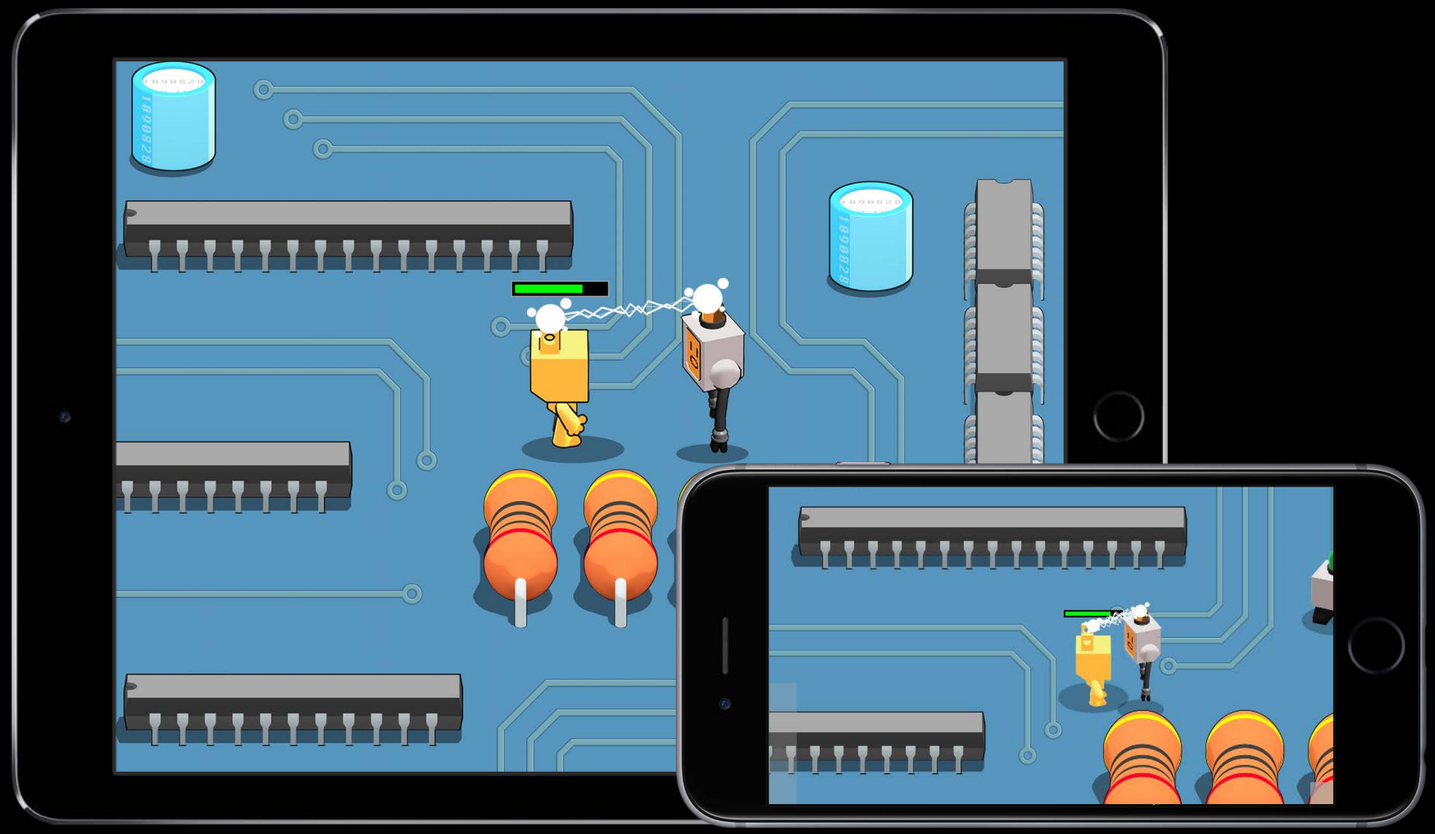
Game Technologies for Apple Watch

Session 612

Christy Warren Game Technologies Engineer
Fatima Broom Game Technologies Engineer
Tyler Casella Game Technologies Engineer

iOS

Fantastic platform for games



iOS and watchOS

Fantastic platform for games

Games on Your Wrist

What you will learn

Input and feedback

Graphics frameworks

Social gaming

Tools and best practices

Games on Your Wrist

What you will learn

Input and feedback

Graphics frameworks

Social gaming

Tools and best practices

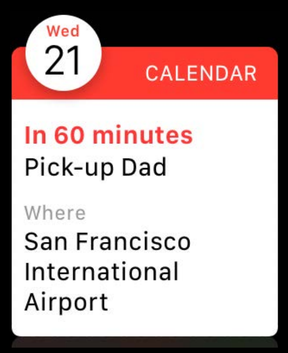
Available Technologies

WatchKit 3

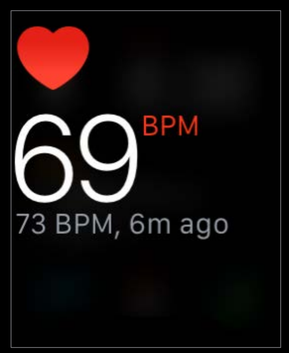
SpriteKit

SceneKit

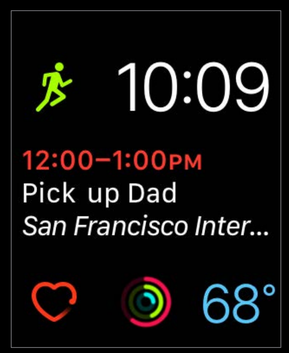
Game Center



Notification



Glances



Complications

Watch Connectivity

Available Technologies

WatchKit 3

SpriteKit

SceneKit

Game Center

Game Technologies

WatchKit

NEW

Interfaces

Input

- Gesture recognizers
- Digital Crown
- Accelerometer

Haptic feedback

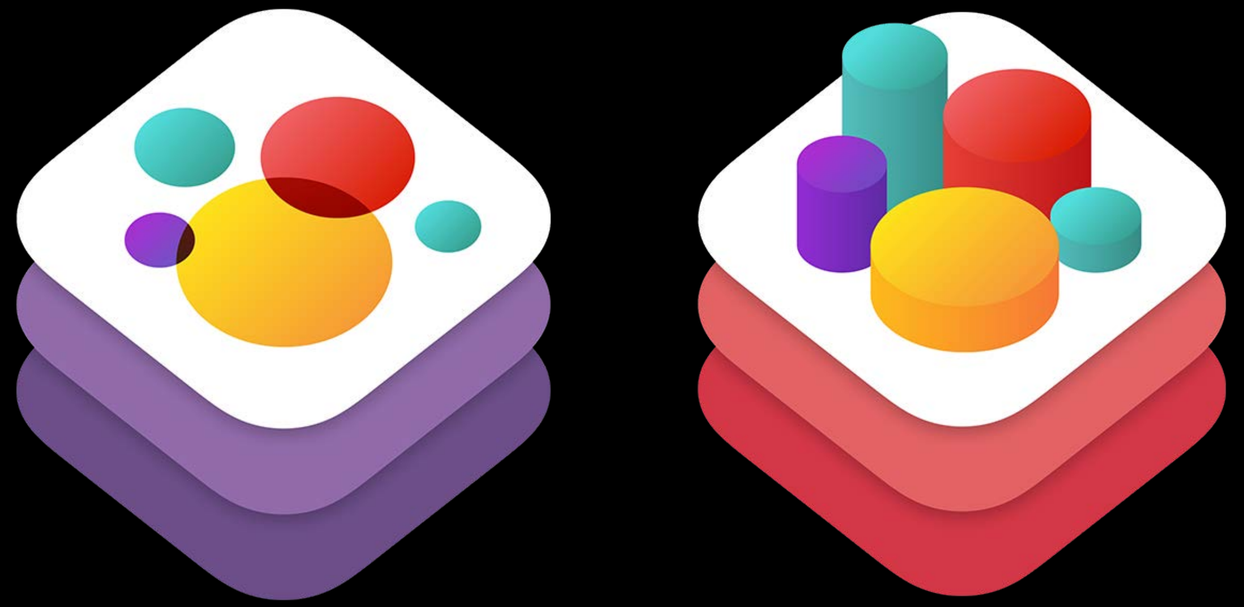
Game Technologies

SpriteKit and SceneKit

NEW

Graphics

Audio



Game Technologies

Game Center

NEW

Social gaming

- Achievements
- Leaderboards
- Multiplayer

Demo

Puzzle 3D

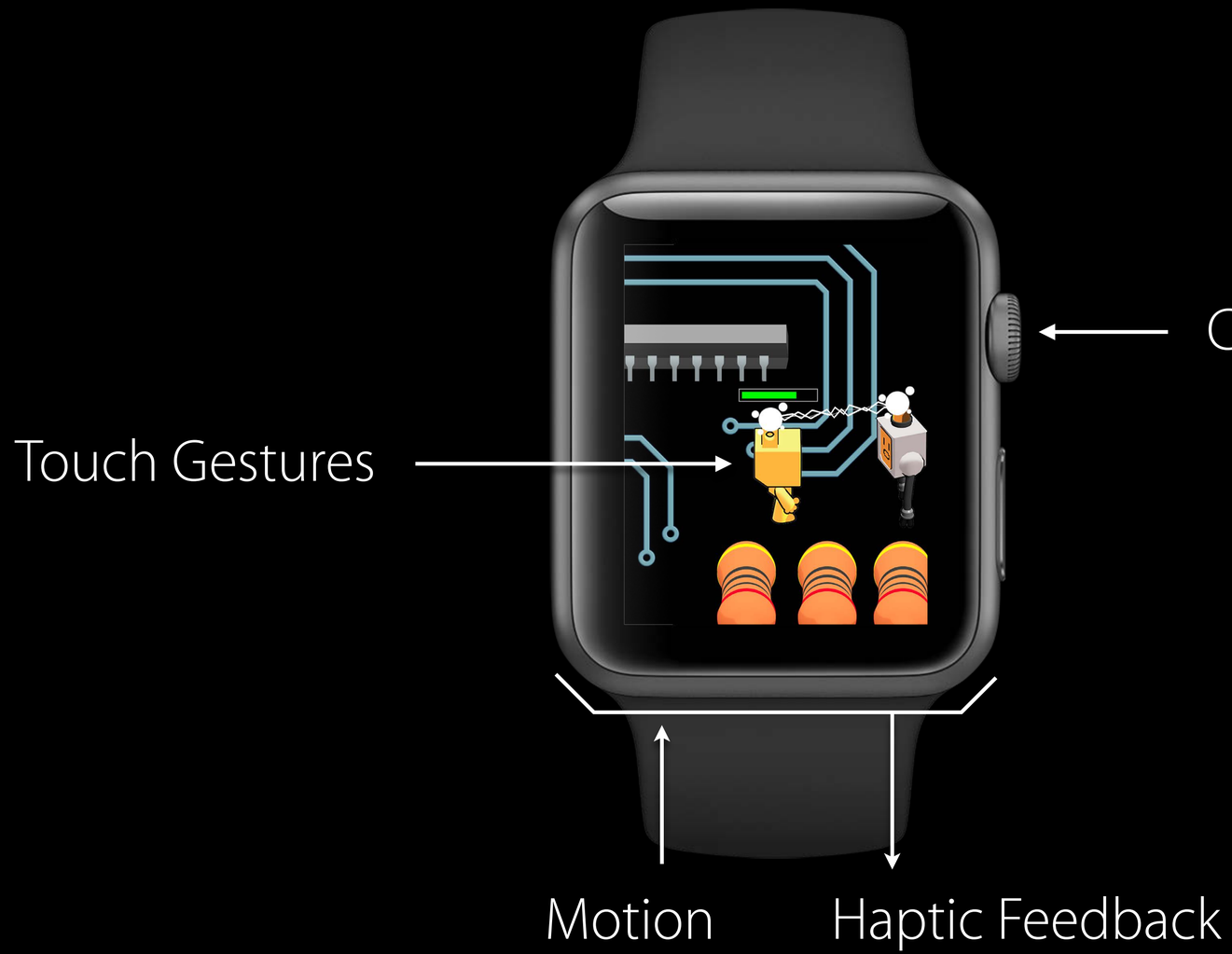
Demo

Puzzle 3D

Input and Feedback

User Interaction

At a glance



Touch Gestures

Types

- Tap
- Pan
- Swipe
- Long press

Touch Gestures

Tap

Target selection

Move to position

Touch Gestures

Tap

Target selection

Move to position

Touch Gestures

Pan

Scrolling

Movement

Touch Gestures

Pan

Scrolling

Movement

Touch Gestures

Pan

Scrolling

Movement

Touch Gestures

Pan

Scrolling

Movement

Touch Gestures

Swipe

Navigation

- Push
- Pop

Touch Gestures

Swipe

Navigation

- Push
- Pop

Touch Gestures

Swipe

Navigation

- Push
- Pop

Touch Gestures

Long press

Bring up menu

Touch Gestures

Long press

Bring up menu

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer
Swipe	WKSwipeGestureRecognizer

