Ch 3: Coding Tidbits and Style that Will Save You
Ch 4: Building your Game

Quiz #2
Discussion
Developing a Game

Game Architecture

Resources:
Chapters 2&3 (Learning XNA 4.0)
A project for creating an XNA Framework 4.0 Windows game.
using System;

namespace Testing1
{
    #if WINDOWS || XBOX
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        static void Main(string[] args)
        {
            using (Game1 game = new Game1())
            {
                game.Run();
            }
        }
    }
    #endif
}

using System;
using System.Collections.Generic;
using System.Linq;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Media;

namespace Testing1
{
    /// <summary>
    /// This is the main type for your game
    /// </summary>
    public class Game1 : Microsoft.Xna.Framework.Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;

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

        /// <summary>
        /// In this method you can add the logic that controls how the Game
        /// class works. The Create method is called when the game should start pr
        /// 
        /// </summary>
        /// <param name="gameTime">A reference to the System.TimeG
        /// 
        /// </param>
        protected override void OnCreateGame(GameTime gameTime)
        {
        }

        /// <summary>
        /// In this method you can add the logic that should be executed eac
        /// 
        /// </summary>
        /// <param name="gameTime">A reference to the System.TimeG
        /// 
        /// </param>
        protected override void Update(GameTime gameTime)
        {
        }

        /// <summary>
        /// This method will be called when Wee
        /// 
        /// </summary>
        /// <param name="gameTime"/>
protected override void Initialize()
{
    // TODO: Add your initialization logic here
    base.Initialize();
}

/// <summary>
/// LoadContent will be called once per game and is the place to load
/// all of your content.
/// </summary>
protected override void LoadContent()
{
    // Create a new SpriteBatch, which can be used to draw textures.
    spriteBatch = new SpriteBatch(GraphicsDevice);

    // TODO: use this.Content to load your game content here
}

/// <summary>
/// UnloadContent will be called once per game and is the place to unload
/// all content.
/// </summary>
protected override void UnloadContent()
{
    // TODO: Unload any non ContentManager content here
}
/// checking for collisions, gathering input, and playing audio.
/// <param name="gameTime">Provides a snapshot of timing values.</param>
protected override void Update(GameTime gameTime)
{
    // Allows the game to exit
        this.Exit();

    // TODO: Add your update logic here

    base.Update(gameTime);
}

/// <summary>
/// This is called when the game should draw itself.
/// </summary>
/// <param name="gameTime">Provides a snapshot of timing values.</param>
protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.CornflowerBlue);

    // TODO: Add your drawing code here

    base.Draw(gameTime);
}
Tutorial moving a 2D sprite on the screen
In Class assignment

 dévelop a game in XNA (2D) called catch the sprite. The objective of the user is to click on the sprite. Each time the user clicks on the sprite, he gains one score point.

Your game should:

- render a sprite
- the sprite should move randomly in a direction
- the sprite should bounce off the screen boundaries
- once clicked score is added
- show the score to the user
Tutorial moving a 2D sprite on the screen
Adding a Sprite

```csharp
namespace ClassAssign1
{
    /// <summary>
    /// This is the main type for your game.
    /// </summary>
    public class Game1 : Microsoft.Xna.Graphics.Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;

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

        /// <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-visual resources
        /// related to the specific platform or game mode being played. Calling base.Initialize will
        /// enumerate through any components and initialize them as well.
        /// </summary>
        protected override void Initialize()
        {
            // TODO: Add your initialization logic here
            base.Initialize();
        }

        /// <summary>
        /// This function will be called once per game and is the place to load
        ```
Loading a Sprite

```csharp
/// <summary>
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;

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

    /// <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-visual 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

        base.Initialize();
    }

    /// <summary>
    /// LoadContent will be called once per game and is the place to load all of your content.
    /// </summary>
    protected override void LoadContent()
    {
        // Create a new SpriteBatch, which can be used to draw textures.
        spriteBatch = new SpriteBatch(GraphicsDevice);
        //loading sprite
        mysprite = Content.Load<Texture2D>("AlphaSprite");
    }

```
Drawing a Sprite

```csharp
{  
    // Allows the game to exit
        this.Exit();

    // TODO: Add your update logic here
    base.Update(gameTime);
    
    /// <summary>
    /// This is called when the game should draw itself.
    /// </summary>
    /// <param name="gameTime">Provides a snapshot of timing values.</param>
    protected override void Draw(GameTime gameTime)  
    {  
        GraphicsDevice.Clear(Color.CornflowerBlue);

