



Part 5

With Lloyd Humphreys
(<http://www.lloydhumphreys.com>)

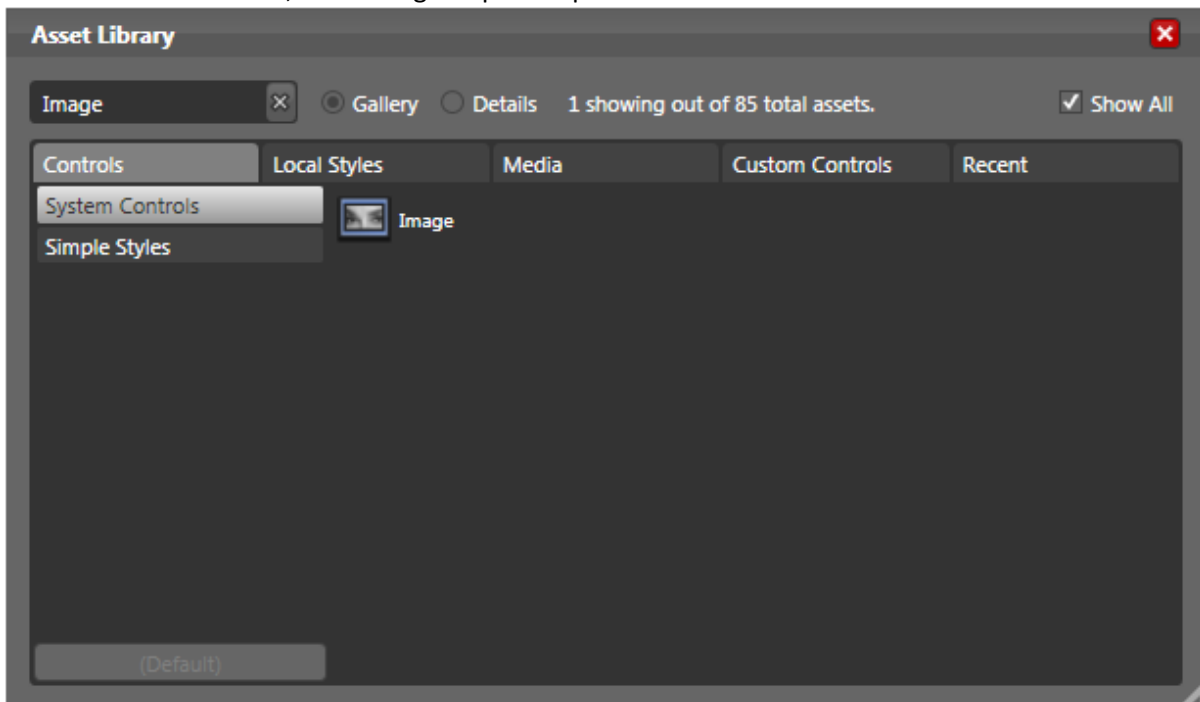
Working With Media

Working with media in Expression Blend is simple, and can be controlled easily. In this tutorial, I'll show you how to embed audio, video and images into Blend, allowing you to play and pause each. There are also some resource materials attached to this document.

To begin with we'll start with images. Images are simply bitmaps, copied into the application and used as a resource.

Adding An Image

Add an image to your Resources folder, or to anywhere you're saving your resources. Then open up the asset library and select "Image". Again, you'll have to make sure "Show All" is checked – I find this somewhat unusual, as an Image is quite important.



After selecting this, drag it and make it the size you'd like onto the canvas. Head over to the Properties panel (with it selected) and under Common Properties, select the dropdown by "Source". If you've already added an image to the project, it'll be there. If not, click the ellipsis next to this, and the image you select will automatically be added to the project.

Adding a Video Clip

Adding a video clip is simple. Head over to the asset library and with “Show All” selected, search for “MediaElement”. Insert this into your project, and set it to the desired size. If you’ve got a video clip in your resources, then head to Properties, and select the Media node. Like above, select your item from the Source dropdown. If you’re adding a new video clip, click the ellipsis. Note that when you’re adding a large video, Blend will appear to freeze for a few seconds, or minutes depending on how good your machine is. If it takes hours, you shouldn’t even be reading this!

What about controlling the video clip? Pausing it? Playing it? Stopping it? Read on into the Controlling Your Media section.

