreDePágina(interfaz))
endmethod
```

pushButton

LUGARES.PáginaDestinos.BotónCancelar.pushButton

Oculta la Barra Rápida personalizada y regresa a VACACIÓN.FSL sin efectuar ninguna otra operación.

```
method pushButton(var eventInfo Event)
   ; Ocultamos el mando
   miMando.Hide()
   ; retornamos a vacación sin realizar procesos
   formreturn(True)
endmethod
```

pushButton

LUGARES.PáginaDestinos.BotónAyuda.pushButton

Visualiza la Ayuda llamando a **helpShowContext**. El valor 20003 es un identificador de contexto para un elemento de archivo de Ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method pushButton(var eventInfo Event)
   helpShowContext(ficheroAyuda, 20003)
endmethod
```

pushButton

LUGARES.PáginaDestinos.BotónAceptar.pushButton

Éste se llama eligiendo el botón Aceptar. Construye la cadena "destino" con los destinos seleccionados en ese momento y lo entrga a VALCONSU.DB como un Valor de Consuta.

```
method pushButton(var eventInfo Event)

  Var
    CadenaDestinos
    String
    DestinosTc, tc    TCursor
    i,
    ContadorLista SmallInt
  endVar
  CadenaDestinos = "" ; inicializa la cadena de destino
  eventInfo.getTarget(interfaz)
  thamlib.LimpiarValoresPágina(NombreFicha.Name,
thamlib.NombreDePágina(interfaz))
  ; debemos rellenar los destinos a valconsu.db
  ContadorLista = ListaDestinos.DestinosEscogidos.list.count
  IF ContadorLista > 0  then ; hay destinos
    tc.Open("valconsu.db")
    if not tc.Locate("N° Consulta", 5) then
      msgStop("Problema", "No pude localizar el destino en valconsu")
    else
      tc.Edit() ; editamos VALCONSU.DB
      if ContadorLista > 1 then
        for i from 1 to ContadorLista
         ; construimos una cadena con la lista de destinos
         ListaDestinos.DestinosEscogidos.list.selection = i
         if CadenaDestinos <> ""  then
               CadenaDestinos = ListaDestinos.DestinosEscogidos.list.value
+" OR " + CadenaDestinos
          else
               CadenaDestinos = ListaDestinos.DestinosEscogidos.list.value
          endif
        endfor
      else
        CadenaDestinos = ListaDestinos.DestinosEscogidos.list.value
      endif
      ; asignamos los destinos a Valor Consulta
      tc."Valor Consulta".value = CadenaDestinos
      tc.endEdit()
      tc.Close()
    endif
  endif
  ; ocultamos el mando
  miMando.Hide()
  formReturn(True)
endmethod
```

LUGARES.PáginaLugares

Métodos

arrive
menuAction
open
LimpiarDatosPágina

Procedimientos

MenúPáginaResultados

ventana Uses

Uses

arrive

LUGARES.PáginaLugares.arrive

Ajusta el titulo de *PáginaLugares* a "Lugares" y muestra un menú desplegable llamando al procedimiento **PáginaLugares.MenúPáginaResultados**.

```
method arrive(var eventInfo MoveEvent)
   ; asigna el título
   setTitle("Lugares")
   ; muestra el menú personalizado
   MenúPrincipal()
endmethod
```

menuAction

LUGARES.PáginaLugares.menuAction

Comprueba la cadena devuelta por una selección del menú *PáginaLugares* y llama al método apropiado. Por ejemplo, eligiendo el elemento Aceptar llama a **BotónAceptar.pushButton.**

```
method menuAction(var eventInfo MenuEvent)
   Var
      OpciónMenú String
   endvar
   ; ¿Qué opción de menú se ha seleccionado?
   OpciónMenú = eventInfo.menuChoice()
   switch
     case OpciónMenú = "&Aceptar"              : BotónAceptar.pushbutton()
     case OpciónMenú = "&Cancelar"             : BotónCancelar.pushbutton()
     case OpciónMenú = "&Eliminar"             :
BotónLimpiarConsulta.pushbutton()
     case OpciónMenú = "&Destinos"                  :
BotónPáginaDestinos.pushbutton()
     case OpciónMenú = "&Contenido"            :
helpShowContext(ficheroAyuda, 20004)
     case OpciónMenú = "&Uso de la Ayuda"      : helpOnHelp()
     case OpciónMenú = "&Acerca de THAM"       : thamlib.EsLlamadaVerdadero()
                                                          if
FichaAcercaDe.open("SobMAST.fsl") then
                                                               ; abre
una caja de diálogo modal

   FichaAcercaDe.Wait()

   ; espera y la cierra

   FichaAcercaDe.Close()

   thamlib.EsLlamadaFalso()
                                                     endif
   endswitch
endmethod
```

open

LUGARES.PáginaLugares.open

Esta ficha debe de llamarse desde VACACIÓN.FSL; no puede ejecutarse directamente. Abre la biblioteca THAMLIB.LSL si ésta se encuentra en el directorio TRABAJO.

```
method open(var eventInfo Event)
   ; Este método no abrirá Lugares.fsl a no ser que haya sido requerida desde
   ; VACACION.fsl. La variable EsLlamada de thamlib debe ser verdadera para
abrir Lugares.
   ; Se asigna el valor verdadero a EsLlamada en EsLlamadaVerdadero() que se
ejecuta
   ; en el método Pushbutton del botón correspondiente de VACACION.
   delayScreenUpdates(Yes)
   ; abre la biblioteca (se asegura de que está e el directorio de trabajo)
   if not thamlib.Open("ThamLib", GlobalToDesktop) then
      msgStop("Fallo", "No pude abrir Thamlib.")
      formReturn(false)
   else
      EsLlamada = thamLib.HaLlamadoaRetornar()
      if EsLlamada = False then
         msgStop("¡Alto!", "No se puede abrir esta ficha directamente. Debe
abrirse desde VACACION.fsl.")
         close()
      else
         ; asocia el nombre de la ficha a la variable NombreFicha
         NombreFicha.attach()
         ; Asegúrese de definir el alias "VACACIONs", en otro caso
         ; la propiedad dataSource empleará VACACIONlug.db en el modelo. Esto
produciría
         ; un problema con lugares en el modelo de datos, lo que contrastaría
         ; con la propiedad DataSource de
LugarPreferencia.ListaLugarPreferencia,
         addAlias("VACACIONs", "standard", getAliasPath("Trabajo"))
      endif
   endif
   delayScreenUpdates(No)
endmethod
```

LimpiarDatosPágina

LUGARES.PáginaLugares.LimpiarDatosPágina

Borra todos los objetos campo de esta página asignándoles el valor "" (cadena vacia).

```
method LimpiarDatosPágina(NombrePágina String) Logical
   ; Este método personalizado limpia todos los objetos de la página
   ; que ha sido seleccionada.

   Var
      ObjetosPágina     TCursor
      ObjetosCampo      UIObject
   endVar

   ; lista todos los campos seleccionables de la página
   thamlib.ListaObjetosDePágina(NombreFicha.name, NombrePágina)
   ObjetosPágina.open("listaObj.db")
   ; limpia los objetos seleccionados
   scan ObjetosPágina :
      ObjetosCampo.attach(ObjetosPágina."Vía de Acceso")
      if ObjetosCampo  <> "" then
         ObjetosCampo = ""
      endIf
   endScan
   ; limpia los valores ficha.página en valconsu.db
   thamlib.LimpiarValoresPágina(NombreFicha.Name, NombrePágina)
   RETURN TRUE
endmethod
```

MenúPrincipal

LUGARES.PáginaLugares.MenúPrincipal

Llamado por **PáginaLugares.arrive** para mostrar un menú desplegable personalizado.

```
proc MenúPrincipal()
   Var
      MenúPrincipal Menu
      MenúEmergente1, MenúEmergente2 popUpMenu
   endVar
   ; este procedimiento de usuario muestra el menú definido para la ficha
Lugares
   ; define el menú
   MenúEmergente1.addText("&Aceptar")
   MenúEmergente1.addText("&Cancelar")
   MenúEmergente1.addText("&Eliminar")
   MenúPrincipal.addPopUp("Selecciones", MenúEmergente1)
   MenúPrincipal.addText("&Destinos")
   MenúEmergente2.addText("&Contenido")
   MenúEmergente2.addText("Uso de la Ayuda")
   MenúEmergente2.addText("Acerca de &THAM")
   MenúPrincipal.addPopUp("&Ayuda", MenúEmergente2)
   MenúPrincipal.show()
endproc
```

LUGARES.PáginaLugares Ventana Uses

En la ventana *Uses* de *PáginaLugares* se declaran rutinas externas empleadas por esta página. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   ListaObjetosDePágina(NombreFicha String, NombrePágina String) Logical
endUses
```

open

LUGARES.PáginaLugares.ListaLugarPreferencia.open

Rellena esta lista con elementos del campo Nombre Situación de LUGARES.DB.

```
method open(var eventInfo Event)
   ; el campo se rellena desde Lugares.db. sin asociarlo
   ; El alias :VACACIONs: se emplea para definir la vía de acceso a
lugares.db
   ; en caso contrario podría emplearse lugares.db del modelo de datos
   ; :VACACIONs: se define como :trabajo: en el método open de la ficha.
   self.dataSource = "[:TRABAJO:lugares.Nombre Situación]"
endmethod
```

pushButton

LUGARES.PáginaLugares.BotónAceptar.pushButton

Éste es llamado utilizando **menuAction** al seleccionar el Botón Aceptar. Emplea **THAMLIB.EstablecerValoresPágina** para enviar los valores de los lugares seleccionados (Alojamiento, Características, y Vida Nocturna) a los campos correspondientes en VALCONSU.DB. El campo (Temperatura media) necesita una gestión especial, por ejemplo, "frio" a "<=69".

```
method pushButton(var eventInfo Event)
  Var
    ValorTemporal String
    Tc TCursor
  endVar

  ; Necesitamos realizar algunos procesos especiales para enviar los campos
  ; correspondientes al alojamiento, características, vida nocturna y
temperatura media
  ; en valconsu.db
  ; Con la excepción de temperatura media - que requiere un tratamiento
especial -
  ; se ejecuta el método EstablecerValoresPágina() de thamlib.lsl para hacer
el proceso

  eventInfo.getTarget(interfaz)
  ThamLib.EstablecerValoresPágina(NombreFicha.name,
thamlib.NombreDePágina(interfaz))
  tc.Open("ValConsu.db")
  tc.Edit()
  ; Busca el registro Temperatura Media en valconsu.db
  if not tc.Locate("N° Consulta", 9)  then
     msgStop("Problemas", "No pude encontrar el registro Temperatura Media
en valconsu.db")
  else
    ValorTemporal = ""
    if fría <> "" then
      ValorTemporal = "<=20"
    endif
    if subTropical <> "" then
      if ValorTemporal <> "" then
        ValorTemporal = "<=30"
      else
        ValorTemporal = ">=21, <=29"
      endif
    endif
    if tropical <> "" then
      if ValorTemporal = "" then
        ValorTemporal = ">=40"
      else
        ValorTemporal= ValorTemporal+" OR >=40"
      endif
    endif
  endif
  tc.endEdit()
```

```
   tc.close()
   ; retorna a VACACION
   formReturn(True)
endmethod
```

pushButton

LUGARES.PáginaLugares.BotónLimpiarConsulta.pushButton

Borra los valores actuales de consulta empleando **THAMLIB.LimpiarDatosPágina**.

```
method pushButton(var eventInfo Event)
   ; ¿Qué objeto es?
   eventInfo.getTarget(interfaz)
   ; limpia los objetos y además limpia los valores de valconsu.db
   LimpiarDatosPágina(thamlib.NombreDePágina(interfaz))
endmethod
```

pushButton

LUGARES.PáginaLugares.BotónCancelar.pushButton

Regresa a VACACIÓN.FSL sin efectuar ningún otro proceso.

```
method pushButton(var eventInfo Event)
   ; retorna a VACACION sin realizar procesos
   formreturn(True)
endmethod
```

pushButton

LUGARES.PáginaLugares.BotónPáginaDestinos.pushButton

Sitúa al usuario en *PáginaDestinos.*

```
method pushButton(var eventInfo Event)
   PáginaDestinos.moveto()
endmethod
```

pushButton

LUGARES.PáginaLugares.BotónAyuda.pushButton

Proporciona ayuda llamando a **helpShowContext**. El valor 20004 es una identificación de contexto de un elemento del archivo ayuda. En aplicaciones mas grandes, es recomendable definir constantes para esos valores en vez de introducirlas directamente en el código.

```
method pushButton(var eventInfo Event)
helpShowContext(helpfile, 20004)
endmethod
```

LISTA

Métodos

open
close
menuAction
keyPhysical
mouseRightUp
MuestraMe

Variables, Constantes, y ventanas Uses

ventana de Variables
ventana Uses

Métodos y procedimientos de THAMLIB.LSL

CogerTítuloMejoras
CogerDatosFuente
EstablecerDatosFuente

open

LISTA.open

Abre la biblioteca THAMLIB.LSL. **GlobalToDesktop** hace que esta ventana de la biblioteca sea accesible por otras fichas, con la condición de que las otras fichas también empleen **GlobalToDesktop**.

```
method open(var eventInfo Event)
   if eventInfo.isPreFilter() then
     ; el código que sitúe aquí lo ejecutarán todos los objetos de la ficha
   else
     ; este código lo ejecuta sólo la propia ficha
     thamlib.open("thamlib.lsl", GlobalToDesktop)
     NombreFicha.attach()
   endIf
endMethod
```

close

LISTA.close

Cierra la biblioteca THAMLIB.LSL.

```
method close(var eventInfo Event)
   if eventInfo.isPreFilter() then
     ; el código que aquí se incluya será ejecutao por todos los objetos de
la ficha
   else
     ; este código lo ejecuta sólo la propia ficha
   thamlib.close()
   endif
endmethod
```

menuAction

LISTA.menuAction

Este código se ejecuta cuando se emplea el menú de control para cerrar la ficha, deshabilitando la respuesta estándar y llamando a **BotónCancelar.pushButton** en su lugar. En consecuencia, cerrar la ficha empleando el menú de control tiene el mismo efecto que seleccionando el botón Cancelar.

```
method menuAction(var eventInfo MenuEvent)
if eventInfo.isPreFilter() then
   ;este código se ejecuta para cada objeto de la ficha.
   if eventInfo.id() = menuControlClose then
      disableDefault
      BotónCancelar.pushButton()
   endIf
else
   ;este código se ejecuta sólo para la propia ficha.
endif
endmethod
```

keyPhysical

LISTA.keyPhysical

Captura la tecla *Esc* para llamar a **BotónCancelar.pushButton.** Captura *Alt+A* para situarse sobre BotónListarTodo.