Gesture Recognizers

Four new classes

Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer
Swipe	WKSwipeGestureRecognizer

Gesture Recognizers

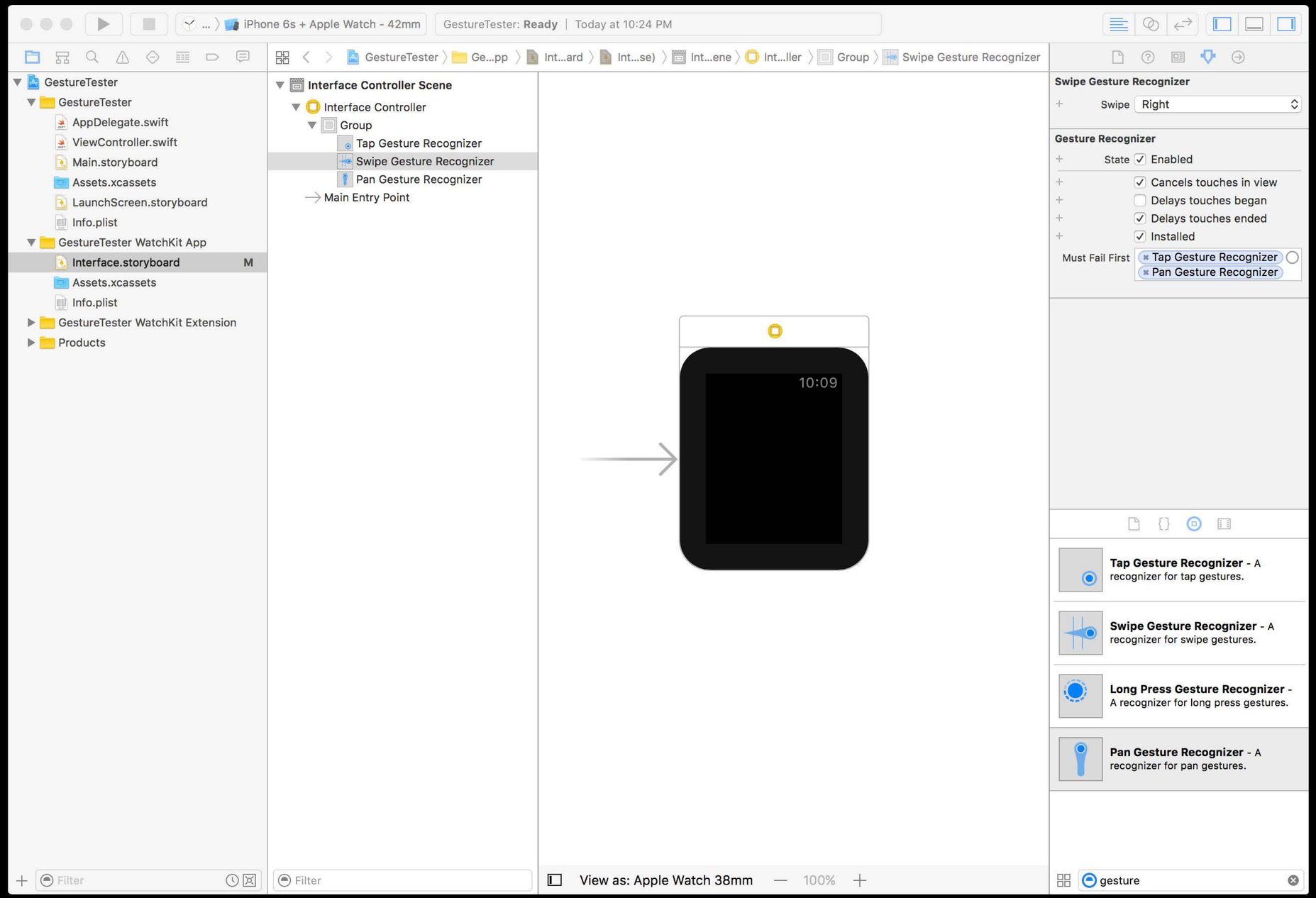
Four new classes

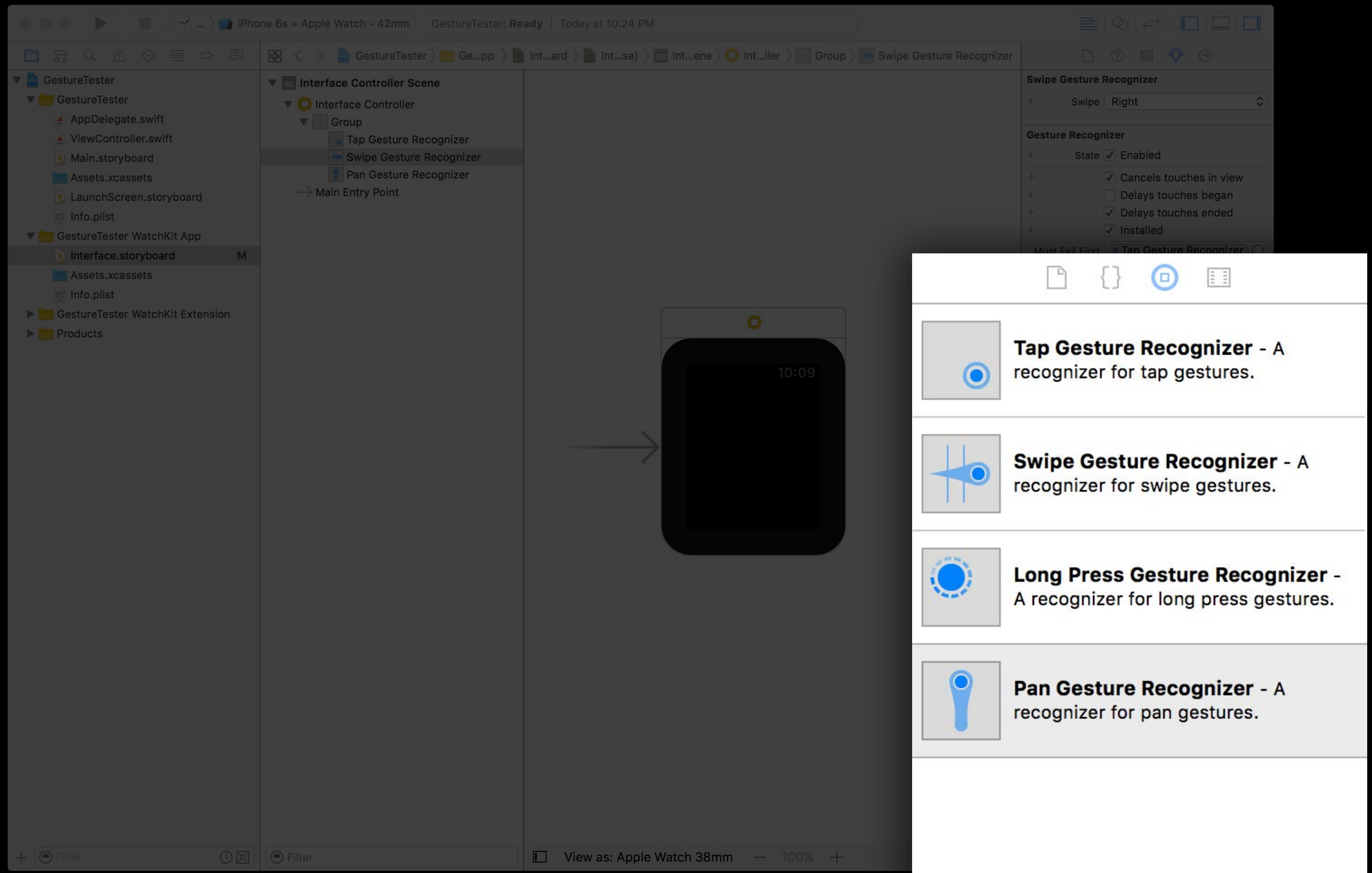
Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer
Swipe	WKSwipeGestureRecognizer
Long Press	WKLongPressGestureRecognizer

