Gaphics and Animation

Mobile Application Development in iOS

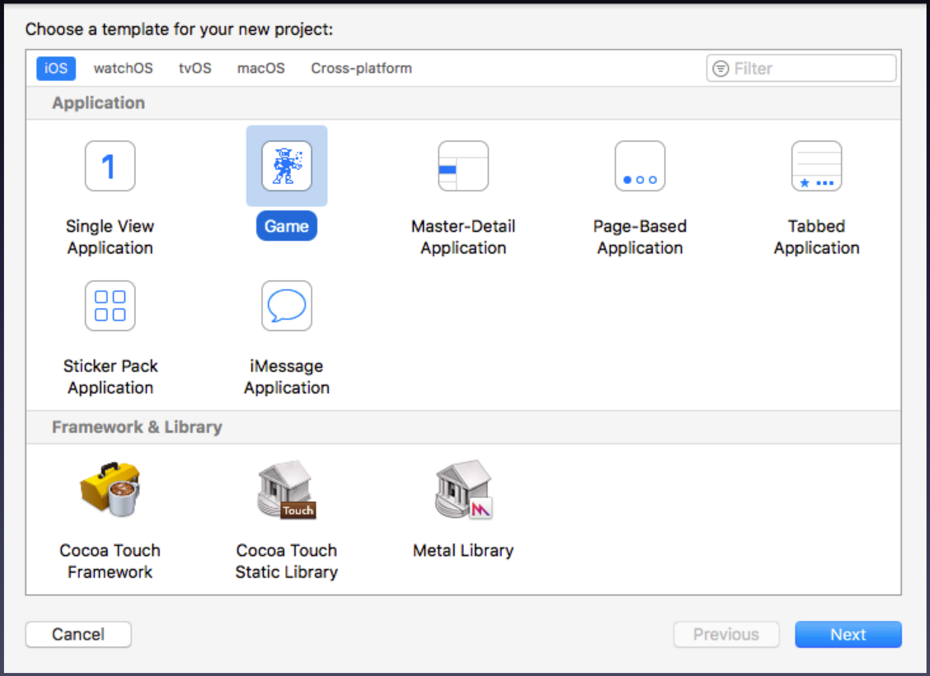
School of EECS

Washington State University

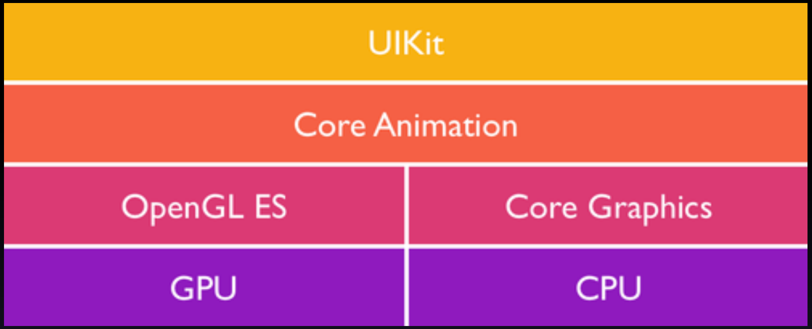
Instructor: Larry Holder

Outline

- iOS frameworks for graphics and animation
- Core Graphics
- SpriteKit
- SceneKit



iOS Frameworks (old)

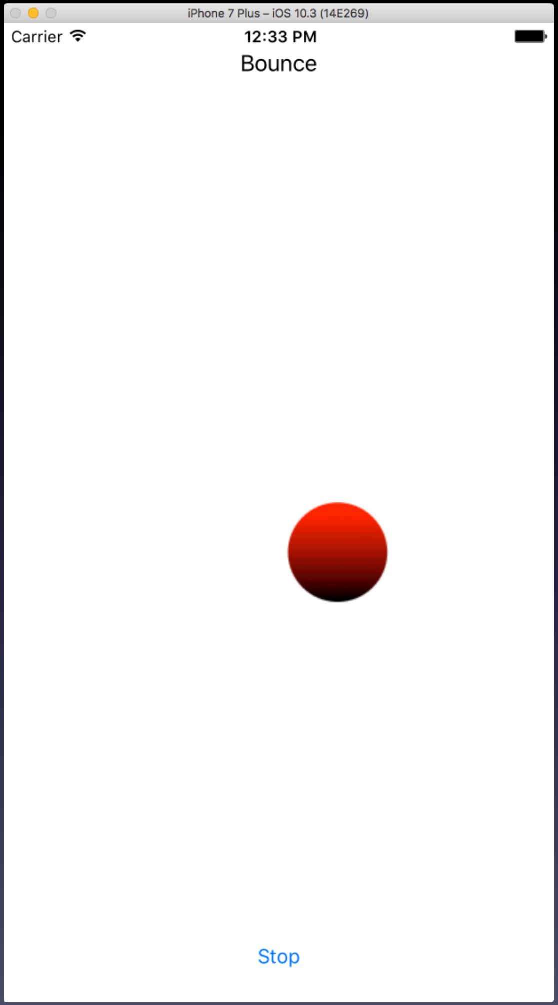


- UIKit graphics
  - Animate elements of view
- Core Graphics and Core Animation
  - 2D graphics and animation engine
  - Part of UIView
- OpenGL ES and GLKit
  - 2D and 3D rendering for GPUs on Embedded Systems (ES)

iOS Frameworks (new)

- SpriteKit
  - 2D game engine
  - Most components accessible via Storyboard
- SceneKit
  - 3D game engine
  - Most components accessible via Storyboard
- Metal
  - More direct access to GPU for graphics and computation

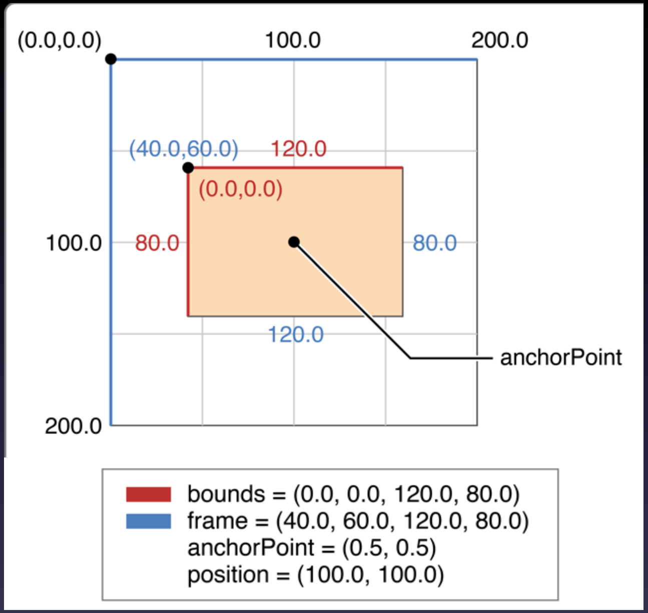
Bounce



Core Graphics

Core Graphics Approach

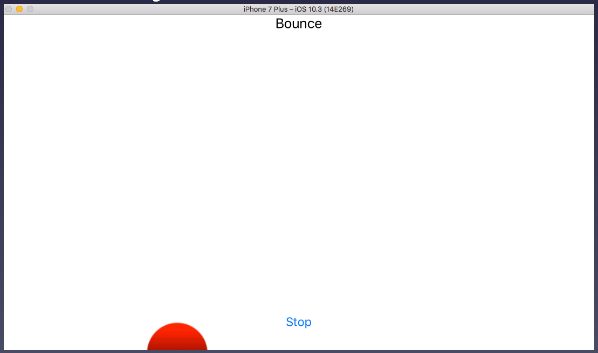
- Coordinate system (upper-left origin)
- Points vs. pixels



- Frame vs. bounds
  - CGRect = {origin.x, origin.y, size.width, size.height}
  - CGRect self.frame, self.bounds