```
method keyPhysical(var eventInfo KeyEvent)
   if eventInfo.isFirstTime() then
      if eventInfo.vCharCode() = VK_ESCAPE then
         disableDefault
         BotónCancelar.pushButton()
      endIf
      if eventInfo.isAltKeyDown() then
         if eventInfo.vChar() = "T" then
            BotónListarTodo.moveTo()
         endIf
      endIf
   endIf
endmethod
```

mouseRightUp

LISTA.mouseRightUp

Llama al método **crearMenú** de la libreria THAMLIB cuando se suelta el botón derecho del raton. Esto genera el menú de la Ayuda de Código. Si se hace clic en un objeto Texto o en un objeto Mapa de bits, se muestra el menú de ayuda de los objetos que contiene debido a que estos objetos no tienen código asociado.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Ejecuta el método thamlib.CrearMenú() para mostrar un menú emergente
   ; para los distintos objetos que admitan el clic del botón derecho
   if eventInfo.isPreFilter()  then
      eventInfo.getTarget(interfaz) ; ¿Qué Objeto se pulsó?
       ; Si no es el bitmap CuadroBandera,
       ; busca el contenedor de los objetos de texto y bitmaps
      if interfaz.name <> "CuadroBandera"  then
        while
           interfaz.class = "Text" or interfaz.class = "Bitmap"
           interfaz.attach(interfaz.ContainerName)
        endWhile
      endif
      thamLib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endmethod
```

MuestraMe

LISTA.MuestraMe

Inicializa el cuadro de diálogo LISTA y lo hace visible.

```
method MuestraMe(const elTítulo String, const ListaDatosFuente String)
   setTitle(elTítulo)
   PáginaLista.CampoVisor.VisorLista.dataSource = ListaDatosFuente
   PáginaLista.CampoVisor.VisorLista.list.selection = 1 ; sobreilumina el
primer valor
   if ListaDatosFuente = "[destipos.Destino]" then
      BotónListarTodo.visible = True
   else
      BotónListarTodo.visible = False
   endIf
   CampoVisor.moveTo()
   bringToTop()
endmethod
```

LISTA Ventana Var

Las variables declaradas en la ventana de Variables de la ficha son accesibles para todos los objetos contenidos en esa ficha.

```
Var
   thamlib  Library
   interfaz       UIObject
   NombreFicha    Form
endVar
```

LISTA Ventana Uses

En la ventana *Uses* se declaran rutinas externas empleadas por esta ficha. Las rutinas pueden almacenarse en una a DLL, en una biblioteca ObjectPAL, o en otra ficha. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, y no en algun otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPAL
  CogerTítuloMejoras() String
  CogerDatosFuente() String
  EstablecerDatosFuente(const Datos String)
  CrearMenú(var NombreFicha Form, NombrePágina String, interfaz UIObject)
  NombreDePágina(interfaz UIObject) string
endUses
```

pushButton

LISTA.PáginaLista.BotónListarTodo.pushButton

Este botón se muestra como un cuadro de comprobación. Cuando el botón está seleccionado , *VisorLista* muestra todos los campos destino en DESTINOS.DB; en caso contrario la lista queda restrigida a los campos seleccionados en DESTIPOS.DB.

```
method pushButton(var eventInfo Event)
   doDefault
   switch
      case self.value = True :
         CampoVisor.VisorLista.dataSource = "[destinos.N° Destino]"
      case self.value = False :
         CampoVisor.VisorLista.dataSource = "[destiPos.Destino]"
   endSwitch
endmethod
```

pushButton

LISTA.PáginaLista.BotónCancelar.pushButton

Devuelve el control al método invocante sin efectuar ninguna otra acción.

```
method pushButton(var eventInfo Event)
   formReturn("")
endmethod
```

pushButton

LISTA.PáginaLista.BotónAceptar.pushButton

Devuelve el control al método invocante junto con el valor del campo seleccionado.

```
method pushButton(var eventInfo Event)
   formReturn(PáginaLista.CampoVisor.VisorLista)
endmethod
```

LISTA.PáginaLista.CampoVisor

Métodos

keyPhysical
newValue

keyPhysical

LISTA.PáginaLista.CampoVisor.keyPhysical

Captura la pulsación de tecla *Return* y llama a **BotónAceptar.pushButton.**

```
method keyPhysical(var eventInfo KeyEvent)
   if eventInfo.vCharCode() = VK_RETURN then
     disableDefault
     BotónAceptar.pushButton()
   endIf
endmethod
```

newValue

LISTA.PáginaLista.CampoVisor.newValue

Muestra el nuevo valor de este campo llamando a **message(self.value).**

```
method newValue(var eventInfo Event)
   message(self.value)
endmethod
```

mouseDouble

LISTA.PáginaLista.ValorLista.mouseDouble

Convierte un doble clic a **BotónAceptar.pushButton.**

```
method mouseDouble(var eventInfo MouseEvent)
   BotónAceptar.pushButton() ; doble clic significa Aceptar
endmethod
```

MANDOVID

Métodos

open
keyChar
mouseRightUp
keyPhysical
EstablecerNombreFichaLlamar

Variables, Constantes, y ventanas Uses

Ventana Var
Ventana Const
Ventana Uses

open

MANDOVID.open

Abre la biblioteca THAMLIB.LSL si la encuentra en el directorio. **GlobalToDesktop** hace que esta ventana sea accesible por otras fichas, dichas fichas tambien usan **GlobalToDesktop**.

```
method open(var eventInfo Event)
   if not eventInfo.isPreFilter() then
      ; abre la biblioteca (asegúrese de que se encuentra en el directorio de
trabajo)
      if not thamLib.open("ThamLib", globalToDesktop) then
        msgStop("Fallo", "No pude abrir Thamlib.")
        close()
      else
        NombreFicha.attach()
      endif
   endif
endmethod
```

keyChar

MANDOVID.keyChar

Captura las pulsaciones de tecla "f", "p", "n", y "l" Para invocar los métodos **pushButton BotóIrPrimero, BotónIrAnterior, BotónIrSiguiente,** y **BotónIrÚltimo** respectivamente.

```
method keyChar(var eventInfo KeyEvent)
   switch
      case eventInfo.char() = "p" : BotónIrPrimero.pushButton()
      case eventInfo.char() = "a" : BotónIrAnterior.pushButton()
      case eventInfo.char() = "s" : BotónIrSiguiente.pushButton()
      case eventInfo.char() = "u" : BotónIrUltimo.pushButton()
   endswitch
endmethod
```

mouseRightUp

MANDOVID.mouseRightUp

Llama al método **crearMenú** de la biblioteca THAMLIB cuando se suelta el botón derecho del ratón. Esto genera el menú de Ayuda de Código. Si se hace clic en un objeto Texto o en un objeto Mapa de bits, se muestra el menú de ayuda de los objetos que contiene debido a que estos objetos no tienen código asociado.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Ejecuta el método CrearMenú() de la biblioteca thamlib que visualiza
   ; un menú de ayuda a ivel de objeto.

   ; evita sucesivas llamadas al método
   if eventInfo.isPreFilter() then
      ; pasa el foco al mando
      bringToTop()
      eventInfo.getTarget(interfaz) ; ¿Qué objeto es?
      ; busca el contenedor de interfaz si se trata de un texto o un bitmap
      while interfaz.class = "Text" or interfaz.class = "Bitmap"
         interfaz.attach(interfaz.ContainerName)
      endWhile
      thamLib.CrearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
      ; devuelve el foco a la ficha que lo llamó
      FichaLlamador.bringToTop()
   endif
endmethod
```

keyPhysical

MANDOVID.keyPhysical

Éste captura pulsaciones de tecla y las convierte en llamadas a métodos, si procede. Los valores 20002, 20003, y 20005 son identificadores de contexto de elementos del archivo de ayuda. En aplicaciones más grandes, es recomendable definir constantes para estos valores.

```
method keyPhysical(var eventInfo KeyEvent)
   ; Si pulsan F1, muestra la ayuda del método que llamó al mando
   if eventInfo.isFirstTime() then
      if eventInfo.vChar() = "VK_F1" then
         disableDefault
        ; ¿Qué ficha abrió el mando?
        Switch
          Case FichaLlamador.name = "VidaMar"        :
helpShowContext(ficheroAyuda, 20002)
          Case FichaLlamador.name = "Lugares"        :
helpShowContext(ficheroAyuda, 20003)
          Case FichaLlamador.name = "Naufragi" :
helpShowContext(ficheroAyuda, 20005)
        endSwitch
      endIf
   endIf
endMethod
```

EstablecerNombreFichaLlamar

MANDOVID.EstablecerNombreFichaLlamar

Ajusta la posición de la Barra Rápida personalizada dependiendo del nombre de la ficha que la llama.

```
method EstablecerNombreFichaLlamar(const NombreFichaLlamador String, xPos
LongInt, yPos LongInt)
   ; la posición del mando depende de la ficha que lo emplee
  Var
     x,
     y,
     w,
     h    LongInt
  endVar
  ; asociamos la ficha que llama al mando
  FichaLlamador.attach(NombreFichaLlamador)
  ; recogemos la posición del mando
  getPosition(x, y, w, h)
  ; lo cambiamos de posición dependiendo de los parámetros de la página
   setPosition(xPos, yPos, w, h)
   ;  pasamos el foco a la ficha que llamó al mando
   FichaLlamador.bringToTop()
endmethod
```

MANDOVID Ventana Var

Las Variables declaradas en la ficha son accesibles por todos los objetos contenidos en la ficha.

```
Var
   FichaLlamador Form
   Interfaz UIObject
   NombreFicha Form
   thamlib Library
   FueLlamado Logical
endVar
```

MANDOVID Ventana Const

Las constantes declaradas en la ventana de Constantes de la ficha son accesibles por todos los objetos contenidos en esa ficha.

```
Const
   FicheroAyuda = ":trabajo:usuario.hlp"
endConst
```

MANDOVID Ventana Uses

En la ventana *Uses* se declaran rutinas externas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran métodos almacenados en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPal
   CrearMenú(var NombreFicha Form, NombrePágina String, interfaz UIObject)
   NombreDePágina(interfaz UIObject) String
   HaLlamadoaRetornar() Logical
    MandoEstáCerrado()
endUses
```

pushButton

MANDOVID.Regleta.BotónIrÚltimo.pushButton

Selecciona el último registro empleando **FichaLlamador.action(DataEnd).** Por ejemplo, si NAUFRAGI.FSL es la ficha invocante, este método muestra el último registro de NAUFRAGI.DB.

```
method pushButton(var eventInfo Event)
   FichaLlamador.bringToTop()
  FichaLlamador.action(DataEnd)
endmethod
```

pushButton

MANDOVID.Regleta.BotónIrSiguiente.pushButton

Muestra el siguiente registro empleando **FichaLlamador.action(DataNextRecord).** Por ejemplo, si VIDAMAR.FSL es la ficha invocante, seleciona el siguiente registro de VIDAMAR.DB empleando **FichaLlamador.action(DataNextRecord)** y mostrándonos toda la información de ese registro.

```
method pushButton(var eventInfo Event)
   FichaLlamador.bringToTop()
   FichaLlamador.action(DataNextRecord)
endmethod
```

pushButton

MANDOVID.Regleta.BotónIrAnterior.pushButton

Selecciona el registro anterior empleando **FichaLlamador.action(DataPriorRecord).** Por ejemplo, si VIDAMAR.FSL es la ficha invocante, este método selecciona el registro anterior de VIDAMAR.DB empleando **FichaLlamador.action(DataNextRecord)** y mostrándonos toda la información de ese registro.

```
method pushButton(var eventInfo Event)
   FichaLlamador.bringToTop()
  FichaLlamador.action(DataPriorRecord)
endmethod
```

pushButton

MANDOVID.Regleta.BotónIrPrimero.pushButton

Selecciona el primer registro empleando **FichaLlamador.action(DataBegin).** Por ejemplo, si VIDAMAR.FSL es la ficha invocante, este método seleccina el primer registro de VIDAMAR.DB empleando **FichaLlamador.action(DataNextRecord)** y mostrándonos toda la información de ese registro.

```
method pushButton(var eventInfo Event)
   FichaLlamador.bringToTop()
  FichaLlamador.action(DataBegin)
endmethod
```

PEDIDOS

Métodos

open
mouseEnter
mouseExit
mouseRightUp
keyPhysical
menuAction

Procedimientos

CorrectoPedido
NúmeroDeSemanas
RecogerMensaje
MontarLista
CosteDestino

Variables, Constantes, y ventanas Uses

Ventana Var
Ventana Const
Ventana Uses

Métodos y procedimientos de THAMLIB.LSL

NombreDePágina
MensajeTHAM
crearMenú
EstablecerTítuloMejoras
EstablecerDatosFuente

open

PEDIDOS.open

Cuando se abre la ficha *PEDIDOS*, se ejecuta el código asociado a su método estándar **open**. Este código realiza varias tareas para inicializar la ficha.

```
method open(var eventInfo Event)
  if not eventInfo.isPreFilter() then ; este código sólo lo emplea la ficha
    tracerOff()
    if not isFile("pedidos.fsl") then
      msgInfo("¡Error de Arranque!",
              "Los ficheros de la aplicación THAM deben estar en el
directorio de trabajo.")
      close()
    endif
    ; abre la biblioteca
    THAMLib.open("ThamLib", GlobalToDesktop)
    if isSpeedBarShowing() then
      hideSpeedBar()
      thamLib.PuedeCerrarseFicha()
      OcultarBarra = True ; la variable de nivel de página OcultarBarra
avisa cuándo oculta la barra
                          ; rápida o cuando se encuentra oculta por acción
de otra ficha.
      maximize()        ; como no nos ha llamado Vacación, maximizamos la
ficha.
    else
      OcultarBarra = False
    endIf
    MarcoTabla.attach(Buceoart)
    mensaje.open("Ayuda.db"); abrimos la tabla de mensajes
    ; empleamos una bandera para si el usuario emplea el botón Cancelar.
    Cancelar = False
  endif
endmethod
```

mouseEnter

PEDIDOS.mouseEnter

Muestra mensajes referidos a campos y a multiregistros. Cuando el cursor del ratón entra en su zona.

```
method mouseEnter(var eventInfo MouseEvent)
  var
    interfaz UIObject
  endvar
  if eventInfo.isPreFilter() then
    ; este código lo ejecutan todos los objetos de la ficha
    eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
    if interfaz.name <> self.name AND  ; si se trata de la ficha elimina el
mensaje
       (interfaz.class = "Field" OR interfaz.class = "Multirecord" OR
interfaz.class = "Button") then
       thamLib.MensajeTHAM(NombreFicha.Name,
thamlib.NombreDePágina(interfaz), interfaz.name)
    endif
  else
    ; el código que se incluya aquí, lo ejecutará sólo la propia ficha
  endIf
endMethod
```

Vea también

PEDIDOS.RecogerMensaje

mouseExit

PEDIDOS.mouseExit

Cuando el ratón abandona un objeto campo o multi-registro, **mouseExit** borra el mensaje generado por una llamada previa a **mouseEnter**.

```
method mouseExit(var eventInfo MouseEvent)
  var
    interfaz UIObject
  endvar
  if eventINfo.isPreFilter() then
    ; este código lo ejecutan todos los objetos de la ficha
    eventInfo.getTarget(interfaz)  ; ¿Qué objeto es?
    if interfaz.name <> self.name AND interfaz.class="Field" then  ; si se
trata de la ficha, elimina el mensaje
      message("")
    endif
  else
    ; el código que aquí se incluya lo ejecutará sólo la propia ficha
  endIf
endMethod
```

mouseRightUp

PEDIDOS.mouseRightUp

Este codigo se ejecuta siempre que se efectúe un clic sobre un objeto en la ficha *PEDIDOS*. Éste reconoce que objeto fue diana del clic, y acto seguido llama al método personalizado **crearMenú** de la biblioteca THAMLIB para crear en el objeto diana un menú desplegable.

Fíjese en que el código está situado en la claúsula **isPreFilter,** de esta forma éste se ejecuta *antes* de que la ficha entregue el suceso al objeto diana. Por ejemplo, cuando se hace clic sobre un objeto campo, el suceso se entrega primero a la ficha, este código se ejecuta en respuesta al suceso antes de que la ficha lo entregue al objeto campo.