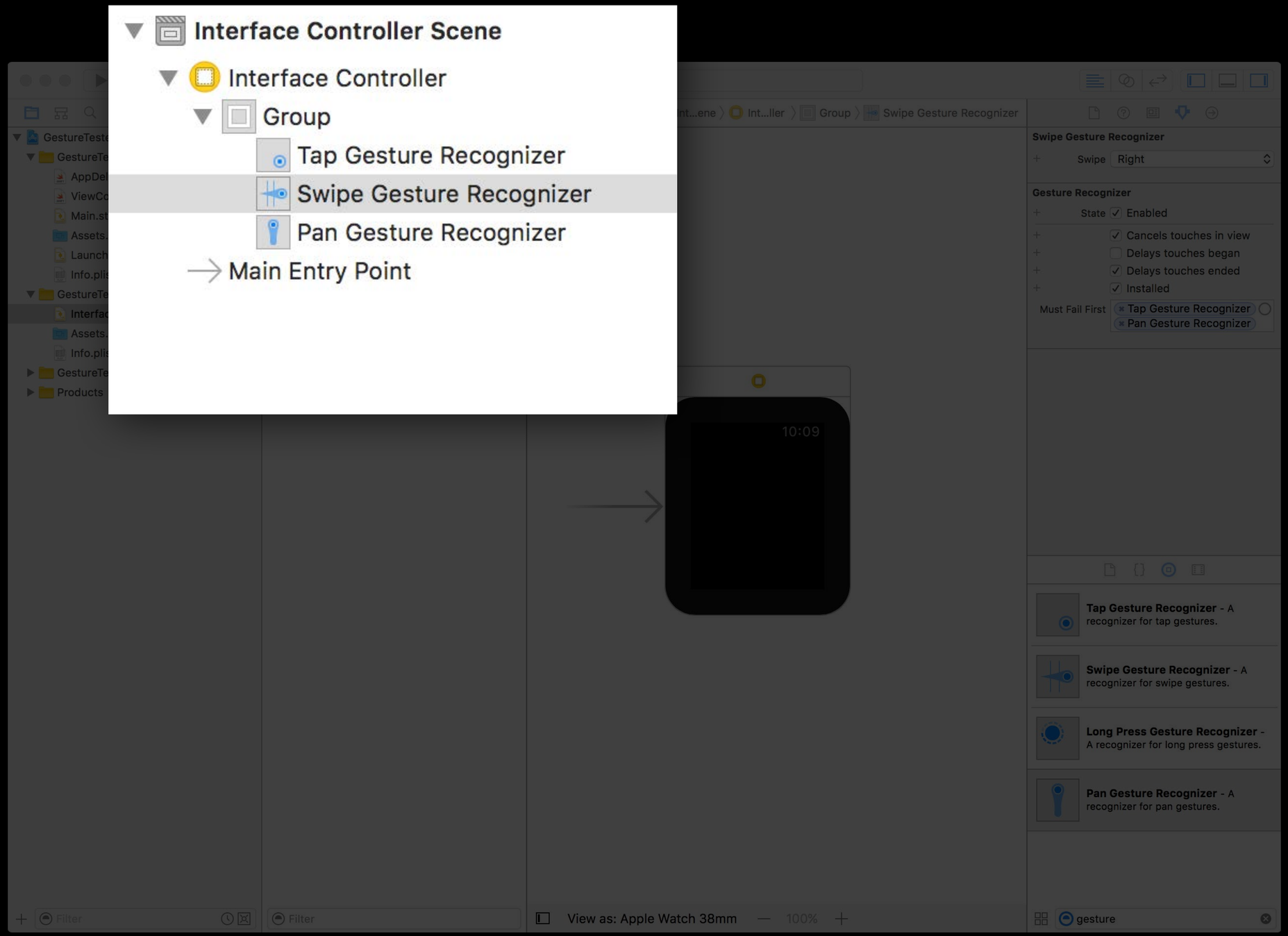
Gesture Recognizers

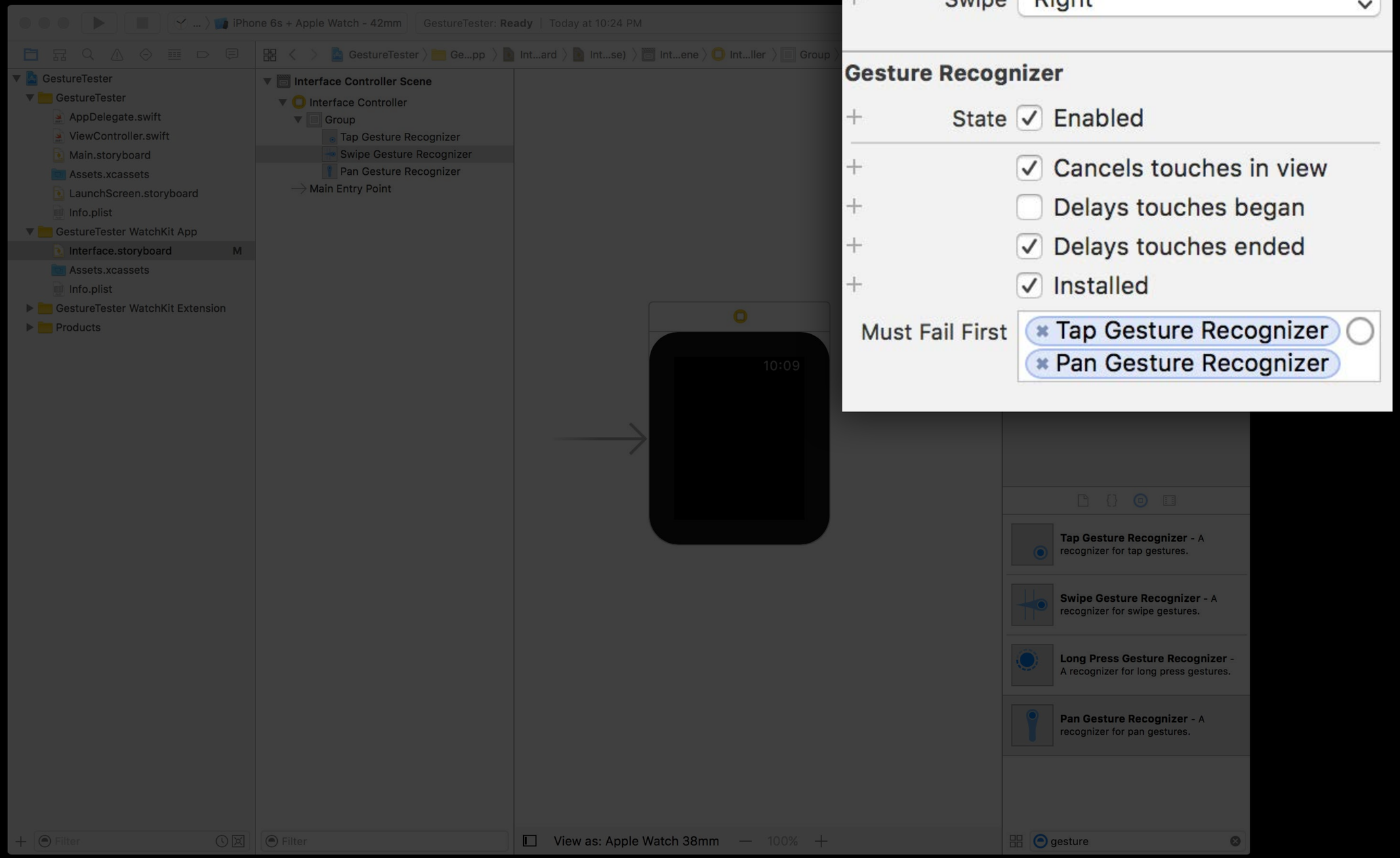
Four new classes

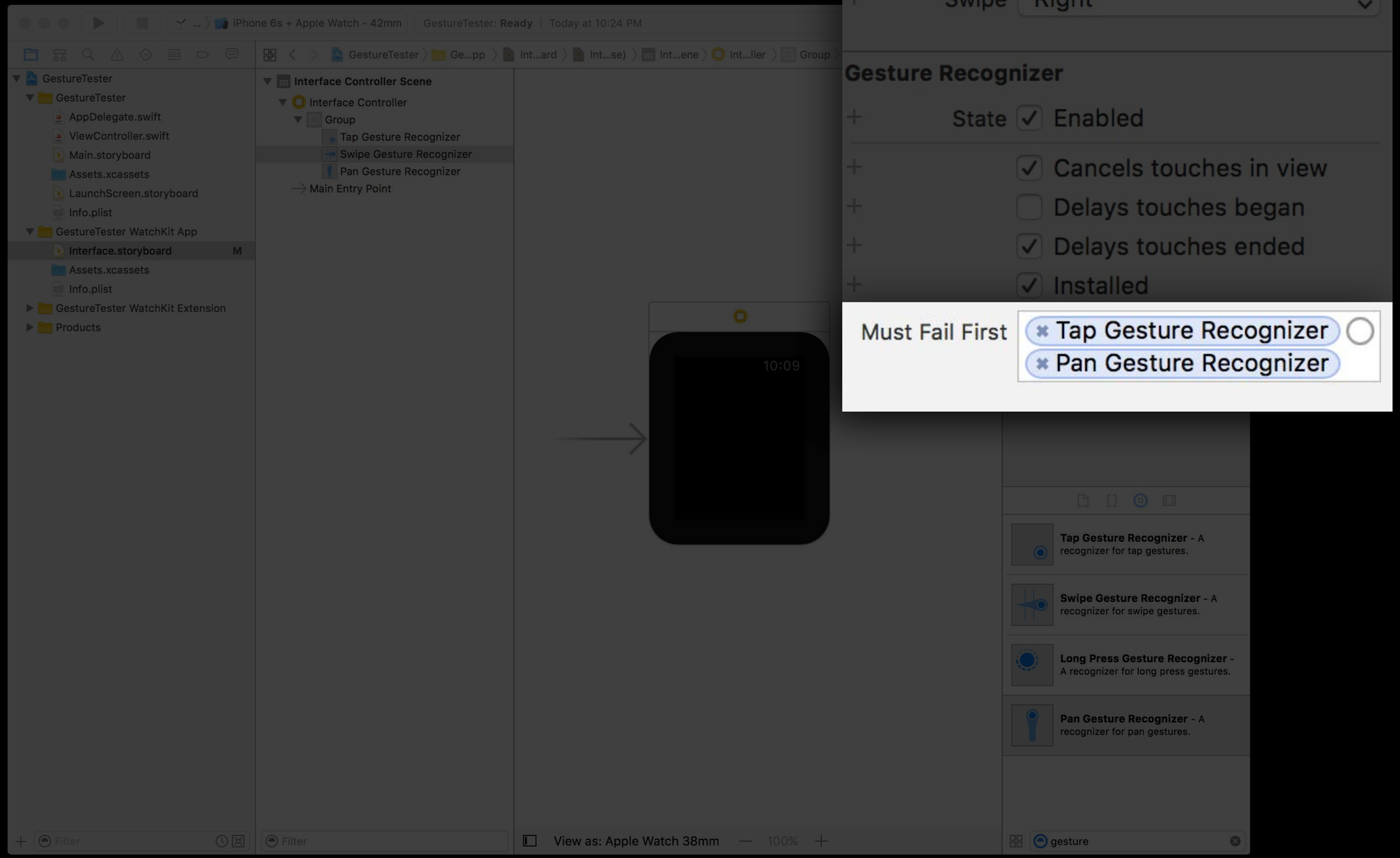
Gesture	Class
Tap	WKTapGestureRecognizer
Pan	WKPanGestureRecognizer
Swipe	WKSwipeGestureRecognizer
Long Press	WKLongPressGestureRecognizer


iPhone 6s + Apple Watch - 42mm
GestureTester: Ready | Today at 10:24 PM
GestureTester
GestureTester
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
GestureTester WatchKit App
Interface.storyboard
M
Assets.xcassets
Info.plist
GestureTester WatchKit Extension
Products
Filter
GestureTester
Ge...pp
Int...ard
Int...se)
Int...ene
Int...ller
Group
Swipe Gesture Recognizer
Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
Filter
10:09
View as: Apple Watch 38mm
100%
Swipe Gesture Recognizer
Swipe
Right
Gesture Recognizer
State
Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
Must Fail First
Tap Gesture Recognizer
Pan Gesture Recognizer
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
gesture


iPhone 6s + Apple Watch - 42mm
GestureTester: Ready | Today at 10:24 PM
GestureTester
GestureTester
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
GestureTester WatchKit App
Interface.storyboard
Assets.xcassets
Info.plist
GestureTester WatchKit Extension
Products
Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
Swipe Gesture Recognizer
Swipe Right
Gesture Recognizer
State Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
10:09
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
gesture
View as: Apple Watch 38mm
100%


Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
10:09
Swipe Gesture Recognizer
Swipe Right
Gesture Recognizer
State Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
Must Fail First
Tap Gesture Recognizer
Pan Gesture Recognizer
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
View as: Apple Watch 38mm
100%
gesture


iPhone 6s + Apple Watch - 42mm
GestureTester: Ready | Today at 10:24 PM
GestureTester
GestureTester
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
GestureTester WatchKit App
Interface.storyboard
Assets.xcassets
Info.plist
GestureTester WatchKit Extension
Products
Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
10:09
Swipe Gesture Recognizer
Swipe
Right
Gesture Recognizer
State
Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
Must Fail First
Tap Gesture Recognizer
Pan Gesture Recognizer
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
Filter
Filter
View as: Apple Watch 38mm
100%
gesture


iPhone 6s + Apple Watch - 42mm
GestureTester: Ready | Today at 10:24 PM
GestureTester
GestureTester
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
GestureTester WatchKit App
Interface.storyboard
Assets.xcassets
Info.plist
GestureTester WatchKit Extension
Products
Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
10:09
Swipe Gesture Recognizer
Swipe
Right
Gesture Recognizer
State
Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
Must Fail First
Tap Gesture Recognizer
Pan Gesture Recognizer
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
View as: Apple Watch 38mm
100%
gesture

