Chapter No. 14
"Adding Sound, Music, and Video"
In this package, you will find:

A Biography of the author of the book
A preview chapter from the book, Chapter NO.14 "Adding Sound, Music, and Video"
A synopsis of the book’s content
Information on where to buy this book

About the Author

**Gastón C. Hillar** has been working with computers since he was eight. He began programming with the legendary Texas TI-99/4A and Commodore 64 home computers in the early 80's.

He has a Bachelor's degree in Computer Science in which he graduated with honors, and he also has an MBA (Master in Business Administration) in which graduated with an outstanding thesis. He has worked as a developer, an architect, and project manager for many companies in Buenos Aires, Argentina. Now, he is an independent IT consultant and a freelance author always looking for new adventures around the world. He also works with electronics (he is an electronics technician). He is always researching new technologies and writing about them. He owns an IT and electronics laboratory with many servers, monitors, and measuring instruments.

He has written another book for Packt, "C# 2008 and 2005 Threaded Programming: Beginner's Guide".

For More Information:
He contributes to Dr. Dobb’s Go Parallel programming portal
http://www.ddj.com/goparallel/ and he is a guest blogger at Intel Software
Network http://software.intel.com

He is the author of more than 40 books in Spanish about computer science, modern
hardware, programming, systems development, software architecture, business
applications, balanced scorecard applications, IT project management, the Internet, and
electronics.

He usually writes articles for the Spanish magazines "Mundo Linux", "Solo
Programadores" and "Resistor".

He lives with his wife, Vanesa, and his son, Kevin. When not tinkering with computers,
he enjoys developing and playing with wireless virtual reality devices and electronics
toys with his father, his son, and his nephew Nico.

You can reach him at: gastonhillar@hotmail.com

You can follow him on Twitter at: http://twitter.com/gastonhillar

Gastón's blog is at: http://csharpmulticore.blogspot.com

For More Information:
www.packtpub.com/3d-game-development-with-microsoft-silverlight-3-
beginners-guide/book
3D Game Development with Microsoft Silverlight 3

Most online interactive content uses 2D graphics. To represent real-life situations, Rich Internet Applications (RIAs) need to show real-time 3D scenes. This book will let you add a new dimension to your Silverlight applications using C# and XAML. The book covers the various tools and libraries needed for giving life to 3D models in a Silverlight viewport—right from editing, exporting, loading, and controlling 3D models up to specific 3D algorithms. It will help you learn to develop 3D games and interactive 3D scenes for a web site with animated models, with numerous examples and clear explanations packed with screenshots to aid your understanding of every process. After all of the code is written and the additional art assets edited, they are all compressed into .zip files for easy availability and use.

What This Book Covers

Chapter 1: Lights, Camera, and Action!: In this chapter, we will cover many topics that will help us to understand the new tools and techniques involved in preparing 2D graphics to be used in Silverlight games. This chapter is all about tools and graphics.

Chapter 2: Working with 2D Characters: In this chapter, we will begin creating 2D characters that move on the screen and we will learn to control their behavior in the 2D space. We will learn about GPU hardware acceleration, 2D vectors, resolutions, sprites, and animation.

Chapter 3: Combining Sprites with Backgrounds: In this chapter, we will control multiple sprites created on demand. We will animate several independent sprites at the same time, while responding to the keys pressed by the player to control characters and some game logic.

Chapter 4: Working with 3D Characters: In this chapter, we will take 3D elements from popular and professional 3D DCC tools and we will show them rendered in real-time on the screen. We will learn a lot about 3D models, meshes, and 3D engines.

Chapter 5: Controlling the Cameras: Giving Life to Lights and Actions: In this chapter, we will learn everything we need to know about 3D cameras to be able to render our models in real-time on a 2D screen from different angles. We will change the values for their most important properties and we will be able to watch their effects in a rendered 3D view.

For More Information:
Chapter 6: Controlling Input Devices to Provide Great Feedback: In this chapter, we will learn everything we need to know about the most widely used gaming input devices. We will be able to read values from them in order to control many aspects of our games.

Chapter 7: Using Effects and Textures to Amaze: In this chapter, we will learn everything we need to know about the process of enveloping a 3D model using textures. We will be able to take 3D elements from popular and professional 3D DCC tools and we will show them rendered in real-time on the screen with different textures and enlightened by many lights.

Chapter 8: Animating 3D Characters: In this chapter, we will learn how to move, rotate, and scale the 3D models in the 3D scenes. We will use object-oriented capabilities to define independent behaviors for simple and complex 3D characters.

Chapter 9: Adding Realistic Motions Using a Physics Engine: In this chapter, we will simulate some laws of 2D and 3D physics. We will learn to define gravity force, mass, drag coefficients, and moment of inertia to represent the physical properties of the bodies that define a model.

Chapter 10: Applying Artificial Intelligence: In this chapter, we will detect collisions between 3D characters and we will define specific behaviors using artificial intelligence, persecution, and evasion algorithms.

Chapter 11: Applying Special Effects: In this chapter, we will use advanced physics and special effects. We will generate gravity effects, we will add fluids with movements, and we will use transitions to determine different states in a game.

Chapter 12: Controlling Statistics and Scoring: In this chapter, we will create gadgets to display different kinds of information to the player on the screen. We will also calculate different kinds of information in order to update the gauges. Also, we will measure and improve the game’s overall performance.