Adding an Audio Clip

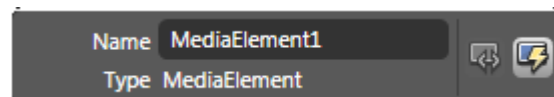
Adding an Audio clip is no more difficult. Using the same MediaElement, when there is no visual content, it just becomes a placeholder. It won’t show up in production, or in debugging. Under Source select your audio clip and run your program (F5). The clip will play once the app is loaded. What about controlling the Audio? Read on...

Controlling Your Media

Now we’re delving deeper into the code behind your application. In order to do this, you’ll need Visual Studio 2005 or later. My tutorials will be shown using 2008. If you don’t have either IDE (Integrated Development Environment), then you can download Visual Studio 2008 Express Edition from Microsoft’s website.

Once you have the MediaElement placeholder on your canvas, navigate to the Media node, and expand its advanced properties. Under “LoadedBehavior” select “Manual”. This controls when your media begins playing. Also, since we’re going to be referencing the MediaElement in code, you’ll need to give it a name. I called mine

“MediaElement1”. You can then tuck it away somewhere on the page and add two buttons.



Select either and push F2, and change their names to Play and Pause, respectively. Then, with one of the buttons selected, head over to the Properties panel, and click the Events button (also shown in the above shot).

A whole new panel will appear. With the Play button selected, click the “Click” event, and enter the value “btn_Play”. This is the name of that button. For Pause, call it “btn_Pause”. After you name your first, Visual Studio will open automatically, and Blend will tell you it can do nothing because it’s interfacing with Visual Studio “at this time”. Once VS loads, you’ll be confronted with the C# File named Window1.xaml.cs. If not, then open the Solution Explorer (to the right) and expand Window1.xaml. The CS file will be in there. Once this is opened, there are 4 lines of code automatically added for you:

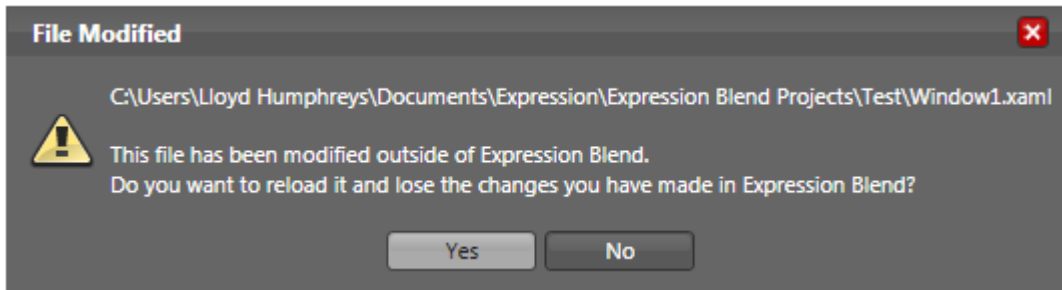
```
private void btn_Play(object sender, RoutedEventArgs e)
{
    MediaElement1.Play();
}

private void btn_Pause(object sender, RoutedEventArgs e)
{
    MediaElement1.Pause();
}
```

In this screenshot, the lines rounded with Red were added by me. This tells the program that

MediaElement1 - the MediaElement we named earlier – wants to play, or pause. The () is there because there are no arguments. Arguments are basically more advanced properties or content.

Save this, and head back to Expression. Expression will show the following message:



Click Yes, and the project will reload.

Now that that is working, how about we make it more interesting? Combining the things we've learned in the past lessons, we'll create an altogether more interesting application to look at.

In my example, I used the song "Believer" by Kill Hannah, one of my current favourite bands. If I could've I'd have included it with the samples, but I was unable to, due to Copyright stuff. Also in the sample is the code behind it. I added an Image of the Album Art, named it KHPic and added the following to the code:

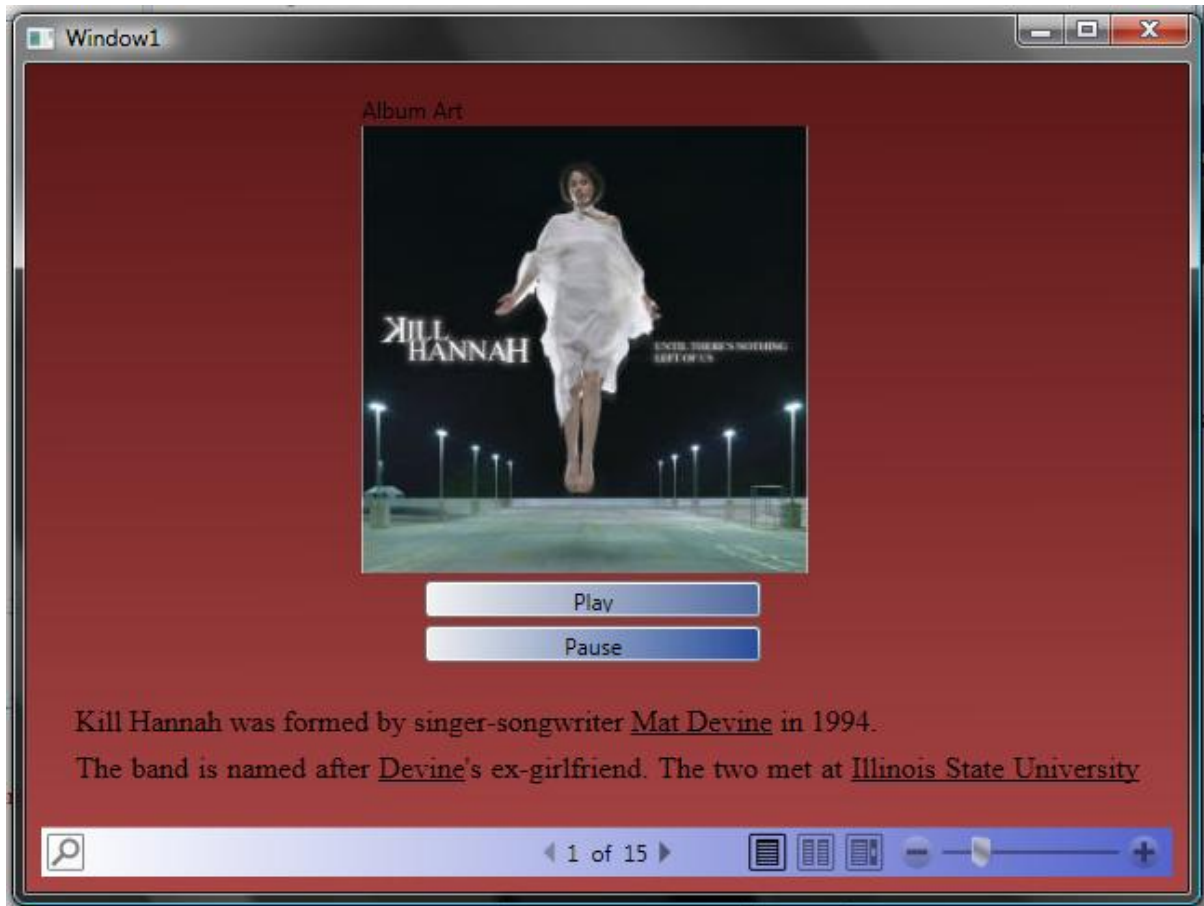
```
private void btn_Play(object sender, RoutedEventArgs e)
{
    MediaElement1.Play();
    KHPic.ToolTip("Now Playing: 'Believer', from the Album 'Until There's Nothing Left Of Us'");
}

private void btn_Pause(object sender, RoutedEventArgs e)
{
    MediaElement1.Pause();
    KHPic.ToolTip("Currently Paused");
}
```

As you can see, "KHPic.ToolTip" has been added to the code. The ToolTip is the caption that pops up when you hover over something. So now, when we click the Play button, the Tooltip of the Album art changes to "Now Playing:.."

Another thing I added was a FlowDocumentReader, and copied some information about Kill Hannah from Wikipedia, and set that as the content. Already, we have a decent looking application. A simple label named "Album Art" finishes the application, and makes it altogether more interesting to look at!

Not the best colours, but it's there to illustrate a point. I created that in just 4



steps!

1. Create it (layout etc)
2. Edit the C#
3. Edit some XAML using my method of deleting tags inside the FlowDocument
4. Run/Debug/Build

About The Included Content

The included content is the whole of the folder in which this exact application was created on my PC. There's also a built version which will run under the \bin\debug directory. This will run, but the music won't play – because it's not there! Due to (damn) copyright issues, I can't distribute the song. What I can do, however, is distribute everything else. If you press Play, the program will either crash or throw an error. If you want, there are 2 ways around this:

1: Open it up in Blend, and add a different song, then rebuild it. (easiest)

2: Place an audio file named (exactly) "Kill Hannah – Believer.wav" into the \Resources folder. (Of course, you'll still need to rebuild it, unless you place it inside the \bin\debug\Resources folder).

Really, there's nothing to learn by listening to the music, but looking at the Window1.xaml.cs file will certainly help.