Touch Gestures

Conflicts

Touch Gestures

Conflicts

Touch Gestures

Conflicts

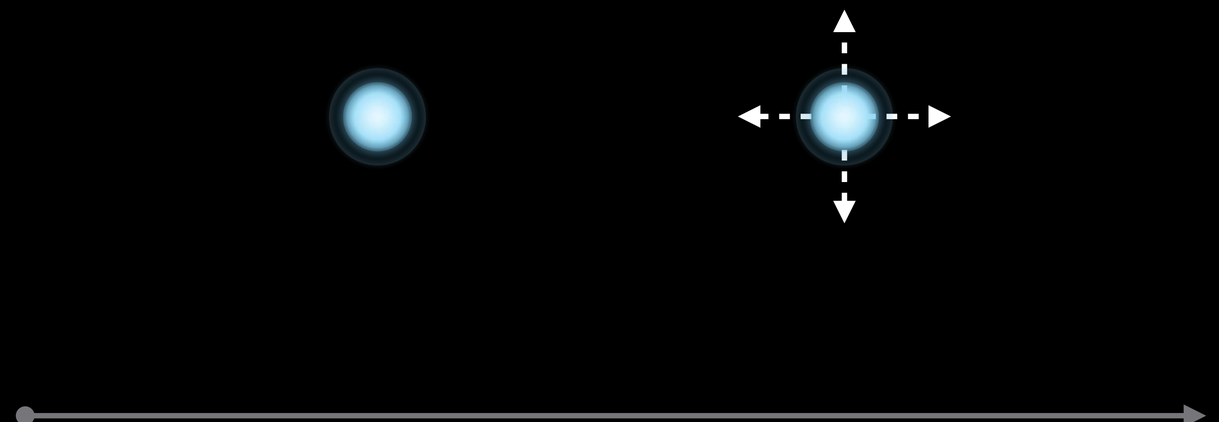
Touch Gestures

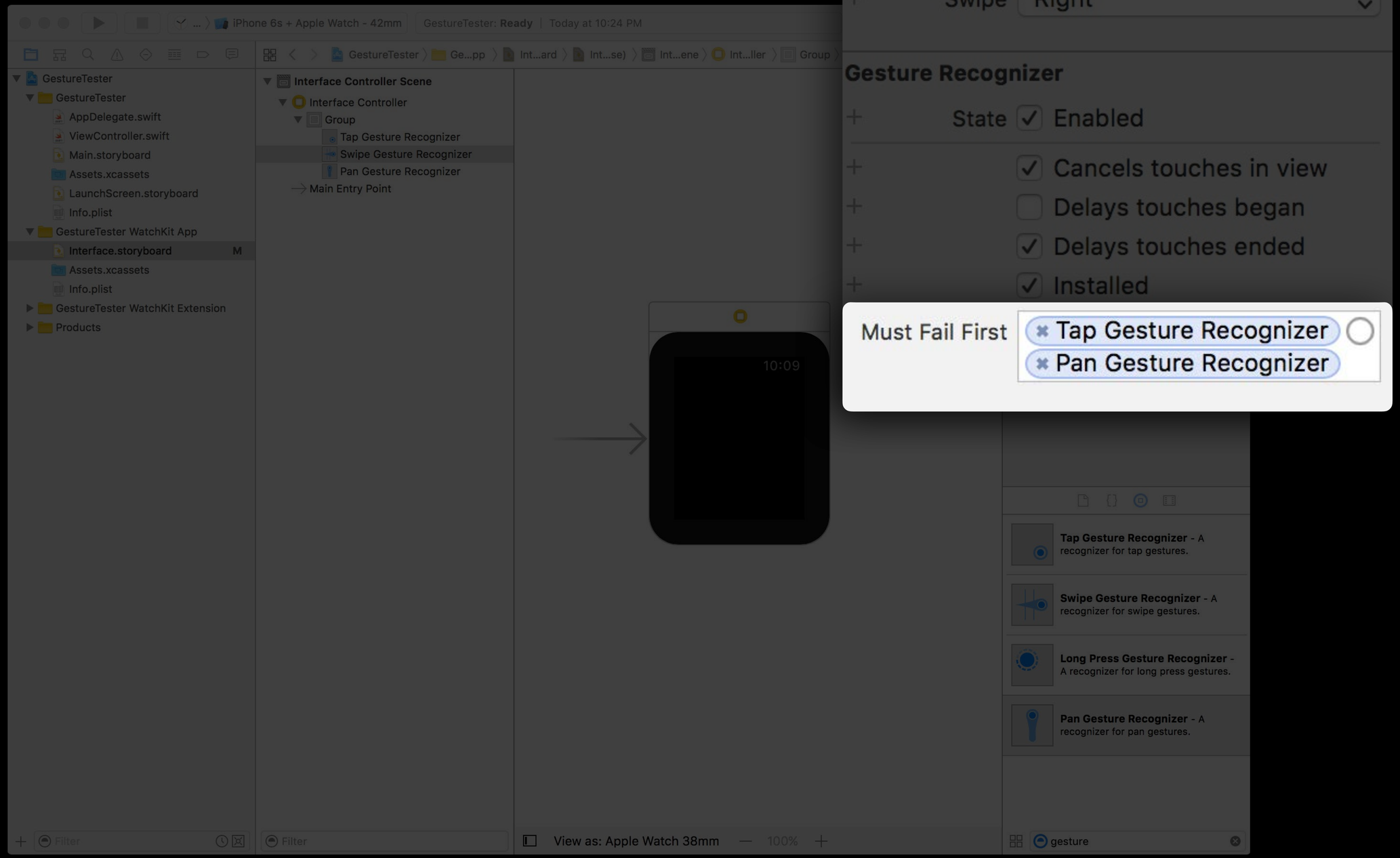
Swipe vs. tap vs. pan

Versus

Touch Gestures

Swipe yields




iPhone 6s + Apple Watch - 42mm
GestureTester: Ready | Today at 10:24 PM
GestureTester
GestureTester
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
GestureTester WatchKit App
Interface.storyboard
Assets.xcassets
Info.plist
GestureTester WatchKit Extension
Products
Interface Controller Scene
Interface Controller
Group
Tap Gesture Recognizer
Swipe Gesture Recognizer
Pan Gesture Recognizer
Main Entry Point
10:09
Swipe Gesture Recognizer
Swipe Right
Gesture Recognizer
State Enabled
Cancels touches in view
Delays touches began
Delays touches ended
Installed
Must Fail First
Tap Gesture Recognizer
Pan Gesture Recognizer
Tap Gesture Recognizer - A recognizer for tap gestures.
Swipe Gesture Recognizer - A recognizer for swipe gestures.
Long Press Gesture Recognizer - A recognizer for long press gestures.
Pan Gesture Recognizer - A recognizer for pan gestures.
View as: Apple Watch 38mm
100%
gesture

WKGestureRecognizer

Handling gestures

Associate recognizer with action method

```
IBAction func handleGesture(recognizer : WKGestureRecognizer?)
```

WKGestureRecognizer

Handling gestures

Associate recognizer with action method

```
IBAction func handleGesture(recognizer : WKGestureRecognizer?)
```

Recognizer properties

```
func locationInObject() -> CGPoint
func objectBounds() -> CGRect
var state : WKGestureRecognizerState
```

```
// Adopting Touch Gestures

// Handle pan gesture
@IBAction func handlePan(panGesture: WKPanGestureRecognizer) {
    let location = panGesture.locationInObject()
    let bounds = panGesture.objectBounds()
    switch panGesture.state {
       case .begin:
            self.setupRotation(withLocation: location, bounds)
       case .changed, .ended, .cancelled:
            self.updateRotation(withLocation: location, bounds)
    }
}
```

```
// Adopting Touch Gestures

// Handle pan gesture
@IBAction func handlePan(panGesture: WKPanGestureRecognizer) {
    let location = panGesture.locationInObject()
    let bounds = panGesture.objectBounds()
    switch panGesture.state {
       case .begin:
            self.setupRotation(withLocation: location, bounds)
       case .changed, .ended, .cancelled:
            self.updateRotation(withLocation: location, bounds)
    }
}
```

```
// Adopting Touch Gestures

// Handle pan gesture
@IBAction func handlePan(panGesture: WKPanGestureRecognizer) {
    let location = panGesture.locationInObject()
    let bounds = panGesture.objectBounds()
    switch panGesture.state {
       case .begin:
            self.setupRotation(withLocation: location, bounds)
       case .changed, .ended, .cancelled:
            self.updateRotation(withLocation: location, bounds)
    }
}
```

```
// Adopting Touch Gestures

// Handle pan gesture
@IBAction func handlePan(panGesture: WKPanGestureRecognizer) {
    let location = panGesture.locationInObject()
    let bounds = panGesture.objectBounds()
    switch panGesture.state {
       case .begin:
            self.setupRotation(withLocation: location, bounds)
       case .changed, .ended, .cancelled:
            self.updateRotation(withLocation: location, bounds)
    }
}
```

Touch Gestures

Considerations

Short interactions

Keep content visible

Discoverable

Touch Gestures

Considerations

Short interactions

Keep content visible

Discoverable

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

Digital Crown

Analog control

Simple analog control

- Selection
- Scrolling
- Game paddle

WKCrownSequencer

Direct access to crown events

Object that reports state of digital crown

Accessed through property on controller

```
public var crownSequencer: WKCrownSequencer
```

WKCrownSequencer

Direct access to crown events

Object that reports state of digital crown

Accessed through property on controller

```
public var crownSequencer: WKCrownSequencer
```

Current state

```
var rotationsPerSecond : Double
var isIdle : Bool
```

WKCrownSequencer

Direct access to crown events

Object that reports state of digital crown

Accessed through property on controller

```
public var crownSequencer: WKCrownSequencer
```

Current state

```
var rotationsPerSecond : Double
var isIdle : Bool
```

WKCrownDelegate

- Reports change deltas

WKCrownDelegate

Protocol to track events

```
optional func crownDidRotate(sequencer : WKCrownSequencer?, rotationalDelta: Double)
```

WKCrownDelegate

Protocol to track events

```
optional func crownDidRotate(sequencer : WKCrownSequencer?, rotationalDelta: Double)
```

Direction of reported values

+ away from user

- towards user

True for left or right wrist

WKCrownDelegate

Protocol to track events

```
optional func crownDidRotate(sequencer : WKCrownSequencer?, rotationalDelta: Double)
```

Direction of reported values

+ away from user

- towards user

True for left or right wrist

```
optional func crownDidBecomeIdle(sequencer : WKCrownSequencer?)
```

Digital Crown

Recap

Great for games

Easy to adopt

Accelerometer

CoreMotion

Enhances other interactions

- Tilt on a pinball game
- Add spin to a pool shot

Accelerometer

CoreMotion

Enhances other interactions

- Tilt on a pinball game
- Add spin to a pool shot

Accelerometer

CoreMotion

Available through `CMMotionManger()`

Accelerometer

CoreMotion

Available through `CMMotionManger()`

Considerations

- Screen may turn off due to user motion
- App put in background

Accelerometer

CoreMotion

Available through `CMMotionManger()`

Considerations

- Screen may turn off due to user motion
- App put in background

Recommendation

- Background processing

```
NSProcessInfo performExpiringActivityWithReason(_:,usingBlock:)
```

```
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
                withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
                if let data = data {
                    self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
                }
        })
    }
}

override func didDeactivate() {
    super.didDeactivate()
    motionManager.stopAccelerometerUpdates()
}
```

```
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
            withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
            if let data = data {
                self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
            }
        })
    }
}

override func didDeactivate() {
    super.didDeactivate()
    motionManager.stopAccelerometerUpdates()
}
```

```
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
            withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
            if let data = data {
                self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
            }
        })
    }
}

override func didDeactivate() {
    super.didDeactivate()
    motionManager.stopAccelerometerUpdates()
}
```

```swift
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
            withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
            if let data = data {
                self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
            }
        })
    }
}

override func didDeactivate() {
    super.didDeactivate()
    motionManager.stopAccelerometerUpdates()
}
```

```
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
            withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
            if let data = data {
                self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
            }
        })
    }
}

override func didDeactivate() {
    super.didDeactivate()
    motionManager.stopAccelerometerUpdates()
}
```

```swift
// Adopting Accelerometer

let motionManager = CMMotionManager()

override func willActivate() {
    super.willActivate()

    if motionManager.isAccelerometerAvailable {

        // set a sampling rate
        motionManager.accelerometerUpdateInterval = 1.0/30.0

        // start getting updates
        motionManager.startAccelerometerUpdates(to: OperationQueue.current()!,
                withHandler: { (data: CMAccelerometerData?, error: NSError?) -> Void in
                if let data = data {
                    self.gameScene.didReceiveAccelerometerUpdate(accelerometerData: data)
                }
            })
    }
}

    override func didDeactivate() {
        super.didDeactivate()
        motionManager.stopAccelerometerUpdates()
    }
```