Chapter 13: Adding Environments and Scenarios: In this chapter, we will create menus and attractive transitions. We will add configuration options and we will save them using an isolated storage.

Chapter 14: Adding Sound, Music, and Video: In this chapter, we will generate sounds associated to game events. We will also add presentation videos and background music.

Appendix: Pop Quiz Answers: This appendix will include answers to all the pop quiz questions chapter-wise.

For More Information:
Adding Sound, Music, and Video

A game needs sound, music and video. It has to offer the player attractive background music. It must also generate sounds associated with certain game events. When a spaceship shoots a laser beam, a sound must accompany this action. Reproducing videos showing high-quality previously rendered animations is a good idea during transitions between one stage and the next.

In this chapter we will add sounds, music and videos to the game. By reading this chapter and following the exercises we will learn how to:

- Add background music to the game
- Generate sounds associated to the game events
- Take advantage of multiple channel audio
- Control the reproduction of concurrent sounds
- Organize the media elements for the game
- Prepare our media elements for their use in Silverlight applications
- Take advantage of Silverlight 3 performance enhancements for videos
- Reproduce videos combined with animated projections

For More Information:
Hear the UFOs coming

So far, we have worked with 3D scenes showing 3D models with textures and different kinds of lights. We took advantage of C# object-oriented capabilities and we animated 3D models and moved the cameras. We have read values from many different input devices and we added physics, artificial intelligence, amazing effects, gauges, statistics, skill levels, environments, and stages. However, the game does not use the speakers at all because there is no background music and there are no in-game sounds. Thus, we have to sort this issue out. Modern games use videos to dazzle the player before starting each new stage. They use amazing sound effects and music custom prepared for the game by renowned artists. How can we add videos, music, and sounds in Silverlight?

We can do this by taking advantage of the powerful multimedia classes offered by Silverlight 3. However, as a game uses more multimedia resources than other simpler applications, we must be careful to avoid including unnecessary resources in the files that must be downloaded before starting the application.

Time for action – installing tools to manipulate videos

The 3D digital artists used Blender to create an introductory video showing a high quality rendered animation for five seconds. They took advantage of Blender’s animation creation features, as shown in the following screenshot:
A spaceship flies in a starry universe for a few seconds. Then, the camera navigates through the stars.
Adding Sound, Music, and Video

Your project manager wants you to add this video as an introduction to the game. However, as the video file is in AVI (Audio Video Interleave) format and Silverlight 3 does not support this format, you have to convert the video to an appropriate format.

The creation of video animations for a game is very complex and requires specialist skills. We are going to simplify this process by using an existing video.

First, we must download and install an additional tool that will help us in converting an existing video to the most appropriate file formats used in Silverlight 3:

The necessary tools will depend on the applications the digital artists use to create the videos. However, we will be using some tools that will work fine with our examples.

1. Download one of the following files:

<table>
<thead>
<tr>
<th>Application's name</th>
<th>Download link</th>
<th>File name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression Encoder 2</td>
<td><a href="http://www.microsoft.com/expression/try-it/try-it-v2.aspx">http://www.microsoft.com/expression/try-it/try-it-v2.aspx</a></td>
<td>Encoder_Trial_en.exe</td>
<td>This is a commercial tool, but the trial offers a free fully functional version for 30 days. This tool will enable us to encode videos to the appropriate format to use in Silverlight 3.</td>
</tr>
<tr>
<td>Expression Encoder 3</td>
<td><a href="http://www.microsoft.com/expression/try-it">http://www.microsoft.com/expression/try-it</a></td>
<td>Encoder_Trial_en.exe</td>
<td>This is the newest trial version of the aforementioned commercial tool.</td>
</tr>
</tbody>
</table>

2. Run the installers and follow the steps to complete the installation wizards.

3. If you installed Expression Encoder 2, download and install its Service Pack 1. The download link for it is http://www.microsoft.com/expression/try-it/try-it-v2.aspx#encodersp1 file name—EncoderV2SP1_en.exe.

4. Once you have installed one of the versions of Expression Encoder, you will be able to load and encode many video files in different file formats, as shown in the following screenshot:

For More Information:

Chapter 14

What just happened?

We installed Expression Encoder. Now we have the necessary tool to convert existing video clipart to the most appropriate file formats to use in Silverlight 3.

Time for action – preparing a video to use it in Silverlight

Now, we are going to use Expression Encoder to convert the video from an AVI format to an **HD 720p** Intranet (High Definition, 1280X720 pixels) encoding profile, compatible with Silverlight 3:

1. Save or copy the original video file (*introduction_HD.avi*) in a new folder (C:\Silverlight3D\Invaders3D\Media).
2. Start Expression Encoder.
4. Now, select File | Import.... Browse to the folder that holds the video file (C:\Silverlight3D\Invaders3D\Media) and select the file *introduction_HD.avi* to import. Then, click on Open. Expression Encoder will analyze the file for a few seconds and then, the first frame for the video will appear on the preview area.

For More Information:

5. Expand **Profile** and select **HD 720p Intranet** from the **Video** combo box. This step defines the desired video profile for the encoding process. The **Video Profile** options will display the output video's width, height, and aspect ratio, among other parameter values, as shown in the following screenshot:

![Video Profile Options](image)

6. Click on the **Output** tab, expand **Job output** and click on the **Browse for output folder** button (…) on the right side of the **Directory** text box. Browse to the folder that holds the original video file (C:\Silverlight3D\Invaders3D\Media) and click on **OK**.

