

## Images On Statictext

- [Alyce](#)

[Images On Statictext](#) | [Why Use Statictext?](#) | [Stylebits](#) | [Setting the Image](#) | [Changing the Image](#) | [StaticText with Bitmap](#) | [Demo of BMP on Statictext](#) | [StaticText with Icon](#) | [Demo of Icon on Statictext](#)

---

## Why Use Statictext?

Why use a statictext control to display images?

When you use a statictext to display images, you can locate them anywhere on a window, with no need for graphicboxes! You can display as many images as you want, and they can be scattered amongst other controls. They don't even need to be flushed to make them stick.

You can use or icons. If you choose to display an icon on a statictext, any transparent areas remain transparent.



Bitmap on statictext.



Icon on statictext.



Icon on statictext, BackgroundColor\$ = "blue"

## Stylebits

Using STYLEBITS to alter the style of a statictext control.

When we create the control with the STATICTEXT command, we needn't be concerned with the caption, since it won't show. We can also put in any values for width and height, because they are ignored when the style is changed to display an image. The control is automatically sized to fit the image. The X and Y location are important for placement, though.

```
statictext #1.s, "",10,10,2,2
```

We need a single line of code to change a statictext control so that it will display a bitmap. We add the `_SS_BITMAP` style to the control like this:

```
stylebits #1.s, _SS_BITMAP, 0, 0, 0
```

We can also display an ICON on the statictext control. To do this, we add the \_SS\_ICON style to the control like this:

```
stylebits #1.s, _SS_ICON, 0, 0, 0
```

## Setting the Image

We must use an API call to cause an image to be displayed on a statictext control. The function is SendMessageA. It requires the handle of the statictext, which we retrieve with the HWND() function.

```
hStatic = hwnd(#1.s)
```

The message to send is \_STM\_SETIMAGE. We need to tell it what type of image is being used. We use either \_IMAGE\_BITMAP or \_IMAGE\_ICON. We also need to tell it the handle of the image. For a bitmap, we use the Liberty BASIC command LOADBMP, then get the handle with HBMP(). Using an icon is a bit more complicated. See the section later in this article for details.

```
loadbmp "image",file$  
hImage=hbmp("image")
```

```
calldll #user32, "SendMessageA",_  
    hStatic as ulong, _      'handle of statictext  
    _STM_SETIMAGE as long, _ 'message to set image  
    _IMAGE_BITMAP as long, _ 'type of image  
    hImage as ulong, _       'handle of bitmap  
    re as long
```

API calls must be placed on one line. If the line continuation character is used, the API call can be written on multiple lines, allowing each argument to be documented. To create shorter code, if you are comfortable with using API calls, write it like this:

```
calldll #user32,"SendMessageA",hStatic as ulong,  
_STM_SETIMAGE as long,_IMAGE_BITMAP as long,hImage as ulong,re as long
```

## Changing the Image

You can use the method outlined above to change the image on the statictext as many times as you'd like during a program's execution. It will always cause the statictext to be resized to match the size of the image.

## StaticText with Bitmap

The following demo allows the user to select a bitmap file on disk. This demonstrates that the control is automatically sized to fit the image. In a program, you would hard-code a path to the bitmap on disk. If the bitmap is in the same folder as the program, no path information is needed.

```
loadbmp "image", "filename.bmp"
or
loadbmp "image", "bmps\filename.bmp"
```

## Demo of BMP on Statictext

Here is the demo program to place a bitmap on a statictext control:

```
nomainwin

'can use hard-coded path such as:
'loadbmp "image", "mybmp.bmp"

filedialog "Open Bmp","*.bmp",file$
if file$="" then end

stylebits #1.s, _SS_BITMAP, 0, 0, 0
statictext #1.s, "",10,10,2,2
open "StaticText Image" for WINDOW as #1
#1 "trapclose [quit]"

loadbmp "image",file$
hImage=hbmp("image")

hStatic=hwnd(#1.s)

calldll #user32, "SendMessageA",_
    hStatic as ulong,_      'handle of statictext
    _STM_SETIMAGE as long,_ 'message to set image
    _IMAGE_BITMAP as long,_ 'type of image
    hImage as ulong,_      'handle of bitmap
```

```
re as long
wait

[quit]
  unloadbmp "image"
  close #1:end
```