Accelerometer

Recap

Enhances other interactions

Screen blanking

Use when needed

Haptic Feedback

Makes game feel real

Combination of vibration and tone

Haptic Feedback

Makes game feel real

Combination of vibration and tone

Haptic Feedback

Makes game feel real

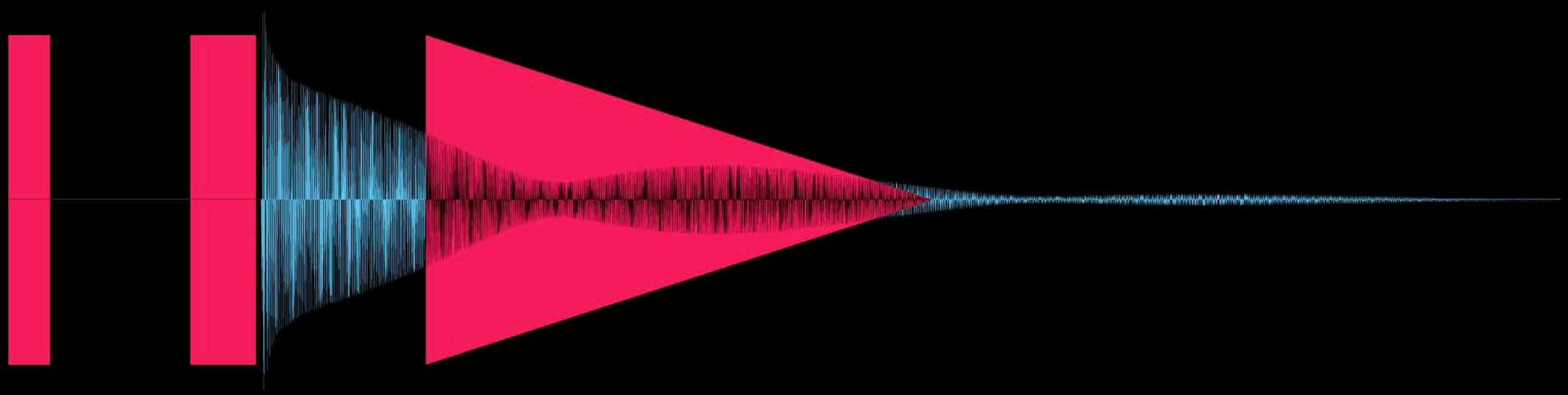
Combination of vibration and tone

Nine types

- Up/down
- Start/stop
- Success/failure/retry
- Notification
- Click

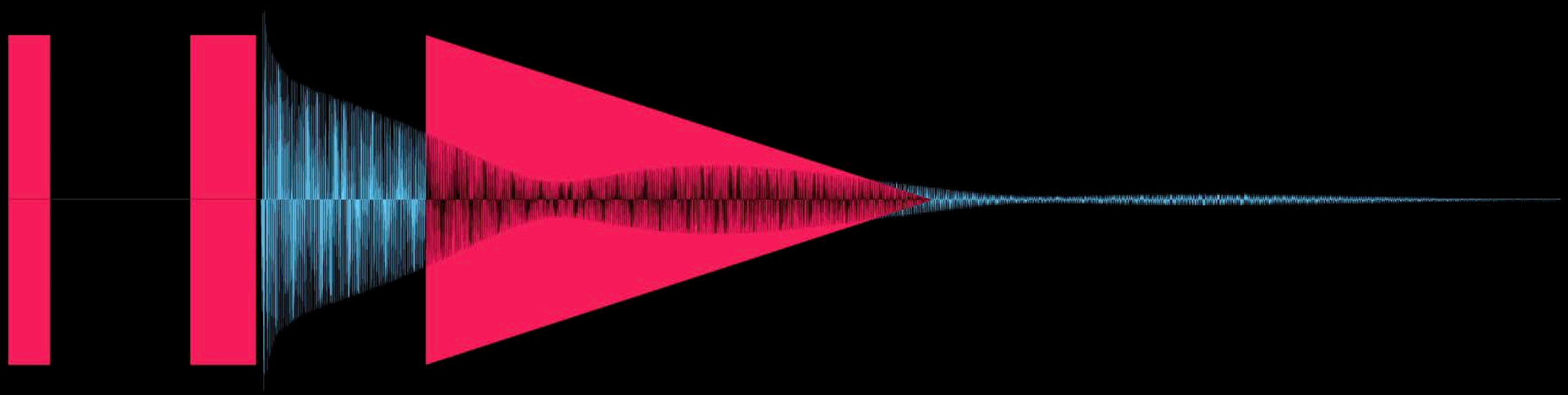
Haptic Feedback

Notification



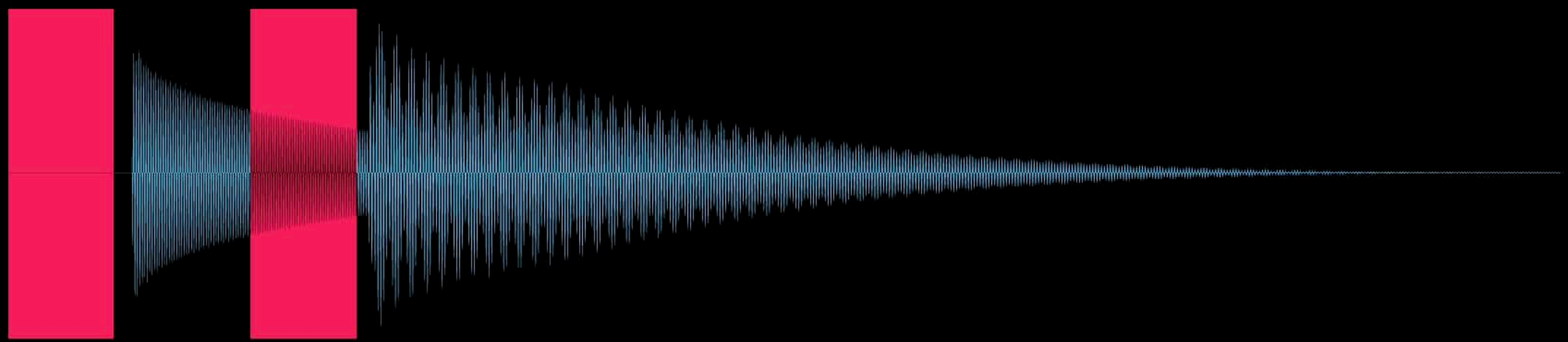
Haptic Feedback

Notification



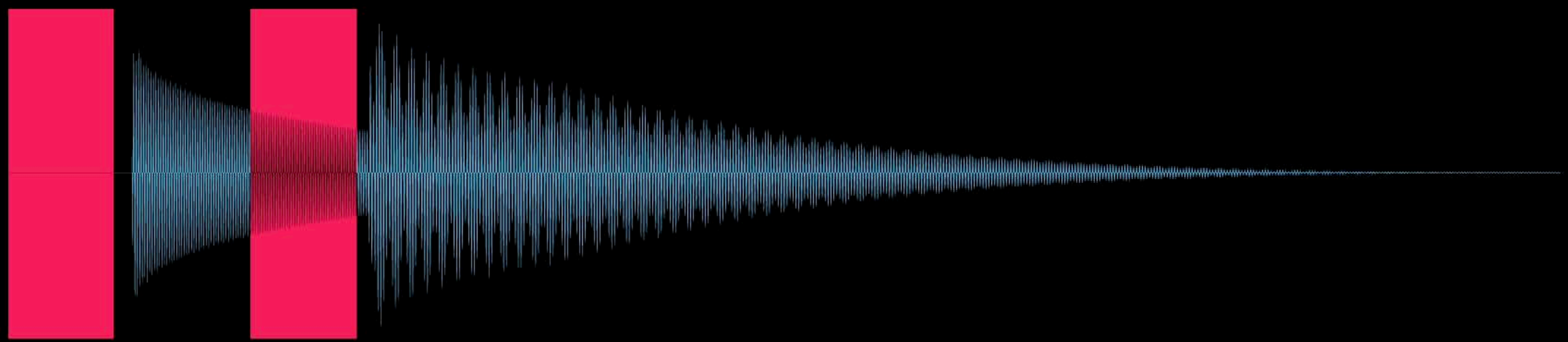
Haptic Feedback

Direction down



