

Uso del control SGrid 2.0 en VFP

1. INTRODUCCIÓN

El Control ActiveX SGrid 2.0 creado con Visual Basic 6.0 (VB) y desarrollado por vbAccelerator (vbaccelerator.com); es un control tipo ListView con muchísimas prestaciones además de su excelente interfaz para el usuario.

2. REQUISITOS

Para utilizar el control SGrid 2.0 es necesario contar con el Automation Object SSubTmr y opcionalmente con el control ImageList (ya se de vbAccelerator o el de Microsoft) para más detalle vea la Tabla 1.

Tabla 1 – Controles necesarios

Control	Descripción	Comentarios
SGrid 2.0	ActiveX tipo ListView, pero con muchas más prestaciones y excelente interfaz	Requerido
SSubTmr	Automation Object para subclasificación	Requerido
ImageList	ActiveX similar al ImageList de Microsoft	Opcional pero recomendado

Todo lo mencionado en la Tabla 1 lo pueden encontrar en <http://vbaccelerator.com>

3. REGISTRANDO COMPONENTES

Como todos los archivos son controles ActiveX y/o Automation Object previamente a ser usados deberán ser registrados en el registro (válgame la redundancia) de Windows; esto se logra utilizando el programa **regsvr32** el cual se puede encontrar en el directorio de Windows y/o en el System32.

Los controles se pueden grabar en cualquier directorio, sólo recuerden que si hacen algún cambio tendrán que actualizar el registro de Windows.

Para registrar los controles lo más sencillo es desde la ventana de comandos y ejecutar las siguientes sentencias:

```
C:> regsvr32 vbalSGrid6.ocx
C:> regsvr32 vbalIm16.ocx
C:> regsvr32 SSubTmr6.dll
```

4. UN VISTAZO A LAS PROPIEDADES

Aunque no encontré documentación específica para cada uno de los métodos, eventos y/o propiedades, enlistaré algunas de las que creo más importantes.

4.1 Propiedad Redraw

Esta propiedad es similar a BeginUpdate y EndUpdate del control ListView, evita el repintado del control lo que ayuda a que la adición de columnas y líneas sea mucho más rápido.

Ejemplo de uso:

```
WITH vbaGrid
    .Redraw = .F. && Evita el repintado

    .AddColumn(...)
    .CellDetails(...)

    .Redraw = .T. && Repinta el control
ENDWITH
```

4.2 Propiedad ImageList

Me imagino que la mayoría ya sabrá qué es esta propiedad, pero me encontré con un pequeño detalle, al querer asignar el control ImageList (de vbaccelerator) a la propiedad ImageList del SGrid, este último como que no lo reconocía (o aceptaba), pero en fin, la solución para esto fue la siguiente:

```
ThisForm.vbaGrid.ImageList = ThisForm.vbaImageList.Object
```

Pues sí, agregando la propiedad `Object` del control ImageList; aunque en la realidad el control ImageList no contiene esa propiedad pero eso es algo de VFP que no tengo ni la menor idea del por qué.

4.3 Método AddColumn

Aquí ya empezamos con lo práctico, como les dije anteriormente, no encontré documentación específica para cada uno de los métodos que contiene el control así que, trataré de explicar para que sirve cada uno de los parámetros:

Para el caso de este método, todos los parámetros son opcionales.

Tabla 4.3.1 – Parámetros de AddColumn

Parámetro	Tipo	Descripción
vKey	String	Similar al cKey en un nodo del control TreeView
sHeader	String	Texto de la columna
eAlign	Integer	Alineación del texto en la columna, los valores que puede tomar se encuentran en el archivo sgrid.rtf en la enumeración <code>ECGTextAlignFlags</code>
ilconIndex	Integer	Índice del icono en un control ImageList
lColumnWidth	Integer	Ancho de la columna
Visible	Boolean	Indica si la columna es visible o no

