Programming Two

Tutorial 6: Intro to C# and XNA

Introduction

This tutorial assumes that you have understood most of the C++ we have done so far and therefore advances at a relatively fast rate.

Stage One: Initialisation

Start a Windows Game Project and then run it (You can press F5)
This should give you a nice blue blank window

Stage Two: Window Properties

Add the code below to either the `Initialize()` function or code your own function called `SetUpXNADevice()` or something similar. This function will need to be called from the `Initialize()` function.

```
graphics.PreferredBackBufferWidth = 500;
graphics.PreferredBackBufferHeight = 500;
graphics.IsFullScreen = false;
graphics.ApplyChanges();
Window.Title = "XNA Tutorial 1";
```

Play around a little with these settings. Change the window title and alter the window size. Look at the following msdn entry to see what other methods/members are available: http://msdn2.microsoft.com/en-us/library/microsoft.xna.framework.graphicsdevicemanager.togglefullscreen.aspx

Stage Three: Loading Images

There are two ways to adding/display images

Before we use either method we need to add the following code:

```
GraphicsDevice device; // <---- added handle to device

Texture2D mTexture; // <---- added handle to texture
```

To our game class, under the pre-generated

```
GraphicsDeviceManager graphics;
SpriteBatch spriteBatch; // <--- needed to draw the sprite
```

Is fine

Method 1: Loading from file
Add the following code:

```
private void InitSprites()
{
    // @ symbol means interpret a string literally
    string path = @"..\..\..\imagename.png";
    if (device != null)
    {
        mTexture = Texture2D.FromFile(device, path);
    }

}
```

And the call it from our `Initialize()` function. The path needs to be the same as where we are storing our image. Don't forget that the image name and extension must match our resource.