```
method mouseRightUp(var eventInfo MouseEvent)
   ; Llama a thamlib.CrearMenú para mostrar el menú emergente
   ; del objeto que ha recibido la pulsación con el botón derecho del ratón.
   if eventInfo.isPreFilter()  then
      eventInfo.getTarget(interfaz) ; ¿Sobre qué objeto ha pulsado?
                                          ; Si no es el bitmap CuadroBandera
                                    ; busca el contenedor cuando se trate de
texto o bitmaps
       if interfaz.name <> "CuaroBandera1"  then
        while
           interfaz.class = "Text" or interfaz.class = "Bitmap"
           interfaz.attach(interfaz.ContainerName)
        endWhile
      endif
      thamLib.crearMenú(NombreFicha, thamlib.NombreDePágina(interfaz),
interfaz)
   endif
endmethod
```

keyPhysical

PEDIDOS.keyPhysical

Éste emplea **hotKey** para capturar la tecla *F1* y varias combinaciones de *Alt*+tecla. La llamada a **isPreFilter** asegura que el código siguiente se ejecute antes de que la ficha entregue el suceso al objeto diana.

```
method keyPhysical(var eventInfo KeyEvent)
  var
    laTecla String
    páginaDestino UIObject
  endvar
  if eventInfo.isPreFilter() then
    laTecla = eventInfo.vChar()
    if eventInfo.isAltKeyDown() then
      páginaDestino.attach(thamLib.NombreDePágina(active))
      if páginaDestino.TeclaRápida(eventInfo) then ; si la página maneja la
tecla (TeclaRápida es un método personalizado asociado a la página)
        disableDefault
        return
      endIf
    endIf
    IF laTecla = "VK_F1" THEN
      disableDefault
      IF thamLib.NombreDePágina(active)="PedidoPg" THEN
        helpShowContext(FICHEROAYUDA, CABECERAAYUDA)
      ELSE
        helpShowContext(FICHEROAYUDA, DETALLEAYUDA)
      ENDIF
    ENDIF
  endif
endMethod
```

menuAction

PEDIDOS.menuAction

Este código evita que el usuario emplee el menú de Control para cerra la ficha.

```
method menuAction(var eventInfo MenuEvent)
   if eventInfo.isPreFilter() then
     ; este código lo ejecutan todos los objetos de la ficha.
     doDefault
   else
     ; este código lo ejecuta sólo la propia ficha.
     ; bloque empleado para cerrar la ficha
     If eventInfo.ID() = MenuCanClose and thamlib.PuedeCerrarseFicha() =
False then
       eventInfo.setErrorCode(1)
       formReturn(True)
       disableDefault
     endIf
   endIf
endMethod
```

ESCORRECTOPEDIDO

PEDIDOS.ESCORRECTOPEDIDO

Verifica que todos los campos requeridos de BUCEOCLI.DB y BUCEOPED.DB tienen un valor. Devolvera *false* en caso contrario.

```
PROC esCorrectoPedido() LOGICAL
  var
    CamposVerificablesDYNARRAY[] String
    clienteTC                    TCURSOR
    x                            ANYTYPE
    NúmeroMaximo            NUMBER
    interfaz                             UIOBJECT
  endVar
  CamposVerificables["Nombre"]="Nombre"
  CamposVerificables["Direción"]="Dirección"
  CamposVerificables["Ciudad"]="Ciudad"
  CamposVerificables["Telefóno"]="Teléfono"
  CamposVerificables["N°_de_personas"]="N°_de_personas"
  CamposVerificables["Fecha_de_salida"]="Fecha_de_salida"
  CamposVerificables["Fecha_de_Llegada"]="Fecha_de_Llegada"
  CamposVerificables["Destino"]="Destino"
  FOREACH i IN CamposVerificables
    interfaz.attach(CamposVerificables[i])
    If interfaz.value = "" THEN
      RETURN FALSE
    ENDIF
  ENDFOREACH
  return True
ENDPROC
```

NúmeroDeSemanas

PEDIDOS.NúmeroDeSemanas

Devuelve el número de semanas, se calculan empleando los valores de los objetos campo *FechaDeSalida* y *FechaDeLlegada*.

```
Proc NúmeroDeSemanas() Number
  var
    númeroSemanas     Number
  endVar
  NúmeroSemanas = ceil(Number(Date(Fecha_de_Llegada.value)-
Date(Fecha_de_Salida.value)))/7
  if númeroSemanas > 0 then
    return(NúmeroSemanas)
  else
    return(1)
  endIf
endProc
```

CogeMensajeObjeto

PEDIDOS.CojeMensajeObjeto

Dado un objeto, busca en la tabla de mensajes, AYUDA.DB, y devuelve un cursor que apunta al mensaje correspondiente. Éste lo emplea **PEDIDOS.mouseEnter** para mostrar el mensaje apropiado según la posición del cursor.

```
PROC CogeMensajeObjeto(interfaz UIOBJECT, tipoMensaje STRING) STRING

  ; comprueba si este nombre de contenedor existe en la tabla de mensajes.
  IF mensaje.locate("Nombre Ficha", "pedido", "Objeto", interfaz.name) THEN
    RETURN mensaje.(tipoMensaje)
  ELSE
    RETURN ""
  ENDIF
endPROC
```

HacerLista

PEDIDOS.HacerLista

Este procedimiento personalizado muestra un cuadro de diálogo que contiene una lista en la que el usuario puede seleccionar un valor. El argumento *elTítulo* especifica el texto que se muestra en la barra del titulo del cuadro de dialogo, y el argumento *listDataSource* especifica la tabla a emplear por la lista como fuente de datos.

El código del bloque **try...onFail** trata de hacer visible el cuadro de dialogo. De esta forma el usuario puede efectuar una seleción, haciéndola invisible cuando el usuario termina. Esto es más rápido que abrir y cerrar el cuadro de dialogo cada vez que se emplea.

MontarLista es invocado por el método estándar **keyPhysical** del objeto campo *Nombre* y del objeto campo *Destino*.

```
proc HacerLista(const elTítulo String, const ListaDatosFuente String) String
  var
    ListaValores String
  endvar
  try
    FichaLista.MuestraMe(elTítulo, listaDatosFuente)
    ; Muestrame es un método personalizado asociado a la ficha lista.
  onFail
    try
      FichaLista.hide()
    onFail
      FichaLista.open("Lista.fsl")
    endTry
    retry
  endTry
  ListaValores = FichaLista.wait()
  try
    Fichalista.hide()
  onFail
    if errorCode() = peFormClosed then
      ; Si el usuario cierra el cuadro de diálogo, no es necesario
ocultarla.
      return ""
    endIf
  endTry
  return ListaValores
endProc
```

CosteDelDestino

PEDIDOS.CosteDelDestino

Este procedimiento personalizado devuelve el coste de un viaje a un destino determinado.

```
proc CosteDelDestino() NUMBER
  var
    destinosTC  TCURSOR
    NuevoCoste  NUMBER
  endVar

  if destino.isBlank() THEN
    RETURN 0
  ENDIF
  destinosTC.open("destinos.db")
  IF destinosTC.locate("Destino", STRING(destino.value)) THEN
    NuevoCoste = destinosTC."Precio del Viaje"
  ELSE
    NuevoCoste = 0
  ENDIF
  destinosTC.close()
  RETURN NuevoCoste
endproc
```

PEDIDOS Ventana Var

La variables declaradas en la ventana de Variables de la ficha son accesibles por todos los objetos contenidos en esa ficha.

```
Var
   THAMLib                    LIBRARY
   NombreFicha,
   FichaLista                       FORM
    interfaz                              UIObject
   Mensaje,
   MarcoTabla                       TCursor
   RegistrosBorrados          String
   OcultarBarra               Logical
   NuevoCliente,
   NuevoPedido,
   SumaArtículos,
    NúmeroSemanas,
   CosteDestino               Number
   Cancelar                         Logical
endVar
```

PEDIDOS Ventana Const

Las constantes declaradas en la ventana de constantes son accesibles por todos los objetos de la ficha.

```
Const
   PORCENTAJEIMPUESTOS = .15 ; 15%
   PORCENTAJEALQUILER  = .15 ; 15%
   FICHEROAYUDA        = ":TRABAJO:usuario.hlp" ; vía de acceso al fichero de
ayuda
   CABECERAAYUDA         = LongInt(20006)
   DETALLEAYUDA        = LongInt(20007)
endConst
```

PEDIDOS Ventana Uses

En la ventana *Uses* se declaran rutinas externas empleadas por esta ficha. Las rutinas pueden estar almacenadas en en una DLL, una biblioteca ObjectPAL, u otra ficha. En el código siguiente se declaran los métodos almacenados en en la biblioteca ObjectPAL THAMLIB. La primera línea incluye la palabra reservada **ObjectPAL** para indicar que los métodos están escritos en ObjectPAL, en vez de en algún otro lenguaje de programación de los empleados para crear una DLL.

```
Uses ObjectPAL
   ; rutinas almacenaas en THAMLIB.LSL (Bilioteca)
   NombreDePágina(interfaz UIObject) STRING
   MensajeTHAM(NombreFicha String, NombrePágina String, objeto String)
   crearMenú(var NombreFicha Form,  NombrePágina String, interfaz UIObject)
   EstablecerTítuloLista(const TítuloLista String)
   EstablecerDatosFuente(const CampoVisor String)
   ; método personalizado asociado a LISTA.FSL (ficha)
   MuestraMe(const elTítulo String, const ListaDatosFuente String)
    ; Dice si puede cerrarse la ficha desde el menú de control
    PuedeCerrarseFicha() Logical
    PuedeCerrarFicha()
endUses
```

PEDIDOS.PedidoPg2

Métodos

open
arrive
menuAction
TeclaRápida

Procedimientos

EstablecerLíneaMenú

Variables, Constantes, y ventanas Uses

Ventana Var
Ventana Const

open

PEDIDOS.PedidoPg2.open

Llama a **setLineOrdMn** para mostrar el menú.

```
method open(var eventInfo Event)
   EstablecerMenúPedidos()
endmethod
```

arrive

PEDIDOS.PedidoPg2.arrive

Muestra el menú correspondiente a esta página y asocia un TCursor al registro actual del marco de tabla *BUCEOART*. El TCursor lo emplean los campos calculados de esta página.

```
method arrive(var eventInfo MoveEvent)
   NúmeroSemanas = NúmeroDeSemanas()
   MarcoTabla.attach(buceoart)
   SumaArtículos = MarcoTabla.cSum("Total")
  doDefault
  MenúLínea.show()
endMethod
```

menuAction

PEDIDOS.PedidoPg2.menuAction

Este código ajusta los atributos de visualización de los items menú y llama a la acción apropiada de una selección de cualquiera de las líneas de ordenes de los menus. Por ejemplo, seleccionando Primero se acciona una llamada a **BUCEOART.action(DataBegin)**. Eligiendo Reservar se devuelve el control a *PedidoPg*, la página principal de PEDIDOS. Los atributos de visualización se ajustan por comprobación de **eventInfo.id = MenuInit**. Cuando se hace clic sobre un elemento en la barra del menú, éste genera un suceso MenuInit justo antes de mostrar el menú desplegable. En ese momento, se pueden especificar los atributos de visualización de las opciones del menú.

```
method menuAction(var eventInfo MenuEvent)
   var
      IdentificadorMenú SmallInt
      ValorRetornado                Logical
   endVar
   IdentificadorMenú = eventInfo.id()
   if eventInfo.id() = MenuInit then
      try
         ValorRetornado = active.Editing
      onFail
         return
      endTry
      if ValorRetornado = True then
         if active.SelectedText <> "" then
            setMenuChoiceAttributeByID(UserMenu + EditarCortar, MenuEnabled)
            setMenuChoiceAttributeByID(UserMenu + EditarCopiar, MenuEnabled)
         else
            setMenuChoiceAttributeByID(UserMenu + EditarCortar,
                                       MenuGrayed + MenuDisabled)
            setMenuChoiceAttributeByID(UserMenu + EditarCopiar,
                                        MenuGrayed + MenuDisabled)
         endIf
      endIf
   endIf
   switch
      ; Menú Vacaciones
      CASE IdentificadorMenú = UserMenu + ReservaVacaciones :
pedidoPg.moveTo()
      ; Menú Editar
      CASE IdentificadorMenú = UserMenu + EditarCortar   :
active.menuAction(MenuEditCut)
      CASE IdentificadorMenú = UserMenu + EditarCopiar  :
active.menuAction(MenuEditCopy)
      CASE IdentificadorMenú = UserMenu + EditarPegar :
active.menuAction(MenuEditPaste)
      ; Menú Registro
      CASE IdentificadorMenú = UserMenu + RegistroPrimero   :
active.action(DataBegin)
      CASE IdentificadorMenú = UserMenu + RegistroUltimo    :
active.action(DataEnd)
      CASE IdentificadorMenú = UserMenu + RegistroSiguiente :
active.action(DataNextRecord)
```

```
     CASE IdentificadorMenú = UserMenu + RegistroAnterior  :
active.action(DataPriorRecord)
     CASE IdentificadorMenú = UserMenu + RegistroInsertar  :
active.action(DataInsertRecord)
     CASE IdentificadorMenú = UserMenu + RegistroBorrar    :
active.action(DataDeleteRecord)
     CASE IdentificadorMenú = UserMenu + RegistroCancelar  :
active.action(DataCancelRecord)
     ; Menú Ayuda
     CASE IdentificadorMenú = UserMenu + AyudaUsoAyuda     :
helpShowContext(FICHEROAYUDA, DETALLEAYUDA)
     CASE IdentificadorMenú = UserMenu + AyudaSistemaAyuda : helpOnHelp()
     CASE IdentificadorMenú = UserMenu + AyudaIndiceAyuda  :
helpShowIndex(FICHEROAYUDA)
  endSwitch
endMethod
```

Vea también

PEDIDOS.PedidoPg2.EstablecerLíneaMenú

TeclaRápida

PEDIDOS.PedidoPg2.TeclaRápida

Un procedimiento auxiliar es empleado por **PEDIDOS.keyPhysical.** éste permite a **keyPhysical** llamar bien a **PedidoPg.TeclaRápida** o a **PedidoPg2.TeclaRápida** dependiendo de la página activa. Esta **TeclaRápida** captura pulsaciones de tecla y efectúa las pertinentes acciones **moveTo** y **pushButton**. Si las acciones se consuman, se devuelve True, en caso contrario se devuelve False. Las pulsaciones de tecla que no se capturan en esta sección del código devuelven False.

```
method TeclaRápida(var eventInfo KeyEvent) Logical
  var
    laTecla String
  endvar
  if eventInfo.isAltKeyDown() then
    laTecla = eventInfo.vChar()
    switch
      case laTecla = "V" : return BotónReservar.pushButton()
      case laTecla = "N" : return BotónNotas.pushButton()
      case laTecla = "C" : return
BUCEOART.RegistroArtículo.Cantidad.moveTo()
      case laTecla = "A" : return
BUCEOART.RegistroArtículo.Alquiler_Venta.moveTo()
      case laTecla = "F" : return CajaSubTotal.Forma_de_Envío.moveTo()
      otherwise          : return False
    endSwitch
  endIf
endmethod
```

EstablecerLíneaMenú

PEDIDOS.PedidoPg2.EstablecerLíneaMenú

Éste es llamado por **PedidoPg2.open** para crear los tres menús emergentes que componen el menú de PEDIDOS.

