

# GD2S03

## Advanced Software Engineering & Programming for Games



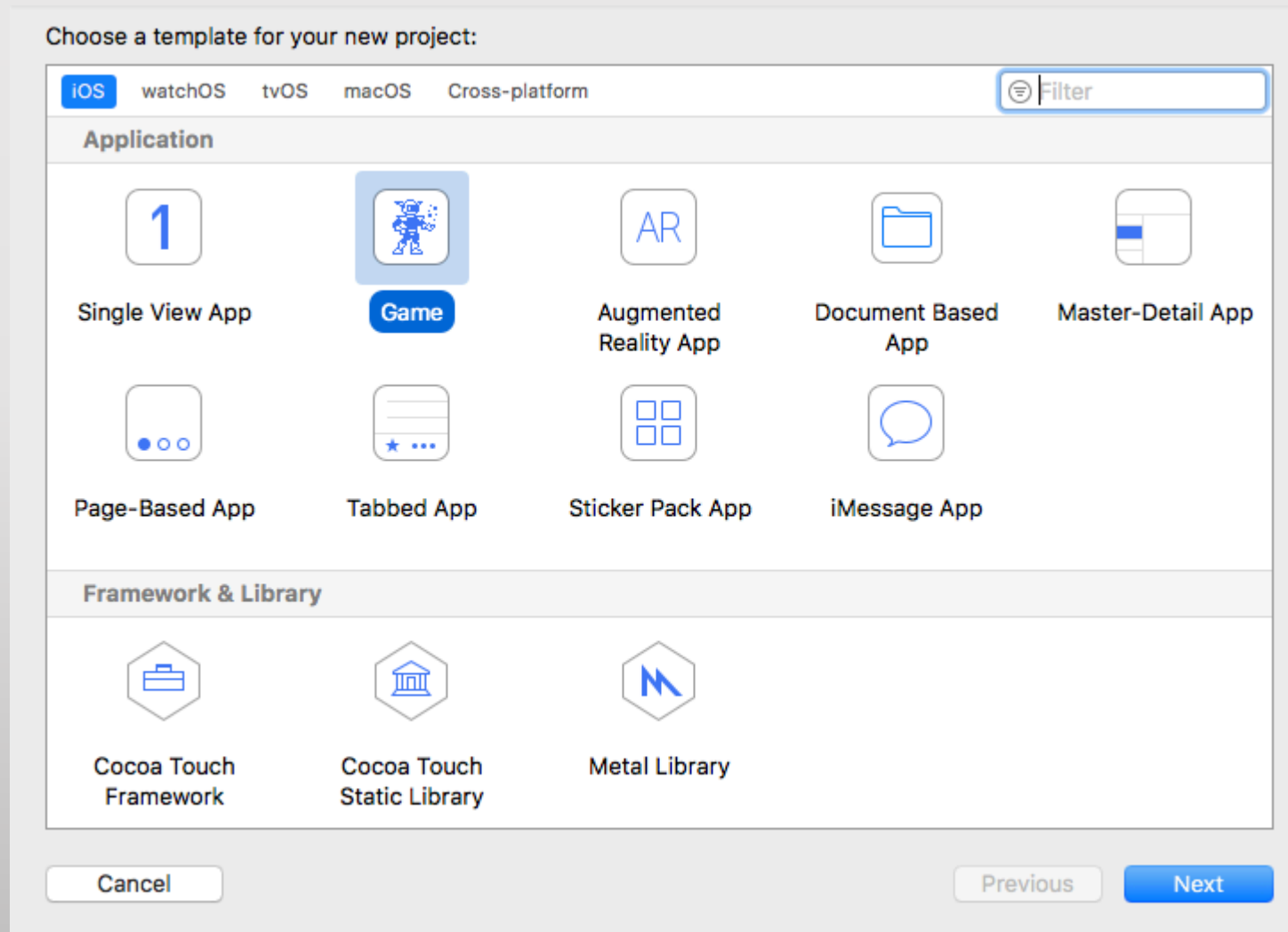
Bachelor of Software Engineering(BSE)  
Game Development

- Overview
  - Project
  - SKView
  - SKNode
  - SKScene
  - SKSpriteNode
  - SKShapeNode
  - SKLabelNode

- Click on Xcode to get the below screen
- Select *“create a new Xcode Project”*



- Select iOS and Game
- Click Next



- Product Name should be the name of your product (e.g SpriteKitDemo)
- Enter *Team*, *Organization Name* and *Organization Identifier* (starts with com)
- Select *Swift* as language and *SpriteKit* as Game Technology
- Click Next