7. Select **File | Encode** or click on the **Encode** button. Expression Blend will begin the encoding job and will display the overall progress as shown in the following screenshot:

![Encoding Progress](image)

---

**For More Information:**

8. After a few seconds (depending on the video length, resolution, and format), Expression Blend will show **Ready** in the Status column.

9. Right-click on the item (C:\Silverlight3D\Invaders3D\Media\introduction_HD.avi) under **Media Content** and select **Open File Location** in the context menu that appears. A new Explorer window will appear showing the folder with a new sub-folder with the default user name as a prefix, the date and the time. Enter this sub-folder and move the new video file (introduction_HD.wmv) to the previously mentioned parent folder.

10. Double-click on the new video file (introduction_HD.wmv) and watch it using your default media player (Windows Media Player or VLC Media Player, among others). Check whether the video quality, resolution, and reproduction are as good as expected, as shown in the following screenshot:

For More Information:
What just happened?

You used Expression Blend to encode the original AVI video into a WMV (Windows Media Video) with an HD 720p Intranet encoding profile. Now, the video has a 1280X720 pixels resolution and it is compatible with Silverlight 3.

In this case, we created a video prepared for an application that runs on the Intranet because we want to test the game. However, we will have to choose a different encoding profile according to the Internet bandwidth offered by the hosting service and the average download speed available for the game’s potential players. The steps to encode the video using a different profile are the same as the ones used in the previously explained procedure. The only step that changes is the one in which we select the desired encoding profile.

Video formats supported in Silverlight 3

Silverlight 3 supports the video encodings shown in the following table:

<table>
<thead>
<tr>
<th>Encoding name</th>
<th>Description and restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Raw video</td>
</tr>
<tr>
<td>YV12</td>
<td>YCrCb(4:2:0)</td>
</tr>
<tr>
<td>RGBA</td>
<td>32-bits Red, Green, Blue, and Alpha</td>
</tr>
<tr>
<td>WMV1</td>
<td>Windows Media Video 7</td>
</tr>
<tr>
<td>WMV2</td>
<td>Windows Media Video 8</td>
</tr>
<tr>
<td>WMV3</td>
<td>Windows Media Video 9</td>
</tr>
<tr>
<td>WMVA</td>
<td>Windows Media Video Advanced Profile (non-VC-1)</td>
</tr>
<tr>
<td>WMVC1</td>
<td>Windows Media Video Advanced Profile (VC-1)</td>
</tr>
<tr>
<td>H.264 (ITU-T H.264 / ISO MPEG-4 AVC)</td>
<td>H.264 and MP43 codecs; base main and high profiles; only progressive (non-interlaced) content and only 4:2:0 chroma sub-sampling profiles</td>
</tr>
</tbody>
</table>

Silverlight 3 does not support interlaced video content.

If we want to use a video with an encoding format that does not appear in the previously shown table, we will have to convert it to one of the supported formats.

For More Information:
Using free applications to convert video formats

Expression Encoder is not the only application capable of converting videos to the encoding profiles supported by Silverlight 3. We can also use many free or open source applications and several online services to convert videos to any of the previously shown formats.

One example of a free and open source audio and video converter is MediaCoder (http://mediacoder.sourceforge.net/).

One example of a free online service to convert audio and video from one format to other is Media Convert (http://media-convert.com/). The advantage of online services dedicated to converting video formats is that they do not require the installation of additional software. Their trade-off is that we have to upload the original video file and then download the converted output video file. If the video files are big, we will need a very fast Internet connection.

If you have to convert too many big files and you do not have a fast Internet connection, installing a free and open source video converter such as MediaEncoder is a good choice.

Time for action – reproducing videos

Now, we are going to reproduce the video converted to a WMV format with an HD 720p Intranet encoding profile, before starting the game:

1. Open the 3DInvadersSilverlight project.
2. Open 3DInvadersSilverlightTestPage.aspx (double-click on it in the Solution Explorer, found under 3DInvadersSilverlight.Web project) and enable hardware acceleration (GPU acceleration).
3. Create a new sub-folder in the ClientBin folder in the 3DInvadersSilverlight.Web project. Rename it to Media.
4. Right-click on the previously mentioned folder and select Add | Existing item... from the context menu that appears.
Adding Sound, Music, and Video

5. Go to the folder in which you have copied the previously encoded video in the WMV format (C:\Silverlight3D\Invaders3D\Media). Select the WMV file and click on Add. This way, the video file will be part of the web project, in the new Media folder, as shown in the following screenshot:

6. Open MainPage.xaml (double-click on it in the Solution Explorer) and insert the following lines of code before the line that defines the cnvMainScreen Canvas:

```xml
<Canvas x:Name="cnvVideo" Visibility="Collapsed" >
    <MediaElement x:Name="medIntroduction" Width="1366" Height="768" AutoPlay="False" CacheMode="BitmapCache" Stretch="UniformToFill" Source="Media/introduction_HD.wmv" MediaEnded="medIntroduction_MediaEnded"/>
</Canvas>
```

7. Add the following lines of code to program the event handler for the MediaElement's MediaEnded event (this code will start the game when the video finishes):

```csharp
private void medIntroduction_MediaEnded(object sender, RoutedEventArgs e)
{
    StartGame();
}
```

8. Add the following private method to play the introductory video:

```csharp
private void PlayIntroductoryVideo()
{
    // Show the Canvas that contains the MediaElement as a child
cnvVideo.Visibility = Visibility.Visible;
    // Play the video
    medIntroduction.Play();
}
```

For More Information:
9. Replace the code in the \texttt{transitionSB\_Completed} method with the following. (We want to play the introductory video instead of starting the game, once the transition finishes.):
   \begin{verbatim}
   //StartGame();
   PlayIntroductoryVideo();
   \end{verbatim}

10. Build and run the solution. Click on the button and the video will start its reproduction after the transition effect, as shown in the following screenshot:

   ![Screenshot](image.png)

   \textbf{What just happened?}
   
   Now, the application shows an introductory video before starting the game. However, your project manager wants you to add some kind of 3D effect to the video. Players must understand that they are going to play a 3D game.