```
proc EstablecerMenúPedidos()
   vacaciónEmer.addText("&Reserva", MenuEnabled, UserMenu +
ReservaVacaciones)
   MenúLínea.addPopup("&Vacaciones", vacaciónEmer)
   EditarEmer.addText("Cor&tar", MenuGrayed + MenuDisabled, UserMenu +
EditarCortar)
   EditarEmer.addText("&Copiar", MenuGrayed + MenuDisabled, UserMenu +
EditarCopiar)
   EditarEmer.addText("&Pegar", MenuGrayed + MenuDisabled, UserMenu +
EditarPegar)
   MenúLínea.addPopUp("&Editar", EditarEmer)
   RegistroEmer.addText("&Primero\tCtrl + F11", MenuEnabled, UserMenu +
RegistroPrimero)
   RegistroEmer.addText("&Ultimo\tCtrl + F12", MenuEnabled, UserMenu +
RegistroUltimo)
   RegistroEmer.addText("&Siguiente\tF12", MenuEnabled,  UserMenu +
RegistroSiguiente)
   RegistroEmer.addText("&Anterior\tF11", MenuEnabled,  UserMenu +
RegistroAnterior)
   RegistroEmer.addSeparator()
   RegistroEmer.addText("&Insertar\tINS", MenuEnabled,  UserMenu +
RegistroInsertar)
   RegistroEmer.addText("&Borrar\tCtrl + DEL", MenuEnabled, UserMenu +
RegistroBorrar)
   RegistroEmer.addText("&Cancelar Cambios\tAlt + Bksp", MenuEnabled,
UserMenu + RegistroCancelar)
   MenúLínea.addPopUp("&Registro", RegistroEmer)
   AyudaEmer.addText("&Uso de la Ayuda", MenuEnabled, UserMenu + 301)
   AyudaEmer.addText("&Indice de Ayuda", MenuEnabled, UserMenu + 302)
   AyudaEmer.addText("&Sistema de Ayuda", MenuEnabled, UserMenu + 303)
   MenúLínea.addPopup("&Ayuda",AyudaEmer)
endProc
```

PEDIDOS.PedidoPg2 Ventana Var

Estas variables son accesibles por todos los objetos de esta página de la ficha, pero no para los objetos de la primera página.

```
Var
   MenúLínea            Menu
   vacaciónEmer,
   editarEmer,
   registroEmer,
   AyudaEmer            PopUpMenu
    NúmeroSemanas Number
endVar
```

PEDIDOS.PedidoPg2 Ventana Const

Estas constantes son accesibles para todos los objetos de esta página de la ficha, pero no para los objetos de la primera página.

```
Const
   ReservaVacaciones = 101
   EditarCortar = 201
   EditarCopiar = 202
   EditarPegar = 203
   RegistroPrimero = 301
   RegistroUltimo = 302
   RegistroSiguiente = 303
   RegistroAnterior = 304
   RegistroInsertar = 305
   RegistroBorrar = 306
   RegistroCancelar = 307
   AyudaUsoAyuda = 401
   AyudaIndiceAyuda = 402
   AyudaSistemaAyuda = 403
endConst
```

PEDIDOS.PedidoPg2.BotónReservar

Métodos

action
pushButton

action

PEDIDOS.PedidoPg2.BotónReservar.action

Este código se ejecuta cuando se pulsa *Enter* para pulsar el botón. Por defecto, Paradox genera una acción FieldEnter y desplaza el foco al siguiente objeto en la ficha (en este caso, el siguiente objeto es BotónAyuda). En este caso, sin embargo, el comportamiento deseado es volver a la primera página. Esto se consigue manipulando la propiedad TabStop de BotónAyuda y llamando a **doDefault** para ejecutar el código por defecto de *action*.

```
method action(var eventInfo ActionEvent)
   if eventInfo.id() = FieldEnter then ; si el usuario pulsa Intro
      BotónAyuda.TabStop = False
      doDefault
      BotónAyuda.TabStop = True
   endIf
endMethod
```

pushButton

PEDIDOS.PedidoPg2.BotónReservar.pushButton

Sitúa al usuario en la página principal de PEDIDOS empleando **PedidoPg.moveTo.** Antes del desplazamiento, este código, se asegura de que el registro está completo y actualiza los valores del campo.

```
method pushButton(var eventInfo Event)
   doDefault
   if CajaSubTotal.Forma_de_Envío.isBlank() then
      msgInfo("Pedido Incompleto", "Introduzca un transportista.")
      CajaSubTotal.Forma_de_Envío.moveTo()
   else
      pedidopg.moveTo()
   endIf
endMethod
```

pushButton

PEDIDOS.PedidoPg2.BotónNotas.pushButton

Abre una ventana de edición permitiendo al usuario introducir y editar notas en el campo memo Notas de BUCEOART.DB para el registro actual de PEDIDOS.

```
method pushButton(var eventInfo Event)
  var
    TextoNotat String
    artículosTC TCursor
  endVar
  doDefault
  ArtículosTc.open("BuceoArt.db")
  ArtículosTc.edit()
  ArtículosTc.locate("N° Pedido", N°_Pedido.value, "N° Artículo",
BUCEOART.RegistroArtículo.N°_Artículo.value)
  TextoNota = ArtículosTc."Nota"
  TextoNota.view("Notas Línea Pedido")
  ArtículosTc."Nota" = TextoNota.subStr(1, 255)
  ArtículosTc.endEdit()
endMethod
```

pushButton

PEDIDOS.PedidoPg2.BotónAyuda.pushButton

Muestra la pantalla del archivo de Ayuda THAMUSER.HLP empleando **helpShowContext** con un identificador de contexto especificado por la constante DETALLEAYUDA. Compárese con **PEDIDOS.PedidoPg.BotónAyuda.pushButton, que** muestra la pantalla de THAMUSER.HLP.

```
method pushButton(var eventInfo Event)
  doDefault
  helpShowContext(FICHEROAYUDA, DETALLEAYUDA)
endMethod
```

PEDIDOS.PedidoPg2.BUCEOART

Métodos

arrive
action
CogerDescripción

ventana de Variables

ventana de Variables

arrive

PEDIDOS.PedidoPg2.BUCEOART.arrive

Cuando el foco se sitúa en BUCEOART, este código se ejecuta y asocia un TCursor al registro actual.

```
method arrive(var eventInfo MoveEvent)
   MarcoTabla.attach(BUCEOART)
endmethod
```

action

PEDIDOS.PedidoPg2.BUCEOART.action

Este código especifica respuestas personalizadas a las siguientes acciones: arrive, delete, lock, unlock, post, e insert.

```
method action(var eventInfo ActionEvent)
  Var
    NuevoValor Number
  endVar
  switch
    case eventInfo.id() = DataDeleteRecord :
      if not N°_Artículo.isBlank() then
        dmGet("buceosto.db", "Disponible", NuevoValor)
         dmPut("buceosto.db", "Disponible", NuevoValor + Cantidad)
      endIf
      RegistrosBorrados = "Si"
      doDefault
      subTotal.action(dataRecalc)


    case eventInfo.id() = DataLockRecord :

      dmGet("buceosto.db", "Disponible", NuevoValor)
       dmPut("buceosto.db", "Disponible", NuevoValor + Cantidad)
      doDefault

    case eventInfo.id() = DataUnlockRecord :

      dmGet("buceosto.db", "Disponible", NuevoValor)
       dmPut("buceosto.db", "Disponible", NuevoValor - Cantidad)
      action(DataPostRecord)
      doDefault

    case eventInfo.id() = DataPostRecord :
      Precio.action(dataRecalc)
      dmPut("buceoart.db", "Precio", Precio.value)
      dmPut("buceoart.db", "Total", Total.value)
      subTotal.action(dataRecalc)

    case eventInfo.id() = DataInsertRecord :
      doDefault
      RegistroArtículo.N°_Artículo.moveTo()
  endSwitch
endMethod
```

CogerDescripción

PEDIDOS.PedidoPg2.**BUCEOART.**CogerDescripción

Abre la ficha LISTA (en el caso de no estar ya abierta) para mostrar una lista de descripciones de los elementos disponibles de BUCEOSTO.DB empleando el procedimiento personalizado **LISTA.MuestraMe.** si éste fuese seleccionado de la lista, el registro correspondiente se recupera de BUCEOSTO.DB.

```
method CogerDescripción()
  var
    DescripciónArtículo AnyType
  endVar
  try
    FichaLista.MuestraMe("Existencias", "[buceosto.descripción]") ; método
personalizado asociado a la ficha lista.fsl
  onFail
    FichaLista.open("lista.fsl")
    retry
  endTry
  DescripciónArtículo = FichaLista.wait()
   if FichaLista.isAssigned() and FichaLista.Focus then
    FichaLista.hide()
  endIf
  BringToTop()
  If DescripciónArtículo <> "" THEN ; si selecciona un artículo
    ; recoge el número de artículo
    ExistenciasTC.open("buceosto.db")
    IF ExistenciasTC.locate("Descripción", DescripciónArtículo) THEN
      BUCEOART.RegistroArtículo.N°_Artículo.value = ExistenciasTC."N°
Artículo"
       ; rellena automáticamente la descripción
      ; ejecuta N°_Artículo.changeValue(), después
      ; Precio.recalcular(), después Total.recalcular(),
      ; después Subtotal.recalcular() y Total_Factura.recalcular()
    ENDIF
    buceoart.RegistroArtículo.Cantidad.moveTo()
  endif
endMethod
```

PEDIDOS.PedidoPg2.BUCEOART Ventana Var

Estas variables son accesibles por todos los objetos contenidos en BUCEOART.

```
Var
   ExistenciasTC TCursor
endVar
```

PEDIDOS.PedidoPg2.Cantidad

Métodos

arrive
changeValue

arrive

PEDIDOS.PedidoPg2.Cantidad.arrive

Si el objeto campo *NúmeroArtículo* del registro actual no contiene valor alguno, este código sitúa el foco en el.

```
method arrive(var eventInfo MoveEvent)
   ; Este método se asegura de que se ha introducido un número de artículo
   ; antes que la cantidad.
   If Nº_Artículo.isBlank() then
      Nº_Artículo.moveTo()
   endIf
endMethod
```

changeValue

PEDIDOS.PedidoPg2.Cantidad.changeValue

Compara la cantidad en stock con la cantidad solicitada. Si se dispone de la cantidad suficiente en el stock, este código calcula el precio y envía una acción Recalcular al objeto campo *TotalLíneaArtículo*. Después, los niveles en stock se actualizan y comprueban sobre el campo Renovar Pedido de BUCEOSTO.DB. Si los niveles de stock son bajos se mostrará el mensaje de renovar.

```
method changeValue(var eventInfo ValueEvent)
{
  ; Este método valida el campo cantidad
  Var
    existenciasTC TCursor
    NuevaCantidad Number
  endVar
  ; Primero recoge el nuevo valor introducido por el usuario
  NuevaCantidad = Number(eventInfo.newValue())
  ; Nos aseguramos de que N°_artículo contiene un número válido, si no es
así
  ; mostramos un error y ponemos el foco en el campo N°_Artículo, evitando
la partida
  ; y desactivamos el código estandar. (Mientras que este método sólo
comprueba un valor
  ; en blanco, el método changeValue asociado a  N°_Artículo comprueba los
valores nuevos.

  if N°_Artículo.isBlank() then
    beep()
    msgInfo("Número de Artículo Vacío", "Debe introducir un número de
artículo " +
            "antes de introducir la cantidad.")
    N°_Artículo.moveTo()
    eventInfo.setErrorCode(CannotDepart)
    disableDefault
  else
    ; El número es válido, pero debemos asegurarnos de que el usuario
    ; no ha introducido una cantidad negativa, decimal o cero.  Si lo ha
hecho
    ; mostramos un error y evitamos la partida.
    if (NuevaCantidad <= 0) or (NuevaCantiad <> NuevaCantidad.truncate(0))
then
         beep()
         msgInfo("Cantidad Incorrecta", "La cantidad debe contener un número
entero " +
                 "mayor que uno (1).  Introduzca un valor correcto " +
                  "o elimine este registro, (pulsando " +
                 "Ctrl+Supr).")
         eventInfo.setErrorCode(CanNotDepart)
    else
      ; Comprobamos si la cantidad es mayor que las existencias
      ; disponibles. Si es así, mostramos un mensaje y evitamos la partida.
```

```
       existenciasTC.attach(Descripción) ; lo asociamos al campo de la tala
buceosto
       existenciasTC.locate("N°_Artículo", N°_Artículo.value)
       if NuevaCantidad > existenciasTC."Disponible" then
         beep()
         msgInfo("Ha pedido demasiadas unidades", "Sólo hay  " +
               String(ExistenciasTC."Disponible") + " unidades disponibles."
+
                  "  Introduzca una cantidad menor.")
         eventInfo.setErrorCode(CanNotDepart)
       else
         ; El valor introducido es menor que las unidades disponibles, pero
         ; comprobamos si las unidades disponiblesquedan por debajo
         ; de la cantidad del campo renovar pedido, Si es así, mostramos un
aviso

         if (existenciasTC."Disponible" - Cantidad.Value) <=
existenciasTC."Renovar Pedido" then
          msgInfo("Es hora de Pedir", "Sólo hay " +
                  String(existenciasTC."Disponible" - Cantidad.Value) +
                           " unidades del artículo " +
String(N°_Artículo.value) +
                           " en existencias.")
         endIf
         ; aceptamos el valor nuevo y actualizamos el total de línea
         doDefault
         BUCEOART.RegistroArtículo.Total.recalcular()
       endIf
     endIf
  endIf
}
endMethod
```

ChangeValue

PEDIDOS.PedidoPg2.Alquiler_Venta.changeValue

Calcula el precio correspondiente a una venta o alquiler, después envía las acciones DataRecalc a los campos calculados correspondientes.

```
method newValue(var eventInfo Event)

   if eventInfo.reason() = editValue then
      if Rental_Sale = "Sale" then
          Price = Sale_Price
       else
          Price = Sale_Price * rentalRate * numWeeks
       endIf
      Line_Total = Price * Qty
      cPrice.action(DataRecalc)
      cLine_Total.action(DataRecalc)
   endIf

endmethod
```

keyPhysical

PEDIDOS.PedidoPg2.BotónVenta.keyPhysical

Este código se ejecuta cuando se pulsan las teclas del cursor, derecha, arriba, y anula la respuesta por defecto.

```
method keyPhysical(var eventInfo KeyEvent)
   If eventInfo.vChar() = "VK_RIGHT" then
      doDefault
      action(FieldForward)
   endIf
endMethod
```

keyPhysical

PEDIDOS.PedidoPg2.BotónAlquiler.keyPhysical

Este código se ejecuta cuando se pulsan las teclas del cursor, izquierda, arriba, y anula la respuesta por defecto.

```
method keyPhysical(var eventInfo KeyEvent)
   disableDefault
  Switch
      Case eventInfo.vChar() = "VK_LEFT" :
       action(FieldBackward)
     Case eventInfo.vChar() = "VK_UP"   :
       action(FieldUp)
      Case eventInfo.vChar() = "VK_DOWN" :
       action(FieldDown)
     Otherwise : doDefault
  endSwitch
endMethod
```

keyPhysical

PEDIDOS.PedidoPg2.Nº_Artículo.keyPhysical

Este código se ejecuta cuando se pulsan *Ctrl* + *Espacio*. Esto llama al método personalizado **CogerDescripción** para mostrar un cuadro de dialogo que se puede emplear para consultar.

```
method keyPhysical(var eventInfo KeyEvent)
   if eventInfo.isControlKeyDown() and eventInfo.vCharCode() = VK_SPACE then
     disableDefault
     cogerDescripción() ; método personalizado asociado al Marco de Tabla
BUCEOART
   endIf
endMethod
```

calcField

PEDIDOS.PedidoPg2.TotalEquipo.calcField

El código siguiente no está asociado a ningun método estándar. *TotalEquipo* es un campo calculado, y este código calcula su valor llamando a **cSum** para introducir la suma de los valores en el campo TotalLíneaArtículo.