Parámetro	Tipo	Descripción
bFixedWidth	Boolean	Indica si se puede o no cambiar el tamaño de la columna
vKeyBefore	String	Agrega una columna después de la columna con el valor del vKey especificado
bIncludeInSelect	Boolean	Indica si la columna será incluida cuando se seleccione una línea
sFmtString	String	Para formatear el texto, el formato deberá de estar en concordancia con vb
RowTextColumn	Boolean	Indica si la columna ocupará todas las columnas previamente agregadas, esto sirve para hacer el efecto del Outlook en que se muestra una pequeña parte del contenido de email en el ListView
eSortType	Integer	Tipo de dato utilizado para ordenar, estos valores están definidos en la enumeración ECGSortTypeConstants
bOwnerDraw	Boolean	Es por si quieren dibujar la columna ustedes mismos

Ejemplo de uso: (parte del código del ejemplo de Outlook de la demostración del uso del control SGrid y "convertido" a código de VFP)

WITH ThisForm.vbaSGrid

```
.AddColumn("urgency", , , 9, 28, , , .F., , CCLSortIcon)
.AddColumn("type", , , 10, 28, , , .F., , CCLSortIcon)
.AddColumn("attach", , , 12, 28, , , .F., , CCLSortIcon)
.AddColumn("flag", , , 11, 28, , , .F., , CCLSortIcon)
.AddColumn("from", "From", , , 96)
.AddColumn("subject", "Subject", , , 256)

.AddColumn("received", "Received", , , 96, , , , "dd/mm/yy
hh:mm", , CCLSortDate)

.AddColumn("to", "To", , , 96)
```

```

.AddColumn("size", "Size", , , 56, , , , , "#,##0", ,
CCLSortNumeric)

*!*' Add two invisible columns to cache status information:

*!* Agregamos dos columnas invisibles para "guardar" información
.AddColumn("read", , , , , .F.)
.AddColumn("ID", , , , , .F.)

*!*' The special "rowcolumnntext" column must be added to the end
*!*' of the available columns. This never appears as a column
*!*' header, but the text in it is drawn underneath the row
(assuming
*!*' the row is high enough for it, starting at the column
*!*' specified by .RowTextStartColumn:

*!* La columna especial "rowcolumnntext" debe ser añadida al
*!* último. Esta no aparecerá como una columna "normal" con
*!* cabecera, pero es texto es dibujado por debajo de la línea
.AddColumn("body", , , , 96 + 256 + 96 + 96, , , , , .T.)
ENDWITH

```

4.4 Propiedad CellDetails

Es importante señalar que para agregar líneas (o Ítems en el caso del control ListView) se utiliza el método CellDetails, aquí el único requisito es que exista la columna en donde poner la celda, el número de línea no tiene por que ser continuo, en otras palabras, pueden colocar texto en la primera línea y después en la 100, sin tener que poner nada entre ellas.

Parámetro	Tipo	Descripción
< Row>	Integer	Número de línea
< Col>	Integer	Número de columna
[sText]	String	Texto de la celda
[eTextAlign]	Integer	Alineación del texto, igual que el parámetro eAlign del método AddColumn
[LIconIndex]	Integer	Índice del icono en un control ImageList
[oBackColor]	Integer	Color de fondo
[oForeColor]	Integer	Color del texto (por decirlo así)

Parámetro	Tipo	Descripción
[oFont]	StdFont	Objeto que Font, que determina las propiedades del tipo de texto a utilizar al dibujar la celda
[lIdent]	Integer	Espacio de izquierda a derecha a dejar en blanco
[lExtralconIndex]	Integer	Indice de un icono extra en la celda
[lItemData]	Integer	Algo así como la propiedad Tag de los controles de VFP

Ejemplo de uso:

```

If (iRow < 16) Then
    .CellDetails(iRow, 6, "Texto", , , , lColour, sFntUnread)
Else
    .CellDetails( iRow, 6, "Texto", , , , lColour)
EndIf

```

5. COMO OBTENER EL TEXTO DE UNA CELDA

Pues es sencillo, la propiedad Cell del control, devuelve un objeto cGridCell, dentro del cual una de sus propiedades es Text, que contiene el valor del texto contenido en la celda.