   \textbf{Locating videos in a related web site}
   
   First, we had to convert the original video to an encoding profile compatible with Silverlight 3. Then, we added it to a new \texttt{Media} sub-folder in the related website \texttt{ClientBin} folder. This way, we do not embed the large video file in the application’s .xap file.

   \begin{center}
   \textit{For More Information:}
   \end{center}
   \begin{quote}
   \end{quote}
Adding Sound, Music, and Video

The .xap file's size determines the time needed to download the Silverlight application. We are using the application's website to hold the media files. This way, we avoid generating a huge .xap file and the application downloads the media files on-demand.

The Source property uses a relative Uri (Media/introduction_HD.wmv) because the root folder for the .xap application is the ClientBin folder and the video is located in ClientBin/Media.

The AutoPlay property was set to False because we did not want the video to begin its reproduction until the transition ended.

**Stretching videos**

The video file prepared for Silverlight 3 uses a resolution of 1280X720 pixels. However, our game was prepared for 1366X768 pixels. Therefore, we created added a Canvas (cnvVideo) with a MediaElement instance (medIntroduction) as a child, using XAML code.

We defined the reproduction area to be 1366X768 pixels and we assigned the UniformToFill value to the Stretch property. Thus, the MediaElement resizes the original 1280X720 pixels video to fill its dimensions (1366X768 pixels) while preserving the video's native aspect ratio.

The following diagram shows the results of using the four possible values in the Stretch property with the same original video:

![Diagram of video stretching options](image)

For More Information:
The following table explains the results of using the previously mentioned values:

<table>
<thead>
<tr>
<th>Stretch value</th>
<th>Description</th>
<th>Aspect ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>The video preserves its original size.</td>
<td>Preserved.</td>
</tr>
<tr>
<td>Uniform</td>
<td>The video is resized to fit in the destination dimensions.</td>
<td>Preserved.</td>
</tr>
<tr>
<td>UniformToFill</td>
<td>The video is resized to fill the destination dimensions. The video content that does not fit in the destination rectangle is clipped.</td>
<td>Preserved.</td>
</tr>
<tr>
<td>Fill</td>
<td>The video is resized to fill the destination dimensions.</td>
<td>Not preserved.</td>
</tr>
</tbody>
</table>

**Taking advantage of GPU acceleration to scale videos**

We configured the web project to start the Silverlight plug-in with hardware acceleration capabilities. As we are using the UniformToFill value for the Stretch property, assigning the BitmapCache value to the MediaElement’s CacheMode property, Silverlight will try to perform the resize operation using the GPU.

The GPU acceleration will work only when the MediaElement has to show the video in a different size. If the value assigned to the Stretch property is None, the video size will not change and there will be no GPU assistance during the reproduction process.

First, we made cnvVideo visible and then we started reproducing the video calling medIntroduction’s Play method, in the PlayIntroductoryVideo method.

As we wanted the game to start once the video ended, we programmed the MediaEnded event handler with the following line:

```
StartGame();
```
Adding Sound, Music, and Video

Time for action – applying projections

We want players to understand that they are going to play a 3D game. Hence, we will add a plane projection to the Canvas that contains the video (the MediaElement instance):

1. Stay in the 3DInvadersSilverlight project.
2. Open MainPage.xaml and insert the following lines of code after the line that defines the medIntroduction MediaElement:
   ```xml
   <Canvas.Projection>
     <PlaneProjection RotationX="-40" RotationY="15" RotationZ="-6"
                        LocalOffsetX="-70" LocalOffsetY="-105" />
   </Canvas.Projection>
   ```
3. Build and run the solution. Click on the button and the video will start its reproduction after the transition effect. However, this time, it will be displayed projected using a perspective transform, as shown in the following screenshot:

![3D video projection](image)

What just happened?

Your project manager is amazed! The 3D digital artists are astonished! You could add a 3D perspective to the 2D video in just a few seconds. Now, the introductory video is even more attractive.

For More Information:

We added a PlaneProjection instance to the Canvas (cnvVideo) that contains the MediaElement (medIntroduction). Then, we assigned the following values to its properties:

- RotationX: -40
- RotationY: 15
- RotationZ: -6
- LocalOffsetX: -70
- LocalOffsetY: -105

The RotationX, RotationY, and RotationZ properties specify the number of degrees to rotate the Canvas in the space. The LocalOffsetX and LocalOffsetY properties specify the distance the Canvas is translated along each axis of the Canvas' plane.

We can apply a perspective transform to any UIElement by setting its Projection property to a PlaneProjection instance. Then, we can assign the desired values to the PlaneProjection's properties.

PlaneProjection is a subclass of the Projection class. The last one allows you to describe how to project a 2D object in the 3D space using perspective transforms.