Core Graphics Approach

- Add a UIView as a subView of the main view
- Implement gameUpdate() method
  - Modify subView's position, etc.
- Use Timer to call gameUpdate() method repeatedly
- Watch out for auto layout and orientation changes



Core Graphics Approach

```
class ViewController: UIViewController {

  let frameRate = 30.0 // updates per seconds
  let ballSpeed = 200.0 // points per second
  var ballDirection = CGPoint(x: 1.0, y: -1.0)
  var ballImageView: UIImageView!
  var gameTimer: Timer!

  func initGame() {
    let ballImage = UIImage(named: "redball.png")!
    ballImageView = UIImageView()
    ballImageView.image = ballImage
    ballImageView.frame = CGRect(x: 0, y: 0, width:
      ballImage.size.width, height: ballImage.size.height)
    self.view.addSubview(ballImageView)
  }
```

Core Graphics Approach

```
func startGame () {
  self.gameTimer = Timer.scheduledTimer(withTimeInterval:
   (1.0 / frameRate), repeats: true, block: updateGame)
}

func pauseGame () {
  self.gameTimer.invalidate()
}
```

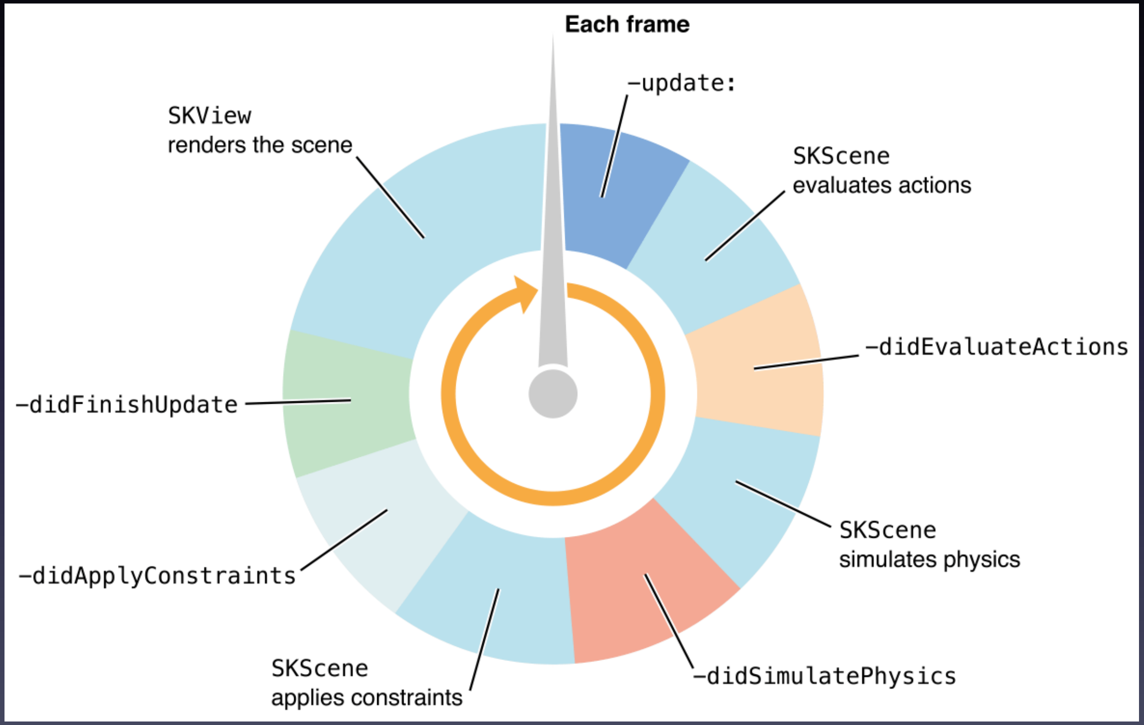
Core Graphics Approach

```swift
func updateGame (timer: Timer) {
  let x = self.ballImageView.frame.origin.x
  let y = self.ballImageView.frame.origin.y
  let width = self.ballImageView.frame.width
  let height = self.ballImageView.frame.height
  // if ball hits wall, then change direction
  if (x < 0) { // Hit left wall
    self.ballDirection.x = -self.ballDirection.x
  }
  if ((x + width) > self.view.frame.width) { // Hit right wall
    self.ballDirection.x = -self.ballDirection.x
  }
  // Handle top and bottom walls...
  // Update ball location
  let xOffset = CGFloat(self.ballSpeed / self.frameRate) * self.ballDirection.x
  let yOffset = CGFloat(self.ballSpeed / self.frameRate) * self.ballDirection.y
  self.ballImageView.frame.origin.x = x + xOffset
  self.ballImageView.frame.origin.y = y + yOffset
}
```

SpriteKit

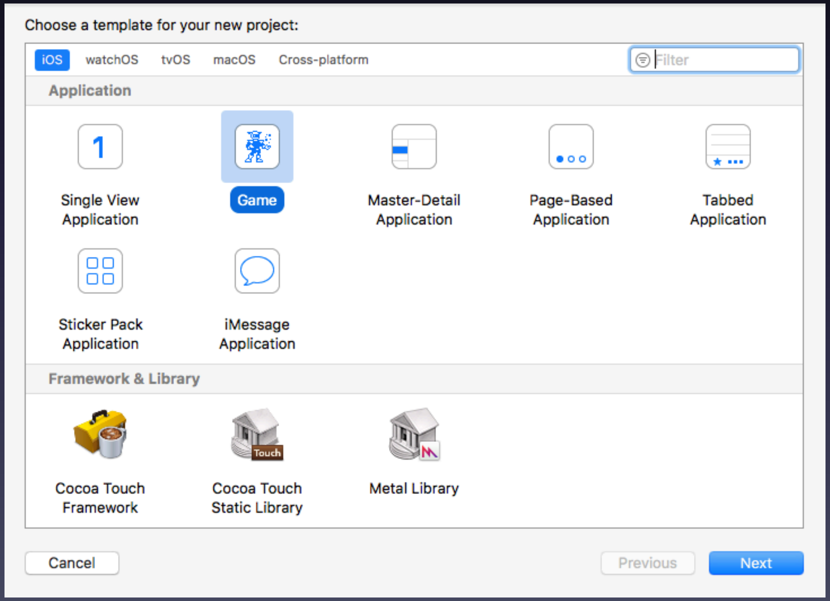
SpriteKit Approach

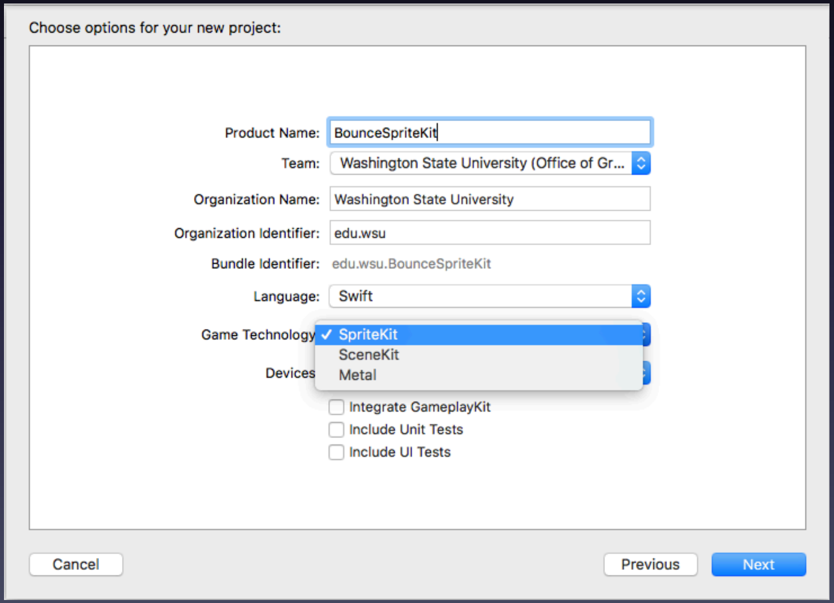
- Update/render loop



SpriteKit Approach

- Create new Game project
  - Game Technology: SpriteKit


Choose a template for your new project:

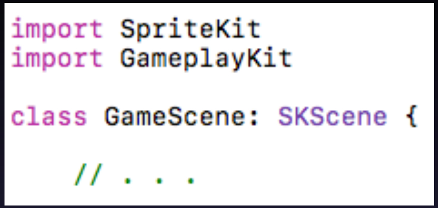

Choose options for your new project:

Product Name: BounceSpriteKit
Team: Washington State University (Office of Gr...
Organization Name: Washington State University
Organization Identifier: edu.wsu
Bundle Identifier: edu.wsu.BounceSpriteKit
Language: Swift
Game Technology: ✓ SpriteKit

SpriteKit Organization

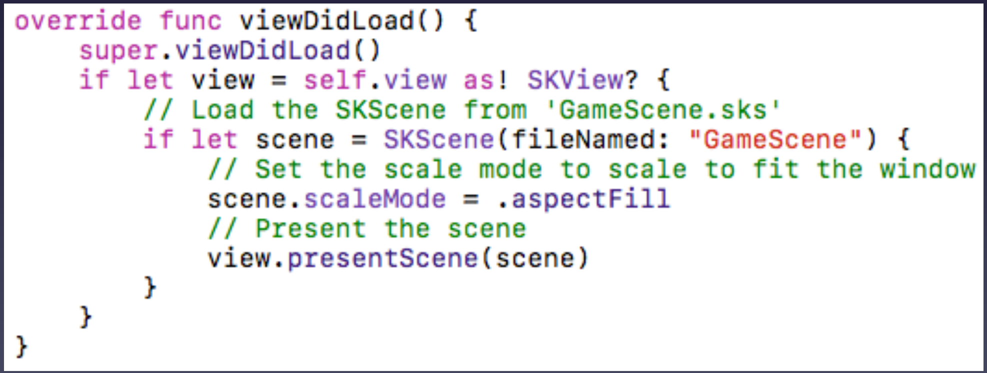
- Scene(s) of type SKScene
  - Edit in Sprite Editor (.sks file)
- Main view of type SKView
- Present SKScene in SKView

GameScene.swift



```swift
import SpriteKit
import GameplayKit

class GameScene: SKScene {

    // . . .
```

GameViewController.swift



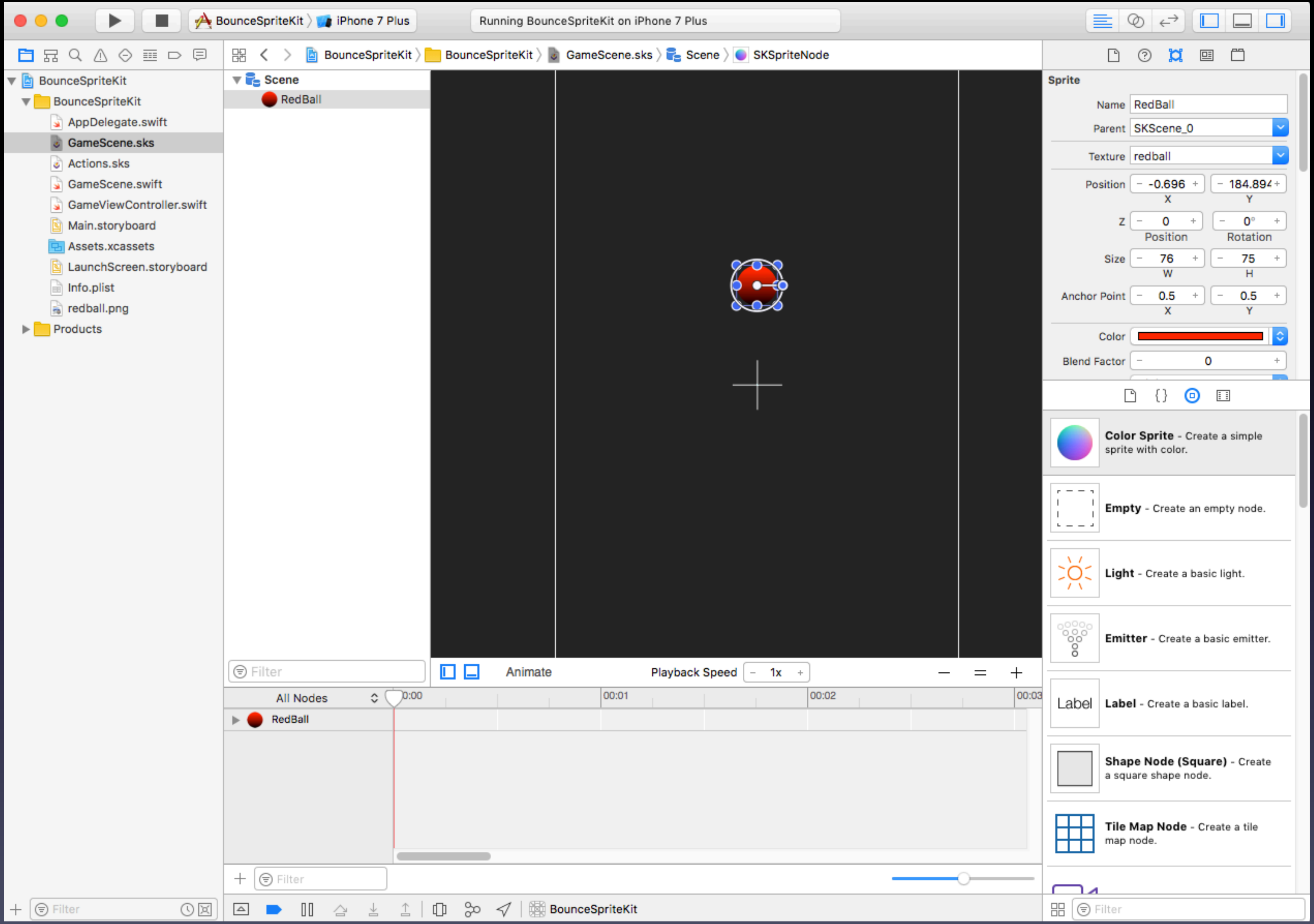
```swift
override func viewDidLoad() {
    super.viewDidLoad()
    if let view = self.view as! SKView? {
        // Load the SKScene from 'GameScene.sks'
        if let scene = SKScene(fileNamed: "GameScene") {
            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill
            // Present the scene
            view.presentScene(scene)
        }
    }
}
```

Sprites

redball.png

- Sprite is a rectangle with a texture (image)
- SKSpriteNode is a sprite with lots of properties
  – SKAction for actions to execute (e.g., movement)
  – SKPhysicsBody for physical effects (e.g., gravity)
- Other types of SKNode's (e.g., SKLabelNode)
- SKScene is a collection of SKNode's

SpriteKit Scene Editor



Sprite

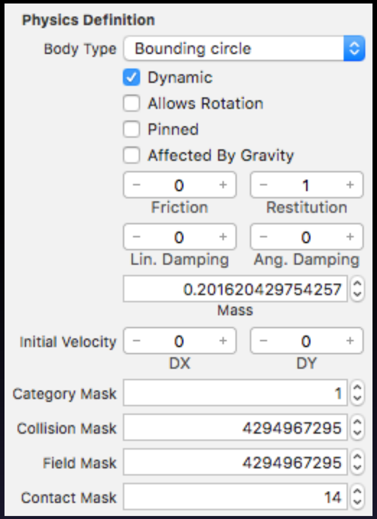
Name RedBall
Parent SKScene_0
Texture redball
Position -0.696 X, 184.894 Y
Z Position 0, Rotation 0°
Size 76 W, 75 H
Anchor Point 0.5 X, 0.5 Y
Color
Blend Factor 0

Color Sprite - Create a simple sprite with color.
Empty - Create an empty node.
Light - Create a basic light.
Emitter - Create a basic emitter.
Label - Create a basic label.
Shape Node (Square) - Create a square shape node.
Tile Map Node - Create a tile map node.

