

ECE4893A/CS4803MPG:

MULTICORE AND GPU PROGRAMMING FOR VIDEO GAMES

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

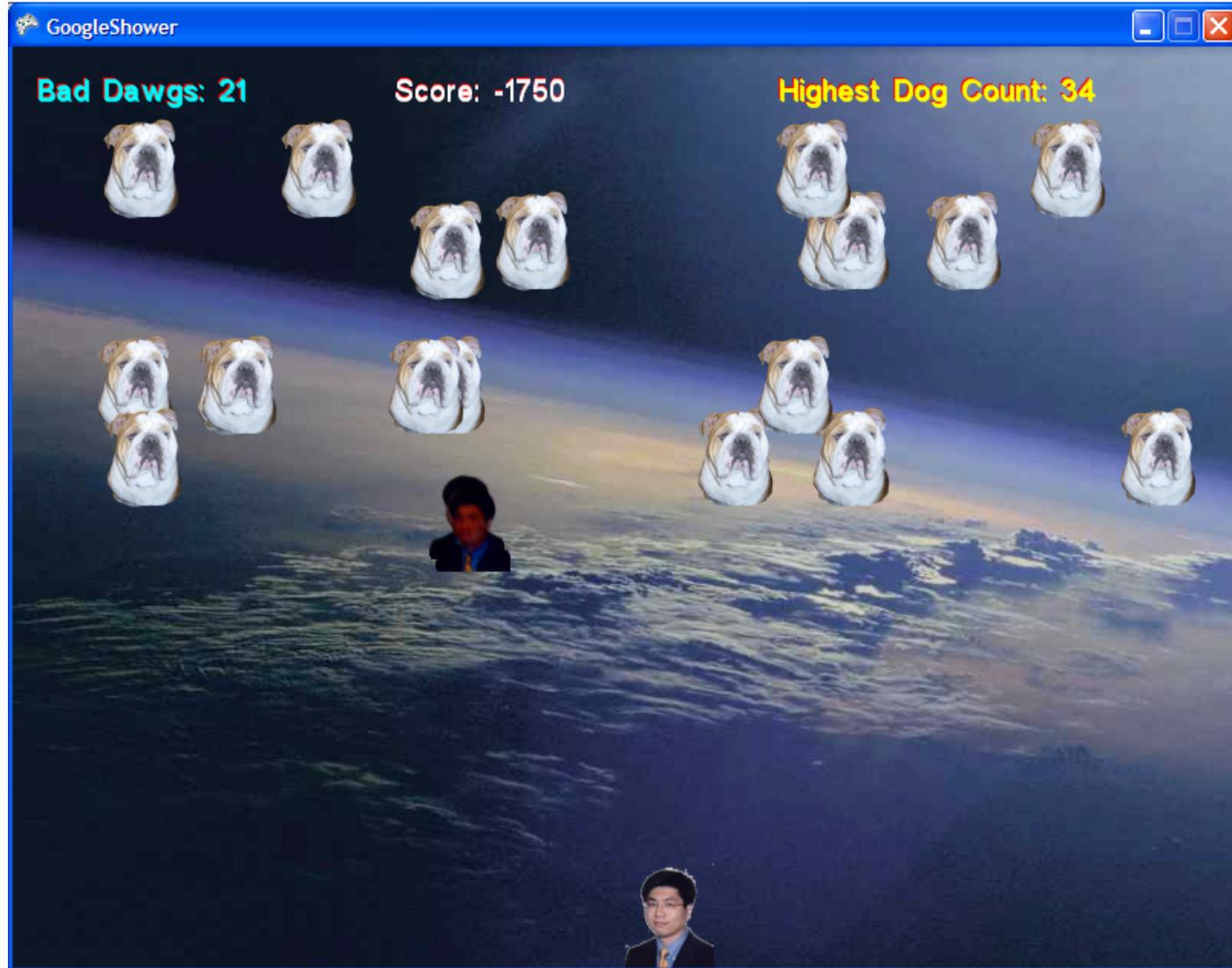
- Using “Sprites”
 - All textures
 - Simple to make or obtain
- Early games before 3D revolution
 - Space Invaders, Lode Runner, Donkey Kong, Pac-Man
- Do not require high performance accelerators
- Simple enough for your grandparents to enjoy