**Time for action – animating projections**

Your project manager wants you to animate the perspective transform applied to the video while it is being reproduced.

We are going to add a StoryBoard in XAML code to animate the PlaneProjection instance:

1. Stay in the project, 3DInvadersSilverlight.
2. Open MainPage.xaml and replace the PlaneProjection definition with the following line (we have to add a name to refer to it):

```xml
<PlaneProjection x:Name="proIntroduction" RotationX="-40"
RotationY="15" RotationZ="-6" LocalOffsetX="-70"
LocalOffsetY="-105" />
```

For More Information:

3. Add the following lines of code before the end of the definition of the cnvVideo Canvas:

```xml
<Canvas.Resources>
  <Storyboard x:Name="introductionSB">
    <DoubleAnimation Storyboard.TargetName="proIntroduction"
                     Storyboard.TargetProperty="RotationX"
                     From="-40" To="0" Duration="0:0:5"
                     AutoReverse="False" RepeatBehavior="1x" />
  </Storyboard>
</Canvas.Resources>
```

4. Now, add the following line of code before the end of the PlayIntroductoryVideo method (to start the animation):

```csharp
introductionSB.Begin();
```

5. Build and run the solution. Click on the button and the video will start its reproduction after the transition effect. While the video is being played, the projection will be animated, as shown in the following diagram:

What just happened?
Now, the projection that shows the video is animated while the video is being reproduced.

For More Information:
Working with a StoryBoard in XAML to animate a projection

First, we added a name to the existing PlaneProjection (proIntroduction). Then, we were able to create a new StoryBoard with a DoubleAnimation instance as a child, with the StoryBoard's TargetName set to proIntroduction and its TargetProperty set to RotationX. Thus, the DoubleAnimation controls proIntroduction's RotationX value.

The RotationX value will go from -40 to 0 in five seconds—the same time as the video's duration:

```
From="-40" To="0" Duration="0:0:5"
```

The animation will run once (1x) and it won't reverse its behavior:

```
AutoReverse="False" RepeatBehavior="1x"
```

We added the StoryBoard inside <Canvas.Resources>. Thus, we were able to start it by calling its Begin method, in the PlayIntroductionVideo procedure:

```
introductionSB.Begin();
```

We can define StoryBoard instances and different Animation (System.Windows.Media.Animation) subclasses instances as DoubleAnimation, using XAML code. This way, we can create amazing animations for many properties of many other UIElements defined in XAML code.

Time for action – solving navigation problems

When the game starts, there is an undesired side effect. The projected video appears in the right background, as shown in the following screenshot:

For More Information:
Adding Sound, Music, and Video

This usually happens when working with projections. Now, we are going to solve this small problem:

1. Stay in the 3DInvadersSilverlight project.

2. Open MainPage.xaml.cs and add the following line before the first one in the medIntroduction_MediaEnded method:
   
   ```csharp
   cnvVideo.Visibility = Visibility.Collapsed;
   ```

3. Build and run the solution. Click on the button and after the video reproduction and animation, the game will start without the undesired background, as shown in the following screenshot:

   ![Screenshot](image)

   **What just happened?**

   Now, once the video finishes its reproduction and associated animation, we have hidden the Canvas that contains it. Hence, there are no parts of the previous animation visible when the game starts.

   For More Information:
   
Time for action – reproducing music

Great games have appealing background music. Now, we are going to search and add background music to our game:

As with other digital content, sound and music have a copyright owner and a license. Hence, we must be very careful when downloading sound and music for our games. We must read licenses before deploying our games with these digital contents embedded.

1. One of the 3D digital artists found a very cool electro music sample for reproduction as background music. You have to pay to use it. However, you can download a free demo (Distorted velocity. 1) from http://www.musicmediatracks.com/music/Style/Electro/. Save the downloaded MP3 file (distorted_velocity.1.mp3) in the previously created media folder (C:\Silverlight3D\Invaders3D\Media).

You can use any other MP3 sound for this exercise. The aforementioned MP3 demo is not included in the accompanying source code.

2. Stay in the 3DInvadersSilverlight project.

3. Right-click on the Media sub-folder in the 3DInvadersSilverlight.Web project and select Add | Existing item... from the context menu that appears.

4. Go to the folder in which you copied the downloaded MP3 file (C:\Silverlight3D\Invaders3D\Media). Select the MP3 file and click on Add. This way, the audio file will be part of the web project, in the Media folder, as shown in the following screenshot:

For More Information:

5. Now, add the following lines of code at the beginning of the btnStartGame button's Click event. This code will enable the new background music to start playing:

```csharp
// Background music
MediaPlayer backgroundMusic = new MediaPlayer();
LayoutRoot.Children.Add(backgroundMusic);
backgroundMusic.Volume = 0.8;
backgroundMusic.Source =
    new Uri("Media/distorted_velocity._1.mp3", UriKind.Relative);
backgroundMusic.Play();
```

6. Build and run the solution. Click on the button and turn on your speakers. You will hear the background music while the transition effect starts.

**What just happened?**

You discovered that the speakers worked! Now, the game has attractive background music. Leave the speakers on, because your project manager wants more sound effects in the game.