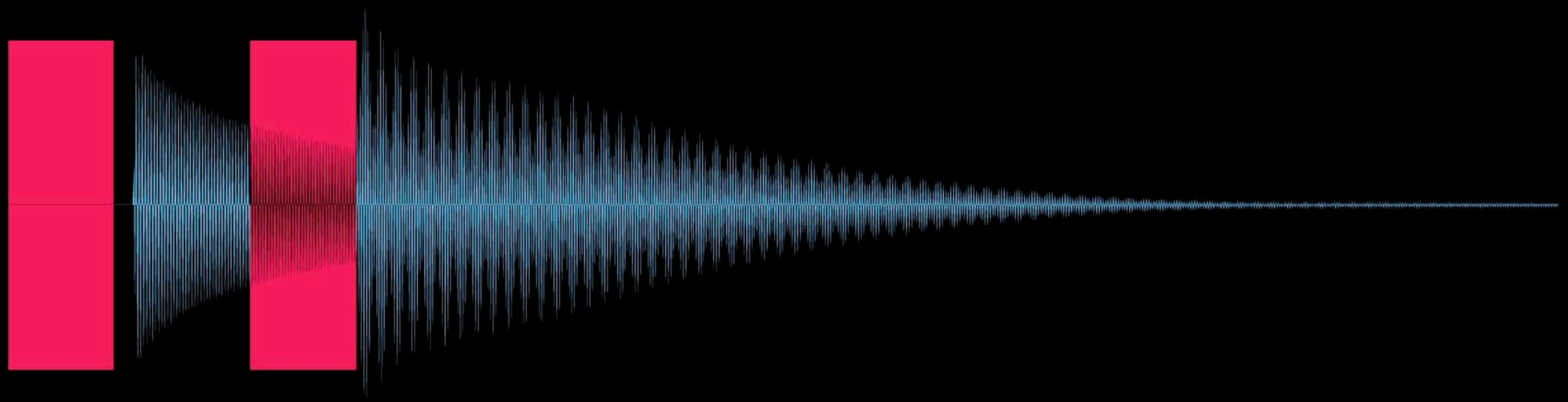
Haptic Feedback

Direction down



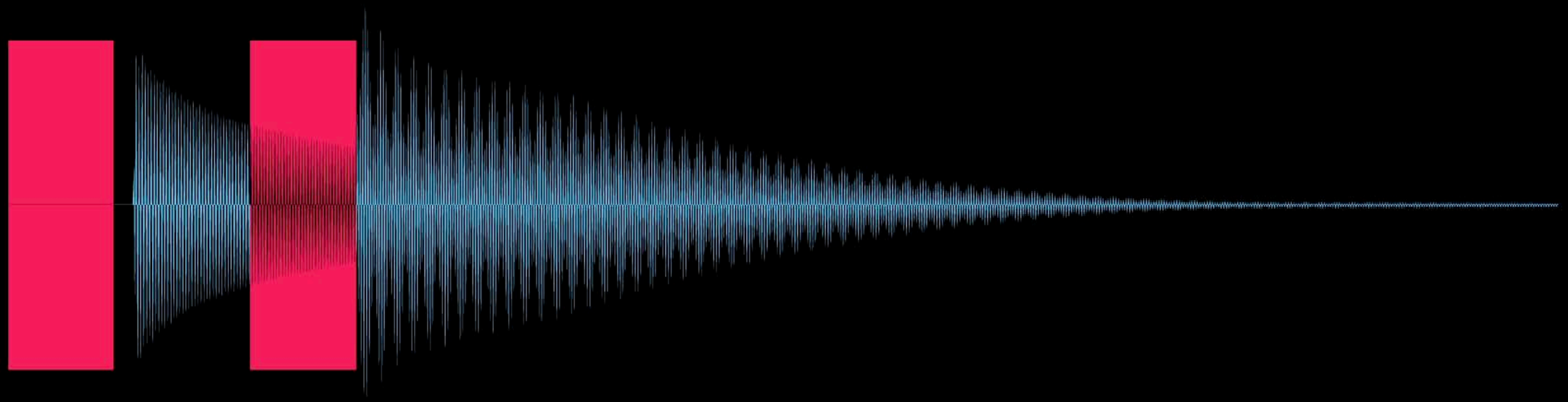
Haptic Feedback

Direction up



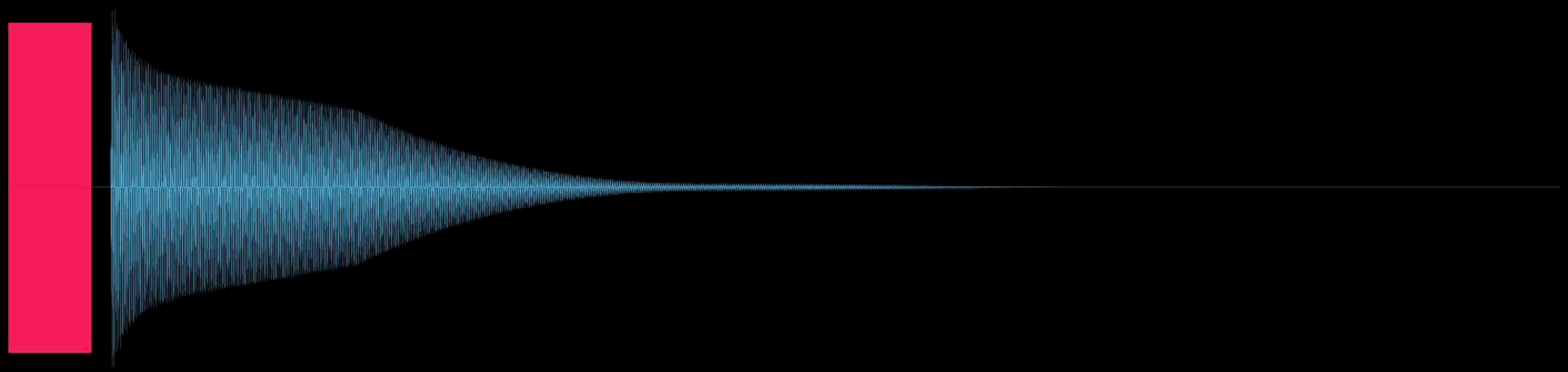
Haptic Feedback

Direction up



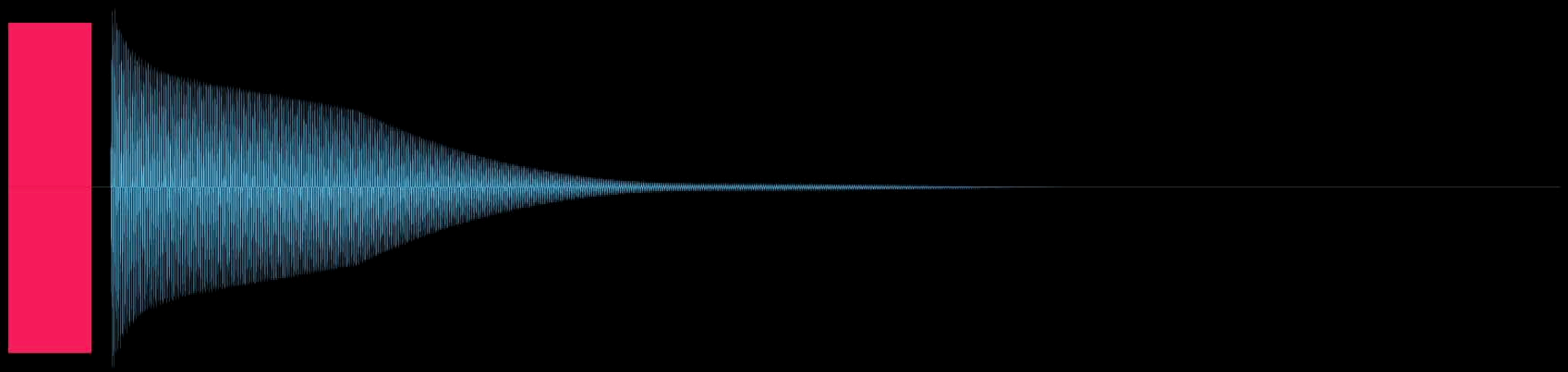
Haptic Feedback

Start



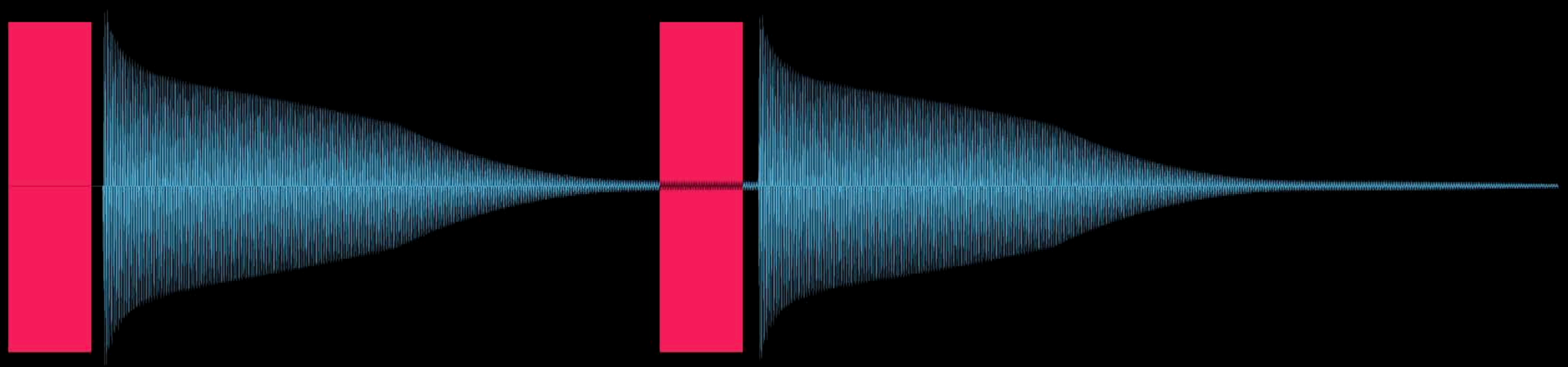
Haptic Feedback

Start



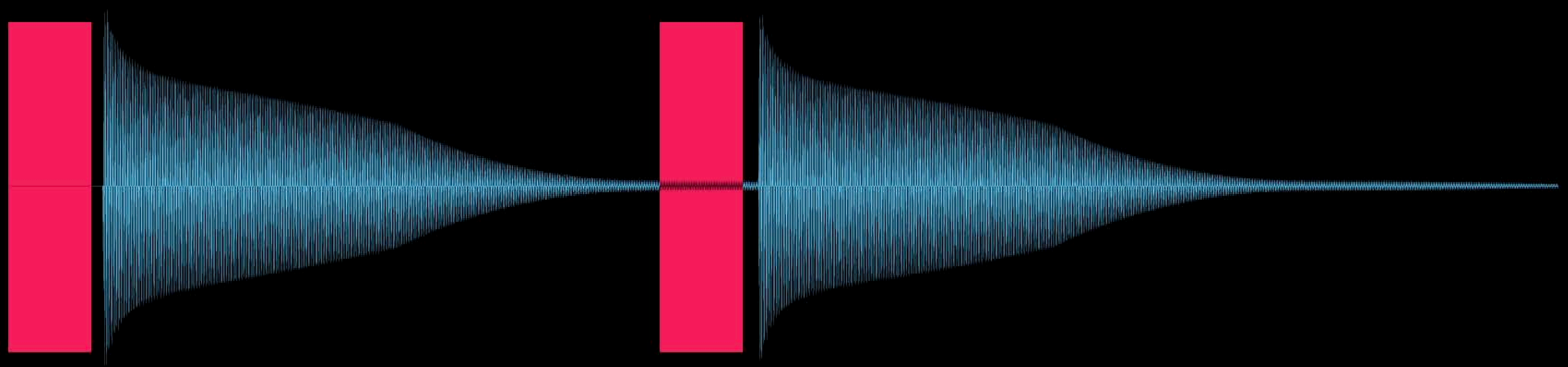
Haptic Feedback

Stop



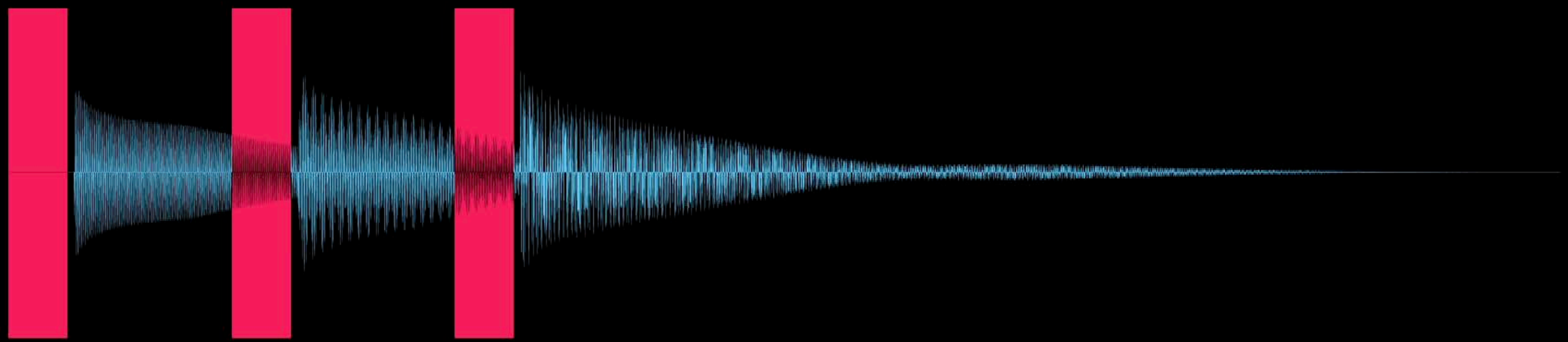
Haptic Feedback

Stop



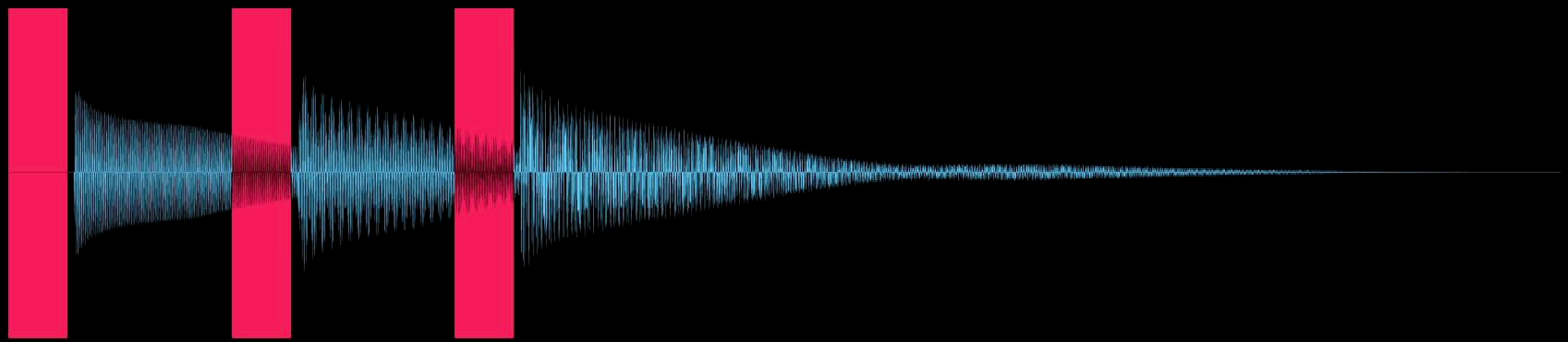
Haptic Feedback

Success



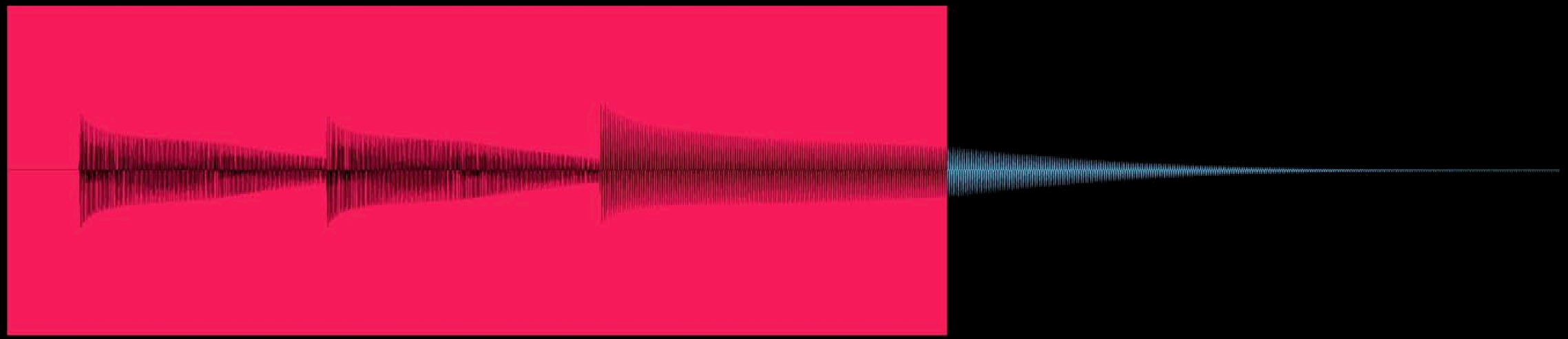
Haptic Feedback

Success



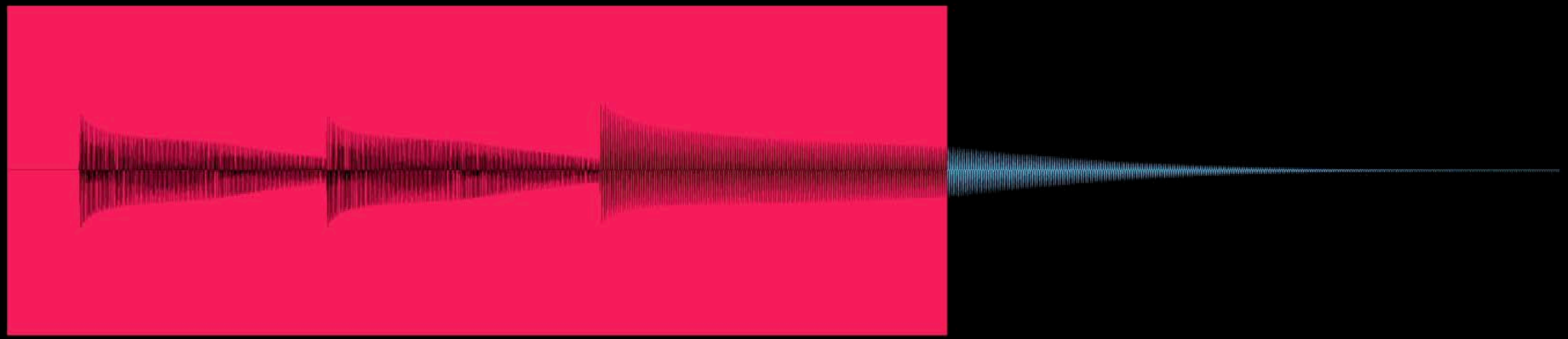
Haptic Feedback