- 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

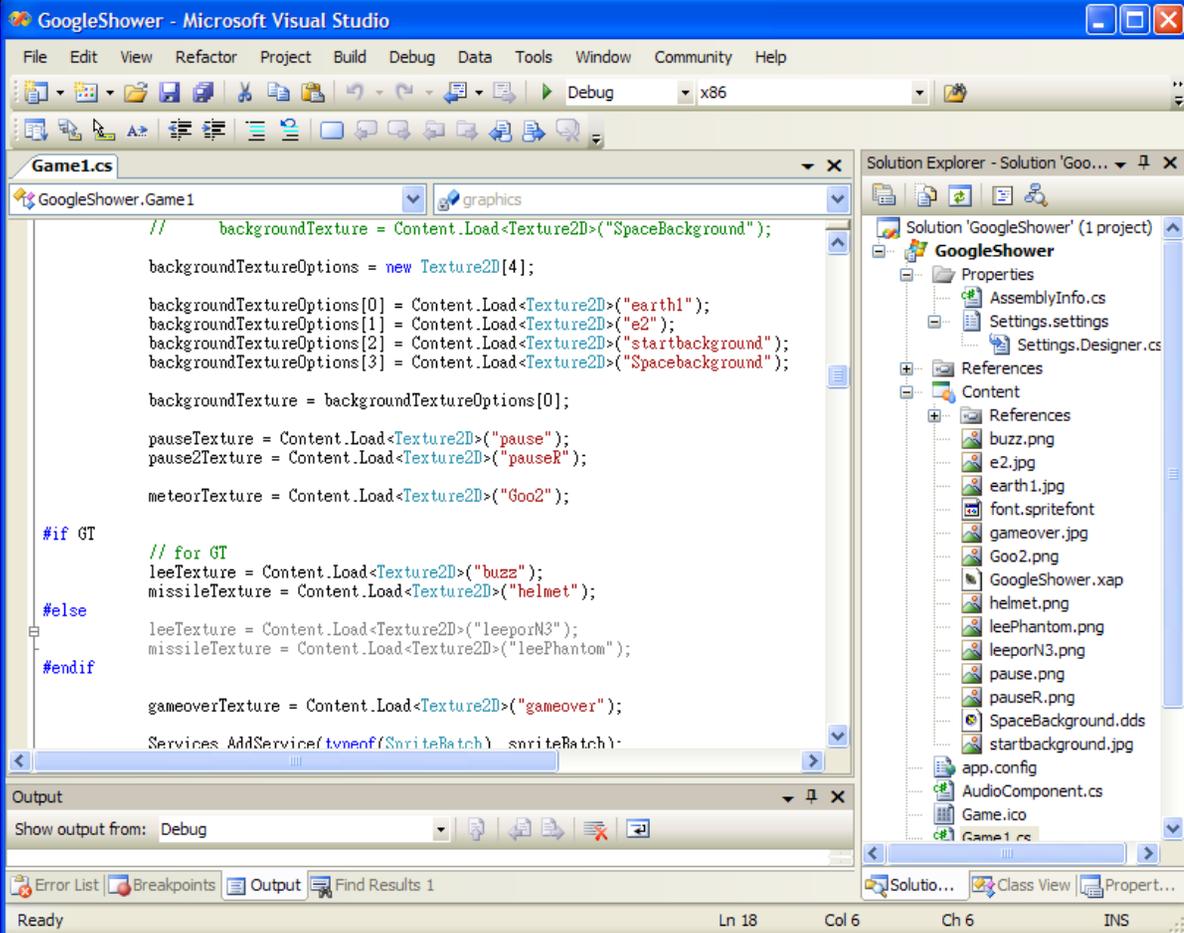
Screen Shot



Demo GoogleShower Game Example

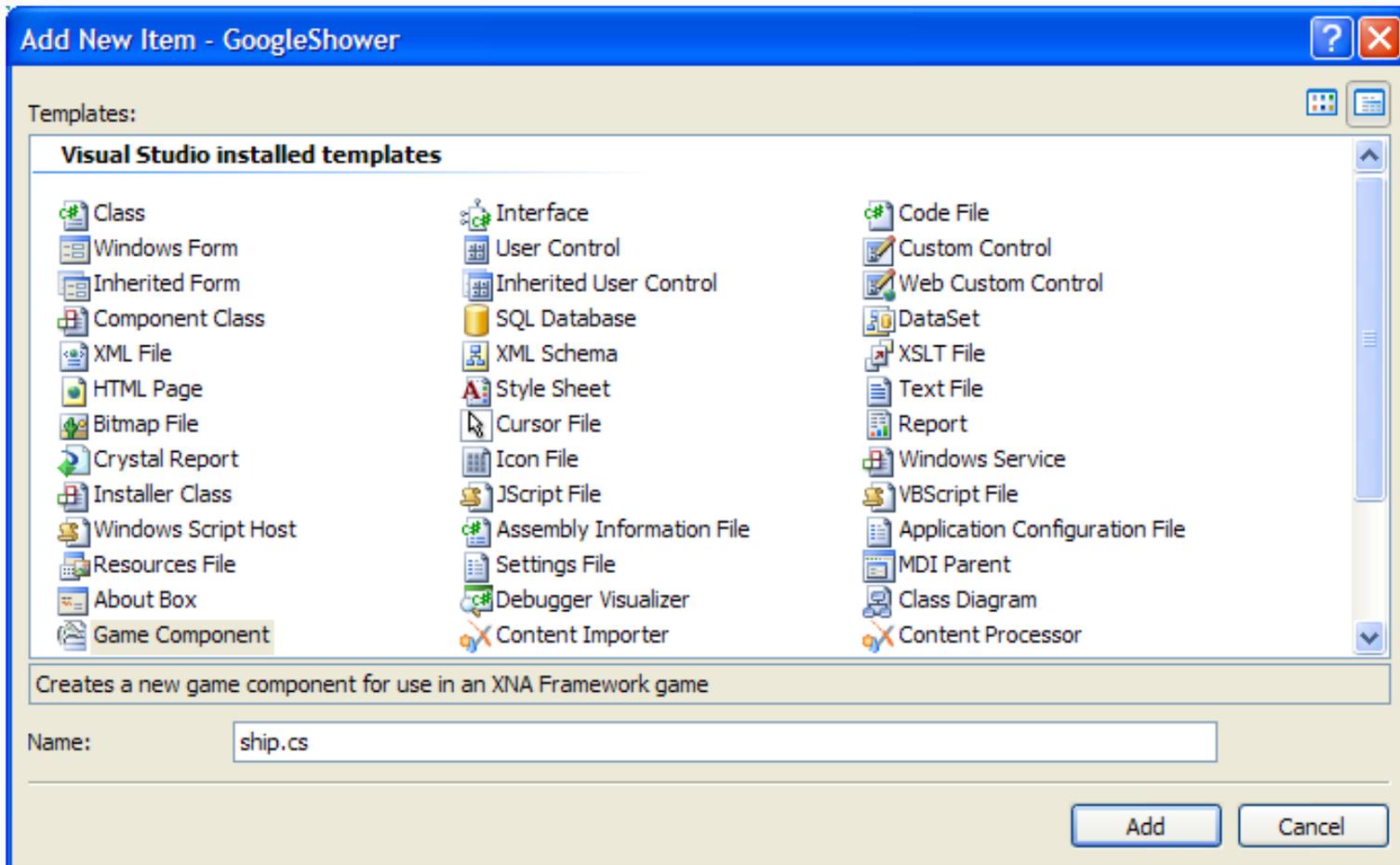
Drawing Sprite Using XNA

- Use “texture”
- Store with XNA’s texture2D class
- Include new texture images into the Content Pipeline
- Use “Content.Load” to associate texture variables



```
GoogleShower - Microsoft Visual Studio
File Edit View Refactor Project Build Debug Data Tools Window Community Help
Debug x86
Game1.cs
GoogleShower.Game1
graphics
// backgroundTexture = Content.Load<Texture2D>("SpaceBackground");
backgroundTextureOptions = new Texture2D[4];
backgroundTextureOptions[0] = Content.Load<Texture2D>("earth1");
backgroundTextureOptions[1] = Content.Load<Texture2D>("e2");
backgroundTextureOptions[2] = Content.Load<Texture2D>("startbackground");
backgroundTextureOptions[3] = Content.Load<Texture2D>("Spacebackground");
backgroundTexture = backgroundTextureOptions[0];
pauseTexture = Content.Load<Texture2D>("pause");
pause2Texture = Content.Load<Texture2D>("pauseR");
meteorTexture = Content.Load<Texture2D>("Goo2");
#if GT
// for GT
leeTexture = Content.Load<Texture2D>("buzz");
missileTexture = Content.Load<Texture2D>("helmet");
#else
leeTexture = Content.Load<Texture2D>("leeporN3");
missileTexture = Content.Load<Texture2D>("leePhantom");
#endif
gameoverTexture = Content.Load<Texture2D>("gameover");
Services.AddService(typeof(SpriteBatch), spriteBatch);
Output
Show output from: Debug
Error List Breakpoints Output Find Results 1
Ready Ln 18 Col 6 Ch 6 INS
```

Adding Game Component (C# source file)



- Adding new [DrawableGameComponent](#) class of objects in the game
- Project → Add Components

Full Screen Mode

```
GoogleShower.Game1 Game1()
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`