We created a new `MediaPlayer` instance (`backgroundMusic`). However, this time, we used C# to create it, instead of working on XAML code. We had to add the new `MediaPlayer` to a parent container:

```csharp
LayoutRoot.Children.Add(backgroundMusic);
```

Then, we defined the desired `Volume` level and the `Source` as a new relative `Uri` (`Uniform Resource Identifier`):

```csharp
backgroundMusic.Volume = 0.8;
backgroundMusic.Source = new Uri("Media/distorted_velocity._1.mp3", UriKind.Relative);
```

The `Volume` ranges from 0 to 1. It uses a linear scale. We used 0.8 because we want the future sound effects to be louder than the background music.

The first parameter for the new `Uri` is the relative path (`ClientBin` is our base path in the web project). The second one is the `UriKind`. In this case, we are working with a `Relative Uri`.

Once we set up all the necessary parameters, we called the `Play` method and Silverlight started playing the MP3 file:

```csharp
backgroundMusic.Play();
```

The code goes on running while the music file is being played. Hence, the game starts and we can still hear the music.
Chapter 14

**Time for action – preparing audio files to use them in Silverlight**

As with video files, Silverlight 3 does not support all audio formats. Now, we are going to use Expression Encoder to convert audio files video from WAV (Waveform audio format) format to a WMA (Windows Media Audio) format with an adaptive streaming audio encoding profile, compatible with Silverlight 3:

1. Your project manager found two excellent WAV files to use as sound effects for the game on The Freesound Project website (http://www.freesound.org/):
   a. The first one is the Ufo atmosphere. This can be downloaded from http://www.freesound.org/samplesViewSingle.php?id=235—filename 235__Erratic__ufo_atmosphere.wav
   b. The second one is that of a thunder clap. This can be downloaded from http://www.freesound.org/samplesViewSingle.php?id=2525—filename 2525__RHumphries__rbh_thunder_03.wav

   The Freesound Project website offers high quality sounds with a Creative Commons License. The website offers thousands of samples. However, it does not offer songs.

2. Save or copy the original audio files (235__Erratic__ufo_atmosphere.wav and 2525__RHumphries__rbh_thunder_03.wav) in a new folder (C:\Silverlight3D\Invaders3D\Media).

3. Start Expression Encoder.

4. Select File | New Job.

5. Now, select File | Import.... Browse to the folder that holds the audio files (C:\Silverlight3D\Invaders3D\Media) and select the files to import, 235__Erratic__ufo_atmosphere.wav and 2525__RHumphries__rbh_thunder_03.wav. Then, click on Open. Expression Encoder will analyze the files for a few seconds, and then it will display a Ready Status for both items.

For More Information:

6. Expand Profile and select WMA High Quality Audio from the Audio combo box. This step defines the desired audio profile for the encoding process. The expanded Audio options will display the output audio's codec, mode, bitrate, sample rate, bits per sample, and channels, among other parameter values, as shown in the following screenshot:

7. Click on the Output tab, expand Job output and click on the Browse for output folder button (…) on the right side of the Directory text box. Browse to the folder that holds the original audio files (C:\Silverlight3D\Invaders3D\Media) and click on OK.

8. Select File | Encode or click on the Encode button. Expression Blend will begin the encoding job and will display the overall progress as shown in the following screenshot:

For More Information:
9. After a few seconds (depending on the audio files length and format), Expression Blend will show Ready in the Status columns.

10. Right-click on one of the items (235__Erratic_ufo_atmosphere.wma) under Media Content and select Open File Location in the context menu that appears. A new Explorer window will appear showing the folder with a new sub-folder with the default user name as a prefix, the date and the time. Enter in this sub-folder and move the audio files (235__Erratic_ufo_atmosphere.wma and 2525__RHumphries__rbh_thunder_03.wma) to the previously mentioned parent folder.

11. Double-click on the new audio files and listen to them using your default media player (Windows Media Player or VLC Media Player, among others). Check whether audio quality, channels and sampling are as good as expected, as shown in the following screenshot:

For More Information:
**What just happened?**

You used Expression Blend to encode the original WAV audio files into WMA with adaptive streaming audio encoding profiles. Now, the audio files are compatible with Silverlight 3.

In this case, we created audio files that were very high quality because we want to test the game locally. However, we will have to choose a different encoding profile according to the Internet bandwidth offered by the hosting service and the average download speed available for the game’s potential players.

**Audio formats supported in Silverlight 3**

Silverlight 3 supports the audio encodings shown in the following table:

<table>
<thead>
<tr>
<th>Encoding name</th>
<th>Description and restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCM</td>
<td>Linear 8 or 16-bits Pulse Code Modulation.</td>
</tr>
<tr>
<td>WMA Standard</td>
<td>Windows Media Audio 7, 8, and 9 Standard</td>
</tr>
<tr>
<td>WMA Professional</td>
<td>Windows Media Audio 9 and 10 Professional; Multichannel (5.1 and 7.1 surround) is automatically mixed down to stereo; it supports neither 24 bit audio nor sampling rates beyond 48 kHz</td>
</tr>
<tr>
<td>MP3</td>
<td>ISO MPEG-1 Layer III</td>
</tr>
<tr>
<td>AAC</td>
<td>ISO Advanced Audio Coding; AAC-LC (Low Complexity) is supported at full fidelity (up to 48 kHz); HE-AAC (High Efficiency) will decode only at half fidelity (up to 24 kHz); Multichannel (5.1) audio content is not supported</td>
</tr>
</tbody>
</table>

If we want to use an audio file with an encoding that does not appear in the previously shown table, we will have to convert it to one of the supported formats.