Failure



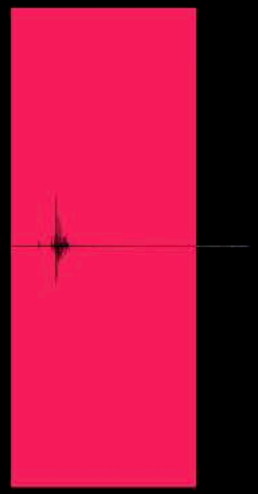
Haptic Feedback

Failure



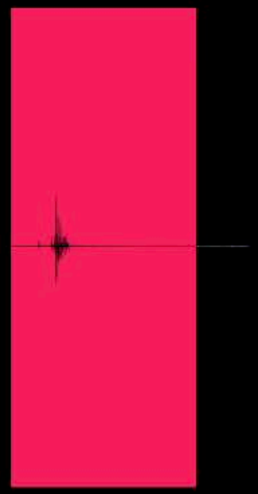
Haptic Feedback

Click



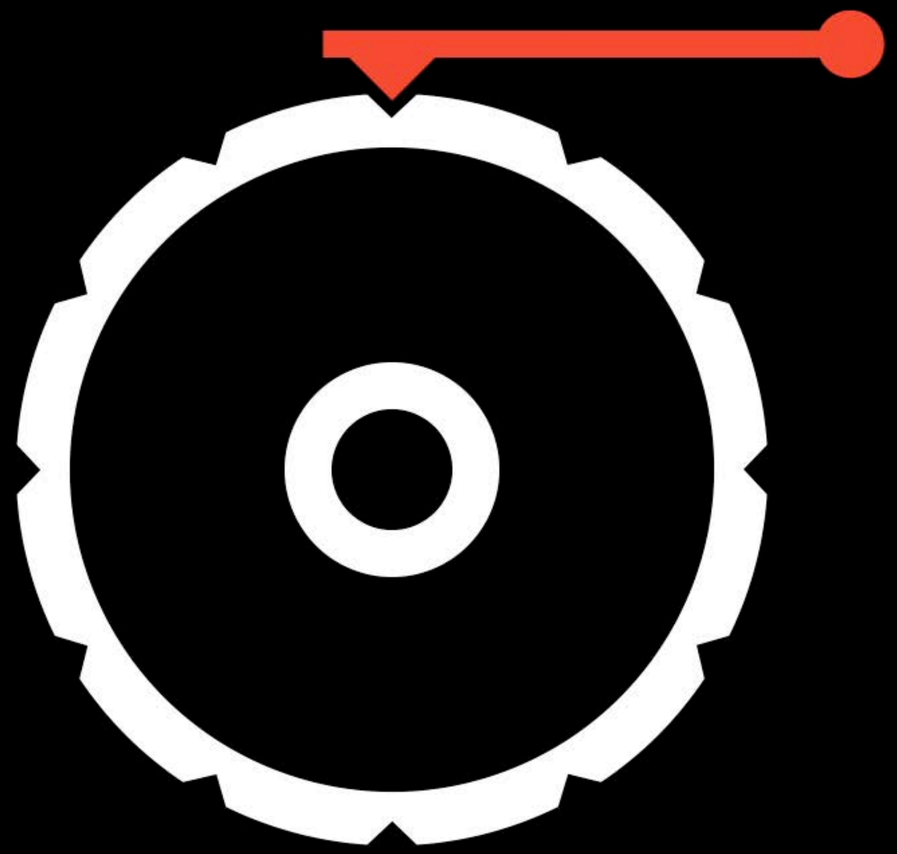
Haptic Feedback

Click



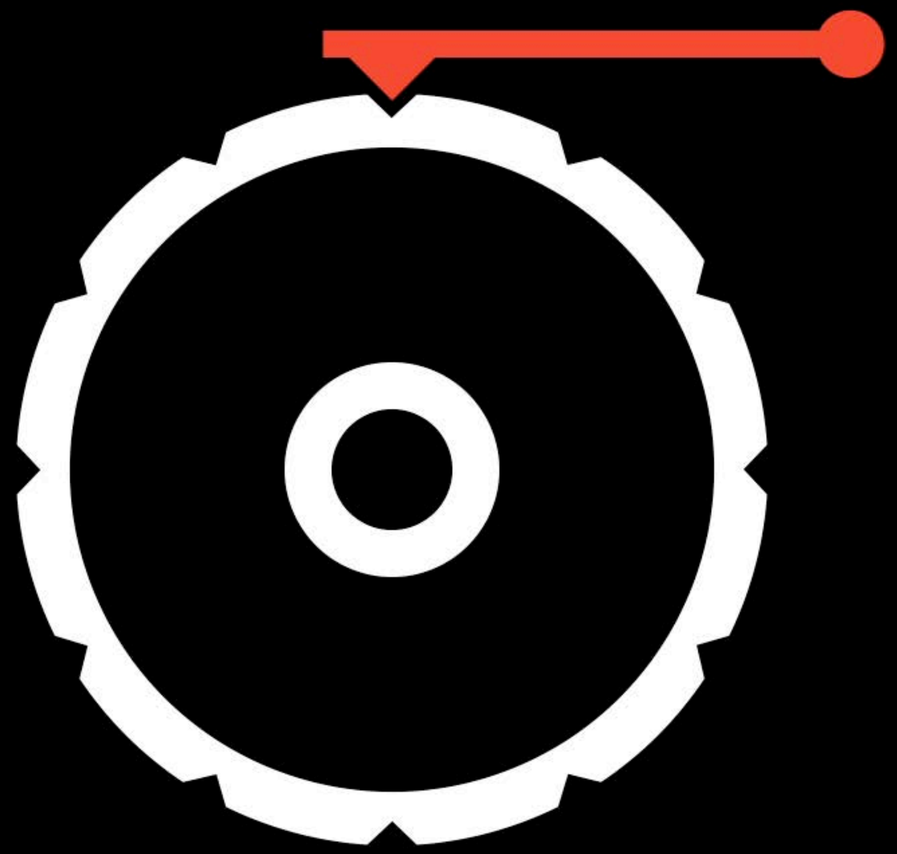
Click

Open a lock



Click

Open a lock



```
// Haptic Feedback
// Example

WKInterfaceDevice.currentDevice().play(.success)
```

```
// Haptic Feedback
// Example

WKInterfaceDevice.currentDevice().play(.success)

enum WKHapticType : Int {
    case notification
    case directionUp
    case directionDown
    case success
    case failure
    case retry
    case start
    case stop
    case click
}
```