```
iif(BUCEOART.nRecords > 0, tc.cSum("TotalLíneaArtículo"), 0)
```

newValue

PEDIDOS.PedidoPg2.Forma_De_Envío.changeValue

Sitúa el foco en *BotónInformaciónVacaciones*.

```
method newValue(var eventInfo Event)
   Var
     envíosTC TCursor
   endVar
   if eventInfo.reason() = EditValue then
     doDefault
     envíosTC.open("envio.db")
     envíosTC.locate("Forma de Envío", STRING(Forma_de_Envío.value))
     dmPut("buceoped.db", "Coste", envíosTC."Coste")
     Coste1.action(dataReCalc)
     PedidoPg2.CajaSubTotal.Coste.recalcular()
   endIf
endMethod
```

campoCalculado

PEDIDOS.PedidoPg2.SubTotal2.calcField

El código siguiente no está asociado a ningun método estándar. *SubTotal2* es un campo calculado, y su código asociado calcula su valor. Éste muestra el valor del campo CosteVacaciones para el registro actual.

```
CosteVacaciones
```

campoCalculado

PEDIDOS.PedidoPg2.Total.calcField

El código siguiente no está asociado a ningun método estándar. *Total* es un campo calculado, y este código calcula su valor sumando los valores de otros tres campos.

```
CosteVacaciones + GastosDeEnvío + TotalEquipo
```

PEDIDOS.PedidoPg

Métodos

open
close
arrive
menuAction
TeclaRápida

Procedimientos

MenúPedidos

Variables, Constantes, y ventanas Uses

ventana de Variables
ventana de Constantes

open

PEDIDOS.PedidoPg.open

Llama a **MenúPedidos** para visualizar el menú de *PedidoPg*, pone la ficha en modo edición, y asocia la variable *NombreFicha* a esta ficha.

```
method open(var eventInfo Event)
   delayScreenUpdates(Yes)
   setMouseShape(MouseWait)
   EstablecerMenúPrincipalPedidos()
   action(DataBeginEdit)                                ; pone la ficha en modo
edición
   BUCEOCLI.action(DataInsertRecord)      ; inserta un registro blanco en la
tabla maestra
                                                                ; ejecuta
BUCEOCLI.action(), y retorna aquí
   setMouseShape(MouseArrow)
   BUCEOCLI."Nombre".moveTo()
   doDefault
    NombreFicha.attach()
   delayScreenUpdates(No)
endMethod
```

close

PEDIDOS.PedidoPg.close

Cierra cualquier ventana *LISTA* que estuviese abierta antes de cerrar la ficha PEDIDOS, cierra el sistema de ayuda y restaura la Barra Rápida.

```
method close(var eventInfo Event)
  try
    FichaLista.close()
  onFail
    errorClear() ; Ya está cerrada, correcto
    doDefault
  endTry
  helpQuit(FICHEROAYUDA)
  ; OcultarBarra es una variable de nivel de página. Indica si la ficha
oculta
  ; la barra o si ha sido ocultada por otra ficha.
  if OcultarBarra then
    showSpeedBar()
  endIf
endMethod
```

arrive

PEDIDOS.PedidoPg.arrive

Muestra el menú de *PedidoPg* llamando a **PrincipalMenú.show**.

```
 method arrive(var eventInfo MoveEvent)
   MarcoTabla.attach(BuceoArt)
    sumaArtículos = MarcoTabla.cSum("Total")
   MenúPrinci.show()
endMethod
```

menuAction

PEDIDOS.PedidoPg.menuAction

Llama a la acción que proceda basándose en la opción elegida del menú *PedidoPg*. Por ejemplo, seleccionando Primero se acciona una llamada a **active.action(DataBegin)** . Los atributos de visualización se ajustan mediante la comprobación de **eventInfo.id = MenuInit**. Cuando se hace clic sobre un elemento en la barra del menú, éste genera un suceso MenuInit justo antes de mostrar el menú desplegable. En ese momento, se pueden especificar los atributos de visualización de las opciones del menú.

```
method menuAction(var eventInfo MenuEvent)
  Var
    NúmeroMenú SmallInt
    NúmeroRegistros, i Number
    tc TCursor
    ValorRetornado Logical
  endVar
  NúmeroMenú = eventInfo.id()
  if eventInfo.id() = MenuInit then
    try
      ValorRetornado = active.Editing
    onFail
      return
    endTry
    if ValorRetornado = True then
      if active.SelectedText <> "" then
        setMenuChoiceAttributeByID(UserMenu + EditarCortar, MenuEnabled)
        setMenuChoiceAttributeByID(UserMenu + EditarCopiar, MenuEnabled)
      else
        setMenuChoiceAttributeByID(UserMenu + EditarCortar,
                                   MenuGrayed + MenuDisabled)
        setMenuChoiceAttributeByID(UserMenu + EditarCopiar,
                                  MenuGrayed + MenuDisabled)
      endIf
    endIf
  endIf
  switch
  ; Menú Pedidos
    case NúmeroMenú = UserMenu + PedidoBuscarCliente :
BUCEOPED."Nombre".BuscarCliente() ; método personalizado asociado a Nombre
    case NúmeroMenú = UserMenu + PedidoPedirEquipo   : PedidoPg2.moveTo()
    case NúmeroMenú = UserMenu + PedidoGuardar       : GuardarYSalir()   ;
procedimiento personalizado asociado a la página
    case NúmeroMenú = UserMenu + PedidoCancelar      :
  CancelarYSalir()  ; procedimiento personalizado asociado a la página
    case NúmeroMenú = UserMenu + PedidoImprimir      :
menuAction(MenuFilePrint) ; imprime una instantánea de la ficha (ambas
páginas)
  ; Menú Editar
    case NúmeroMenú = UserMenu + EditarCortar        :
active.menuAction(MenuEditCut)
    case NúmeroMenú = UserMenu + EditarCopiar        :
active.menuAction(MenuEditCopy)
```

```
     case NúmeroMenú = UserMenu + EditarPegar         :
active.menuAction(MenuEditPaste)
   ; Menú Registro
     case NúmeroMenú = UserMenu + RegistroPrimero     :
active.action(DataBegin)
     case NúmeroMenú = UserMenu + RegistroUltimo      :
active.action(DataEnd)
     case NúmeroMenú = UserMenu + RegistroSiguiente   :
active.action(DataNextRecord)
     case NúmeroMenú = UserMenu + RegistroAnterior    :
active.action(DataPriorRecord)
     case NúmeroMenú = UserMenu + RegistroInsertar    :
active.action(DataInsertRecord)
     case NúmeroMenú = UserMenu + RegistroBorrar      :
active.action(DataDeleteRecord)
     case NúmeroMenú = UserMenu + RegistroCancelar    :
active.action(DataCancelRecord)
   ; Menú Ayuda
     case NúmeroMenú = UserMenu + AyudaUsoAyuda       : helpOnHelp()
     case NúmeroMenú = UserMenu + AyudaSistemaAyuda   :
helpShowContext(FICHEROAYUDA, CABECERAAYUDA)
     case NúmeroMenú = UserMenu + AyudaIndiceAyuda    :
helpShowIndex(FICHEROAYUDA)
   endSwitch
endMethod
```

Vea también

MenúPedidos

TeclaRápida

PEDIDOS.PedidoPg.TeclaRápida

Este es un procedimiento auxiliar empleado por el método **keyPhysical** de la ficha. Ésta llama bien a **PedidoPg.TeclaRápida** o bien a **PedidoPg2.TeclaRápida** dependiendo de la página activa. Esta **TeclaRápida** captura pulsaciones de tecla y efectúa las pertinentes acciones **moveTo** y **pushButton**. Si las acciones se consuman, se devuelve True, en caso contrario se devuelve False. Las pulsaciones de tecla que no se capturan en esta sección del código devuelven False. Por ejemplo, Alt+N llama a **CLIENTES."Nombre".moveTo.**

```
method TeclaRápida(var eventInfo KeyEvent) Logical
  Var
    LaTecla String
  endVar
  if eventInfo.isAltKeyDown() then
    LaTecla = eventInfo.vChar()
    switch
      case LaTecla = "E" : return BUCEOCLI.Dirección.moveTo()
      case LaTecla = "C" : return BotónCancelar.pushButton()
      case LaTecla = "D" : return
BUCEOPED.DatosPedido.DatosViaje.Destino.moveTo()
      case LaTecla = "B" : return BUCEOCLI."Nombre".buscarCliente()
      case LaTecla = "U" : return BUCEOCLI.Ciudad.moveTo()

      case LaTecla = "T" : return BUCEOCLI.Teléfono.moveTo()
      case LaTecla = "F" : return
BUCEOPED.DatosPedido.Forma_de_Pago.moveTo()
      case LaTecla = "N" : return BUCEOCLI."Nombre".moveTo()
      case LaTecla = "S" : return
BUCEOPED.DatosPedido.DatosViaje.Fecha_De_Salida.moveTo()
      case LaTecla = "P" : return BotónPedidoEquipo.pushButton()

      case LaTecla = "G" : return BotónGuardar.pushButton()
      case LaTecla = "J" : return
BUCEOPED.DatosPedido.DatosViaje.N°_de_Personas.moveTo()
      case LaTecla = "L" : return
BUCEOPED.DatosPedido.DatosViaje.Fecha_De_Llegada.moveTo()

      case LaTecla = "I" : return BotónInformaciónVacaciones.pushButton()
      case LaTecla = "A" : return BUCEOCLI.País.moveTo()
      case LaTecla = "O" : return BUCEOCLI.Código_Postal.moveTo()
      case LaTecla = "V" : return BUCEOCLI.Provincia.moveTo()
      otherwise          : return False
    endSwitch
  endIf
endMethod
```

EstablecerMenúPrincipalPedidos

PEDIDOS.PedidoPg.EstablecerMenúPrincipalPedidos

Éste es llamado por **PedidoPg.open** para crear los menús desplegables que componen el menú de *PedidoPg*.

```
proc EstablecerMenúPrincipalPedidos()
   PedidoEmer.addText("&Buscar Cliente", MenuEnabled, UserMenu +
PedidoBuscarCliente)
   PedidoEmer.addText("&Pedir Equipo", MenuEnabled, UserMenu +
PedidoPedirEquipo)
   PedidoEmer.addSeparator()
   PedidoEmer.addText("&Guardar", MenuEnabled, UserMenu + PedidoGuardar)
   PedidoEmer.addText("&Cancelar", MenuEnabled, UserMenu + PedidoCancelar)
   PedidoEmer.addSeparator()
   PedidoEmer.addText("&Imprimir", MenuEnabled, UserMenu + PedidoImprimir)
   MenúPrinci.addPopup("&Pedidos", PedidoEmer)
   EditarEmer.addText("Cor&tar", MenuGrayed + MenuDisabled, UserMenu +
EditarCortar)
   EditarEmer.addText("&Copiar", MenuGrayed + MenuDisabled, UserMenu +
EditarCopiar)
   EditarEmer.addText("&Pegar", MenuEnabled, UserMenu + EditarPegar)
   MenúPrinci.addPopup("&Editar", EditarEmer)
   RegistroEmer.addText("&Primero\tCtrl + F11", MenuEnabled, UserMenu +
RegistroPrimero)
   RegistroEmer.addText("&Ultimo\tCtrl + F12", MenuEnabled, UserMenu +
RegistroUltimo)
   RegistroEmer.addText("&Siguiente\tF12", MenuEnabled,  UserMenu +
RegistroSiguiente)
   RegistroEmer.addText("&Anterior\tF11", MenuEnabled,  UserMenu +
RegistroAnterior)
   RegistroEmer.addSeparator()
   RegistroEmer.addText("&Insertar\tINS", MenuEnabled,  UserMenu +
RegistroInsertar)
   RegistroEmer.addText("&Borrar\tCtrl + DEL", MenuEnabled, UserMenu +
RegistroBorrar)
   RegistroEmer.addText("&Cancelar Cambios\tAlt + Bksp", MenuEnabled,
UserMenu + RegistroCancelar)
   MenúPrinci.addPopUp("&Registro", RegistroEmer)
   AyudaEmer.addText("&Uso de la Ayuda", MenuEnabled, UserMenu +
AyudaUsoAyuda)
   AyudaEmer.addText("&Indice de Ayuda", MenuEnabled, UserMenu +
AyudaIndiceAyuda)
   AyudaEmer.addText("Sistema de &Ayuda", MenuEnabled, UserMenu +
AyudaSistemaAyuda)
   MenúPrinci.addPopup("&Ayuda", AyudaEmer)
endProc
```

PEDIDOS.PedidoPg Ventana Var

Estas variables son accesibles por todos los objetos contenidos en esta página, pero no por los objetos contenidos en *PedidoPg2*.

```
Var
   pedidoEmer,
   EditarEmer,
   EquipoEmer,
   RegistroEmer,
   AyudaEmer                  PopUpMenu
   MenúPrinci           Menu
endVar
```

PEDIDOS.PedidoPg Ventana Const

Estas constantes son accesibles por todos los objetos contenidos en esta página, pero no por los objetos contenidos en *PedidoPg2*.

```
Const
   PedidoBuscarCLiente  = 101
   PedidoPedirEquipo          = 102
   PedidoGuardar              = 103
   PedidoCancelar             = 104
   PedidoImprimir             = 105
   EditarCortar               = 201
   EditarCopiar               = 202
   EditarPegar                      = 203
   RegistroPrimero            = 301
   RegistroUltimo             = 302
   RegistroSiguiente          = 303
   RegistroAnterior           = 304
   RegistroInsertar           = 305
   RegistroBorrar             = 306
   RegistroCancelar           = 307
   AyudaUsoAyuda              = 401
   AyudaIndiceAyuda           = 402
   AyudaSistemaAyuda    = 403
endConst
```

pushButton

PEDIDOS.PedidoPg.BotónAyuda.pushButton

Muestra la pantalla del archivo THAMUSER.HLP empleando **helpShowContext** con el valor del identificador de contexto asignado a la constante CABECERAAYUDA (declarada en la ventana de constantes de ficha). Compárese con **PEDIDOS.PedidoPg2.BotónAyuda.pushButton,** que muestra la pantalla de THAMUSER.HLP.

```
method pushButton(var eventInfo Event)
  doDefault
  helpShowContext(FICHEROAYUDA, CABECERAAYUDA)
endMethod
```

action

PEDIDOS.PedidoPg.BuceoCli.action

BuceoCli es un objeto multi-registro que agrupa la cabecera de PEDIDOS (Nombre Cliente y Dirección, Nº de Cliente, etc.). El código asociado a su método estándar **action** especifica las respuestas a ciertas acciones (identicadas empleando **eventInfo.id**).

