XNA Sound

305890
Spring 2013
5/31/2013
Kyoung Shin Park
kpark@dankook.ac.kr

Two Choices for Sound

- Default Sound API
  - If your game is to use a few sound files, then the **SoundEffect, SoundEffectInstance, DynamicSoundEffectInstance** classes will provide everything you need to play and stream audio during gameplay. For background music, **Song, MediaLibrary, MediaPlayer** classes can be used.
- Microsoft XACT (Cross-platform Audio Creation Tool) Audio Authoring
  - For developers that plan on working with multiple sets of sound files during the gameplay, XNA Game Studio provides **XACT**, an advanced audio content creation and management tool. XACT provides audio designers with the ability to load sound files into groups, to organize those files into discrete cues that can be activated by in-game events, and to create sound transitions.

Default Sound API

- SoundEffect and SoundEffectInstance classes
- Capable of streaming audio, must use wav format.
  - Slower runtime than using the content pipeline.
- May build audio in the content pipeline, source does not matter.
  - Mp3, wma, wav and others may be loaded through the ContentManager, thus converting them into the XNB (XNA Framework Content Pipeline Binary file) format.

SoundEffect

- SoundEffect
  - To play a sound without using XACT, you will use SoundEffect or SoundEffectInstance class
  - **A SoundEffect class provides a loaded sound resource.**
  - A SoundEffect contains the audio data and metadata (such as wave data and loop information) loaded from a sound file.
  - **You can create a SoundEffect by calling ContentManager.Load.**
  - **You can play or modify the volume, panning, and pitch** of the SoundEffectInstance by setting the **Volume**, **Pitch**, and **Pan** properties.

SoundEffectInstance

- Increased control through SoundEffectInstance class
  - **SoundEffectInstance's are defined from an instance of the SoundEffect class.**
  - If creating multiple SoundEffectInstances from a single defined SoundEffect object, all SoundEffectInstances will use the memory resources reserved by the original SoundEffect object. Destroying the SoundEffect will break all SoundEffectInstances created from it.
  - Various additional methods and properties become available when using the SoundEffectInstance.

SoundEffectInstance

- SoundEffectInstance
  - **SoundEffectInstance class provides a single playing, paused, or stopped instance of a SoundEffect sound.**
  - **You can create a SoundEffectInstance by calling SoundEffect.CreateInstance**. Initially, the SoundEffectInstance is created as stopped, but you can play or pause it or stop it by calling **Play** or **Pause** or **Stop**.
  - **Pan (ratio of left to right speaker output), IsLooped, Pitch, Volume** properties may be either retrieved or set.
  - Further properties include **IsDisposed** (check if resources are still held), **State** (returns the SoundState.Playing, Paused, Stopped)
  - On Zune, a game can have a maximum of 16 total playing SoundEffectInstance instances at one time, combined across *all* loaded SoundEffect objects.

Playing a Sound using SoundEffect

- Declare SoundEffect

```
SoundEffect soundKaboom;
string soundName = "kaboom";     // .wav sound
```

- Play the sound
  - SoundEffect plays a sound in a **"fire and forget" fashion**; therefore, the lifetime of these sounds is managed by the framework.
  - These sounds will play once and then stop. They cannot be looped or 3D positioned.

```
soundKaboom = Content.Load<SoundEffect>(soundName );
soundKaboom.Play();                  // play a sound

// play a sound based on volume, pitch, and panning
soundKaboom.Play(0.5f, 0.0f, 0.0f);
```

Playing a Sound using SoundEffect & SoundEffectInstance

- Declare SoundEffect & SoundEffectInstance

```
SoundEffect soundEngine;
SoundEffectInstance soundEngineInstance = null;
```