Choose options for your new project:

Product Name:

Team:

Organization Name:

Organization Identifier:

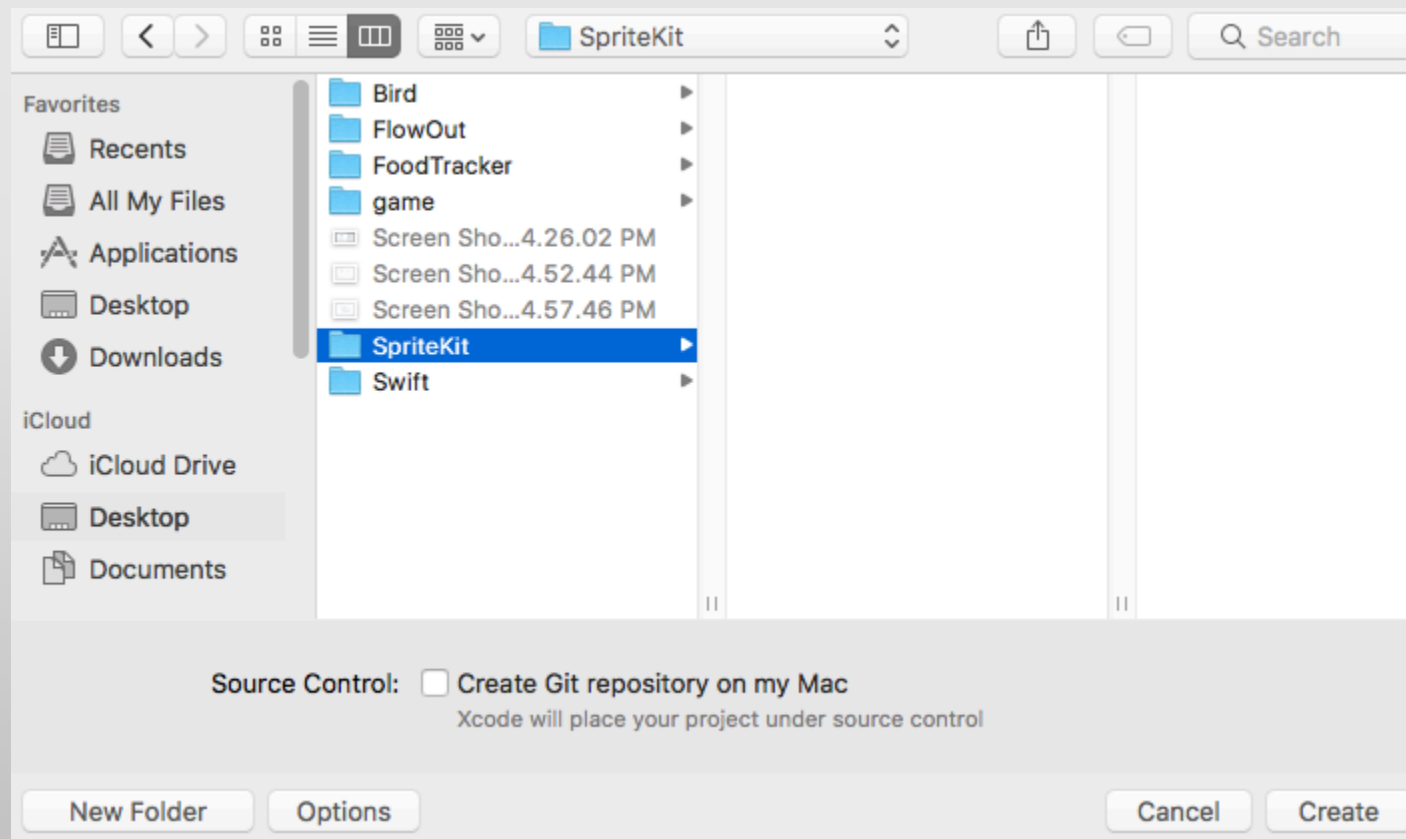
Bundle Identifier:

Language:

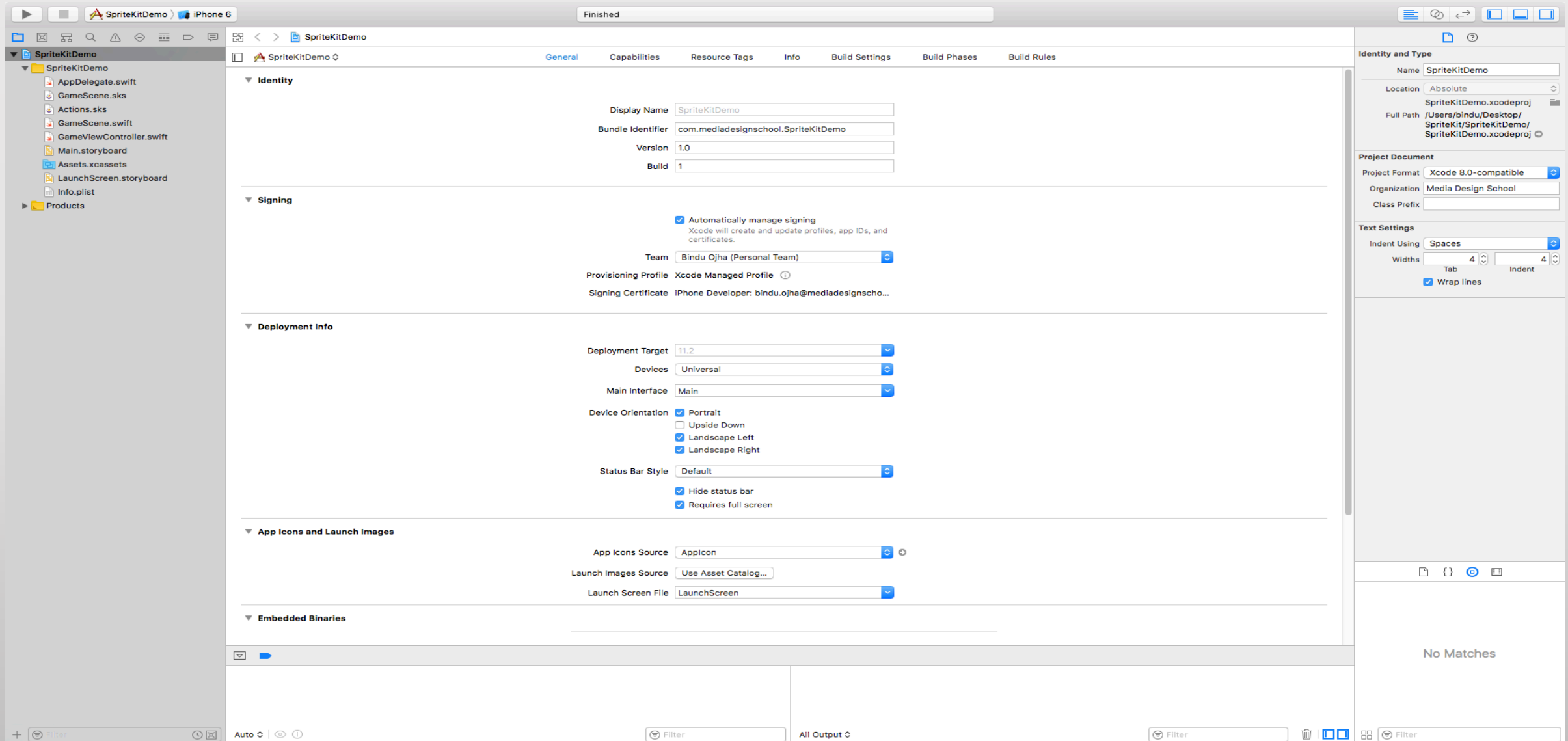
Game Technology: ☒ SpriteKit  
☐ SceneKit  
☐ Metal

☐ Include Unit Tests  
☐ Include UI Tests

- Select the folder where project has to be stored.
- Uncheck *Git repository on my mac* unless you have one
- Click *Create*



- Modify section *signing* and *Deployment Info* as required



The screenshot shows the Xcode interface for a project named 'SpriteKitDemo'. The 'General' tab is selected, displaying various settings for the project. The left sidebar shows the project's file structure, including source files like AppDelegate.swift, GameScene.sks, and GameScene.swift, as well as assets like Main.storyboard and Assets.xcassets.

**Identity Section:**

- Display Name:** SpriteKitDemo
- Bundle Identifier:** com.mediadesignschool.SpriteKitDemo
- Version:** 1.0
- Build:** 1

**Signing Section:**

- ☒ **Automatically manage signing**  
Xcode will create and update profiles, app IDs, and certificates.
- Team:** Bindu Ojha (Personal Team)
- Provisioning Profile:** Xcode Managed Profile
- Signing Certificate:** iPhone Developer: bindu.ojha@mediadesignscho...