```
// 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()`

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

Use `GetService` to acquire service

Drawing Background in One Pass

```
hower.Game1
    base.Update(gameTime);
}

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

    spriteBatch.Begin();
    spriteBatch.Draw(backgroundTexture, new Rectangle(0, 0, graphics.GraphicsDevice.DisplayMode.Width,
        graphics.GraphicsDevice.DisplayMode.Height),
        Color.LightGray);

    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);

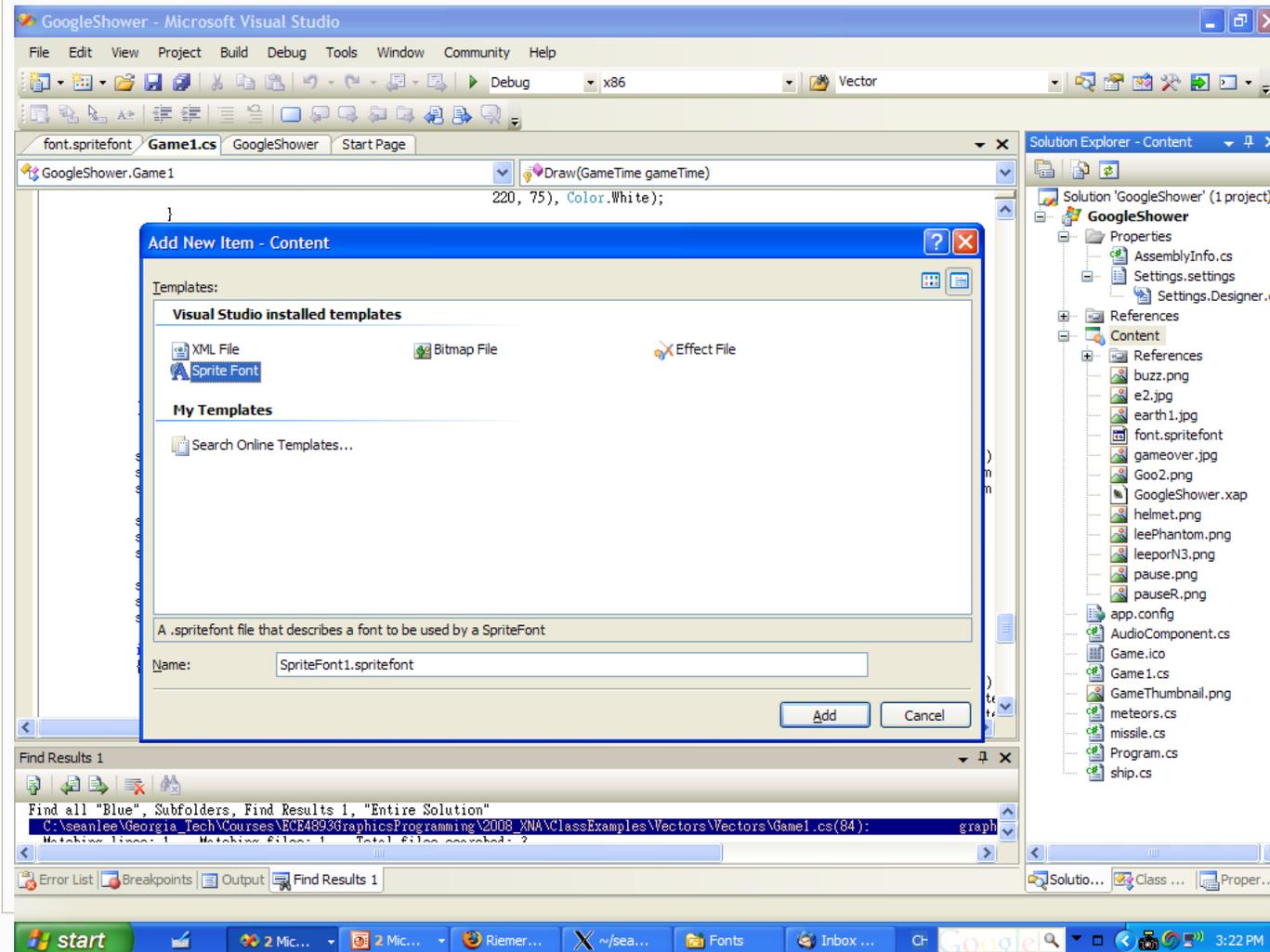
    spriteBatch.End();

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

Draw other
sprites

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

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

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

```
private void CheckMissileFired()
{
    // add component is SPACE bar was hit
    if (player.shootMissile())
    {
        audioComponent.PlayCue("shoot");
        Components.Add(new missile(this, ref missileTexture, player.GetPosition()));
        player.resetShoot();
    }
}
```

- `Components.RemoveAt(j)` removes j^{th} component

```
if (hasCollision)
{
    audioComponent.PlayCue("explosion");
    // remove collided meteros (Google)
    score++;
    Components.RemoveAt(j);
}
```

Game Logic