- Play the sound

```
soundEngine = Content.Load<SoundEffect>("engine_2" );
soundEngineInstance = soundEngine.CreateInstance();
soundEngineInstance.Volume += 0.1f;
soundEngineInstance.IsLooped = true;
// set the pan speed so a full pan (-1.0 to 1.0 = 2.0)
// is achieved over the duration of the sound
panSpeed = 2.0f / (float)soundEngine.Duration.TotalSeconds;
// randomly choose panning direction
panSpeed = rnd.NextDouble() > 0.5 ? panSpeed : -panSpeed;
soundEngineInstance.Pan = panSpeed;
```

Playing a Sound using SoundEffect & SoundEffectInstance

```
if (keyboard.IsKeyDown(Keys.M)&&!(lastKeyboard.IsKeyDown(Keys.M))) {
        if(soundEngineInstance.State == SoundState.Playing)
                soundEngineInstance.Stop();
        else
                soundEngineInstance.Resume();
}
if (keyboard.IsKeyDown(Keys.Delete)) {
        soundEngineInstance.Dispose();
}
if (!soundEngineInstance.IsDisposed) {
        if (soundEngineInstance.State == SoundState.Playing)
                soundEngineInstance.Stop();
        else
                soundEngineInstance.Play();
}
```

MediaLibrary

- MediaLibrary
  - **MediaLibrary** provides the following properties that return media collections: Albums, Artists, Genres, Pictures, Playlists, and Songs.
  - **On Windows, MediaLibrary cannot find any songs unless the Windows Media Player previously found songs on the system.** That means that Windows Media Player must first search the system for music before any songs can be accessed through MediaLibrary.

MediaPlayer

- MediaPlayer **(static class)**
  - **MediaPlayer** provides methods and properties for playing songs in the media library.
  - To control song playback, use the Play, Pause, Stop and Resume methods. MoveNext and MovePrevious methods move to the next or previous song in the queue.

Song

- Song
  - **Song** provides information about a song, including the song's Name, Artists, Album.
  - You can obtain a Song object through the SongCollection.Item indexer and the MediaQueue.ActiveSong property.
  - Before XNA 3.0, the only way to add audio to your game was to use XACT. All audio files had to be .wav files, which can be extremely large. Now, with XNA 3.0, they've provided another way that allows us to use .mp3 files and .wma files.

Playing a Song using MediaPlayer

- Declare MediaLibrary

```
MediaLibrary library = new MediaLibrary();
```

- Play the song

```
MediaPlayer.Stop();          // stop current audio playback
SongCollection songs = library.Songs;
Song song = songs[0];        // select the first song in the list
MediaPlayer.Play(song);      // play the song

MediaPlayer.Play(library.Albums[0].Songs[0]); // play a song
```

Playing a Song using MediaPlayer

- Declare Song

```
Song bgm;
```

- Play the Sound

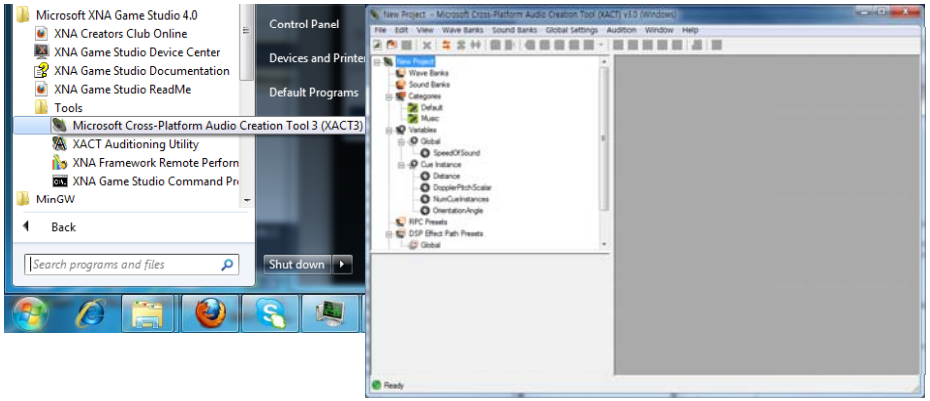
```
bgm = Content.Load<Song>("엘린+숲");
MediaPlayer.IsRepeating = true;     // looping sound
MediaPlayer.Volume = 0.5f;          // set volume (0.0 ~ 1.0)
MediaPlayer.Play(bgm);      // play a background music

MediaPlayer.Stop();         // stop the music
```

XACT (Cross-Platform Audio Creation Tool)

- May only import uncompressed audio (pcm **wav** and **aiff)**
- Allows separation of Audio engineer's and programmer's responsibilities.