Method 2: Loading from resource
The first thing we need to do is add our image as a resource to our content directory structure. Make sure you have solution explorer view selected. To do this add the image to the relevant directory. Then right click our content solution tab and navigating to the add option select existing item.
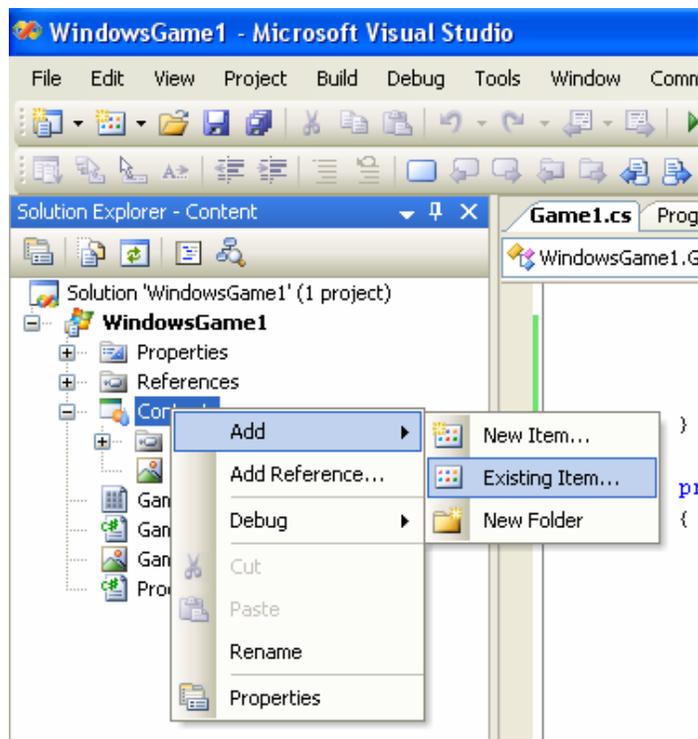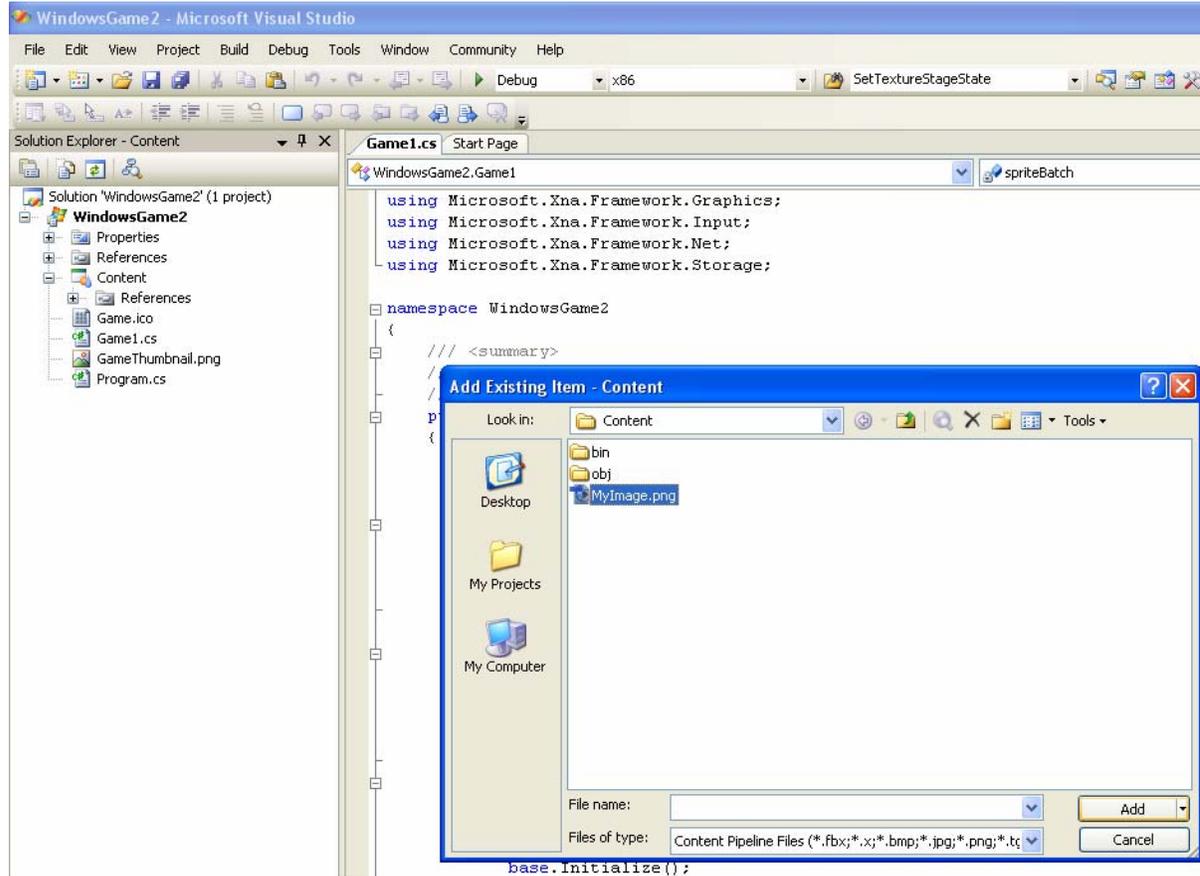
Then locate the image you wish to add in the browser window and click add. The image should now be in your content listing when you are viewing with solution explorer.