```
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()`

```
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

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

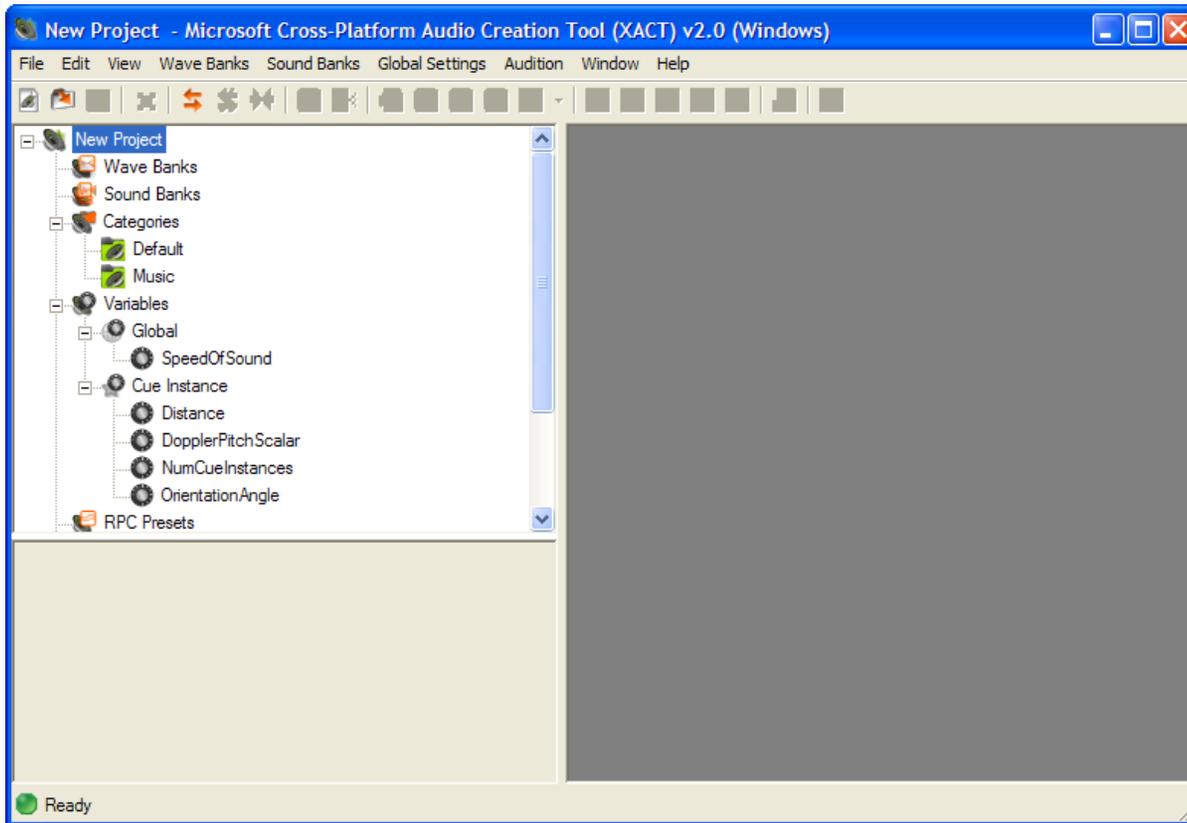
- BoundingSphere also provides similar method

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

```
public bool Intersects (
    BoundingSphere sphere
)
```

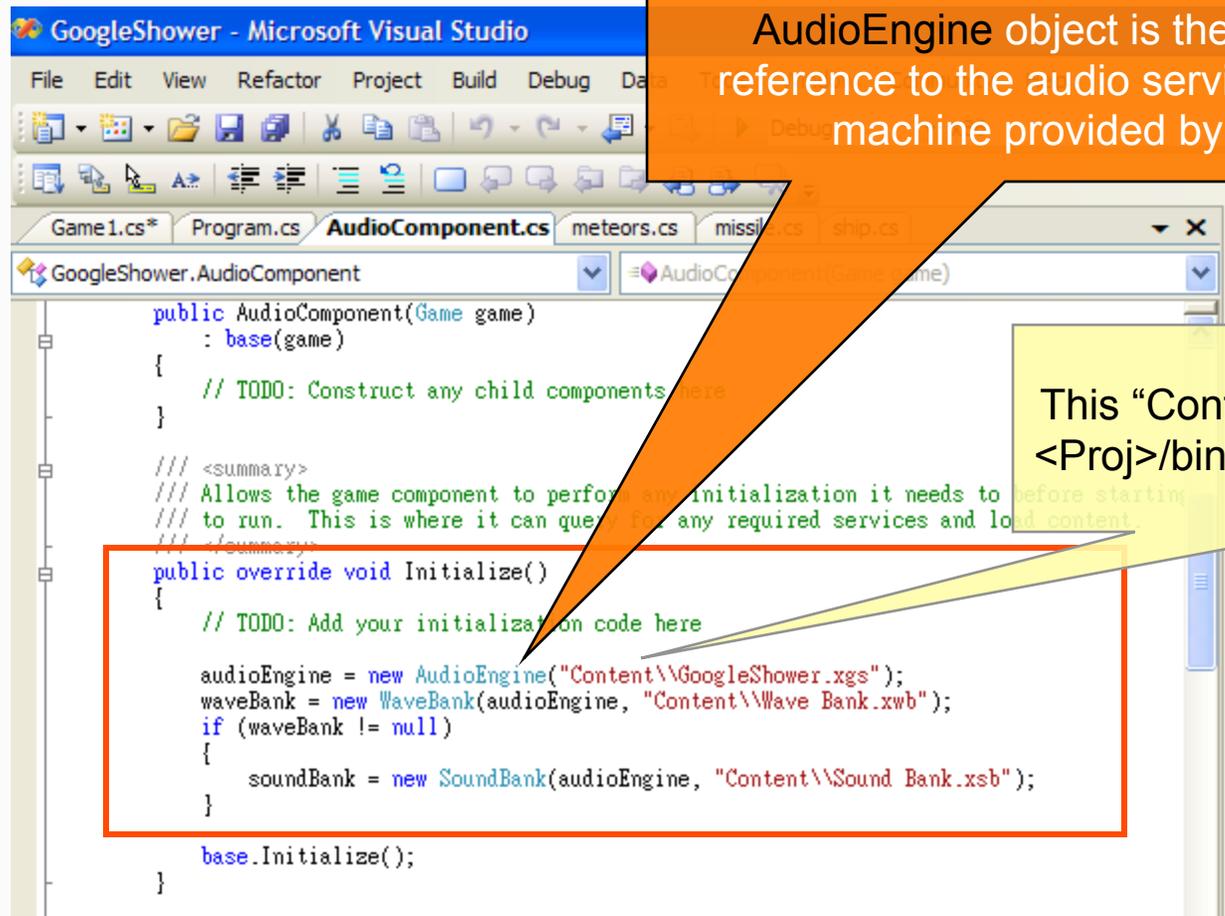
XNA Game Audio Component

- 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 a new **GameComponent** for audio
- Initialize WaveBank and SoundBank in the C# code