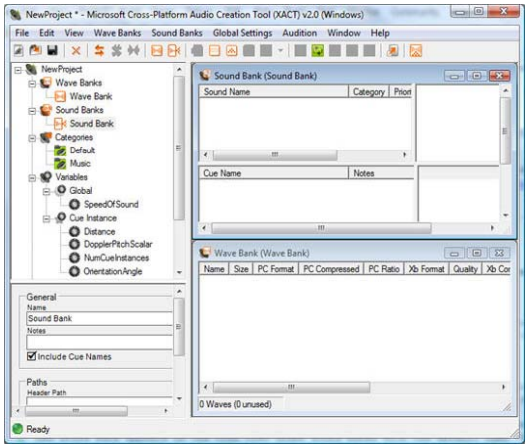
XACT

- XNA Game Studio 4.0 => Tools => Microsoft Cross Platform Audio Creation Tool 3 (XACT3)



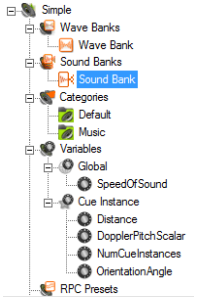
Adding Sounds to XACT

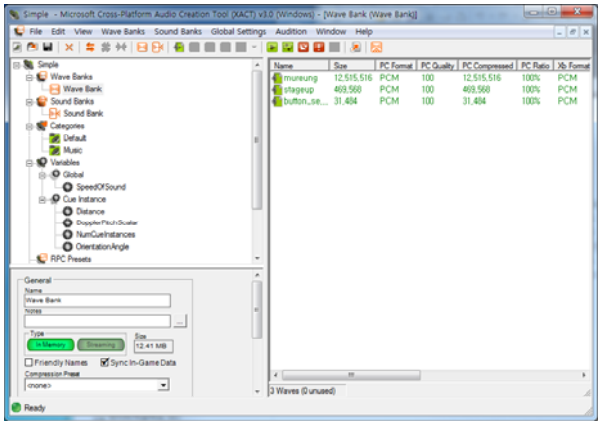
- Create a new XACT project (e.g., simple.xap)



Adding Sounds to XACT

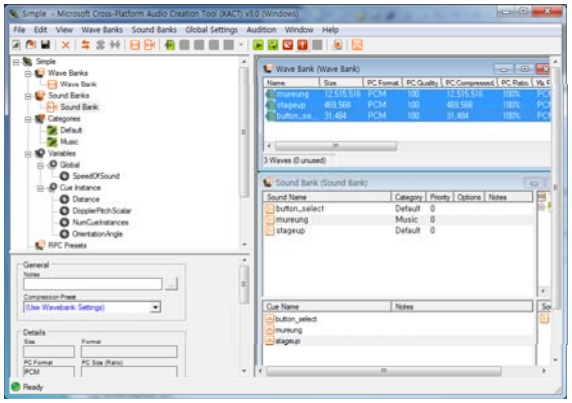
- Right click both Wave Bank and Sound Bank, create a new file for each. Then insert your wave files in Wave Bank.





Adding Sounds to XACT

- Drag the wave from the Wave Bank into the bottom half of the Sound Bank. This will populate the top half of the Sound Bank at the same time.