SpriteKit Physics

- Handling collisions and contacts
- Each SKSpriteNode has masks
  - Category mask
    - Unique power of 2 for each object type
    - E.g., ball: 0001=1, brick: 0010=2, wall: 0100=4, paddle: 1000=8
  - Collision mask
    - More for how objects react, than to detect contact
    - E.g., all 1's = 4294967295, everything collides with everything
  - Contact mask
    - Detect when two objects touch (contactMask & categoryMask > 0)
    - Send message to delegate
    - E.g., ball: 0111=14, detect contacts between ball and paddle, bricks and walls
    - E.g., other objects: 0001=1, detect contacts with ball

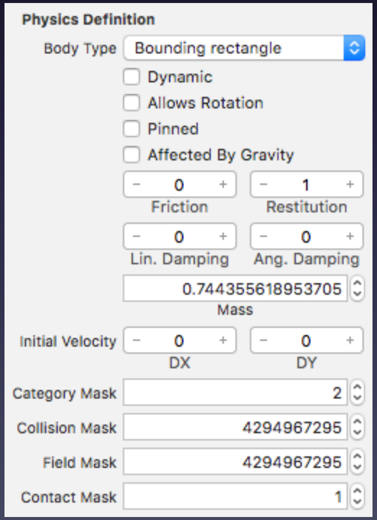
ball:



Physics Definition

Body Type	Bounding circle
	[✓] Dynamic
	[ ] Allows Rotation
	[ ] Pinned
	[ ] Affected By Gravity
Friction	0
Restitution	1
Lin. Damping	0
Ang. Damping	0
Mass	0.201620429754257
Initial Velocity DX	0
Initial Velocity DY	0
Category Mask	1
Collision Mask	4294967295
Field Mask	4294967295
Contact Mask	14

brick:

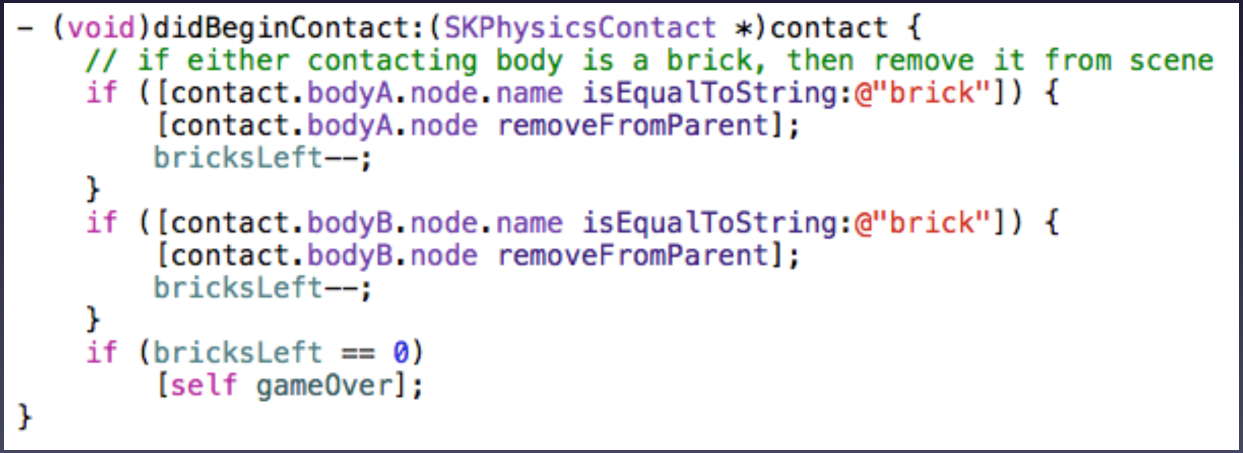


Physics Definition

Body Type	Bounding rectangle
	[ ] Dynamic
	[ ] Allows Rotation
	[ ] Pinned
	[ ] Affected By Gravity
Friction	0
Restitution	1
Lin. Damping	0
Ang. Damping	0
Mass	0.744355618953705
Initial Velocity DX	0
Initial Velocity DY	0
Category Mask	2
Collision Mask	4294967295
Field Mask	4294967295
Contact Mask	1

SpriteKit Physics

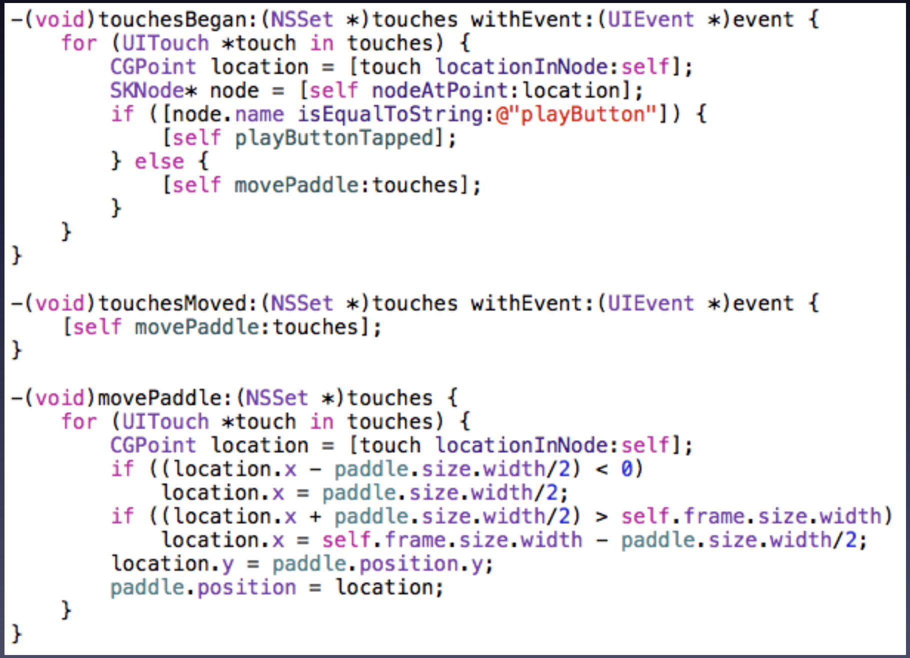
- Handling contacts
  - SKPhysicsContactDelegate for SKScene
  - didBeginContact:(SKPhysicsContact*)contact



```
- (void)didBeginContact:(SKPhysicsContact *)contact {
    // if either contacting body is a brick, then remove it from scene
    if ([contact.bodyA.node.name isEqualToString:@"brick"]) {
        [contact.bodyA.node removeFromParent];
        bricksLeft--;
    }
    if ([contact.bodyB.node.name isEqualToString:@"brick"]) {
        [contact.bodyB.node removeFromParent];
        bricksLeft--;
    }
    if (bricksLeft == 0)
        [self gameOver];
}
```

