C# How-to 12: Add a toolbox bitmap for a component or control

Introduction

A previous tutorial on this site showed how to build a simple .NET component in C#. However, although the component displayed .NET"s default bitmap in the VS.NET toolbox, the tutorial did not show how to make your component display a bitmap of your choice. I wanted to do this to make the KeyState control developed in the tutorial more professional, and here is how it is done.

Steps to take

1. Create a component

First, create a new component (in VS.NET, choose New Project ->Visual C# Projects -> Class Library) and name the project 'TestComp'. Add a reference to the System.Windows.Forms DLL. Then modify the code to look like this:

```
using System;
using System.ComponentModel;
namespace Microbion.TestComp {

public class MyComp : System.ComponentModel.Component {
public MyComp() {
// constructor logic here
}

// a method of the class
public void DoSomething() {
System.Windows.Forms.MessageBox.Show("This is a message!");
}
}
}
```

As you can see, this component does nothing except display a message when its DoSomething() method is called.

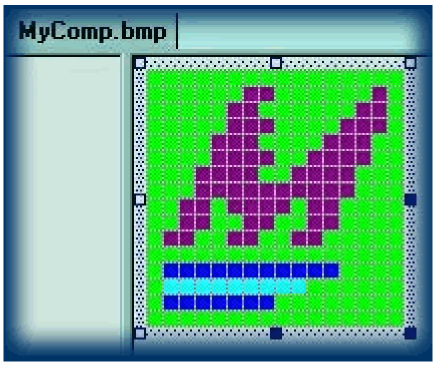
2. Create a bitmap

If you build this component, it will compile without problems. If you then go to the Customize Toolbox... dialog box, and navigate to the TestComp.dll, then selecting this file will show that it has the default icon (a cog wheel). You need to create a better bitmap for this component.

To do this, go into Solution Explorer, right-click the project name (TestComp) and select 'Add new item...' from the menu. In the dialog box, expand 'Local project items', then click 'Resources'. Click on 'Bitmap file' and change the file name to 'MyComp.bmp' (this is very important! The bitmap must have the same name as the component class.). When you click the Open button, the bitmap editor appears (this is actually pretty basic; there's no reason why you can't create the bitmap in a fully-featured graphics editor and instead of 'Add new item..' choose 'Add existing item...' from the menu described above - just be sure that class and bitmap have the same name, just different extensions).

Bring up the properties window for the bitmap and change it to a size of 16 by 16 pixels, with 16 colours. Now you can create the bitmap. There's one caveat: the colour bright green (0x00FF00) is treated as a transparent colour and the toolbox's background colour will show through wherever this colour is used.

As an example, my magnificent creation looked like this:



Save the bitmap when you are done.

3. Embed the bitmap in the executable

You now need to tell the compiler that the bitmap should be embedded in the DLL itself, rather than kept as a separate file. You could, in fact, keep it separate, but then there is a risk that the bitmap file would become separated from the component; it's better to embed it. This is easy: in Solution Explorer, right click the bitmap's name, and choose 'Properties' from the popup menu. Under the 'Advanced' heading you will see a property called 'Build Action'. This has a drop-down list attached to it and the current setting is probably 'Content'. Change this to 'Embedded Resource' to embed it in the executable file at compile time.

But how do you tell the framework that there is a toolbox bitmap to display? In fact, you don't have to; the framework will automatically look for a bitmap of the same name as the class in the compiled executable, and will use that bitmap if it finds one (this is why the bitmap must have the same name as the component class!).

4. Change the namespace of the bitmap file

If you now load this component in Customize Toolbox, you will see that it hasn't worked - the component still has the old cog wheel bitmap! The reason for this is that the framework could not find the bitmap in the DLL, because in effect it was looking in the wrong place.The bitmap must not only have the same name as the component class, it must also be in the same namespace. You can see from the code that the component's namespace is Microbion.TestComp. What namespace is the bitmap currently in? Well, from Solution Explorer you see that it's actually in the namespace TestComp. In this case, you need to change this, but you won't always need to do so - it depends on how you arrange your namespaces.

If you right-click the project name in Solution Explorer,and choose 'Properties', a large dialog box appears; one of the entries is 'Default Namespace' which is currently set to 'TestComp'. Change this to 'Microbion.TestComp' and rebuild the component. Load the component again, and this time the bitmap appears correctly.

That's really all there is to it. Just remember to make sure the name and namespace of the bitmap and component class are the same, and it should work fine.