Waves, Sounds and Cues

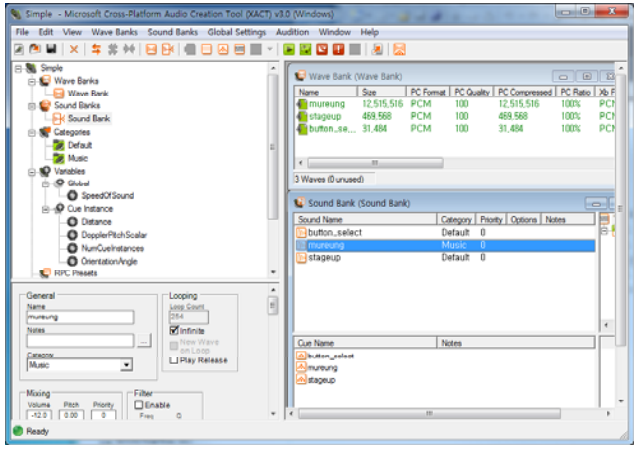
- Waves are uncompressed audio data.
- Sounds are collections of instructions to regulate how the wave files are played.
- Cues are the objects called to play specified Sounds.

With XACT you should only be referring to Cue objects, to play and control your audio. You can either save the Cue objects in memory or retrieve them from your xact project for every event.

//An example modifying a Cue to simulate 3D positioning:
Cue.Apply3D(AudioListner listener, AudioEmitter emitter);
//AudioListener and AdioEmitter simply contain: Speed,
//Magnitude and Direction of both the noise's origin and our listener

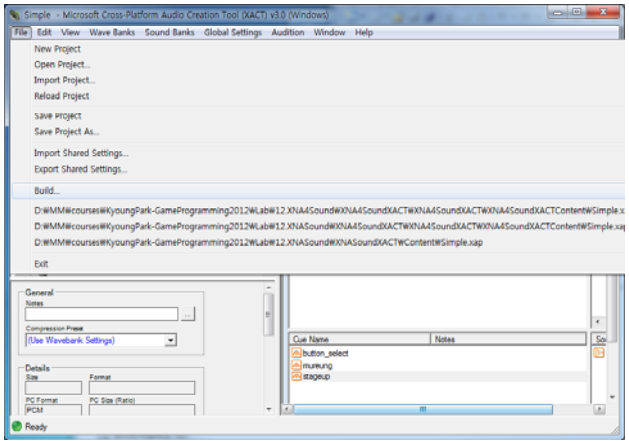
Looping or Backgroud Music in XACT

- Click the sound that you wish to designate as background music in Sound Bank. In the Category drop down box, select "Music", and save the project.



Adding Sounds to XACT

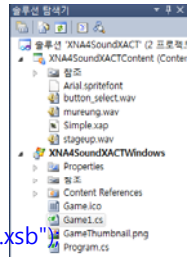
- To save the project, File->Save Project
- To build the project, File->Build



Playing Sounds using XACT

- Load the XACT project (e.g., simple.xap) into the Content
- Declare AudioEngine, SoundBank, WaveBank

```
// Audio objects
AudioEngine engine;
SoundBank soundBank;
WaveBank waveBank;
engine = new AudioEngine("Content/Simple.xgs");
soundBank = new SoundBank(engine, "Content/Sound Bank.xsb");
waveBank = new WaveBank(engine, "Content/WaveBank.xwb");
```



- Update audio

```
audioEngine.Update();
```

- Play sound cues

```
soundBank.PlayCue("stageup");
soundBank.PlayCue("button_select");
```

Playing Background Sound with XACT

- Play background sound

```
Cue cue;
// Play the background sound.
cue = soundBank.GetCue("mureung");
cue.Play();
if (keyboardState.IsKeyDown(Keys.Left))
                cue.Stop(AudioStopOptions.AsAuthored);
else if (keyboardState.IsKeyDown(Keys.Right)) {
        if (cue.IsPaused) cue.Resume();
        else if (cue.IsPlaying) cue.Pause();
        else {
                // If stopped, create a new cue.
                cue = soundBank.GetCue(cue.Name);
                cue.Play();
        }
}
```