SpriteKit Touches

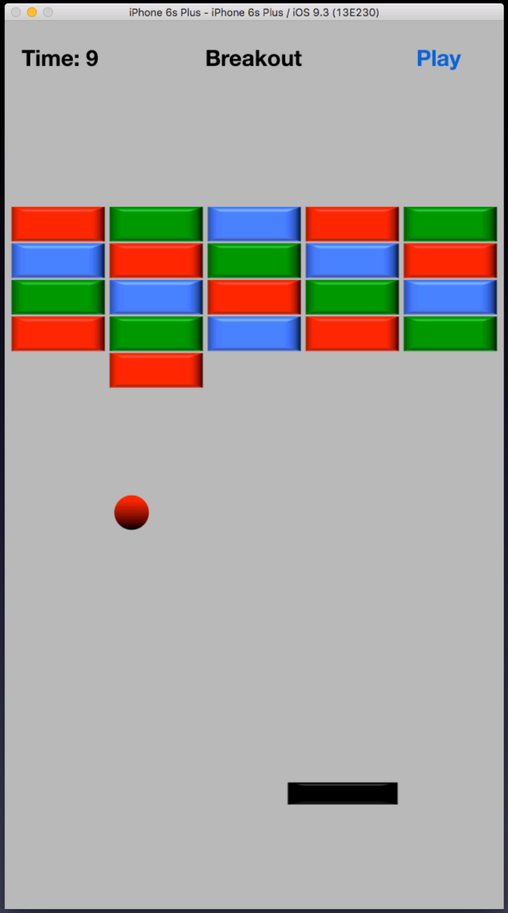
- Same as for UIView
  - touchesBegan:(NSSet*)touches withEvent(UIEvent*)event
  - touchesMoved:(NSSet*)touches withEvent(UIEvent*)event



```
-(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
    for (UITouch *touch in touches) {
        CGPoint location = [touch locationInNode:self];
        SKNode* node = [self nodeAtPoint:location];
        if ([node.name isEqualToString:@"playButton"]) {
            [self playButtonTapped];
        } else {
            [self movePaddle:touches];
        }
    }
}

-(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    [self movePaddle:touches];
}

-(void)movePaddle:(NSSet *)touches {
    for (UITouch *touch in touches) {
        CGPoint location = [touch locationInNode:self];
        if ((location.x - paddle.size.width/2) < 0)
            location.x = paddle.size.width/2;
        if ((location.x + paddle.size.width/2) > self.frame.size.width)
            location.x = self.frame.size.width - paddle.size.width/2;
        location.y = paddle.position.y;
        paddle.position = location;
    }
}
```

Breakout2D with SpriteKit

- resetGame
- playGame
- pauseGame
- gameOver
- update
  - Check for game over
  - Nudge stuck ball