        //Drawing the Sprite
        spriteBatch.Begin();
        spriteBatch.Draw(mysprite, Vector2.Zero, Color.White);
        spriteBatch.End();

        base.Draw(gameTime);
    }
}
Coordinate system in 2D
Moving Sprite in random direction

- Set random Position
- How to translate in space:
  
  New Position =

  Old Position + (direction * Speed)

- Need to continuously update position, where would you put that?
Moving Sprite in random direction

```csharp
using Microsoft.Xna.Framework.Media;

namespace Class1Assign1
{
    /// <summary>
    /// This is the main type for your game.
    /// </summary>
    public class Game1 : Microsoft.Xna.Framework.Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;

        // Adding a sprite
        Texture2D mySprite;

        // position and direction
        Vector2 position;
        Vector2 direction;
        int speed = 5;

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

        /// <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-graphics
        /// related content. Calling base.Initialize will enumerate through any components
        /// and initialize them as well.
        /// </summary>
        protected override void Initialize()
        {
            Random randomdir = new Random();
            int randomdirx = randomdir.Next(1, 10);
            int randomdiry = randomdir.Next(1, 10);
            direction = new Vector2(randomdirx, randomdiry);
            direction.Normalize();
        }
    }
}```
Moving Sprite in random direction

```csharp
base.Initialize();

/// <summary>
/// LoadContent will be called once per game and is the place to load
/// all of your content.
/// </summary>
protected override void LoadContent()
{
    // Create a new SpriteBatch, which can be used to draw textures.
    spriteBatch = new SpriteBatch(GraphicsDevice);
    // loading sprite
    mySprite = Content.Load<Texture2D>("AlphaSprite");

    // Random position
    Random randomvarpos = new Random();
    int randomx = randomvarpos.Next(0, Window.ClientBounds.Width - mySprite.Width);
    int randomy = randomvarpos.Next(0, Window.ClientBounds.Height - mySprite.Height);
    position = new Vector2(randomx, randomy);
}

/// <summary>
/// UnloadContent will be called once per game and is the place to unload
/// all content.
/// </summary>
protected override void UnloadContent()
{
    // TODO: Unload any non-ContentManager content here
}

/// <summary>
/// Allows the game to run logic such as updating the world,
/// checking for collisions, gathering input, and playing audio.
/// </summary>
/// <param name="gameTime">Provides a snapshot of timing values.</param>
protected override void Update(GameTime gameTime)
{
    // Allows the game to exit
        this.Exit();
    // set new position
```
Moving Sprite in random direction

```csharp
protected override void Update(GameTime gameTime)
{
    // Allows the game to exit
        this.Exit();

    // set new position
    position.X += direction.X * speed;
    position.Y += direction.Y * speed;

    base.Update(gameTime);
}
```
Bouncing

- If the position is out of bounds, change direction
Bouncing

Not Ideal, why?

```csharp
if (position.X > (Window.ClientBounds.Width - mySprite.Width)) || (position.X < 0))
    direction.X = -1;
if (position.Y > (Window.ClientBounds.Height - mySprite.Height)) || (position.Y < 0))
    direction.Y = -1;
```
Check Mouse + Add Score

```csharp
//mouse state
MouseState mouseState;

//score
int score = 0;

//Font
SpriteFont font;

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

/// <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()
{
    // set random direction then normalize
    Random randomDir = new Random();
    int randomDirX = randomDir.Next(1, 10);
    int randomDirY = randomDir.Next(1, 10);
}
```
Check Mouse + Add Score
Check Mouse + Add Score

```csharp
if ((position.Y > (Window.ClientBounds.Height - mysprite.Height)) || (position.Y < 0))
direction.Y *= -1;

mouseState = Mouse.GetState();
if (mouseState.LeftButton == ButtonState.Pressed)
{
    if ((mouseState.X > (position.X - (mysprite.Width / 2))) &&
        (mouseState.X < (position.X + (mysprite.Width / 2))) &&
        (mouseState.Y > (position.Y - (mysprite.Height / 2))) &&
        (mouseState.X < (position.X + (mysprite.Height / 2))))
    score += 10;

    spriteBatch.DrawString(font, score.ToString(), Vector2.Zero, Color.White);
}
```
Putting it in the architecture from last class

- View
  - What goes here?
- Logic
  - What goes here?
- Application
  - What goes here?