**Deployment Info Section:**

- Deployment Target:** 11.2
- Devices:** Universal
- Main Interface:** Main
- Device Orientation:**
  - ☒ Portrait
  - ☐ Upside Down
  - ☒ Landscape Left
  - ☒ Landscape Right
- Status Bar Style:** Default
- ☒ Hide status bar
- ☒ Requires full screen

**App Icons and Launch Images Section:**

- App Icons Source:** Appicon
- Launch Images Source:** Use Asset Catalog...
- Launch Screen File:** LaunchScreen

**Embedded Binaries Section:**

- (Empty list)

**Right Sidebar (Identity and Type):**

- Name:** SpriteKitDemo
- Location:** Absolute
- Full Path:** /Users/bindu/Desktop/SpriteKit/SpriteKitDemo/SpriteKitDemo.xcodeproj
- Project Document:**
  - Project Format:** Xcode 8.0-compatible
  - Organization:** Media Design School
  - Class Prefix:**
- Text Settings:**
  - Indent Using:** Spaces
  - Widths:** Tab: 4, Indent: 4
  - ☒ Wrap lines

**Bottom Panel:**

- Auto:** Auto
- Filter:** Filter
- All Output:** All Output
- Filter:** Filter
- Filter:** Filter

- Delete *GameScene.sks* and *Actions.sks*
- In *GameScene.swift* file, delete everything except function *didMove*
- Delete the body of function *didMove*
- From file *GameViewController.swift*, delete everything except function *viewDidLoad*
- Execute to see the result

```
import UIKit
import SpriteKit

class GameViewController: UIViewController {

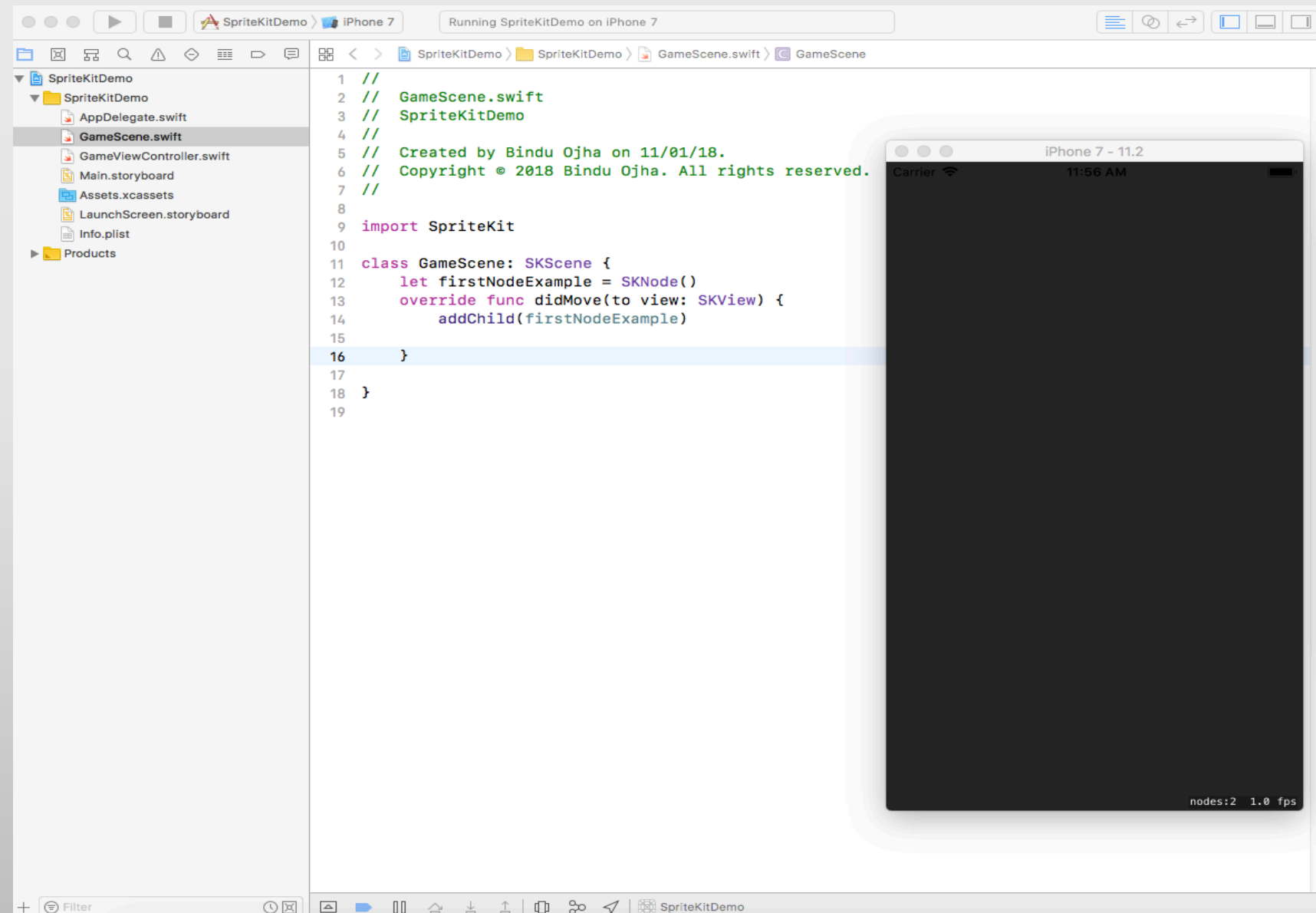
    override func viewDidLoad() {
        super.viewDidLoad()

        if let view = self.view as! SKView? {
            // Load the SKScene from 'GameScene.sks'
            let scene = GameScene(size: view.bounds.size)
            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill
            // Present the scene
            view.presentScene(scene)
            view.ignoresSiblingOrder = true
            view.showsFPS = true
            view.showsNodeCount = true
            view.showsPhysics = true
        }
    }
}
```