```
method action(var eventInfo ActionEvent)
   ; Este méto recoge las acciones sobre buceoclis y realiza gran variedad de
acciones
   ; dependiendo de la acción que lo haya provocado.
   Var
     ClientesTB Table
   endVar
   Switch
     case eventInfo.id() = DataUnlockRecord :
       action(dataPostRecord)
       doDefault
     case eventInfo.id() = DataPostRecord :
       if buceoart.nRecords > 0 and buceoart.touched then
         buceoart.action(dataPostRecord)
       endIf
       if buceoped.nRecords > 0 and buceoped.touched then
         buceoped.action(dataPostRecord)
       endIf
       doDefault
     case eventInfo.id() = DataArriveRecord :
       if buceoped.DatosPedido.DatosViaje.N°_Pedido.isAssigned() and
            not NúmeroSemanas.isAssigned() then
         NúmeroSemanas = NúmeroDeSemanas()
       endIf
       if buceoped.DatosPedido.DatosViaje.N°_Pedido.isAssigned() and
          not CosteDestino.isAssigned() then
         CosteDestino = CosteDelDestino()
       endIf
     ; Si se inserta un registro, calculamos el nuevo número de cliente
     ; y devolvemos el registro que contiene el número.
     case eventInfo.id() = DataInsertRecord :
       doDefault
       ClientesTB.attach("buceocli.db")
       ; Bloqueamos la tabla para evitar que se escriba mientras
       ; obtenemos el nuevo número de cliente.
       if lock(ClientesTB, "Write") then
         N°_Cliente.value = ClientesTB.cMax("N° Cliente") + 1
         unlock(ClientesTB, "Write")
       endIf
       NuevoCliente = N°_Cliente.value
       self.action(DataPostRecord)
       self."Nombre".moveTo()
     ; si hay un registro borrado, comprobamos si contiene información útil
     ; Si no es así, lo borramos. Si tiene información útil information,
     ; mostramos un cuadro de diálogo y dejamos que el usuario selccione lo
que quiera.
     case eventInfo.id() = DataDeleteRecord:
```

```
        if BUCEOCLI."Nombre".isBlank() then
          RegistrosBorrados = "Si"
        else
          RegistrosBorrados = msgYesNoCancel("¿Borrar?",
                    "¿Borro la información de " +
                       BUCEOCLI."Nombre".value + "?")
        endIf
        delayScreenUpdates(Yes)
        switch
          case RegistrosBorrados = "Si":
           ; If there are orders attached to this customer, then
           ; delete them.
           while buceoped.nRecords > 0
                 buceoped.action(DataDeleteRecord)
                 if RegistrosBorrados <> "Si" then
                       quitLoop
                 endIf
           endWhile
           ; si el usuario cancela el borrado del pedido,
           ; no borramos al cliente, si no, lo borramos
           if RegistrosBorrados <> "Si" then
                 disableDefault
           else
                 doDefault
           endIf
          case RegistrosBorrados = "No":
          disableDefault
          case RegistrosBorrados = "Cancelar":
           disableDefault
           eventInfo.setErrorCode(cannotDepart)
        endSwitch
        delayScreenUpdates(No)
    endSwitch
endMethod
```

PEDIDOS.PedidoPg.Nombre

Métodos

keyPhysical
BuscarCliente

keyPhysical

PEDIDOS.PedidoPg.Nombre.keyPhysical

Captura la pulsación de las teclas *Ctrl+Espacio* para llamar al método personalizado **BuscarCliente**.

```
method keyPhysical(var eventInfo KeyEvent)
  if eventInfo.isControlKeyDown() and eventInfo.vCharCode() = VK_SPACE then
    disableDefault
    BuscarCliente() ; método personalizado (no es un precedimiento porque
                ; también lo llaman objetos externos a la herencia de
contenedores)
            ; asociado al propio objeto
  endIf
endMethod
```

BuscarCliente

PEDIDOS.PedidoPg.Nombre.BuscarCliente

Es un método personalizado llamado por **NombreCliente.keyPhysical** cuando se detecta la pulsación de teclas *Ctrl+Espacio*. **BuscarCliente** comprueba primero el objeto campo *Nombre* del registro actual de CLIENTES.DB y borra el registro si el campo *Nombre* está vacío. Después mediante **listarFicha.MuestraMe** muestra una lista de consulta de clientes, permitiendo hacer una selección de clientes. El cliente seleccionado es localizado en CLIENTES.DB y se inserta un nuevo registro para PEDIDOS en CLIENTES.DB empleando para ello **Clientes.action(DataInsertRecord).**

```
method BuscarCliente() Logical
  Var
    NombreListaClientes AnyType
  endVar
  ; Borra el registro blanco que no se emplea,
  ; utiliza un registro de cliente existente en su lugar.
  if BUCEOCLI."Nombre".isBlank() then
    BUCEOCLI.action(DataDeleteRecord)
    ; ejecuta BUCEOCLI.action() para borrar el registro vacío
    ; y retorna aquí.
  endIf
  ; el valor RegistrosBorrados se asigna en BUCEOCLI.action() cuando
  ; se borra un registro vacío.
  if not RegistrosBorrados.isAssigned() then
    RegistrosBorrados = "No"
  endIf
  ; No busca si RegistrosBorrados = "Cancelar"
  if RegistrosBorrados = "Yes" or RegistrosBorrados = "No" then
    NombreListaClientes = HacerLista("Clientes", "[buceocli.Nombre]") ;
procedimiento personalizado asociado a la ficha
    if NombreListaClientes <> "" then
      ; recoge los datos del cliente de NombreListaClientes
      if BUCEOCLI.locate("Nombre", NombreListaClientes) then
        BUCEOART.action(DataInsertRecord)
        ; inserta un registro para el nuevo pedido
        return True
      else
        msgInfo("Buscar Cliente", "No pude encontrar " +
NombreListaClientes)
      endIf
    endIf
  return False
  endIf
endmethod
```

PEDIDOS.PedidoPg.BotónGuardar.pushButton

Este código se asegura de que el registro actual esté completo, después devuelve el control a la ficha VACACIÓN (o cierra la ficha *PEDIDOS*, si ésta estaba ejecutándose en solitario).

```
method pushButton(var eventInfo Event)
   Tasas.action(dataRecalc)
   doDefault
   GuardarYSalir() ; procedimiento personalizado asociado a la ficha
endMethod
```

PEDIDOS.PedidoPg.BotónInformaciónVacaciones

Métodos

arrive
depart
keyPhysical
pushButton

arrive

PEDIDOS.PedidoPg.BotónInformaciónVacaciones.arrive

Cambia el color de etiqueta del boton a blanco cuando el cursor se sitúa en *BotónInformaciónVacaciones*.

```
method arrive(var eventInfo MoveEvent)
   self.etiqueta.color = White
endMethod
```

depart

PEDIDOS.PedidoPg.BotónInformaciónVacaciones.depart

Restaura el color de etiqueta del botón a gris cuando cursor del ratón abandona *BotónInformaciónVacaciones*.

```
method depart(var eventInfo MoveEvent)
   self.etiqueta.color = Gray
endMethod
```

keyPhysical

PEDIDOS.PedidoPg.BotónInformaciónVacaciones.keyPhysical

Desactiva la respuesta por defecto de la tecla cursor abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   if eventInfo.vChar() = "VK_DOWN" then
      disableDefault
      Pedidos.Destino.moveTo()
   endIf

endmethod
```

pushButton

PEDIDOS.PedidoPg.BotónInformaciónVacaciones.pushButton

Sugiere que se seleccione la información de destino empleando **PEDIDOS.PedidoPg.Destino.moveTo.**

```
method pushButton(var eventInfo Event)
   doDefault
   Destino.moveTo()
endmethod
```

keyPhysical

PEDIDOS.PedidoPg.Provincia.ChangeValue

Ejecuta el recáculo de impuestos de la factura si encuentra un número de registros mayor que cero.

```
method changeValue(var eventInfo ValueEvent)
   doDefault
   delayScreenUpdates(Yes)
   If buceoped.nRecords > 0 then
      Tasas.recalcular()
      self.moveTo()
      action(fieldForward)
   endIf
   delayScreenUpdates(No)
endMethod
```

PEDIDOS.PedidoPg.**BUCEOPED**

Métodos

action

ventana de Variables

ventana de Variables

action

PEDIDOS.PedidoPg.**BUCEOPED.**action

El código asociado al método estándar **action** de *BUCEOPED* especifica las respuestas a ciertas acciones (identificadas empleando **eventInfo.id**). Por ejemplo, si la acción es DataDeleteRecord, un registro no vacio se borrará después de un cuadro de dialogo que pedirá confirmación. Los registros vacíos son borrados sin solicitud de confirmación y la bandera *RegistrosBorrados* se ajusta a Yes.

```
method action(var eventInfo ActionEvent)
  Switch
    case eventInfo.id() = DataInsertRecord:
      doDefault
      N°_Pedido.construir()
      NuevoPedido = N°_Pedido.value
      Fecha_de_Venta.construir()
      Destino.moveTo()
    case eventInfo.id() = DataDeleteRecord :
      if N°_Pedido.isBlank() or Destino.isBlank() then
        RegistrosBorrados = "Si"
        doDefault
      else
        RegistrosBorrados = msgYesNoCancel("¿Borrar?",
                  "¿Borro la información del pedido " +
                     N°_Pedido.value + "?")
        If RegistrosBorrados = "SI" then
         ; Primero borramos los equipos incluidos en el pedidos
           while buceoart.nRecords > 0
               buceoart.action(DataDeleteRecord)
         endwhile
         doDefault
        else
         disableDefault
        endIf
      endIf
    case eventInfo.id() = DataUnlockRecord :
      subTotal.action(dataRecalc)
      Coste1.action(dataRecalc)
      Precio_del_viaje.action(dataRecalc)
      Tasas.action(dataRecalc)
      action(DataPostRecord)
      doDefault
    case eventInfo.id() = DataPostRecord :
      if N°_Pedido.isAssigned() then
        dmPut("buceoped.db", "Coste", Coste1.value)
        dmPut("buceoped.db", "Subtotal", Total.value)
        dmPut("buceoped.db", "Tasas", Tasas.value)
        dmPut("buceoped.db", "Importe Total", Importe_Total.value)
        dmPut("buceoped.db", "Precio del Viaje", Precio_del_Viaje.value)
        doDefault
      else
        beep()
        msgInfo("Pedido incompleto", "Falta información en este " +
                "pedido. Rellene todos los campos para guardarlo o " +
```

```
                "eliminelo para cancelar el proceso.")
          eventInfo.setErrorCode(cannotDepart)
        endIf
      case eventInfo.id() = DataNextRecord:
        doDefault
        if N°_Pedido.isBlank() then
          N°_Pedido.construir()
          Fecha_de_Venta.construir()
          Destino.moveTo()
        endIf
      case eventInfo.id() = DataArriveRecord :
        CosteDestino = CosteDelDestino()
        Precio_del_Viaje.action(dataRecalc)
    endSwitch
endMethod
```

PEDIDOS.PedidoPg.BUCEOPED Ventana Var

Las variables declaradas en una ventana de Variables de página, son accesibles por todos los objetos contenidos en ésta. No hay objetos visibles en otras páginas.

```
Var
  NuevoCoste Number
endVar
```

PEDIDOS.PedidoPg.**BotónPedidoEquipo**

Métodos

arrive
depart
pushButton

arrive

PEDIDOS.PedidoPg.BotónPedidoEquipo.arrive

Cambia el color de etiqueta del botón a blanco cuando este objeto recibe el foco.

```
method arrive(var eventInfo MoveEvent)
   self.etiqueta.color = White
endMethod
```

depart

PEDIDOS.PedidoPg.BotónPedidoEquipo.depart

Restaura el color de etiqueta del botón a gris cuando este objeto pierde el foco.

```
method depart(var eventInfo MoveEvent)
   self.etiqueta.color = Gray
endMethod
```

pushButton

PEDIDOS.PedidoPg.BotónPedidoEquipo.pushButton

Sitúa al usuario en *PEDIDOS.PedidoPg2* donde se introducen los pedidos.

```
method pushButton(var eventInfo Event)
   doDefault
   pedidoPg2.moveTo()
endmethod
```

Construir

PEDIDOS.PedidoPg.Fecha_De_Venta.Construir

Ajusta el valor del objeto campo *FechaDeVenta* a la fecha actual devuelta por **today.**

```
method construir()
   ; rellena la fecha de venta con la fecha actual
   self.value = today()
endMethod
```

Construir

PEDIDOS.PedidoPg.Nº_Pedido.Construir

Incrementa el valor del objeto campo *Nº_Pedido* en 1 y envía el resultado a PEDIDO.DB. La tabla se bloquea durante el proceso de actualización.

```
method construir()
   var
     pedidoTB Table
   endVar
   pedidoTB.attach("buceoped.db")
   ; Bloque al tabla para evita que se escriba
   ; mientras recoge un nuevo número de pedido.
   if lock(pedidoTB, "Write") then
     self.value = pedidoTB.cMax("N° Pedido") + 1
     unlock(pedidoTB, "Write")
   endIf
   self.action(DataPostRecord)
endMethod
```

PEDIDOS.PedidoPg.FechaDeLlegada

Métodos

keyPhysical
changeValue

changeValue

PEDIDOS.PedidoPg.Fecha_De_Llegada.changeValue

Tiene en cuenta el nuevo valor. Comprueba que el nuevo valor de *FechaDeLlegada* es mayor que el valor de *FechaDeSalida* (a menos que ambos estén vacíos). Se muestra un cuadro de diálogo y la posibilidad de escape la bloquea **eventInfo.setErrorCode(CanNotDepart)** en caso de que *FechaDeLlegada* no fuese válido.

```
method changeValue(var eventInfo ValueEvent)
  var
    NuevaFecha Date
  endVar
  NuevaFecha = eventInfo.newValue()
  If not NuevaFecha.isBlank() and NOT Fecha_de_Salida.isBlank() and
        NuevaFecha < Fecha_de_Salida.value then
    msgInfo("Fecha de Llegada Incorrecta", "Introduzca una fecha de llegada
posterior " +
            "a la fecha de salida.")
    eventInfo.setErrorCode(canNotDepart)
  else
    if NuevaFecha <> self.value then
      doDefault
      Precio_del_Viaje.action(dataRecalc)
    endIf
  endIf
endMethod
```

changeValue

PEDIDOS.PedidoPg.FechaDeSalida.changeValue

Tiene en cuenta el nuevo valor. Comprueba que el nuevo valor del objeto campo *FechaDeSalida* es menor o igual que el valor de *FechaDeLlegada* (a menos que ambos estén vacíos). Se muestra un cuadro de dialogo y la posibilidad de escape (departure) la bloquea **eventInfo.setErrorCode(CanNotDepart)** en caso de que *FechaDeSalida* no fuese válido.

```
method changeValue(var eventInfo ValueEvent)
  var
    NuevaFecha  Date
  endVar
  NuevaFecha = eventInfo.newValue()
  IF NOT NuevaFecha.isBlank() and NOT Fecha_de_Llegada.isBlank() AND
        NuevaFecha > Fecha_de_Llegada then
    beep()
    msgInfo("Fecha de Salida Incrorrecta", "Introduzca una fecha de salida
anterior " +
            "a la fecha de llegada.")
    eventInfo.setErrorCode(canNotDepart)
    disableDefault
  else
    if NuevaFecha <> self.Value then
      doDefault
      Precio_del_viaje.action(dataRecalc)
    endIf
  endIf
endMethod
```