```
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");
    }

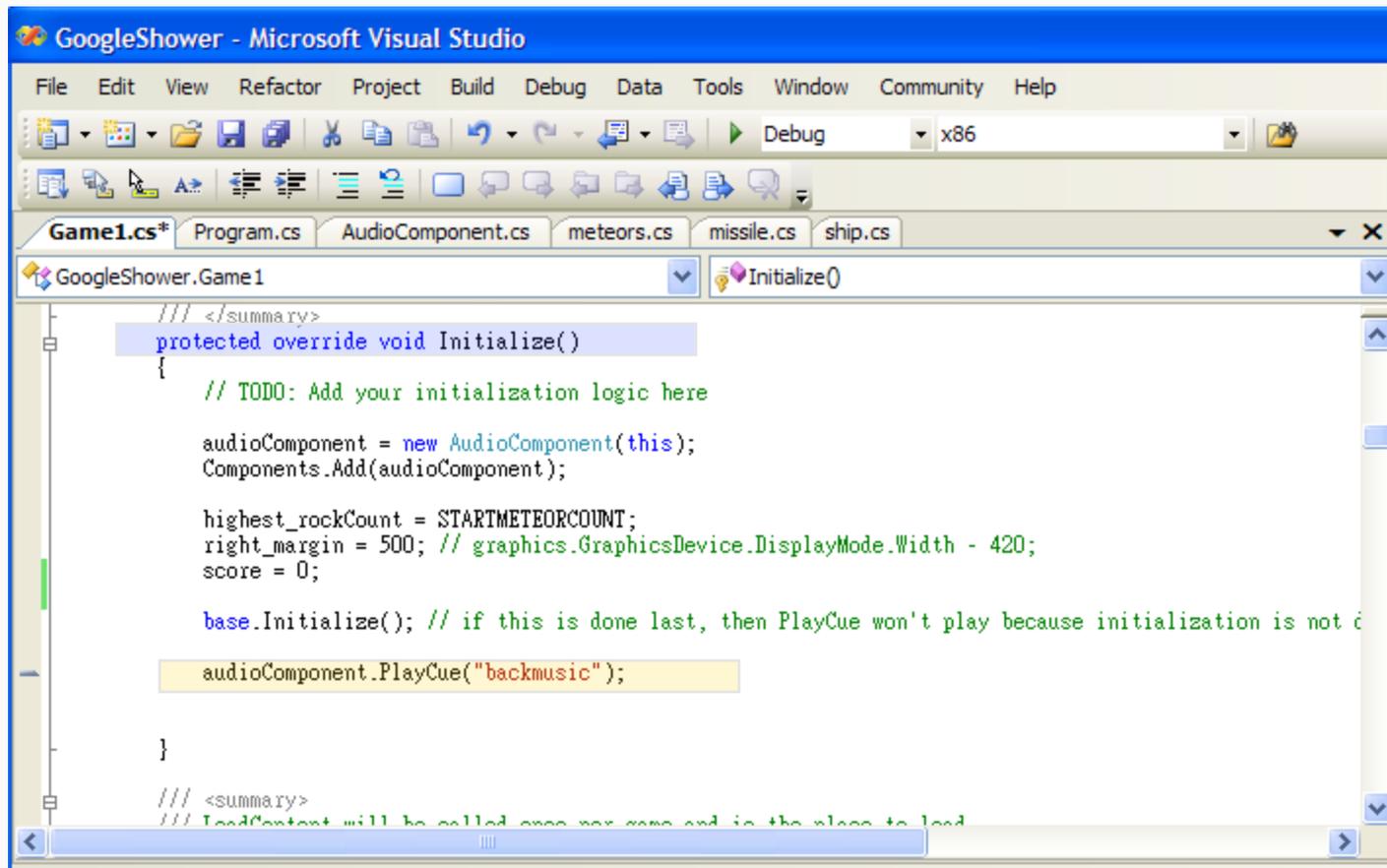
    base.Initialize();
}
```

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

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



The screenshot shows the Microsoft Visual Studio IDE with the following details:

- Title Bar:** GoogleShower - Microsoft Visual Studio
- Menu Bar:** File, Edit, View, Refactor, Project, Build, Debug, Data, Tools, Window, Community, Help
- Toolbar:** Includes icons for File Explorer, Solution Explorer, Code View, and a Debug button.
- Toolbox:** Shows a 'Debug' dropdown menu and 'x86' architecture.
- Tab Bar:** Displays several open files: Game1.cs*, Program.cs, AudioComponent.cs, meteors.cs, missile.cs, and ship.cs.
- Class Explorer:** Shows 'GoogleShower.Game1' with a dropdown menu for 'Initialize()'. A small icon indicates a warning or error.
- Code Editor:** Contains the following C# code:

```
/// </summary>
protected override void Initialize()
{
    // TODO: Add your initialization logic here

    audioComponent = new AudioComponent(this);
    Components.Add(audioComponent);

    highest_rockCount = STARTMETEORCOUNT;
    right_margin = 500; // graphics.GraphicsDevice.DisplayMode.Width - 420;
    score = 0;

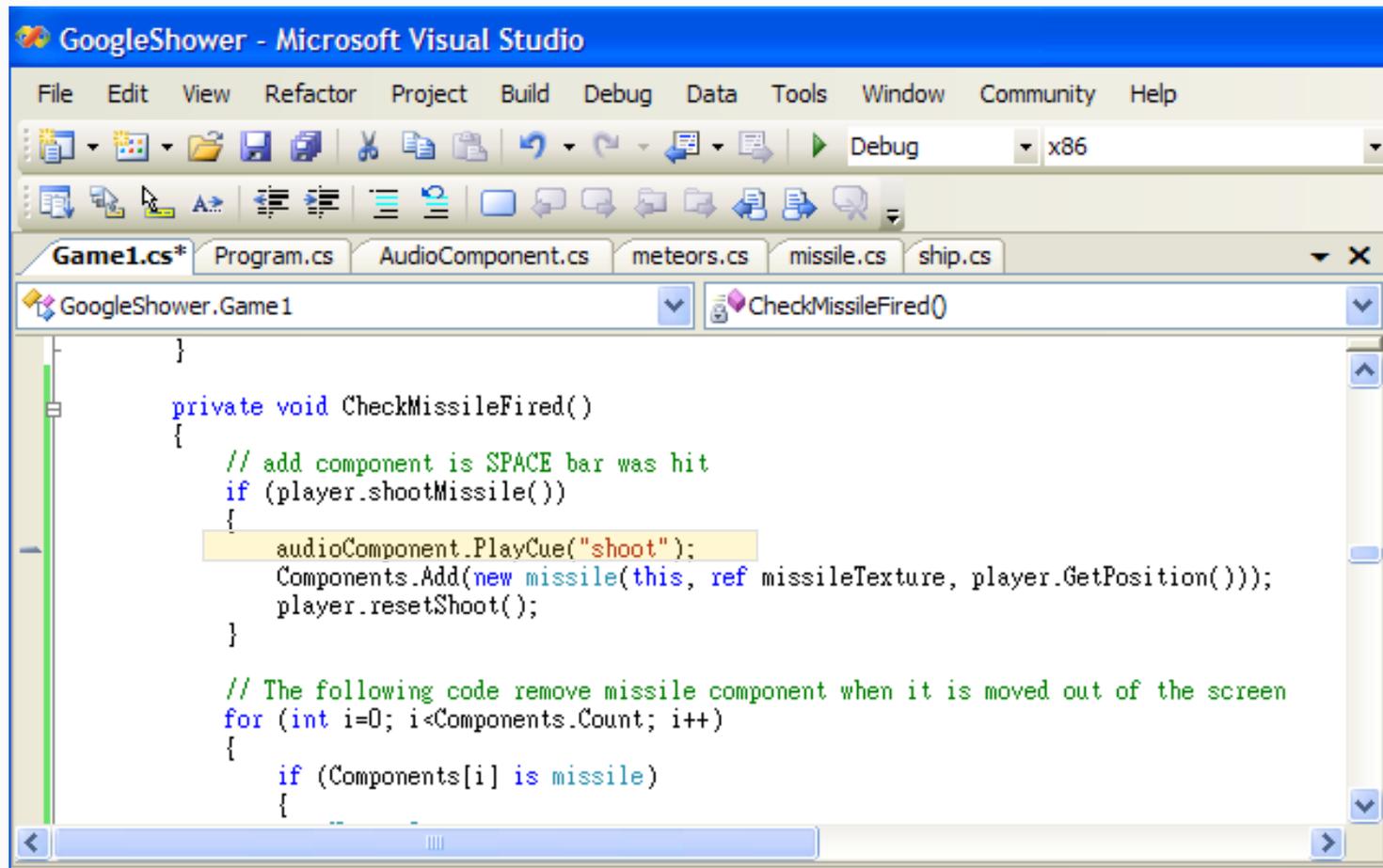
    base.Initialize(); // if this is done last, then PlayCue won't play because initialization is not c

    audioComponent.PlayCue("backmusic");
}

/// <summary>
/// LoadContent will be called once per game and is the place to load
```

Play Sound On Event

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