- SKView is an object that displays SpriteKit content.
- **SKView** object is used for rendering and animation purpose. A view object is placed inside a window to render its contents.
- Contents in a game are organized into scenes, represented by **SKScene** objects. A scene holds sprites and other contents to be rendered.
- At any given time, a view presents only one scene and **SKTransition** class can be used to animate between two scenes.
- **SKScene** class is inherited from **SKNode** class. Nodes are fundamental building blocks for all contents.
- In a scene, **SKScene** object is used as the root node and all the contents of that scene is represented by tree of node objects (**SKNode**)
- Parent node passes its coordinate system to its descendants' and child's position is specified in the coordinate system defined by its parent.
- A node also applies other properties to its content and the content of its descendants. For example, when a node is rotated, all of its descendants are rotated also.
- The **SKNode** class does not draw anything but the scene (**SKScene** object) implements the per-frame logic and content processing. **SKNode** class apply its properties to its descendants.

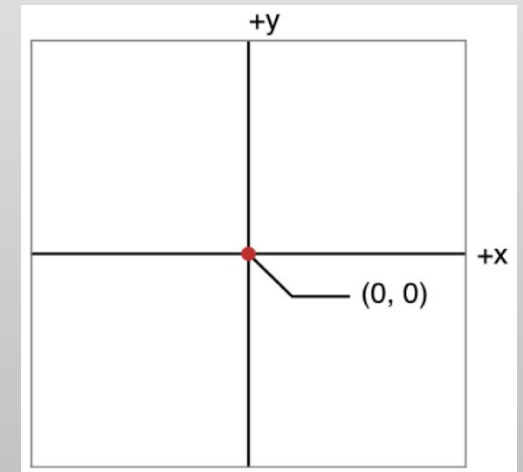
- addChild function is used to add child to a node
- didMove function is called when scene is first loaded
- Node count is 2 as scene itself represents a node.
- Assets must be stored in Assets.xcassets