**Using free applications to convert audio formats**

Expression Encoder is not the only application capable of converting audio files to the encoding profiles supported by Silverlight 3. We can also use many free or open source applications and several online services to convert audio files to any of the previously shown formats.

The same applications mentioned for converting video formats are capable of converting audio files to the encoding profiles supported by Silverlight 3.

---

**For More Information:**

Time for action – creating a class to handle audio concurrency

Now, your project manager wants you to add different sound effects associated with different game events. Sometimes, these sounds have to be played concurrently. For example, a thunder can happen at the same time as the player applies an impulse to an UFO.

It is time to create a simple yet useful sound manager class. It must be able to handle many concurrent sound banks:

1. Stay in the 3DInvadersSilverlight project.
2. Create a new class—SoundManager.
3. Add the following lines of code at the beginning of the class definition (as we are going to use the System.Collections.Generic.List class):
   using System.Collections.Generic;
4. Add the following private and static variables:
   // The target for the new MediaElement instances (it must be a Panel subclass)
   private static Panel _target;
   // The last sound bank used
   private int _lastSoundBank = -1;
   // The number of sound banks available
   private static int SoundBanks = 5;
   // The default media folder
   public static String MediaFolder = "Media/";
5. Add the following private list of media elements that will hold many MediaElement instances:
   // The list of media elements
   private List<MediaElement> _soundBanks;
6. Add the following property to control the Volume (it must be improved later with additional code):
   public double Volume { get; set; }
7. Add the following constructor with a parameter:
   public SoundManager(Panel target)
   {
       _target = target;
       _soundBanks = new List<MediaElement>(SoundBanks);
       for (int i = 0; i < SoundBanks; i++)
   {

For More Information:
Adding Sound, Music, and Video

```csharp
    _soundBanks.Add(new MediaElement());
    _target.Children.Add(_soundBanks[i]);
}
}

8. Add the following public method to play a sound taking advantage of the banks using a round robin algorithm:

```csharp
public void Play(Uri uri)
{
    _lastSoundBank++;
    if (_lastSoundBank >= SoundBanks)
    {
        // A simple round robin algorithm
        _lastSoundBank = 0;
    }
    _soundBanks[_lastSoundBank].Stop();
    _soundBanks[_lastSoundBank].Source = uri;
    _soundBanks[_lastSoundBank].Volume = Volume;
    _soundBanks[_lastSoundBank].Play();
}
```

9. Add the previously encoded audio files (235__Erratic__ufo_atmosphere.wma and 2525__RHumphries__rbh_thunder_03.wma) to the Media folder in the web project.

10. Add the following two public methods to reproduce the previously added audio files without the need to call the generic Play method specifying a Uri:

```csharp
public void PlayAtmosphere()
{
    Play(new Uri(MediaFolder + "235__Erratic__ufo_atmosphere.wma", UriKind.Relative));
}
public void PlayThunder()
{
    Play(new Uri(MediaFolder + "2525__RHumphries__rbh_thunder_03.wma", UriKind.Relative));
}
```

What just happened?
The code to manage multiple concurrent audio playback is now held in the new SoundManager class. It uses some static private variables because there will be just one sound manager for the game. Thus, we will need just one instance of this class.

For More Information:
The class is quite easy to understand. The constructor receives a Panel as a parameter. It will use it as a container for the multiple MediaElement instances that it is going to create.

The _soundBanks list holds the number of MediaElement instances defined in the static variable SoundBanks (5).

**Using a round robin algorithm to work with concurrent sounds**

The Play method is responsible of playing the audio file, which Uri receives as a parameter. It uses a simple round robin algorithm to assign the available MediaElement instances (sound banks) for each new audio file that has to be played.

Initially, there are 5 sound banks available.

If we call the Play method 7 times, the following sequence will take place:

1. Play audio file #1 with MediaElement instance #0 (_soundBanks[0]).
2. Play audio file #2 with MediaElement instance #1 (_soundBanks[1]).
3. Play audio file #3 with MediaElement instance #2 (_soundBanks[2]).
4. Play audio file #4 with MediaElement instance #3 (_soundBanks[3]).
5. Play audio file #5 with MediaElement instance #4 (_soundBanks[4]).
6. Play audio file #6 with MediaElement instance #0 (_soundBanks[0]). The round starts again here.
7. Play audio file #7 with MediaElement instance #1 (_soundBanks[1]).

Each time the Play method is called, the value in the _lastSoundBank increases by 1:

```
_lastSoundBank++;
```

If the value is equal or greater than the maximum number of sound banks, it is time to start using the first sound bank available again (a new round):