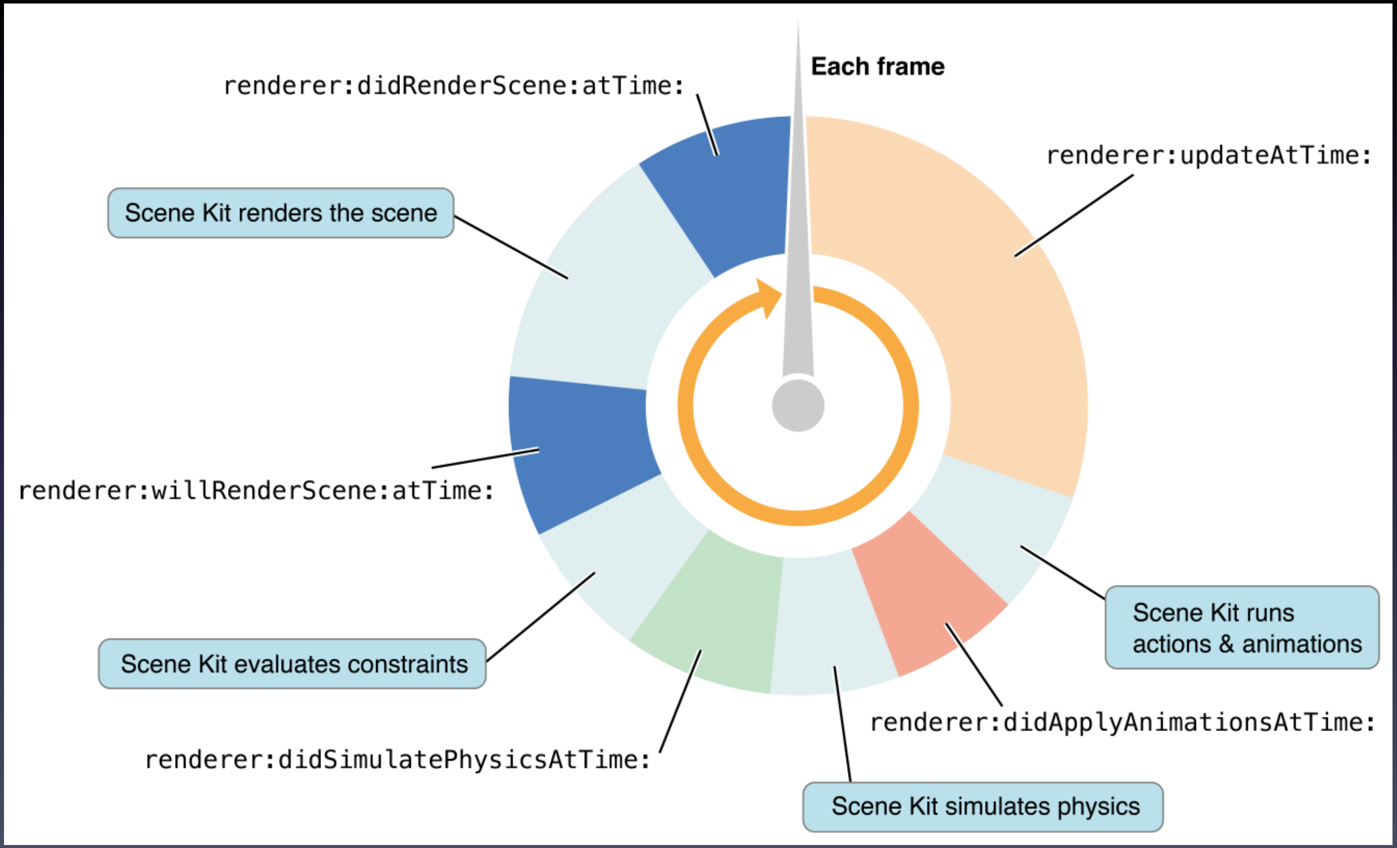


SceneKit

SceneKit Approach

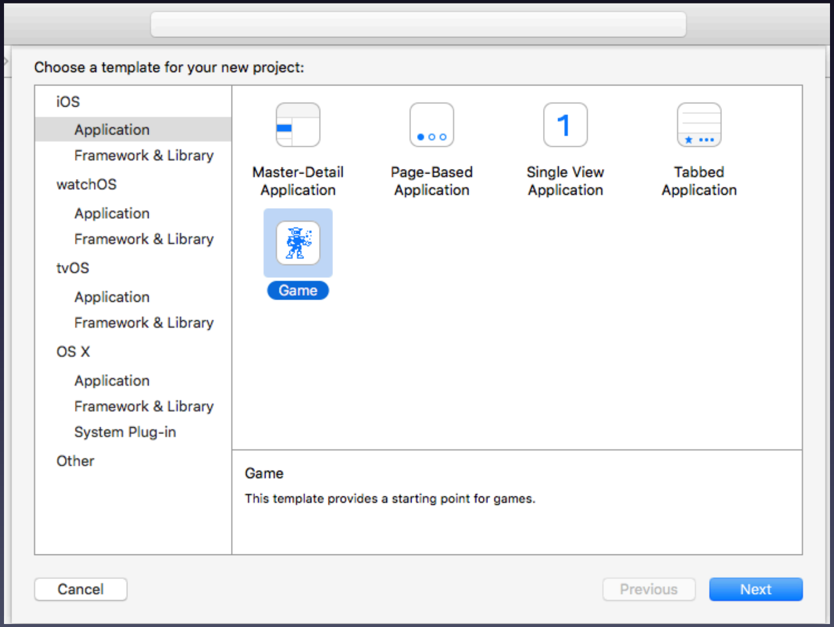
SceneKit Approach

- Update/render loop

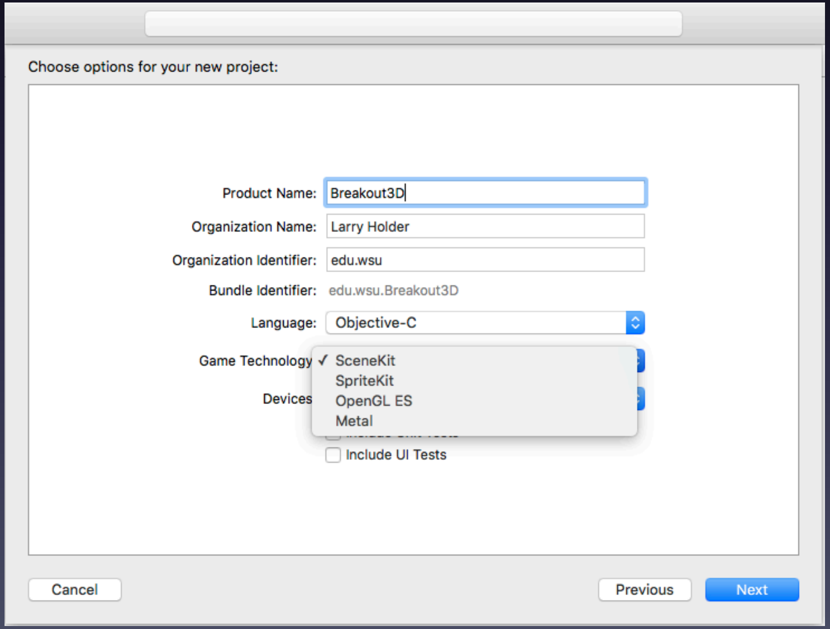

Each frame
renderer:updateAtTime:
Scene Kit runs actions & animations
renderer:didApplyAnimationsAtTime:
Scene Kit simulates physics
renderer:didSimulatePhysicsAtTime:
Scene Kit evaluates constraints
renderer:willRenderScene:atTime:
Scene Kit renders the scene
renderer:didRenderScene:atTime:

SceneKit Approach

- Create new Game project
  - Game Technology: SceneKit


Choose a template for your new project:

Game
This template provides a starting point for games.

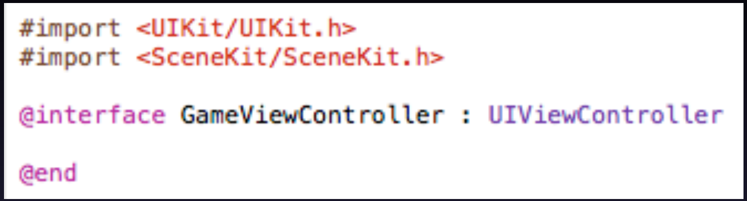

Choose options for your new project:

Product Name:	Breakout3D
Organization Name:	Larry Holder
Organization Identifier:	edu.wsu
Bundle Identifier:	edu.wsu.Breakout3D
Language:	Objective-C
Game Technology	✓ SceneKit

SceneKit Organization

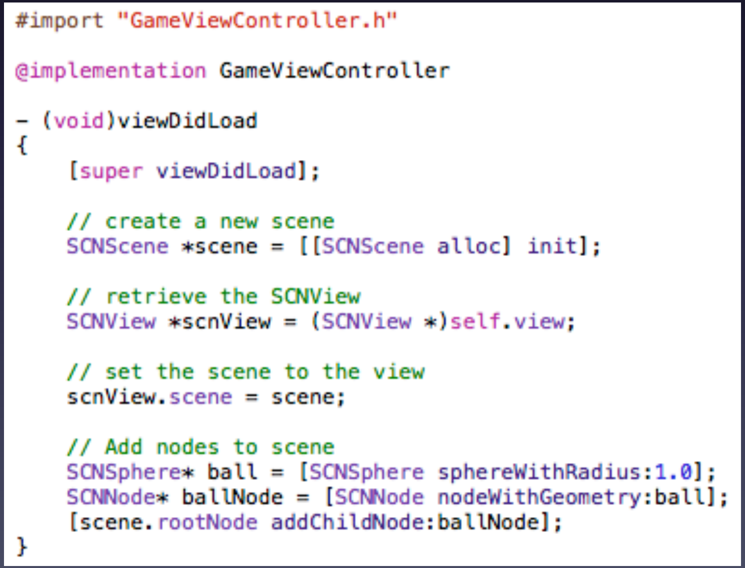
- Main view of type SCNView (available in StoryBoard)
- Scene(s) of type SCNScene
  - Edit in Scene Editor (.scn file)
  - Or, create programmatically

GameController.h



```
#import <UIKit/UIKit.h>
#import <SceneKit/SceneKit.h>

@interface GameViewController : UIViewController

@end
```

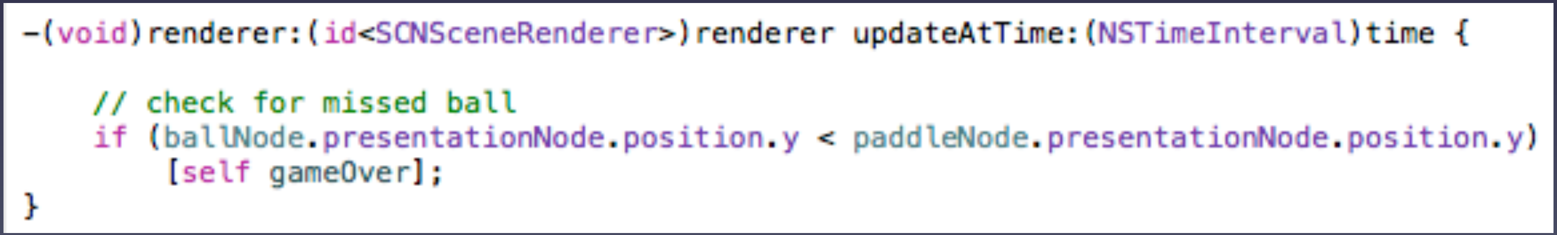
GameController.m



```
#import "GameViewController.h"

@implementation GameViewController

- (void)viewDidLoad
{
    [super viewDidLoad];

    // create a new scene
    SCNScene *scene = [[SCNScene alloc] init];

    // retrieve the SCNView
    SCNView *scnView = (SCNView *)self.view;

    // set the scene to the view
    scnView.scene = scene;

    // Add nodes to scene
    SCNSphere* ball = [SCNSphere sphereWithRadius:1.0];
    SCNNode* ballNode = [SCNNode nodeWithGeometry:ball];
    [scene.rootNode addChildNode:ballNode];
}
```

