Lecture 10: A Walkthrough of an XNA 2D game

Prof. Hsien-Hsin Sean Lee
School of Electrical and Computer Engineering
Georgia Institute of Technology
2D Games

• Using “Sprites”
  – All textures
  – Simple to make or obtain

• Early games before 3D revolution
  – Space Invaders, Lode Runner, Donkey Kong, Pac-Man

• Do not require high performance accelerators

• Simple enough for your grandparents to enjoy

• Easy to do using XNA framework

• No “effect” (.fx) used
My Game: GoogleShower

• Shoot the bad dawgs!

• Consist of four main objects
  – The shooter (ship.cs)
  – The bad dawgs (meteros.cs)
  – The missile (missile.cs)
  – Music (AudioComponent.cs)

• The moving objects are all made of sprites

Adapted from Chapter 3 of XNA 2.0 Game Programming
Screen Shot
Demo GoogleShower Game Example
Drawing Sprite Using XNA

- Use “texture”
- Store with XNA’s `texture2D` class
- Include new texture images into the Content Pipeline
- Use “Content.Load” to associate texture variables
Adding Game Component (C# source file)

- Adding new `DrawableGameComponent` class of objects in the game
- Project → Add Components
Full Screen Mode

```csharp
private int right_margin;
private int score;
private AudioComponent audioComponent;
private SpriteFont gamefont;

public Game1()
{
    graphics = new GraphicsDeviceManager(this);
    Content.RootDirectory = "Content";

    // for running at Full Screen mode
    graphics.PreferredBackBufferWidth = 1024;
    graphics.PreferredBackBufferHeight = 768;
    graphics.IsFullScreen = true;
}

/// <summary>
/// Allows the game to perform any initialization it needs to before starting to run.
/// This is where it can query for any required services and load any non-graphic
/// related content. Calling base.Initialize will enumerate through any components
/// and initialize them as well.
/// </summary>
protected override void Initialize()
{
    // TODO: Add your initialization logic here
```
Game Services

- Game services maintain loose coupling between objects that need to interact with each other
- Register a “global” `SpriteBatch` for drawing all sprites
- `Draw()` method will look for an active `SpriteBatch` in GameServices
- All `GameComponents` will use this `SpriteBatch`

```csharp
public override void Draw(GameTime gameTime)
{
    SpriteBatch sBatch = (SpriteBatch)Game.Services.GetService(typeof(SpriteBatch));
    sBatch.Draw(texture, position, spriteRectangle, Color.White);
    base.Draw(gameTime);
}
```

```
// Create a new SpriteBatch, which can be used to draw textures.
spriteBatch = new SpriteBatch(GraphicsDevice);
Services.AddService(typeof(SpriteBatch), spriteBatch);
```

**Registering a Game Service In LoadContent()**

**Use GetService to acquire service**
Drawing Background in One Pass

```csharp
base.Update(gameTime);

protected override void Draw(GameTime gameTime)
{
    graphics.GraphicsDevice.Clear(Color.CornflowerBlue);

    spriteBatch.Begin();
    spriteBatch.DrawString(gameFont, "Bad Dawgs: " + rockCount.ToString(), new Vector2(16, 16), Color.Red);
    spriteBatch.DrawString(gameFont, "Bad Dawgs: " + rockCount.ToString(), new Vector2(17, 17), Color.Cyan);
    spriteBatch.DrawString(gameFont, "Bad Dawgs: " + rockCount.ToString(), new Vector2(18, 18), Color.Cyan);
    spriteBatch.DrawString(gameFont, "Highest Dog Count: " + highest_rockCount.ToString(), new Vector2(right_margin, 16), Color.Red);
    spriteBatch.DrawString(gameFont, "Highest Dog Count: " + highest_rockCount.ToString(), new Vector2(right_margin + 1, 17), Color.Yellow);
    spriteBatch.DrawString(gameFont, "Highest Dog Count: " + highest_rockCount.ToString(), new Vector2(right_margin + 2, 18), Color.Yellow);
    spriteBatch.DrawString(gameFont, "Score: " + score.ToString(), new Vector2(250, 16), Color.Red);
    spriteBatch.DrawString(gameFont, "Score: " + score.ToString(), new Vector2(250, 17), Color.White);
    spriteBatch.DrawString(gameFont, "Score: " + score.ToString(), new Vector2(250, 18), Color.White);

    spriteBatch.End();
    spriteBatch.Begin(SpriteBlendMode.AlphaBlend);
    base.Draw(gameTime);
    spriteBatch.End();
}
```
DrawString (Scoreboard)