```
GoogleShower - Microsoft Visual Studio
File Edit View Refactor Project Build Debug Data Tools Window Community Help
Debug x86
Game1.cs* Program.cs AudioComponent.cs meteors.cs missile.cs ship.cs
GoogleShower.Game1 CheckMissileFired()
}
private void CheckMissileFired()
{
    // add component is SPACE bar was hit
    if (player.shootMissile())
    {
        audioComponent.PlayCue("shoot");
        Components.Add(new missile(this, ref missileTexture, player.GetPosition()));
        player.resetShoot();
    }

    // The following code remove missile component when it is moved out of the screen
    for (int i=0; i<Components.Count; i++)
    {
        if (Components[i] is missile)
        {
```

Game Over

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



```
private void RemoveGame()
{
    for (int i = 0; i < Components.Count; i++)
    {
        Components.RemoveAt(i);
        i--;
    }
}

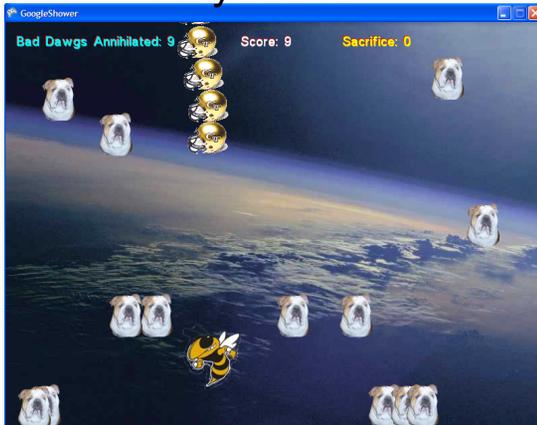
private void GameOver()
{
    if (dead >= DEAD_COUNT)
    {
        if (highestscore < score)
            highestscore = score;

        RemoveGame();
        backgroundTexture = gameoverTexture;
        gameover = true;
        player = null;
        backmusic.Pause();
    }
}
```

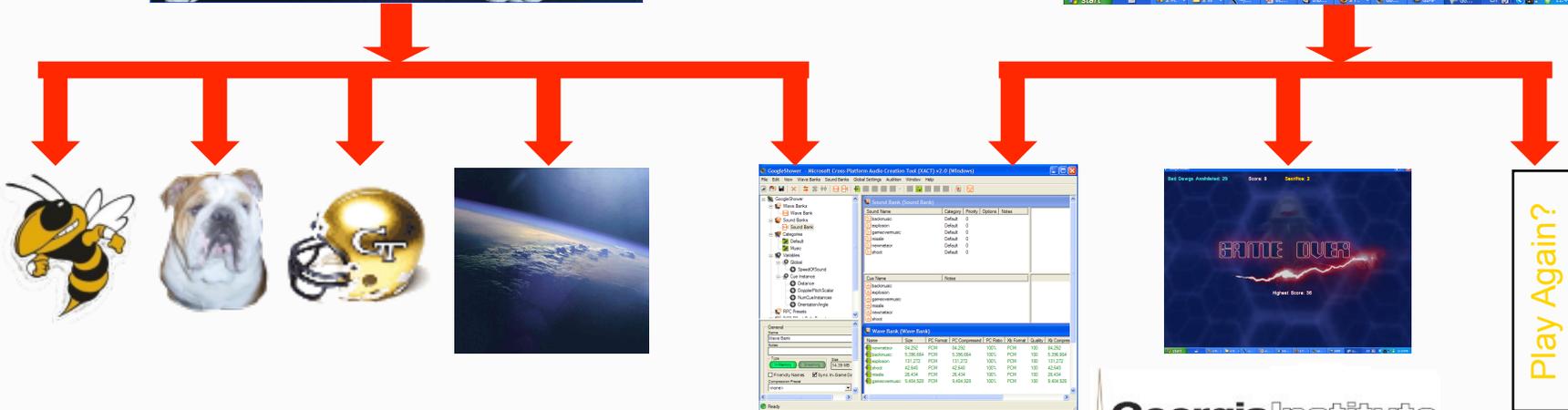
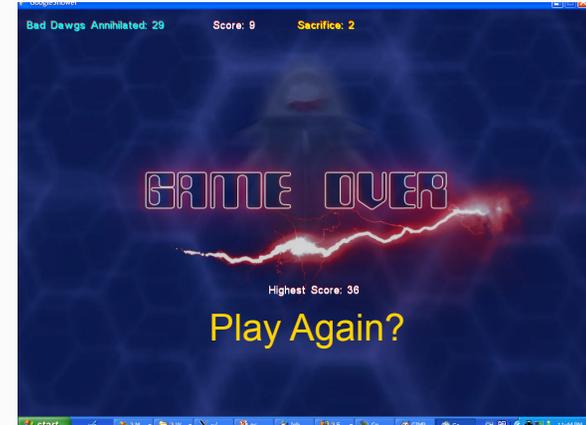
Organized Game Structure

Class GameScene : DrawableGameComponent

PlayScene



EndScene



SoundBank