SceneKit Frame Update

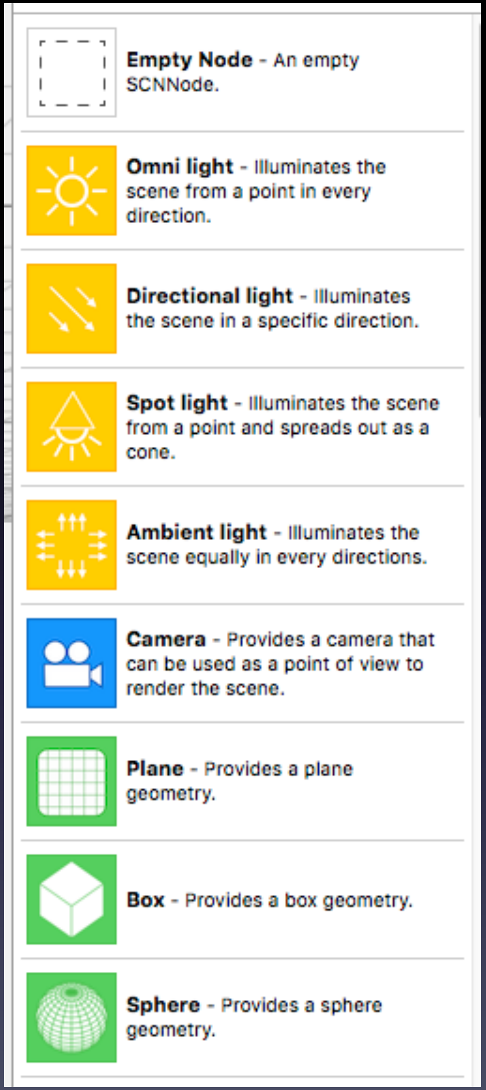
- Add SCNSceneRendererDelegate to scnView
- scnView.delegate = self
- scnView.playing = YES
- Implement updateAtTime delegate method



```
-(void)renderer:(id<SCNSceneRenderer>)renderer updateAtTime:(NSTimeInterval)time {

    // check for missed ball
    if (ballNode.presentationNode.position.y < paddleNode.presentationNode.position.y)
        [self gameOver];
}
```

SCNNode Types

- Lighting: Ambient, Directional, Omni, Spot
- Camera
- Geometry: Plane, Box, Sphere, Text, …
- Fields: Drag, Gravity, Electric, Magnetic, ...
- Particle system
- Actions: Move, Scale, Rotate, Fade



Empty Node - An empty SCNNode.

Omni light - Illuminates the scene from a point in every direction.

Directional light - Illuminates the scene in a specific direction.

Spot light - Illuminates the scene from a point and spreads out as a cone.

Ambient light - Illuminates the scene equally in every directions.

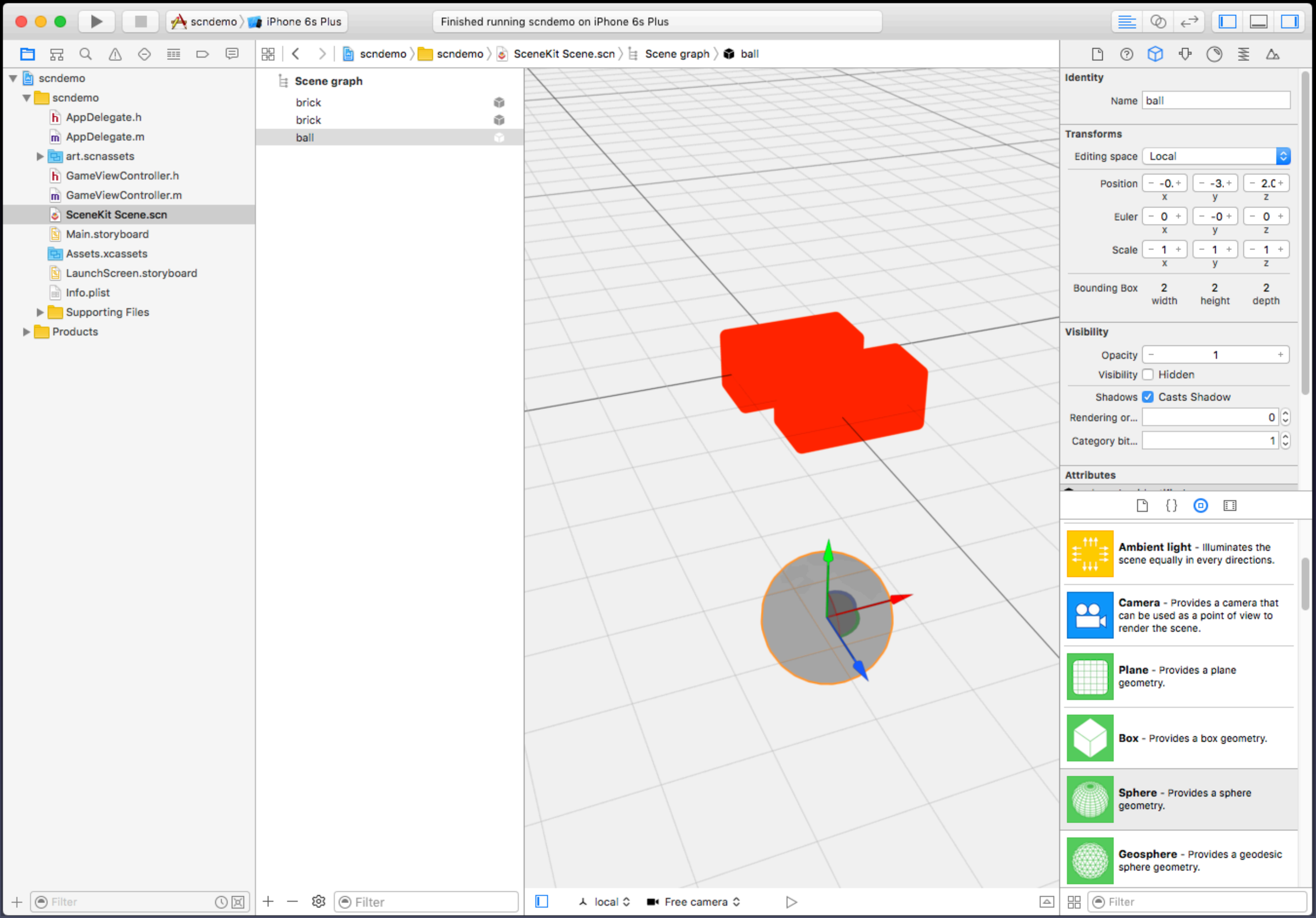
Camera - Provides a camera that can be used as a point of view to render the scene.

Plane - Provides a plane geometry.

Box - Provides a box geometry.

Sphere - Provides a sphere geometry.

SceneKit Scene Editor



Finished running scndemo on iPhone 6s Plus

Identity
Name: ball

Transforms
Editing space: Local
Position: x -0. y -3. z 2.0
Euler: x 0 y -0 z 0
Scale: x 1 y 1 z 1
Bounding Box: 2 width, 2 height, 2 depth

Visibility
Opacity: 1
Visibility: Hidden
Shadows: ✓ Casts Shadow
Rendering or...: 0
Category bit...: 1

Attributes

Ambient light - Illuminates the scene equally in every directions.

Camera - Provides a camera that can be used as a point of view to render the scene.

Plane - Provides a plane geometry.

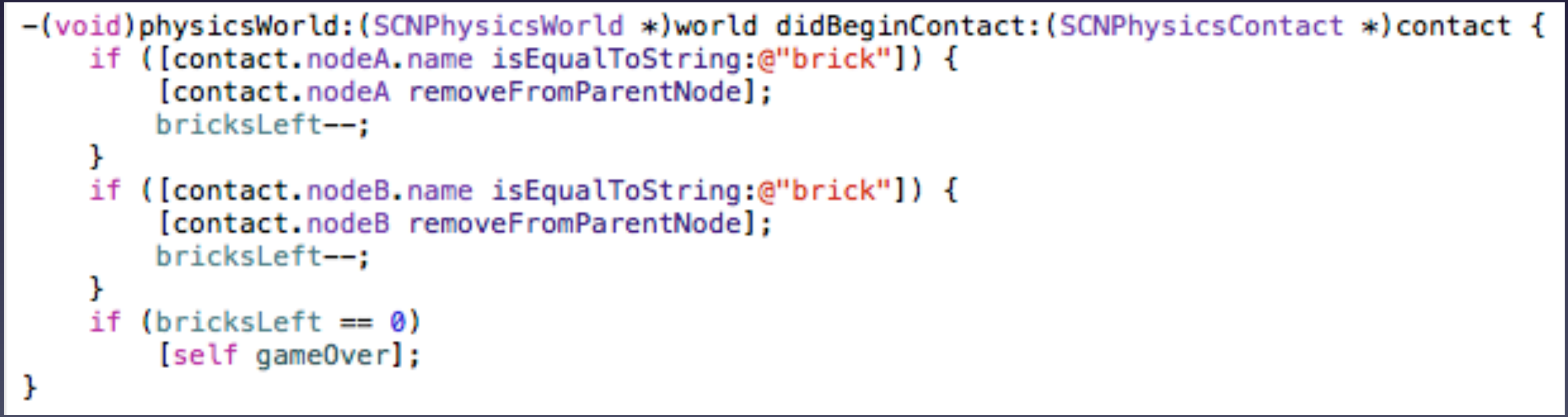
Box - Provides a box geometry.