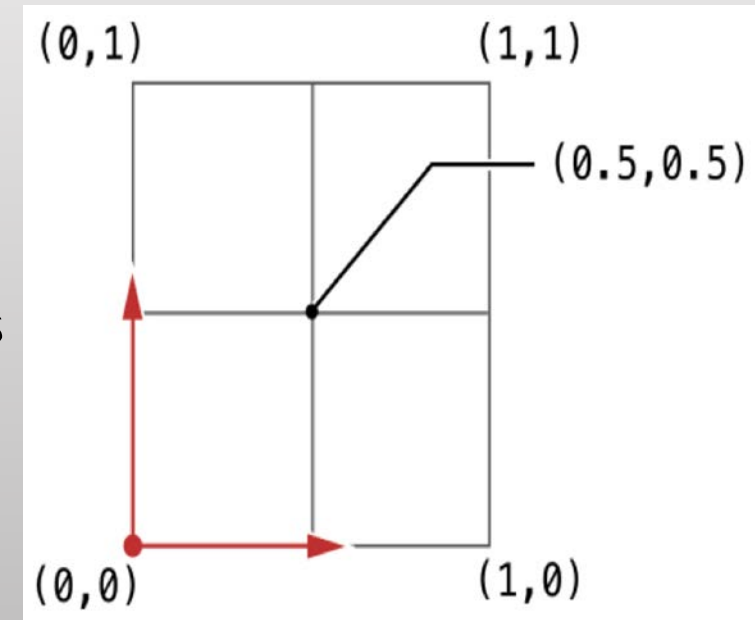
- An SKSpriteNode is used to draw a texture, an image or a colored square.
- An SKSpriteNode object can be drawn either as a rectangle with a texture mapped onto it or as colored, untextured rectangle.
- Texture sprites are more common as it represents the custom artwork which represents the characters in the game or even backgrounds etc.

## Co-ordinate System

- When a SKNode is placed in the node tree, its *position* property places it within a coordinate system provided by its parent.
- SpriteKit uses the same coordinate system on both iOS and macOS.
- A positive x coordinate goes to the right and a positive y coordinate goes up the screen.



- **AnchorPoint**
  - Anchor points are specified in the unit coordinate system, shown in the following illustration.
  - The unit coordinate system places the origin at the bottom left corner of the frame and  $(1,1)$  at the top right corner of the frame.
  - A sprite's anchor point defaults to  $(0.5,0.5)$ , which corresponds to the center of the frame.
  - When an SKSpriteNode is added to an SKNode, by default it's anchorpoint is placed at node's position
  - When an SKSpriteNode is added to another SKSpriteNode, by default it's anchorpoint is placed at anchorpoint of parent node.
  - An SKSpriteNode's origin is the bottom left point of the frame
- 
- **Frame**
  - A rectangle in the parent's coordinate system that contains the node's content, ignoring the node's children.



```
import SpriteKit

class GameScene: SKScene {
    let firstNode = SKNode()
    let nonTexturedSriteNodeFirst = SKSpriteNode()
    let nonTexturedSriteNodeSecond = SKSpriteNode()
    let texturedSpriteNode = SKSpriteNode(imageNamed: "surprised")

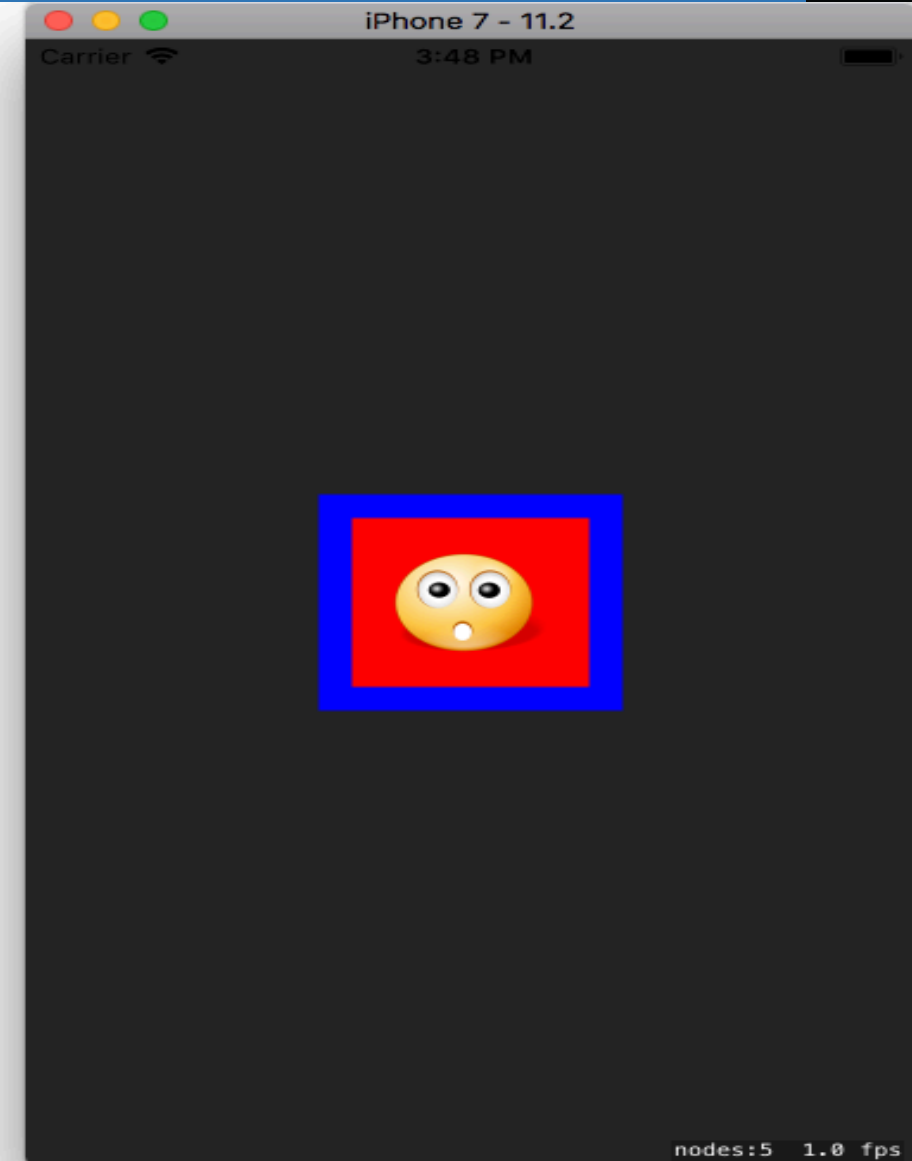
    override func didMove(to view: SKView) {
        nonTexturedSriteNodeSecond.addChild(texturedSpriteNode)
        nonTexturedSriteNodeFirst.addChild(nonTexturedSriteNodeSecond)
        firstNode.addChild(nonTexturedSriteNodeFirst)
        addChild(firstNode)

        firstNode.position = CGPoint(x: self.frame.midX, y: self.frame.midY)
        nonTexturedSKSpriteNode()
        texturedSKSpriteNode()
    }

    func texturedSKSpriteNode(){
        texturedSpriteNode.size = CGSize(width: 64, height: 64)
        texturedSpriteNode.anchorPoint = CGPoint(x: 0.5, y: 0.5)
    }

    func nonTexturedSKSpriteNode(){
        nonTexturedSriteNodeFirst.name = "first"
        nonTexturedSriteNodeFirst.color = UIColor.blue
        nonTexturedSriteNodeFirst.size = CGSize(width: 128, height: 128)
        nonTexturedSriteNodeFirst.anchorPoint = CGPoint(x: 0.5, y: 0.5)

        nonTexturedSriteNodeSecond.name = "second"
        nonTexturedSriteNodeSecond.color = UIColor.red
        nonTexturedSriteNodeSecond.size = CGSize(width: 100, height: 100)
        nonTexturedSriteNodeSecond.anchorPoint = CGPoint(x: 0.5, y: 0.5)
        nonTexturedSriteNodeSecond.position = CGPoint(x: 0, y: 0)
    }
}
```