With your image selected in the solution explorer click the properties window tab to see your image properties, including its asset name – which is the filename by default.
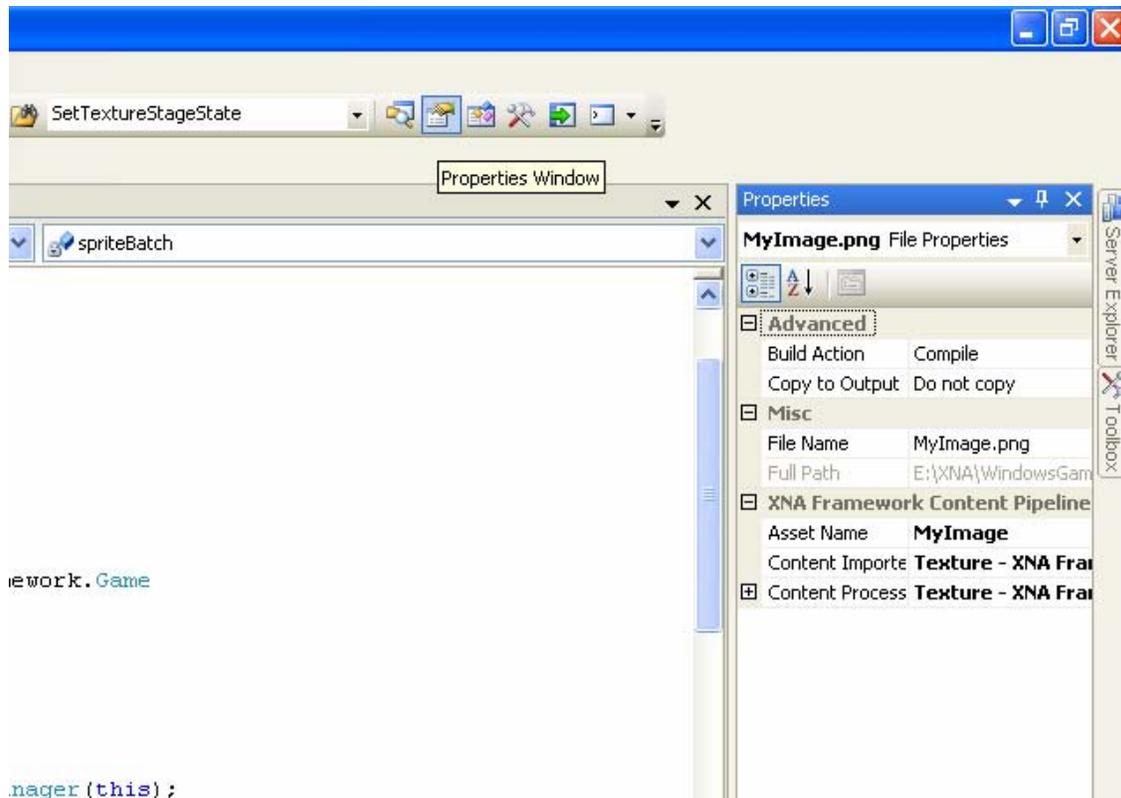
In the function `LoadContent()` add the following code:

```
spriteBatch = new SpriteBatch(GraphicsDevice);
mTexture = Content.Load<Texture2D>("JumpingJelly");
```

This code loads the texture from the content via the asset name.

Stage Four: Drawing the Images

We need to add the following code to our `Draw(GameTime gameTime)` function.

```
spriteBatch.Begin(SpriteBlendMode.AlphaBlend);
spriteBatch.Draw(mTexture, new Rectangle(0, 0, 100, 100), Color.White);
spriteBatch.End();
```

This displays the whole of the texture. Play around altering the rectangle to change where the image is drawn, and at what size. Play with the blending modes to see if you can determine any difference. You can also alter the background colour via the following function:

```
graphics.GraphicsDevice.Clear(Color.CornflowerBlue);
```

Find or make an texture that contains two or more images. Google "sprite sheet" if necessary.

```
spriteBatch.Begin(SpriteBlendMode.AlphaBlend);
spriteBatch.Draw(mTexture, new Rectangle(0, 0, 50, 50), new
      Rectangle(0, 0, 256, 256) , Color.White);
spriteBatch.End();
```

Play with the `spriteBatch.Draw` function to get one of the two (or more images) on the texture to display. The Draw function is overloaded (remember overloading) so that we have multiples of the same function with different parameters.
See: http://msdn2.microsoft.com/en-us/library/microsoft.xna.framework.graphics.spritebatch.draw.aspx
For the list of Draw operations.

Stage Five: Animation Attempt

Attempt to get the image changing from one of the sprites on the texture to another. If you have a sprite sheet loop through the contained images. To do this you will likely need a global rectangle that you use to specify what section of the image to display. You will also need to make your own function `protected void UpDateSprite()` in here you should update which of the images on the texture is being displaying. This function should be called from the `Update(GameTime gameTime)` function.