March, 1995

xSprite Control

Overview

In this context, a sprite is an image object with a transparent background. The sprite may a single image or several related images called frames. The xSprite control is used with the xWinG control as an easy way to manipulate sprites. The control itself is invisible at runtime and must be placed on an xWinG control in order to function. The sprite image is contained in one of the xWinG buffers. When the sprite is made visible, the control will first copy the xWinG bitmap rectangle where the sprite is to be located into an internal buffer. The sprite frame image will then be pasted onto the WinG bitmap. When the sprite is moved or made invisible the original image rectangle will be restored from the internal buffer.

The Sprite Image

The sprite image may contain several frames which are arranged in a column and row format. The image must have a width that is an even multiple of the frame width. For example, if a sprite is 100 pixels wide the sprite image may be 400 pixels wide but not 402 pixels wide. In this case there are four frames per image row. If the sprite is 80 pixels high and the image is 185 pixels high, then there are two rows of sprite frames for a total of 8 frames. There is also a 400 x 15 pixel area at the bottom of the sprite image that is not used by the sprite except that the lower left pixel of the sprite image defines the transparent sprite color index. The xWinG control may use any and all of the sprite image for normal xWinG operations.

Properties

Buffer

Integer.

This is the buffer number that corresponds to the xWinG control property of the same name which contains the sprite image. This image may be shared by more than one sprite.

Draw

Boolean.

If this property is true then the sprite rectangle will immediately be drawn whenever it is changed. If the property is false then the sprite rectangle will be added to the Windows GDI dirty rectangle and updated win the normal course of Windows painting messages. In order to force an immediate update the programmer can use the **xSprite.Refresh** method.

Frame

Integer. Must be greater than 0 and less than or equal to the number of frames in the sprite image. Read only at run time.

If a sprite image is 400 x 250 pixels and the sprite is 100 x 80 pixels, then a frame value of 10 refers to the image rectangle located at 100 x 160. That is, the second column in the third row.

OnBitmap

Boolean. This is the value of the control.

This property indicates whether the sprite has been placed on the bitmap. Setting this property to true will cause the sprite frame to be placed on the bitmap. Setting the property to false will replace the sprite with the contents of the internal buffer. Setting the property to true if it is already true or if at least one pixel of the sprite is not on the WinG image will generate an error.

PixWide PixHigh

Integers. Read only at run time. Must be 4 or greater.

These define the size of the sprite frame. The sprite image must be a multiple of **PixWide** wide. The sprite image must be at least **PixHigh** high. The internal backup buffer requires (**PixWide** + 6) times **PixHigh** bytes of memory.

X Y

Integers. May be negative. Read only at run time.

This is the location of the sprite on the xWinG bitmap. These properties determine the upper left corner of the sprite. A negative value results in the sprite being partly off the bitmap. For example, a 100 pixel wide sprite with an X value of -40 will only have the rightmost 60 pixels showing on the bitmap.

Methods

Move

The move method is used to display the sprite on the bitmap. The method may be used with one, two, or three arguments. If one argument is used then that is the sprite frame. If two arguments are used then they are the X and Y position of the sprite. If three arguments are used then they are the X and Y position and the sprite frame. e.g.

```
xSprite1.Move Frame
xSprite1.Move X, Y
xSprite1.Move X, Y, Frame
```

If the sprite is already on the bitmap then this method will replace the bitmap pixels and place the sprite in the new location. If the old location and the new location overlap, then the two rectangles will be combined when the screen is repainted. The repainting style is still determined by the **Draw** property.

Refresh

When the sprite **Draw** property is false, the sprite rectangle is not immediately redrawn when the sprite is moved. Instead the dirty rectangle or rectangles are passed to the Windows GDI to be placed in the drawing queue. Using the **xSprite**.**Refresh** method will force Windows to update all dirty rectangles on the sprite's parent **xWinG** window. This will just update the dirty rectangles whereas the **xWinG.Refresh** method will redraw the entire control window. The redraw time difference may be considerable especially if there are other controls on the **xWinG** control that are not in the dirty area.