## StaticText with Icon

There is only one difference in the above technique when an icon is used instead of a bitmap. It is in the way the image is loaded. There is no native Liberty BASIC command to load an icon. To use the ExtractIconA API call, we need to know the instance handle of the window. We get this easily with GetWindowLongA.

```
CallDLL #user32, "GetWindowLongA",_
hWindow As long,_           'window handle
_GWL_HINSTANCE As long,_    'type
hInstance As long           'returns instance handle
```

The function also requires the disk filename of the icon. If it is in the program's directory, there is no need for path information in the filename. The function returns the handle of the icon, which is then used to set the image for the statictext control.

```
CallDLL #shell32, "ExtractIconA",_
hInstance As long,_ 'instance handle of window
file$ As ptr,_      'filename of ico,dll,exe
index As long,_     '0-based icon index
hIcon As long       'returns handle of icon
```

When the program ends, we need to remove the icon from memory with a call to DestroyIcon.

```
calldll #user32, "DestroyIcon",_
hIcon as long,_ 'handle of icon
result as long
```

It is also possible to load an icon using LoadImageA. It will allow you to load an icon from disk. The ExtractIconA function has more flexibility, since it allows you to extract the icon from a file on disk, from an executable, from a DLL, or from an icon library (\*.icl). This allows you to place all of your icons in a single icon library file, or to extract an icon from the program executable to display on the window.

## Demo of Icon on Statictext

The icon API functions in the demo have been wrapped in Liberty BASIC functions to make their use very easy. Here is the demo to place an icon on a statictext control:

```
nomainwin
BackgroundColor$="blue"

'can use hard-coded path such as:
'icofile$="myicon.ico"

filedialog "Open Icon","*.ico;*.exe",icofile$
if icofile$="" then end

stylebits #1.s, _SS_ICON, 0, 0, 0
statictext #1.s, "",10,10,2,2
open "StaticText Image" for WINDOW as #1
#1 "trapclose [quit]"

hImage=ExtractIcon(hwnd(#1),icofile$,0)

hStatic=hwnd(#1.s)

calldll #user32, "SendMessageA",_
    hStatic as long,_      'handle of statictext
    _STM_SETIMAGE as long,_ 'message to set image
    _IMAGE_ICON as long,_  'type of image
    hImage as long,_       'handle of icon
    re as long
wait

[quit]
    re=DestroyIcon(hImage)
    close #1:end

Function ExtractIcon(hW, file$, index)
    hInst=GetWindowLong(hW, _GWL_HINSTANCE)
    CallDLL #shell32, "ExtractIconA",_
    hInst As long,_ 'instance handle of window
    file$ As ptr,_  'filename of ico,dll,exe
```

```
index As long,_ '0-based icon index
ExtractIcon As long
End Function
```

```
Function DestroyIcon(hIcon)
  calldll #user32, "DestroyIcon",_
  hIcon as long,_ 'handle of icon
  DestroyIcon as long
End Function
```

```
Function GetWindowLong(hW, type)
  CallDLL #user32, "GetWindowLongA",_
  hW As long,_ 'window handle
  type As long,_ 'flag for type
  GetWindowLong As long
End Function
```

---

[Images On Statictext](#) | [Why Use Statictext?](#) | [Stylebits](#) | [Setting the Image](#) | [Changing the Image](#) | [StaticText with Bitmap](#) | [Demo of BMP on Statictext](#) | [StaticText with Icon](#) | [Demo of Icon on Statictext](#)