Sphere - Provides a sphere geometry.

Geosphere - Provides a geodesic sphere geometry.

SceneKit Physics

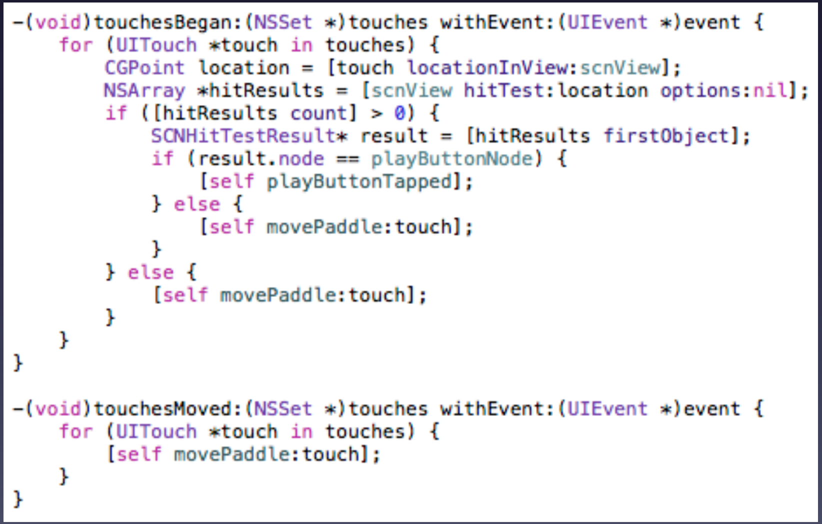
- Handling collisions and contacts
  - Essentially same as SpriteKit (masks)
  - SCNPhysicsContactDelegate for SCNScene
  - didBeginContact:(SCNPhysicsContact*)contact



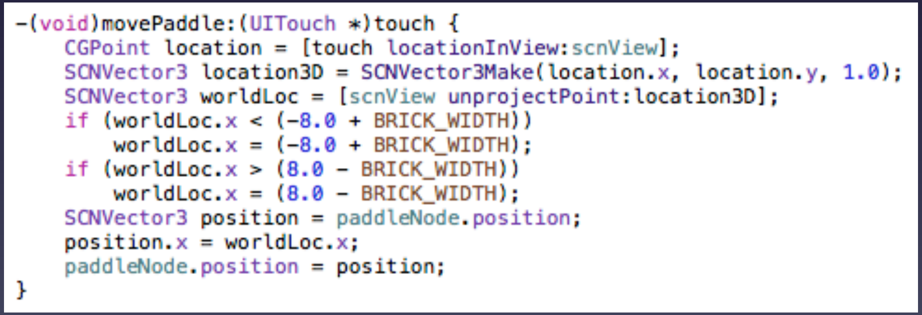
```
-(void)physicsWorld:(SCNPhysicsWorld *)world didBeginContact:(SCNPhysicsContact *)contact {
    if ([contact.nodeA.name isEqualToString:@"brick"]) {
        [contact.nodeA removeFromParentNode];
        bricksLeft--;
    }
    if ([contact.nodeB.name isEqualToString:@"brick"]) {
        [contact.nodeB removeFromParentNode];
        bricksLeft--;
    }
    if (bricksLeft == 0)
        [self gameOver];
}
```

SceneKit Touches

- Same as for UIView
  - touchesBegan:(NSSet*)touches withEvent(UIEvent*)event
  - touchesMoved:(NSSet*)touches withEvent(UIEvent*)event
  - Except “hitTest” instead of “nodeAtPoint”



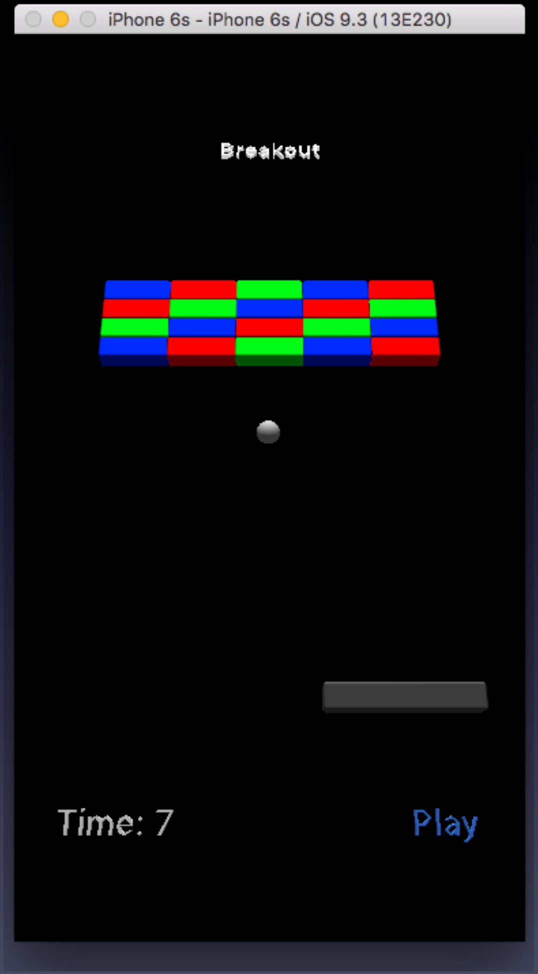
```
-(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
    for (UITouch *touch in touches) {
        CGPoint location = [touch locationInView:scnView];
        NSArray *hitResults = [scnView hitTest:location options:nil];
        if ([hitResults count] > 0) {
            SCNHitTestResult* result = [hitResults firstObject];
            if (result.node == playButtonNode) {
                [self playButtonTapped];
            } else {
                [self movePaddle:touch];
            }
        } else {
            [self movePaddle:touch];
        }
    }
}

-(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    for (UITouch *touch in touches) {
        [self movePaddle:touch];
    }
}
```



```
-(void)movePaddle:(UITouch *)touch {
    CGPoint location = [touch locationInView:scnView];
    SCNVector3 location3D = SCNVector3Make(location.x, location.y, 1.0);
    SCNVector3 worldLoc = [scnView unprojectPoint:location3D];
    if (worldLoc.x < (-8.0 + BRICK_WIDTH))
        worldLoc.x = (-8.0 + BRICK_WIDTH);
    if (worldLoc.x > (8.0 - BRICK_WIDTH))
        worldLoc.x = (8.0 - BRICK_WIDTH);
    SCNVector3 position = paddleNode.position;
    position.x = worldLoc.x;
    paddleNode.position = position;
}
```

Breakout3D with SceneKit

- resetGame
- playGame
- pauseGame
- gameOver
- update
  - Check for game over
  - Nudge stuck ball



Resources

- Core Graphics
  - developer.apple.com/reference/coregraphics
- Sprite Kit
  - developer.apple.com/spritekit/
- Scene Kit
  - developer.apple.com/scenekit/
- Gameplay Kit
  - developer.apple.com/reference/gameplaykit