PEDIDOS.PedidoPg.Destino

Métodos

canDepart
changeValue
keyPhysical

Procedimientos

cogerListaDestinos

canDepart

PEDIDOS.PedidoPg.Destino.canDepart

Se asegura de que este campo contiene un destino válido después permite el desplazamiento a otro campo.

```
method canDepart(var eventInfo MoveEvent)
  var
    DestinoTC   TCursor
  endvar
   if not DestinoTC.open("destinos.db") then
    msgInfo("Error", "No pude abrir la tabla de destinos")
    Return
  endif
   if Self.recordStatus("Modified") then
    if DestinoTC.locate("Destino", self.value) then
      Precio_del_Viaje.action(dataRecalc)
    else
      if self.value <> "" then
        beep()
        message("Ese destino no está en la tabla.")
        eventInfo.setErrorCode(CanNotDepart)
      endIf
    endIf
  endIf
endMethod
```

changeValue

PEDIDOS.PedidoPg.Destino.changeValue

Este código se asegura de que el valor del objeto campo *Destino* es un destino válido (uno de los almacenados en DESTINOS.DB).

```
method changeValue(var eventInfo ValueEvent)
{
   var
     DestinoTC   TCursor
   endvar
   ;doDefault
   ;disableDefault
    if not DestinoTC.open("destinos.db") then
     msgInfo("Error", "No pude abrir la tabla de destinos")
     Return
   endif
   msginfo("Prueba",EditValue)
   if DestinoTC.locate("Destino",EditValue) then
     ;enableDefault
     doDefault
     Precio_de_Viaje.recalcular()
   else
     msgStop("¡Espere!", "Este no es uno de los destinos disponibles. Pulse
Ctrl-Espacio para seleccionar un destino de la lista.")
     disableDefault
     moveto(Destino)
   endif
}
endMethod
```

keyPhysical

PEDIDOS.PedidoPg.Destino.keyPhysical

Captura la combinación de teclas *Ctrl+Espacio* para llamar a **Destino.CogerListaDestinos.**

```
method keyPhysical(var eventInfo KeyEvent)
   if eventInfo.isControlKeyDown() and eventInfo.vCharCode() = VK_SPACE then
      disableDefault
      cogerListaDestinos() ; procedimiento personalizado asociado a este
objeto
   endIf
endMethod
```

CogerListaDestinos

PEDIDOS.PedidoPg.Destino.CogerListaDestinos

Busca DESTIPOS.DB (destinos posibles) para las selecciones previas hechas en LUGARES. Si no se encuentra ninguna, esos destinos se listan por Nombre Destino en la ficha LISTA por selección del usuario. Si DESTIPOS está vacío, cada destino de DESTINOS.DB se lista. El método llama a **MontarLista** y espera una selección que es asignada al valor *Destino*. Después se oculta la lista y se devuelve el control al objeto invocante.

```
proc cogerListaDestinos()
  var
    NombreDestino,
    ListaDatosFuente STRING
    PosiblesDestinosTB Table
    PosiblesDestinosTC TCursor
  endVar
  ; Si tenenmos registros en la tabla destipos.db, el usuario seleccionará
los destinos
  ; de dicha tabla. Si no los tenemos, empleará la tabla destinos.db
  ; Esto permite usar la ficha pedidos en solitario o en conjunción con las
otras fichas.
  PosiblesDestinosTB.attach("destipos.db")
  If PosiblesDestinosTB.isEmpty() THEN ; selecciona de destinos.db
    ListaDatosFuente = "[destinos.Destino]"
  else
    ListaDatosFuente = "[destipos.Destino]"
  endIf
  NombreDestino = HacerLista("Destinos", ListaDatosFuente) ; procedimiento
personalizado asociado a la ficha
  self.value = NombreDestino
endProc
```

changeValue

PEDIDOS.PedidoPg.Nº_de_Personas.changeValue

Tiene en cuenta el nuevo valor de este objeto campo y actualiza el Precio del viaje empleando **action(DataRecalc)**.

```
method changeValue(var eventInfo ValueEvent)
   doDefault
   precio_del_Viaje.action(dataRecalc)
endMethod
```

PEDIDOS.PedidoPg.FormaDePago

Métodos

open
canDepart
keyPhysical

Procedimientos

CogerListaFormaPago
CogeInformaciónTarjeta
CogeNúmeroTarjeta
CogeFechaExpedición

ventana de Variables

ventana de Variables

open

PEDIDOS.PedidoPg.FormaDePago.open

Cuando este objeto campo se abre, el código siguiente se ejecuta para inicializar una matriz de formas de pago.

```
method open(var eventInfo Event)
   ; Inicializa el array TiposTarjeta decalrado en la ventana de variables de
este objeto.
   ; Lo emplean algunos procedimientos personalizados asociados al mismo
objeto.
   TiposTarjeta[1] = "AmEx"
   TiposTarjeta[2] = "Diners Club"
   TiposTarjeta[3] = "Discover"
   TiposTarjeta[4] = "Master Card"
   TiposTarjeta[5] = "Visa"
endMethod
```

changeValue

PEDIDOS.PedidoPg.FormaDePago.changeValue

Este código se asegura de que el valor de los objetos campo es una forma de pago válida antes de permitir el desplazamiento a cualquier otro objeto.

```
mmethod changeValue(var eventInfo ValueEvent)
  var
    MétodosPago Array[3] String
    CómoPaga String
  endVar
  If Cancelar then
    disableDefault
  else
    MétodosPago[1] = "Caja"
    MétodosPago[2] = "Cheque"
    MétodosPago[3] = "Transferencia"
    CómoPaga = eventInfo.newValue()
    switch
      case TiposTarjeta.contains(CómoPaga) :
        cogeInformaciónTarjeta() ; procedimiento personalizado asociado a
este objeto
      case MétodosPago.contains(CómoPaga) :
        Cuenta_Corriente.blank()
        Fecha_Expedición.blank()
      otherwise :
        msgStop("Seleccione una forma de pago correcta", "No aceptamos " +
CómoPaga)
        eventInfo.setErrorCode(CanNotDepart)
    endSwitch
  endIf
endMethod
```

keyPhysical

PEDIDOS.PedidoPg.FormaDePago.keyPhysical

Este código captura la combinación de teclas *Ctrl+Espacio* para llamar a **FormaDePago.CogerListaFormaPago**. También desactiva la respuesta por defecto de la tecla cursor derecha.

```
method keyPhysical(var eventInfo KeyEvent)
   switch
     case eventInfo.isControlKeyDown() and
          eventInfo.vCharCode() = VK_SPACE :
       disableDefault
       CogeListaFormasDePago() ; procedimiento personalizado asociado a este
objeto
     case eventInfo.vCharCode() = VK_RIGHT :
       disableDefault
       buceoped.action(FieldForward)
   endSwitch
endMethod
```

CogerListaFormaPago

PEDIDOS.PedidoPg.FormaDePago.CogerListaFormaPago

Crea y muestra un menú emergente de formas de pago. Cuando se elige un elemento del menú emergente, éste se asigna al objeto campo *FormaDePago*.

```
proc cogeListaFormasDePago()
  var
    PrincipalPago,
    CuentaPago                          PopUpMenu
    SelecciónPago                 String
    ListaPagox,
    ListaPagoy                    LongInt
    posiciónAnterior,
    ListaPagoPosición Point
  endVar
  CuentaPago.addArray(TiposTarjeta)
  PrincipalPago.addStaticText("Formas de Pago")
  PrincipalPago.addSeparator()
  PrincipalPago.addText("Caja")
  PrincipalPago.addText("Cheque")
  PrincipalPago.addPopUp("Tarjeta de Crédito", CuentaPago)
  PrincipalPago.addText("Transferencia")
  ListaPagoPosición = self.position
  ListaPagox = ListaPagoPosición.x()
  ListaPagoy = ListaPagoPosición.y()
  posiciónAnterior = self.position
  BUCEOPED.convertPointWithRespectTo(PedidoPg, posiciónAnterior,
ListaPagoPosición)
  ListaPagox = ListaPagoPosición.x()
  ListaPagoy = ListaPagoPosición.y()
  SelecciónPago = PrincipalPago.show(ListaPagox, ListaPagoy)
  self.value = SelecciónPago
endproc
```

CogeInformaciónTarjeta

PEDIDOS.PedidoPg.FormaDePago.CogeInformaciónTarjeta

Este procedimiento personalizado llama a otro procedimiento personalizado (**CogeNúmeroTarjeta**) para tomar el número de la tarjeta de crédito. Si el número introducido es válido, la llamada a **CogeFechaExpedición** tomará la fecha de caducidad.

```
proc cogeInformaciónTarjeta()
  if cogeNúmeroTarjeta() then     ; procedimiento personalizado definido a
continuación
    cogeFechaExpedición()      ; procedimiento personalizado definido a
continuación
  else
    return
  endIf
endProc
```

CogeNúmeroTarjeta

PEDIDOS.PedidoPg.FormaDePago.CogeNúmeroTarjeta

A Este procedimiento le llama otro procedimiento personalizado, **CogeInformaciónTarjeta**. Éste emplea un cuadro de diálogo **view** para mostrar un número de tarjeta de crédito.

```
proc cogeNúmeroTarjeta() Logical
   var
     NúmeroTarjeta String
   endvar

   NúmeroTarjeta = "Introduzca aquí el número:"
   NúmeroTarjeta.view("Introduzca el número:")
   if NúmeroTarjeta <> "" and NúmeroTarjeta <> "Introduzca aquí el número:"
then
     Cuenta_Corriente.value = NúmeroTarjeta.subStr(1, 25)
     return True
   else
     Cuenta_Corriente.blank()
     return False
   endIf
endProc
```

CogeFechaExpedición

PEDIDOS.PedidoPg.FormaDePago.CogeFechaExpedición

Este procedimiento personalizado es llamado por el procedimiento personalizado **CogeInformaciónTarjeta**. Éste emplea un cuadro de diálogo **view** para mostrar una fecha de caducidad. El código siguiente se asegura de que se ha introducido una fecha válida y que la tarjeta de crédito no ha expirado.

```
proc cogeFechaExpedición()
   var
      FechaTarjeta AnyType
   endVar
   FechaTarjeta = "01/01/99"
   FechaTarjeta.view("Introduzca la fecha de expedición:")
   try
      Date(FechaTarjeta)
      Fecha_Expedición.value = FechaTarjeta
   onFail
      msgStop("Introduzca una fecha válida", "Fecha incorrecta: " +
String(FechaTarjeta))
      cogeFechaExpedición()
   endTry
endProc
```

PEDIDOS.PedidoPg.FormaDePago Ventana Var

Las variables declaradas aquí sólo son accesibles por el campo objeto *FormaDePago*.

```
Var
   TiposTarjeta Array[5] String
endVar
```

THAMLIB.LSL Library

Métodos

LimpiarValoresPágina
CrearMenú
PuedeCerrarFicha
CogerDatosFuente
CogerTítuloMejoras
PuedeCerrarseFicha
EstáMandoAbierto
ListaObjetosDePágina
MensajeTHAM
open
NombreDePágina
HaLlamadoARetornar
EstablecerDatosFuente
EsLlamadaVerdadero
EsLlamadaFalso
EstablecerTítuloMejoras
EstablecerValoresPágina
MandoEstáAbierto
NombreMando
MandoEstáCerrado

ventanas Variables y Constantes

ventana de Variables
ventana de Constantes

LimpiarValoresPágina

THAMLIB.LimpiarValoresPágina

Borra cualquier valor de consuta para la página (dado por el argumento *NombrePágina*) de la ficha dada por el argumento *NombreFicha*. Entonces el identificador de la consulta sitúa la entrada en VALCONSU.DB. Llamando antes a **EstablecerValoresPágina.**

Vea también

THAMLIB.EstablecerValoresPágina

```
method LimpiarValoresPágina(NombreFicha string, NombrePágina string)
  var
    miConsulta  Query
  endVar
  ; limpia los valores de esta página de ValConsu.db
  miConsulta=Query

    mapconsu.db | Nombre Ficha | Página        | N° Consulta |
             | ~NombreFicha | ~NombrePágina | _qID        |

    valconsu.db | N° Consulta | Valor Consulta  |
                | _qID        | changeto blank  |

  endQuery
  executeQBE(miConsulta)
endmethod
```

CrearMenú

THAMLIB.CrearMenú

Crea y visualiza un menú emergente sensible al objeto. Éste es llamado con los argumentos NombreFicha, NombrePágina, e Interfaz pasados por el método **NombreFicha.mouseRightUp**. El objeto diana se determina por el foco (por ejemplo, la posición del cursor del ratón) pero algunos objetos (como textos o mapas de bits) son excluidos por **mouseRightUp** para algunas fichas particulares. Un objeto menú se crea y se muestra únicamente si en el archivo AYUDA.DB existe una entrada para ese objeto diana. La opción estándar del menú es Código de Ayuda. Las demás opciones dependen del objeto diana. Por ejemplo, si el foco lo tiene ImagenPez, se añadira la opción Acerca de Pez. **CrearMenú** también procesa las selecciones del menú.

Vea también

NAUFRAGI.mouseRightUp,
VIDAMAR.mouseRightUp,
VACACIÓN.mouseRightUp,
PEDIDOS.mouseRightUp
LUGARES.mouseRightUp

```
method crearMenú(var NombreFicha Form, NombrePágina String, Interfaz
UIObject)

; El sistema de mensajes a nivel de ficha examina la pulsación de la tecla
derecha
; sobre los distintos objetos de la página. Proporciona la ayuda
correspondiente al
; objeto afectado mediante la localización de su nombre en el fichero
Ayuda.db
; ofreciendo un menú emergente con la opción estándar "Ayuda del Objeto"
; Si el usuario selecciona esta opción del menú se muestra un cuadro
informativo
; que muestra el texto contenido en el fichero Ayuda

; PASOS:
; Primero, determina si MenAyuda está abierto y, si es así,
; localiza el objeto dentro de la tabla. No todos los objetos tienen ayuda.
Si el
; objeto se encuentra dentro de MenAyuda, muestra el menú emergente
estándar.
; Después añade al menú emergente las entradas adicionales pertenecientes al
objeto y
; finalmente procesa la opción seleccionada por el usuario.

  var
    MenúEmergente PopUpMenu
  endvar
  ; la tabla de ayuda ya está abierta
  ; ahora buscamos NombreFicha.NombrePágina
  if ObjetoAyuda.Locate("Nombre Ficha", NombreFicha.Name, "Nombre Página",
NombrePágina, "Objeto", interfaz.name) then
    ; construimos el menú
    MenúEmergente.empty() ; Primero vaciamos los elementos antiguos
```

```
     MenúEmergente.addStaticText("Propiedades del objeto")
     MenúEmergente.addSeparator()
     MenúEmergente.addText("Ayuda del código")
     ; añadimos los elementos a "Acerca del Objeto"
     switch
       case interfaz.name = "BotónProcesar"    : MenúEmergente.addText("Ver
Consulta")
       case interfaz.name = "CuadroDeBandera"  :
MenúEmergente.addText("Acerca de THAM")
       case interfaz.name = "BotónVidaMarina"  :
MenúEmergente.addText("&Acerca de los Peces")
       case interfaz.name = "BotónNaufragio"   :
MenúEmergente.addText("&Seleccionar Naufragio")
     endswitch
     selección=MenúEmergente.Show()
     ; procesa la opción de menú seleccionada
     switch
       case selección = "Ayuda del código"          :
  helpShowContext(AyudaPrograma, ObjetoAyuda."Contexto")
       case selección = "Ver Consulta"                    :
  Informe.open("valconsu.rsl")

      Informe.setTitle("Ver Consulta")
       case selección = "Acerca de THAM"                  :
  esLlamadaVerdadero()

     if AcercaDe.open("SobTham.fsl") then

          ; abre un cuadro de diálogo modal

          AcercaDe.Wait()

          ; espera y lo cierra

          AcercaDe.Close()

          EsLlamadaFalso()

     endif
       case selección = "&Acerca de los Peces"      :
  NombreFicha.PáginaVidaMar.notas.moveTo()
       case selección = "&Seleccionar Naufragio" :  NombreFicha.naufragio =
self."Nombre del Barco".value

       NombreFicha.naufragio.moveto()
     endswitch
  endif
  ; esto es todo
endmethod
```

PuedeCerrarFicha

THAMLIB.PuedeCerrarFicha

Ajusta una bandera para indicar que se puede cerrar una ficha.

```
method PuedeCerrarFicha()
    PuedeCerrar = True
endmethod
```

CogerDatosFuente

THAMLIB.CogerDatosFuente

Devuelve el elemento almacenado en el índice ListadoFuente del DynArray *TítuloYFuente* declarado en THAMLIB.LSL.