- Draw using **SpriteBatch**
- Create a font sprite
- Based on available Fonts in the system
- Add font in **LoadContent()**
Manage Components

- Components: member of GameComponentCollection (i.e., Microsoft.XNA.Framework.Game)
- Use `Components.Add()` to add a new component to the list

```csharp
protected override void Initialize()
{
    audioComponent = new AudioComponent(this);
    Components.Add(audioComponent);
}
```

```csharp
private void Start()
{
    if (player == null)
    {
        player = new ship(this, ref leeTexture);
        Components.Add(player);
    }

    if (player.shootMissile())
    {
        audioComponent.PlayCue("shoot");
        Components.Add(new missile(this, ref missileTexture, player.GetPosition()));
        player.resetShoot();
    }
}
```
Game Logic

```csharp
private void DoGameLogic()
{
    bool hasCollision = false;
    Rectangle shipRectangle = player.GetBounds();
    foreach (GameComponent gc in Components)
    {
        if (gc is meteors)
        {
            hasCollision = ((meteors)gc).CheckCollision(shipRectangle);
            if (hasCollision)
            {
                audioComponent.PlayCue("missile");
                score -= PENALTY;
                dead++;
                RemoveAllMeteors();
                Start();
                break;
            }
        }
    }
    CheckforNewMeteor();
    CheckMissileFired();
    CheckMissileHit();
    AdvanceLevel();
}
```

- Embedded inside `Update()` function
- Logistics check
- Check collision
- Check if a missile is fired
- Check if should advance levels
Pause the Game

• Poll the keyboard state
• Simply do **not** perform any update inside the `Update()`

```csharp
if (!pause && keyboard.IsKeyDown(Keys.Tab))
{
    pause = true;
}
else if (pause && keyboard.IsKeyDown(Keys.LeftControl))
{
    pause = false;
}
if (pause == false)
{
    DoGameLogic();
    base.Update(gameTime);
}
```
Built-in Test for Collision Detection

- Test bounding boxes of given rectangular sprites
- Return a boolean result

```csharp
public bool CheckCollision(Rectangle rect)
{
    Rectangle spriterect = new Rectangle((int)position.X, (int)position.Y,
                                           MISSILEWIDTH, MISSILEHEIGHT);
    return spriterect.Intersects(rect);
}
```

- BoundingSphere also provides similar method

```csharp
public BoundingSphere(
    Vector3 center,
    float radius
)

public bool Intersects(
    BoundingSphere sphere
)
```
XNA Game Audio Component

- Use “Content pipeline” again
- Use Microsoft Cross-Platform Audio Creation Tool (or XACT) in XNA Game Studio 2.0
XNA Game Audio Component Cont’d

• Create wave banks and sound banks
• Compile them into XAP file used by the content manager

“checked” for looping background music
XNA Game Audio Component Cont’d

• Create a new `GameComponent` for audio
• Initialize WaveBank and SoundBank in the C# code

AudioEngine object is the program reference to the audio services in your machine provided by XNA

```csharp
public AudioComponent(Game game) : base(game)
{
    // TODO: Construct any child components here
}

/// <summary>
/// Allows the game component to perform any initialization it needs to before starting to run. This is where it can query for any required services and load content.
/// </summary>
public override void Initialize()
{
    // TODO: Add your initialization code here
    audioEngine = new AudioEngine("Content\GoogleShower.xgs");
    waveBank = new WaveBank(audioEngine, "Content\Wave Bank.xwb");
    if (waveBank != null)
    {
        soundBank = new SoundBank(audioEngine, "Content\Sound Bank.xsb");
    }
    base.Initialize();
}
```

This “Content” directory is located in `<Proj>/bin/x86/<Sln Config>/Content`
Background Looping Music

- Add in Initialize code of the Game
- PlayCue is a method of SoundBank
Play Sound On Event

- Shoot.wav in SoundBank was not set to “infinite”, thus will only be played once
Game Over

- Remove all components
- Replace background canvas
- Pause the music

```csharp
private void GameOver()
{
    if (dead >= DEAD_COUNT)
    {
        highestscore = score;
        RemoveGame();
        backgroundTexture = gameOverTexture;
        gameover = true;
        player = null;
        backmusic.Pause();
    }
}
```
Organized Game Structure

Class GameScene : DrawableGameComponent

PlayScene

EndScene

Play Again?

SoundBank

Play Again?