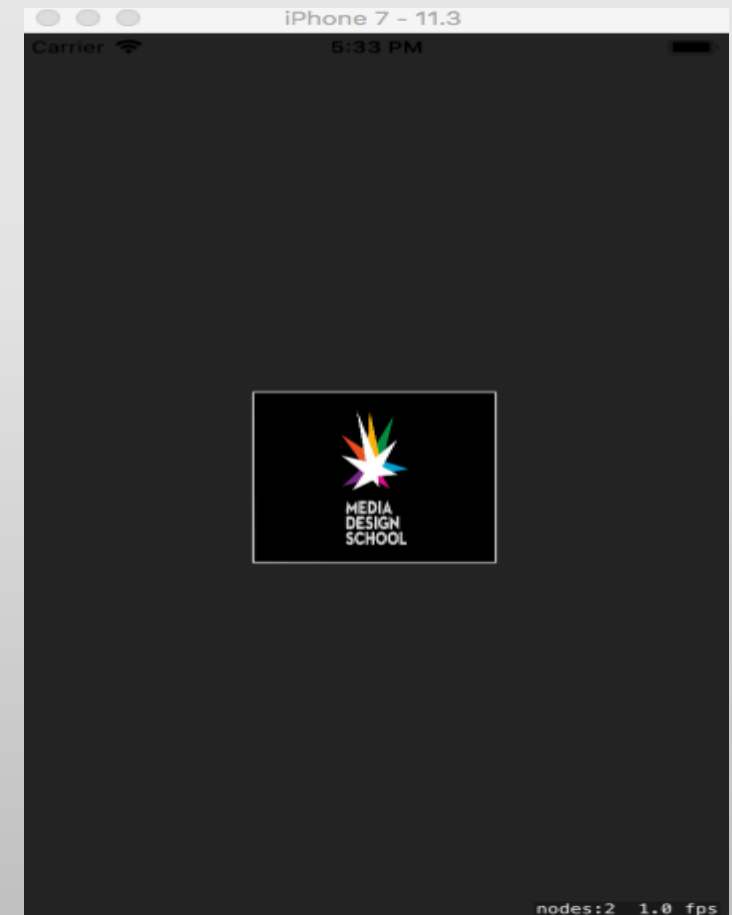
- Shape nodes are useful for content that cannot be easily decomposed into simple textured sprites.
- Shape nodes are also very useful for building and displaying debugging information on top of your game content.
- However, the SKSpriteNode class offers higher performance than this class, so use shape nodes sparingly.

```
import UIKit
import SpriteKit

class GameScene: SKScene {
    var shape: SKShapeNode!

    override func didMove(to view: SKView) {
        createShape()
    }

    func createShape(){
        shape = SKShapeNode(rectOf: CGSize(width: 128, height: 128))
        shape.fillColor = SKColor.white
        shape.fillTexture = SKTexture(imageNamed: "MDSlogo")
        shape.position = CGPoint(x: self.frame.midX, y: self.frame.midY)
        self.addChild(shape)
    }
}
```