Vea también

THAMLIB.EstablecerDatosFuente

```
method CogerDatosFuente() string
   return TítuloyFuente["ListadoFuente"]
   ; devuelve el valor almacenado en el Dynarray
endmethod
```

CogerTítuloMejoras

THAMLIB.CogerTítuloMejoras

Devuelve el elemento almacenado en el índice TítuloMejora del DynArray *TitleAndSource* declarado en THAMLIB.LSL.

Vea también

THAMLIB.EstablecerTítuloMejoras

```
method CogerTítuloMejoras() string
   return TítuloyFuente["TítuloMejora"]
endmethod
```

PuedeCerrarseFicha

THAMLIB.PuedeCerrarseFicha

Comprueba el valor de la variable *PuedeCerrar* para informar cuando se puede cerrar una ficha.

```
method PuedeCerrarseFicha() Logical
   if isAssigned(PuedeCerrar) then
      return PuedeCerrar
   else
      return False
   endif
endmethod
```

EstáMandoAbierto

THAMLIB.EstáMandoAbierto

Informa si una Barra Rápida personalizada está abierta.

```
method EstáMandoAbierto() Logical
   ; ¿está abierto el mando empleado para desplazarse por los registros?
   if isAssigned(MandoAbierto) then
      ; el mando ha sido abierto
      return MandoAbierto
   ; el mando no ha sido abierto
   else
      return False
   endif
endmethod
```

ListaObjetosDePágina

THAMLIB.ListaObjetosDePágina

Toma los objetos valores de la página actual y los escribe en una tabla.

```
method ListaObjetosDePágina(NombreFicha String, NombrePágina String) Logical
   ; este método ejecuta una consulta (ConsultaObjetosPágina) para encontrar
   ; todos los objetos de la ficha y página activa, enciándo los resultados a
listObje.db
   var
     ConsultaObjetosPágina Query
   endVar
   ConsultaObjetosPágina = Query

        Mapconsu.db | Nombre Ficha | Página         | Vía de Acceso |
                    | ~NombreFicha | ~NombrePágina | Check         |

        endQuery

   executeQBE(ConsultaObjetosPágina, "listaObj.db")
   RETURN TRUE
endmethod
```

MensajeTHAM

THAMLIB.MensajeTHAM

Abre AYUDA.DB e intenta localizar una entrada que se corresponda con los argumentos *NombreFicha*, *NombrePágina*, and *NombreObjeto*. Si se encuentra, el valor de *Mensaje* se muestra como un mensaje.

```
method MensajeTHAM(NombreFicha String, NombrePágina String, NombreObjeto
String)
   ; busca NombreFicha.NombrePágina.NombreObjeto
   If MiMensaje.locate("Nombre Ficha",NombreFicha,"Nombre
Página",NombrePágina,"Objeto",NombreObjeto) then
      message(MiMensaje."Mensaje".value)
   endif
endmethod
```

open

THAMLIB.open

Abre AYUDA.DB, ésta es una tabla que almacena mensajes de ayuda.

```
method open(var eventInfo Event)
   if not ObjetoAyuda.Open("Ayuda.db") then
     ; fallo al abrir la tabla de ayuda -- informamos al usuario
     msgStop("Problema", "No pude abrir Ayuda.db")
   endif
   if not MiMensaje.open("Ayuda.db") then
     msgStop("¡Dios Mío!", "No pude abrir Ayuda.db")
   endif
endmethod
```

NombreDePágina

THAMLIB.NombreDePágina

Devuelve el nombre de la página que contiene el interfazUsuario especificado. Una cadena vacía es devuelta si no se encuentra la página.

```
method NombreDePágina(interfazUsuario UIObject) String
  ; este método devuelve el nombre de la página que contiene el objeto
activo
  Var
    esLapágina string
  endVar
  while interfazUsuario.Class <> "Page" and interfazUsuario.ContainerName <>
""
    interfazUsuario.attach(interfazUsuario.ContainerName)
  endWhile
  if interfazUsuario.class = "Page" then
    esLaPágina = interfazUsuario.name
   else
    esLaPágina = ""
  endif
  return esLaPágina
endmethod
```

HaLlamadoARetornar

THAMLIB.HaLlamadoARetornar

Devuelve el valor True o False que esté asignado a la variable *esLlamada* de THAMLIB. En caso de no tener ningún valor asignado devuelve False. *esLlamada* se comprueba mediante los métodos **open** de LUGARES, NAUFRAGI, VIDAMAR, y SOBTHAM para asegurarse de que esas fichas sólo pueden ser abiertas por VACACIÓN.

```
method HaLlamadoaRetornar() Logical
   if not esLlamada.isAssigned() then
        esLlamada = False
   endif
   return esLlamada
endmethod
```

EstablecerDatosFuente

THAMLIB.EstablecerDatosFuente

Asigna la cadena argumento *NuevoValor* al índice ListadoFuente de *TítuloYFuente*, un DynArray declarado en THAMLIB.LSL

Vea también

THAMLIB.CogerDatosFuente

```
method EstablecerDatosFuente(const NuevoValor String)
   TítuloyFuente["ListadoFuente"] = NuevoValor
endmethod
```

EsLlamadaVerdadero

THAMLIB.EsLlamadaVerdadero

Ajusta la variable *esLlamada* de THAMLIB a True. *esLlamada* se emplea para evitar que ciertas fichas sean abiertas directamente en lugar de por VACACIÓN.

Vea también

THAMLIB.HaLlamadoARetornar

```
method EsLlamadaVerdadero()
   EsLlamada = True
endmethod
```

EsLlamadaFalso

THAMLIB.EsLlamadaFalso

Ajusta la variable *esLlamada* de THAMLIB a falso. *esLlamada* se emplea para evitar que ciertas fichas sean abiertas directamente en lugar de por VACACIÓN.

Vea también

THAMLIB.HaLlamadoARetornar

```
method EsLlamadaFalso()
   EsLlamada = False
endmethod
```

EstablecerTítuloMejoras

THAMLIB.EstablecerTítuloMejoras

Asigna el argumento cádena *NuevoValor* al índice TítuloMejora en el DynArray *TítuloYFuente* (declarado en THAMLIB.LSL).

Vea también

THAMLIB.CogerTítuloMejoras

```
method EstablecerTítuloMejoras(const NuevoValor String)
   TítuloyFuente["TítuloMejora"] = NuevoValor ; asigna un valor al elemento
del DynArray
endmethod
```

EstablecerValoresPágina

THAMLIB.EstablecerValoresPágina

Ejecuta una consulta para encontrar todos los objetos selecionados en los argumentos entregados *NombreFicha* y *NombrePágina*. Llamados por los métodos **BotónAceptar** de *PáginaNaufragios*, *PáginaDestinos*, y *PáginaVidaMar* para reunir, en VALCONSU.DB, los criterios seleccionados para los destinos posibles. La consulta resultante se envia al TCursor *NuevosValores*. VALCONSU.DB es utilizada por **VACACIÓN.DestinoConsulta** para encontrar los destinos correspondientes en DESTINOS.DB.

Vea también

THAMLIB.LimpiarValoresPágina,
VACACIÓN.DestinoConsulta

```
method EstablecerValoresPágina(NombreFicha String, NombrePágina String)
Logical

  ; este método ejecuta una consulta (ConsultaCogerValores) para encontrar
los objetos
  ; de la ficha y página activa, enviando el resultado a un tipo TCursor
llamado NuevosValores

  var
    NuevosValores, ValorCursor TCursor
    ConsultaCogerValores Query
    ObjetoCampo UIObject
  endVar

  LimpiarValoresPágina(NombreFicha, NombrePágina) ; limpia los valores de
valconsu.db

  ConsultaCogerValores = Query

      Mapconsu.db | Nombre Ficha  | Página        | N° Consulta | Vía de
Acceso |
                  |  ~NombreFicha | ~NombrePágina | _qID        | Check
|

      Valconsu.db | N° Consulta | Nombre Campo |
                    | _qID        | Check        |

  endQuery

  executeQBE(ConsultaCogerValores, NuevosValores)

  ValorCursor.open("Valconsu.db")
  ValorCursor.edit()
  scan NuevosValores :
    ValorCursor.locate("Nombre Campo", NuevosValores."Nombre Campo")
    ; msgInfo("busco en NombreCampo de valconsu ..: ",NuevosValores."Nombre
Campo"+NuevosValores."Vía de Aceeso")
    ObjetoCampo.attach(NuevosValores."Vía de acceso")
    if not isBlank(ObjetoCampo.value) then
```

```
        if isBlank(ValorCursor."Valor Consulta") then
          ValorCursor."Valor Consulta" = ObjetoCampo.value
        else
          ValorCursor."Valor Consulta" = ValorCursor."Valor Consulta" + " OR
" + ObjetoCampo.value
        endIf
      endIf
   endScan
   ValorCursor.endEdit()
   RETURN TRUE
endmethod
```

MandoEstáAbierto

THAMLIB.MandoEstáAbierto

Ajusta una bandera para indicar que una Barra Rápida personalizada está abierta.

```
method MandoEstáAbierto()
   MandoAbierto = True
endmethod
```

NombreMando

THAMLIB.NombreMando

Devueve el texto de la barra de título de una Barra Rápida personalizada.

```
method NombreMando() String
    ; devuelve el título del mando
    return "Pulse un botón para cambiar de registro"
endmethod
```

MandoEstáCerrado

THAMLIB.MandoEstáCerrado

Ajusta la bandera para indicar que una Barra Rápida personalizada está cerrada.

```
method MandoEstáCerrado()
   ; cerrar el mando
   MandoAbierto = False
endMethod
```

THAMLIB Ventana Var

Estas variables son accesibles por todos los métodos y procedimientos de esta biblioteca.

```
Var

   PuedeCerrar,
   EsLlamada,
   MandoAbierto         Logical
   TítuloyFuente        DynArray[] String ; 2 campos: TítuloMejora y
ListadoFuente
   MiMensaje,
   ObjetoAyuda          Tcursor
   Informe              Report
   AcercaDe             Form
endVar
```

THAMLIB Ventana Const

Estas constantes son accesibles para todos los métodos y procedimientos de esta biblioteca.

```
Const
  AyudaPrograma = ":trabajo:Ayudprog.hlp"
endConst
```

changeValue

PEDIDOS.PedidoPg2.Forma_De_Envío.changeValue

Sitúa el foco en *BotónInfomaciónVacaciones*.

```
method changeValue(var eventInfo ValueEvent)
   doDefault
   BotónInformaciónVacaciones.moveTo()
endmethod
```

keyPhysical

PEDIDOS.PedidoPg.Provincia.keyPhysical

Redirecciona la respuesta por defecto del objeto campo para las teclas flecha arriba y flecha abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   disableDefault
   switch

      case eventInfo.vChar() = "VK_DOWN" :
         Provincia.moveTo()

      case eventInfo.vChar() = "VK_UP" :
         Street.moveTo()

      otherwise: doDefault

   endSwitch

endmethod
```

keyPhysical

PEDIDOS.PedidoPg.País.keyPhysical

Redirecciona la respuesta por defecto del objeto campo pra la tecla flecha abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   if eventInfo.vChar() = "VK_DOWN" then
      disableDefault
      País.moveTo()
   endIf

endmethod
```

keyPhysical

PEDIDOS.PedidoPg.Código_Postal.keyPhysical

Redirecciona la respuesta por defecto del objeto campo pra la tecla flecha abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   if eventInfo.vChar() = "VK_DOWN" then
      disableDefault
      Código_Postal.moveTo()
   endIf

endmethod
```

canDepart

PEDIDOS.PedidoPg.Registro53.canDepart

Este método se asegura de que el registro está completo.

```
method canDepart(var eventInfo MoveEvent)

  var
    eReason smallInt
  endVar

  eReason = eventInfo.reason()

  If eReason = userMove or eReason = shutDownMove or
     eReason = palMove then

    if not N°_de_Pedido.isBlank() then
      If not orderOK() then
        beep()
        msgInfo("Pedido incompleto", "Rellene todos" +
                "los campos de la ficha de pedidos " +
                "(para poder almacenarlo) o borre" +
                "(y cancele) el pedido.")
        buceoped.Destino.moveTo()
        eventInfo.setErrorCode(cannotDepart)
      endIf
    endIf
  endIf

endmethod
```

keyPhysical

PEDIDOS.PedidoPg.BotónPedidoEqupo.keyPhysical

Redirecciona la respuesta por defecto del objeto campo pra la tecla flecha abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   if eventInfo.vChar() = "VK_DOWN" then
      disableDefault
      beep()
   endIf

endmethod
```

keyPhysical

PEDIDOS.PedidoPg.FechaDeLlegada.keyPhysical

Redirecciona la respuesta por defecto del objeto campo pra la tecla flecha abajo.

```
method keyPhysical(var eventInfo KeyEvent)

   if eventInfo.vChar() = "VK_DOWN" then
      disableDefault
      action(fieldForward)
   endIf

endmethod
```

canDepart

PEDIDOS.PedidoPg.FormaDePago.canDepart

Este método se asegura que el valor introducido en el campo es válido antes de poder desplazarse a otro objeto.

```
method canDepart(var eventInfo MoveEvent)

  If N°_de_Pedido.isBlank() then
    doDefault
  Else


    Switch

      Case Forma_de_Pago = "AmEx" or
           Forma_de_Pago = "Diners Club" or
           Forma_de_Pago = "Visa" or
           Forma_de_Pago = "Master Card" or
           Forma_de_Pago = "4B" :

        if ccNumber.isBlank() or self.Touched then
         if not getCardNum() then
               beep()
               msgInfo("Número de Tarjeta Incorrecto",
      "Introduzca " +
                       "el número de trajetas de crétido si
        desea emplear" +
                       la tarjeta + ".")
               eventInfo.setErrorCode(cannotDepart)
               disableDefault
         endIf
        endIf

        if Fecha_Expedición.isBlank() or self.Touched then
         if not getExpDate() then
               beep()
               msgInfo("Fecha de expedición incorrecta",
               "Introduzca " +
                       "la fecha de expedición de la tarjeta
               de crédito si desea emplearla" +
                            Payment_Method + ".")
               eventInfo.setErrorCode(cannotDepart)
               disableDefault
         endIf
        endIf

      Case Forma_de_pago = "Caja" or
           Forma_de_pago = "Talón" or
           Forma_de_pago = "Transferencia" :

        if not ccNumber.isBlank() then
         ccNumber.blank()
        endIf
```

```
         if not ccExpDate.isBlank() then
            ccExpDate.blank()
         endIf

      Otherwise:

         beep()
         msgInfo("Forma de pago incorrecta", "Seleccione una " +
                 "de las formas de pago que aparecen al
         pulsar" +
                 "Ctrl-Espacio o seleccione Registro|
          Cancelar cambios para" +
                 "establecer la forma de pago previa.")
         eventInfo.setErrorCode(cannotDepart)
         disableDefault

    endSwitch
  endIf
endmethod
```

depart

PEDIDOS.PedidoPg.Grupo91.depart

Este método comprueba los valores de campo y llama a **action(DataRecalc)** para recalcular el campo *cTotal_Factura*.

```
method depart(var eventInfo MoveEvent)

   If not Destino.isBlank() and
      not N°_de_Personas.isBlank() and
      not Fecha_de_Salida.isBlank() and
      not Fecha_de_Llegada.isBlank() then

     númeroSemanas = númerodesemanas()
     costedestino = costedeldestino()
     Precio_de_Viaje = costedestino * númerosemanas *
                              N°_de_Personas
     cImporte_Total.action(dataRecalc)

   endIf

endmethod
```