Ejemplo de uso: en el evento DbClick del control SGrid pongan el siguiente código.

```

LPARAMETERS lrow, lcol
LOCAL loCell
loCell = This.cell(lrow, lcol)

ThisForm.List1.AddItem(loCell.Text)

```

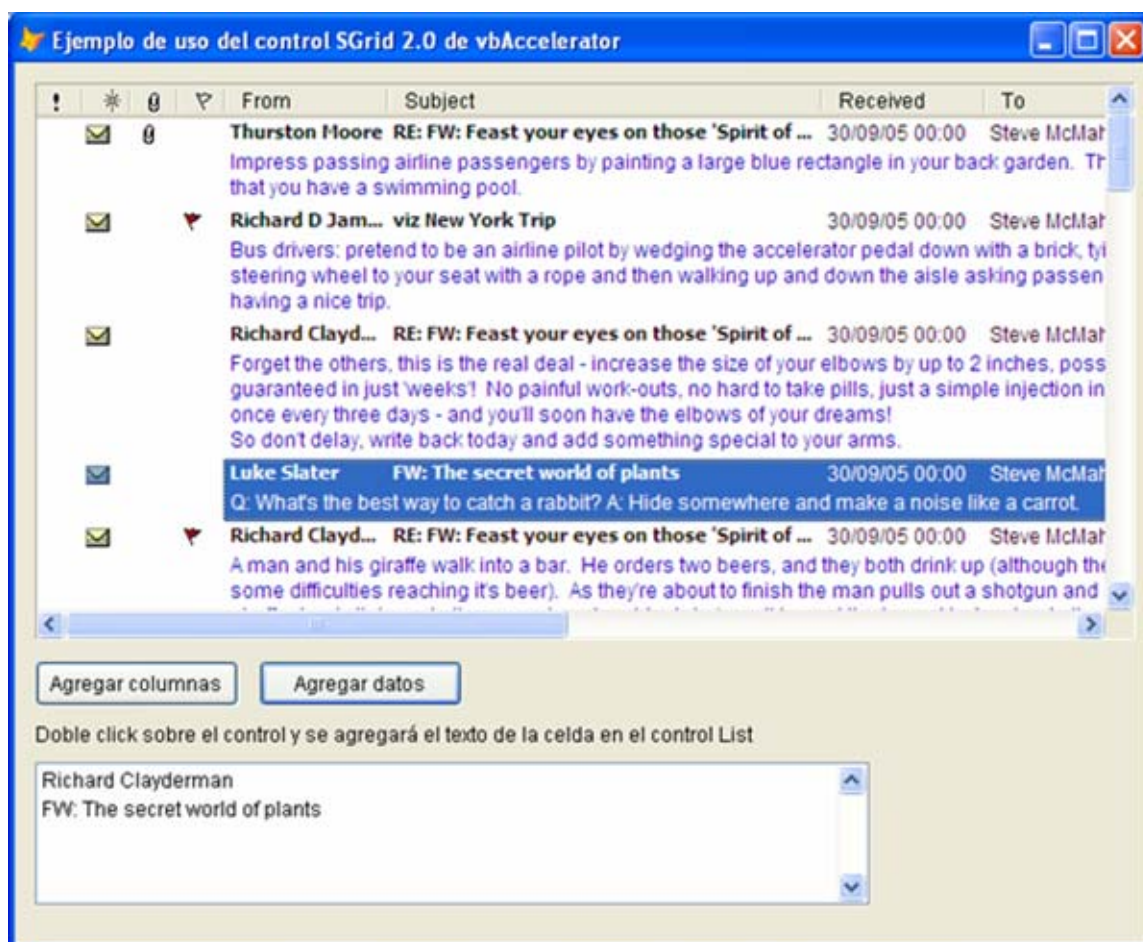
Esto agregará el texto de la celda a un control ListBox.

6. EJEMPLO EN VFP

El artículo es acompañado por un ejemplo sencillo de cómo agregar columnas y datos al control SGrid 2.0 utilizando VFP.

Para que logren ver el formulario como el de la figura de abajo, ejecuten el formulario, presionen el botón "Agregar columnas" y posteriormente el botón "Agregar datos".

Hagan doble clic sobre el control y el texto de la celda se agregará al control List.



Espero que les sea de utilidad, saludos

Denny Infante

denny_infante@hotmail.com