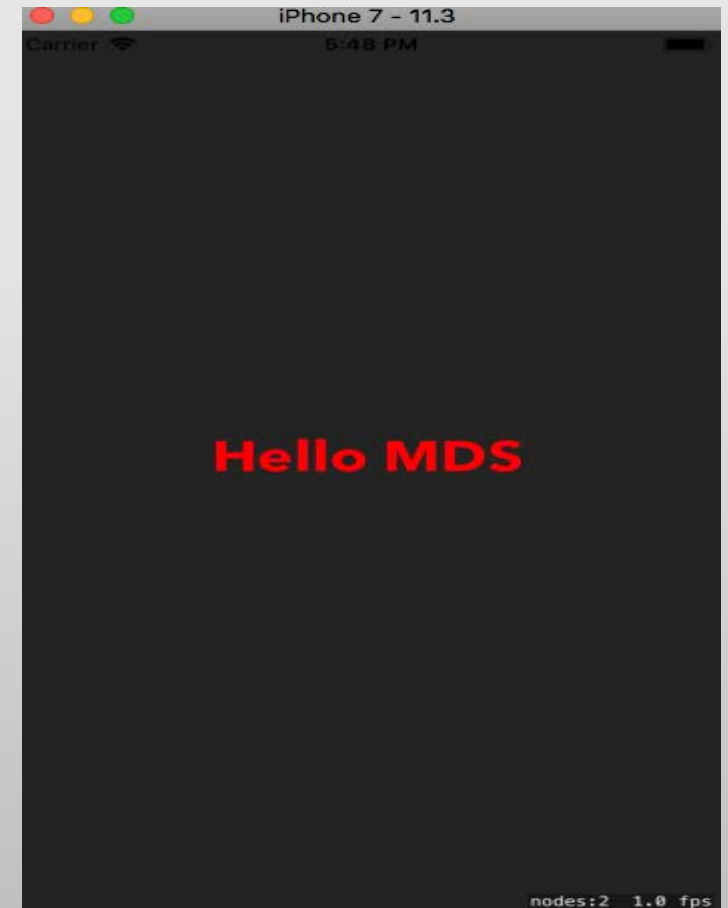
- Just about every game needs to display text at some point, even if it is just to display “Game Over” to the player.
- If you had to implement this yourself in OpenGL, it would take a fair amount of work to get it correct.
- But the SKLabelNode class does all of the work necessary to load fonts and create text for display.

```
import UIKit
import SpriteKit

class GameScene: SKScene {
    var label: SKLabelNode!

    override func didMove(to view: SKView) {
        createLabel()
    }

    func createLabel(){
        label = SKLabelNode()
        label.text = "Hello MDS"
        label.fontSize = 32.0
        label.fontName = "AvenirNext-Bold"
        label.fontColor = UIColor.red
        label.position = CGPoint(x: self.frame.midX, y: self.frame.midY)
        self.addChild(label)
    }
}
```



- Exercise
  - Change the anchor points and see the position of nodes with respect to parent node.
  - Create various shapes using SKShapeNode.
  - Use SKLabelNode to create various game labels.