```
if (_lastSoundBank >= SoundBanks)
{
    _lastSoundBank = 0;
}
```

Then, it is time to play the new audio file using the assigned MediaElement instance (sound bank).

> This way, we can play many audio files concurrently. We do not have to worry about MediaElement instances in the game because we can use the SoundManager instance features.

For More Information:

## Time for action – generating sounds associated to game events

Now, it is time to add concurrent sound effects associated to game events.

1. Stay in the 3DInvadersSilverlight project.
2. Open InvadersGame.cs.
3. Add the following private variable to hold the SoundManager instance:
   ```csharp
   private SoundManager _soundManager;
   ```
4. Add the following private method to create and initialize the sound manager related to the game:
   ```csharp
   private void InitializeSoundManager()
   {
       _soundManager = new SoundManager(_mainPage.LayoutRoot);
       _soundManager.Volume = 1;
   }
   ```
5. Add the following lines of code after the line `base.Initialize();` in the Initialize method:
   ```csharp
   InitializeSoundManager();
   ```
6. Now, add the following lines of code before the end of the UpdateWithTime method (a random thunder):
   ```csharp
   if (_random.Next(20) == 2)
       _soundManager.PlayThunder();
   ```
7. Replace the code that checks the Key.I key in the CheckKeyboard method with these lines:
   ```csharp
   if (KeyboardManager.IsKeyDown(Key.I))
   {
       _ufo1.Body.ApplyImpulse(_levelImpulse);
       // Play a sound when the user applies an impulse to the UFO
       _soundManager.PlayAtmosphere();
   }
   ```
8. Build and run the solution. Click on the button and turn on your speakers again. You will hear the background music. Then, the game will start. If you wait for a few seconds, you will hear the sound of many thunder claps. Sometimes, before a thunder clap finishes, you will hear many others. The music will go on playing in the background.

For More Information:
9. Now, press the / key and you will hear a strange sound like the atmosphere of a UFO. Another thunder clap will scare you. Press the / key again and you will enjoy concurrent sound effects.

**What just happened?**

You are promoted from the position of a game developer to that of a senior game developer!

The game has background music and amazing concurrent sound effects thanks to the simple use of a `SoundManager` class.

We created and initialized a `SoundManager` instance (`_soundManager`). Then, we used its methods to play a random thunder sound and a UFO atmosphere effect when the player presses the / key.

![Using a sound manager it is very easy to fire sounds when certain game events occur.]

**Have a go hero – animating the game over scene**

Now that you have shown your project manager videos with animated projections, he wants you to change the Game Over screen.

You have to use a `VideoBrush` to paint the **GAME OVER** text with an animated video. You do not know how to do it. However, you know about brushes, videos, animations, and timeline management. You can do it with some additional research!

**Have a go hero – configuring sounds and music**

Most games allow the players to configure the desired volume levels for the different sound effects and the background music.

Your project manager wants you to add a new gauge with a button to the game. When the player clicks on this button, the game has to pause and a `Canvas` with a control panel using different sliders must allow the user to control the volumes for sounds and music.

You will have to make some changes to the `SoundManager` class to allow the user to change some properties that define the volume for sounds organized by categories. Also, you have to change the way you play the background music in the game.

---

For More Information:

Pop quiz – working with audio and video in Silverlight 3

1. In order to reproduce an AVI (Audio Video Interleave) video in Silverlight 3 using MediaElement:
   a. You just have to assign the Uri to the MediaElement's Source property and Silverlight will recognize the video format.
   b. You must convert it to one of the video formats supported by Silverlight 3.
   c. You must typecast Uri to AviUri and assign the result to the MediaElement's Source property and Silverlight will recognize the video format.

2. If you want a video or audio file to be available in a website instead of being embedded in the application's .xap file:
   a. You can add it to the ClientBin folder in the application's web project.
   b. You can add it to the Assets folder in the application's main project.
   c. You can add it to the ClientBin folder in the application's main project.

3. In order to start reproducing a video or audio file using a MediaElement instance, you can:
   a. Call its Reproduce method.
   b. Call its Render method.
   c. Call its Start method.

4. When you reproduce multiple concurrent video or audio files using many different MediaElement instances:
   a. Silverlight reproduces only the last started MediaElement's audio.
   b. Silverlight mixes the concurrent sounds.
   c. Silverlight reproduces only the first started MediaElement's audio.

5. In order to reproduce a WAV audio in Silverlight 3 using MediaElement:
   a. You just have to assign the Uri to the MediaElement's Source property and Silverlight will recognize the audio format.
   b. You must typecast Uri to WavUri and assign the result to the MediaElement's Source property and Silverlight will recognize the audio format.
   c. You must convert it to one of the audio formats supported by Silverlight 3.

For More Information:
6. The Projection class allows describing:
   a. How to project a 2D object in the 3D space using perspective transforms.
   b. How to project a 2D object in the 2D space using perspective transforms.
   c. How to project a 3D model in the 3D space using perspective transforms.

7. You can apply a perspective transform to a UIElement:
   a. Setting its 3DEffects property to a PlaneProjection instance.
   b. Setting its Projection property to a PlaneProjection instance.
   c. Setting its Projection property to a Projection instance.

8. Silverlight 3 can use GPU acceleration to:
   a. Reproduce videos on any size.
   b. Reproduce audio files.
   c. Scale or stretch videos.

9. When a MediaElement's Stretch property has the UniformToFill value:
   a. It does not preserve the video's native aspect ratio.
   b. It preserves the video's native aspect ratio.
   c. It reproduces the video using a fixed 16:9 aspect ratio.

Summary

We learnt a lot in this chapter about adding and controlling sound, music, and videos. Specifically, we were able to use Expression Encoder to convert many different audio and video formats to the encoding profiles supported by Silverlight. We added animated plane projections to videos. We created a simple sound manager class capable of reproducing concurrent sound effects. Also, we learnt how to take advantage of hardware acceleration when scaling or stretching videos.

We learnt a lot in this book about developing 3D games using Silverlight and many other exciting applications, engines, libraries, and tools.

Now that we have learned to develop 3D games using Silverlight, we are ready to give life to 3D models in Rich Internet Applications. We can create amazing 3D scenes and games. Besides, when Silverlight is not enough, we have the option to switch to the more powerful XBAP WPF applications.

For More Information:
Where to buy this book

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.