Haptic Feedback

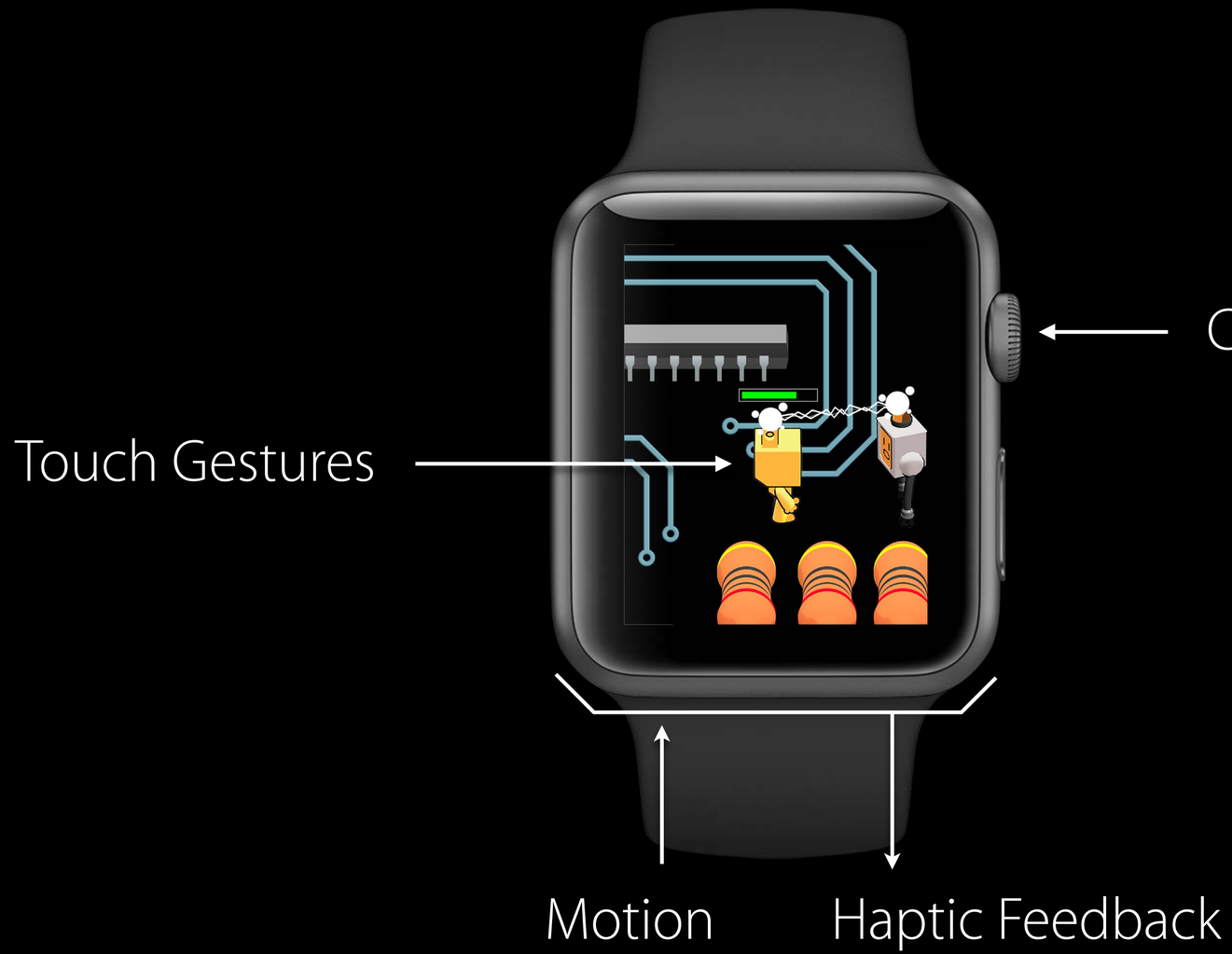
Considerations

Distinct events

Exercise restraint

Input and Feedback

Recap



Graphics Frameworks

Fatima Broom Game Technologies Engineer

SpriteKit and SceneKit

2D and 3D graphics frameworks for games

Super easy to use

Integrated tools and editors

Available on macOS, iOS, tvOS, and watchOS

SpriteKit

SceneKit

SpriteKit and SceneKit

GPU-accelerated, real-time rendering

SpriteKit and SceneKit

GPU-accelerated, real-time rendering

Features

Animations

SpriteKit

SceneKit

Animations

SpriteKit

SceneKit

Particles

SpriteKit

SceneKit

Particles

SpriteKit

SceneKit

Physics

SpriteKit

SceneKit

Physics

SpriteKit

SceneKit

Lighting

SpriteKit

SceneKit

Lighting

SpriteKit

SceneKit

Camera

SpriteKit

SceneKit

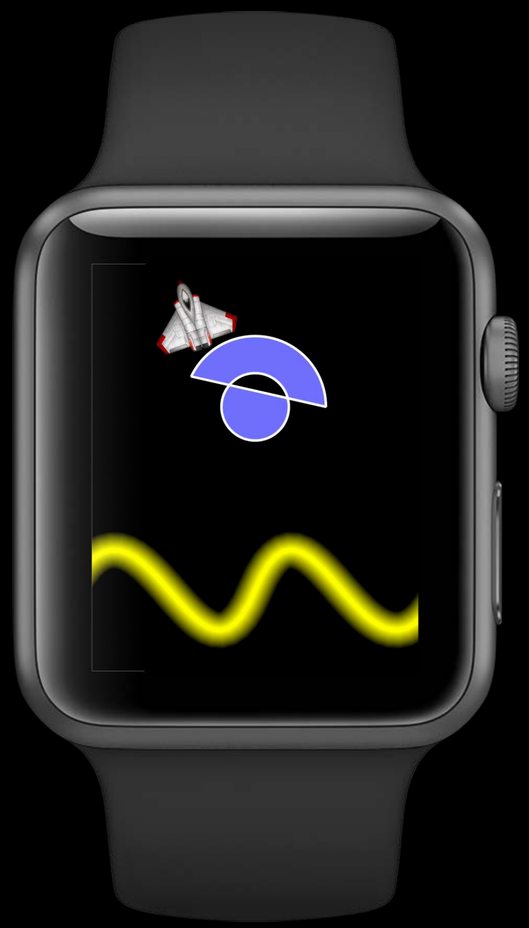
Camera

SpriteKit

SceneKit

Shapes and Text

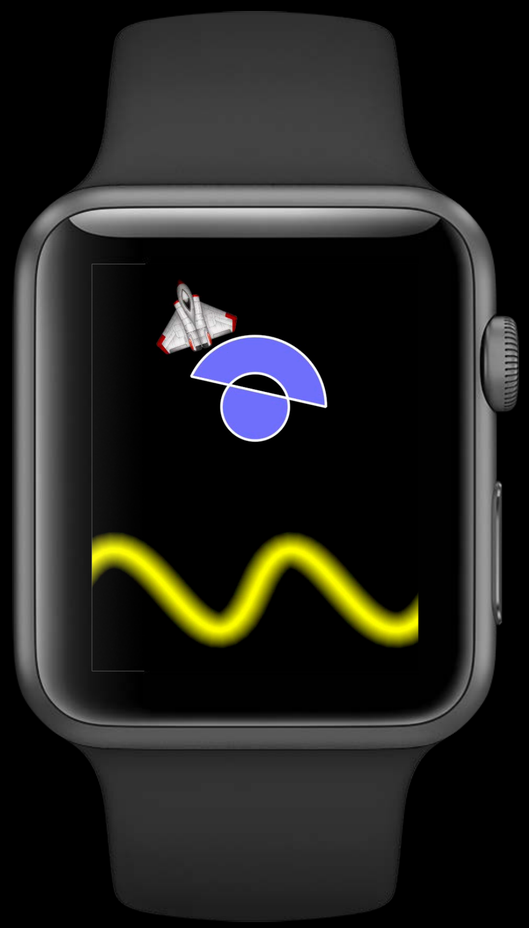
SpriteKit



SceneKit

Shapes and Text

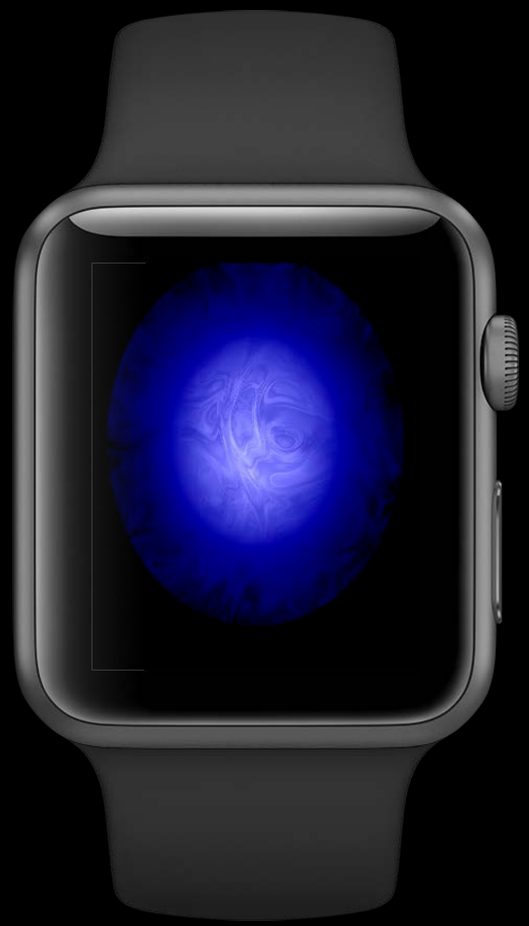
SpriteKit



SceneKit

Shaders

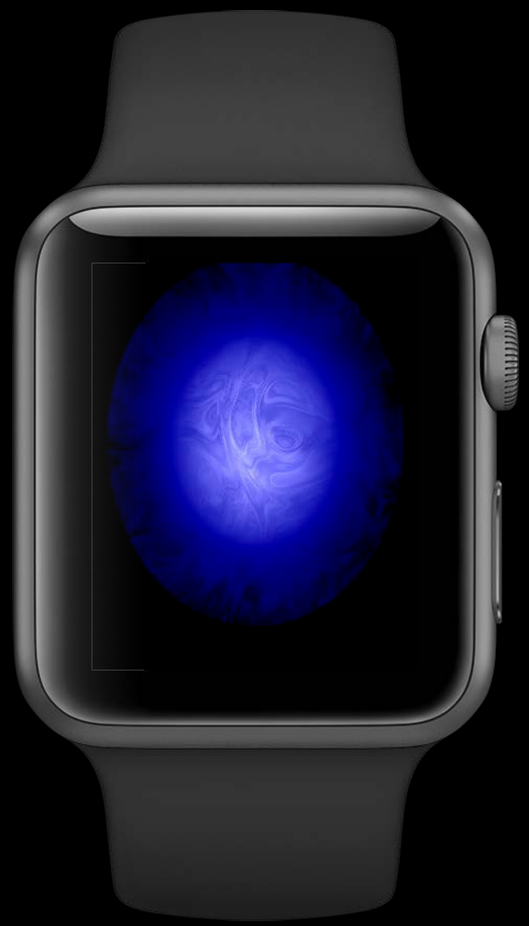
SpriteKit



SceneKit

Shaders

SpriteKit



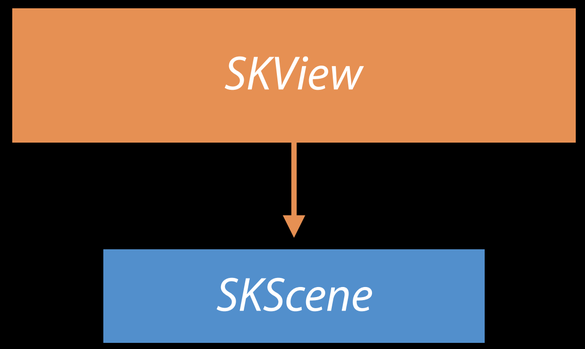
SceneKit

SpriteKit Scene Graph

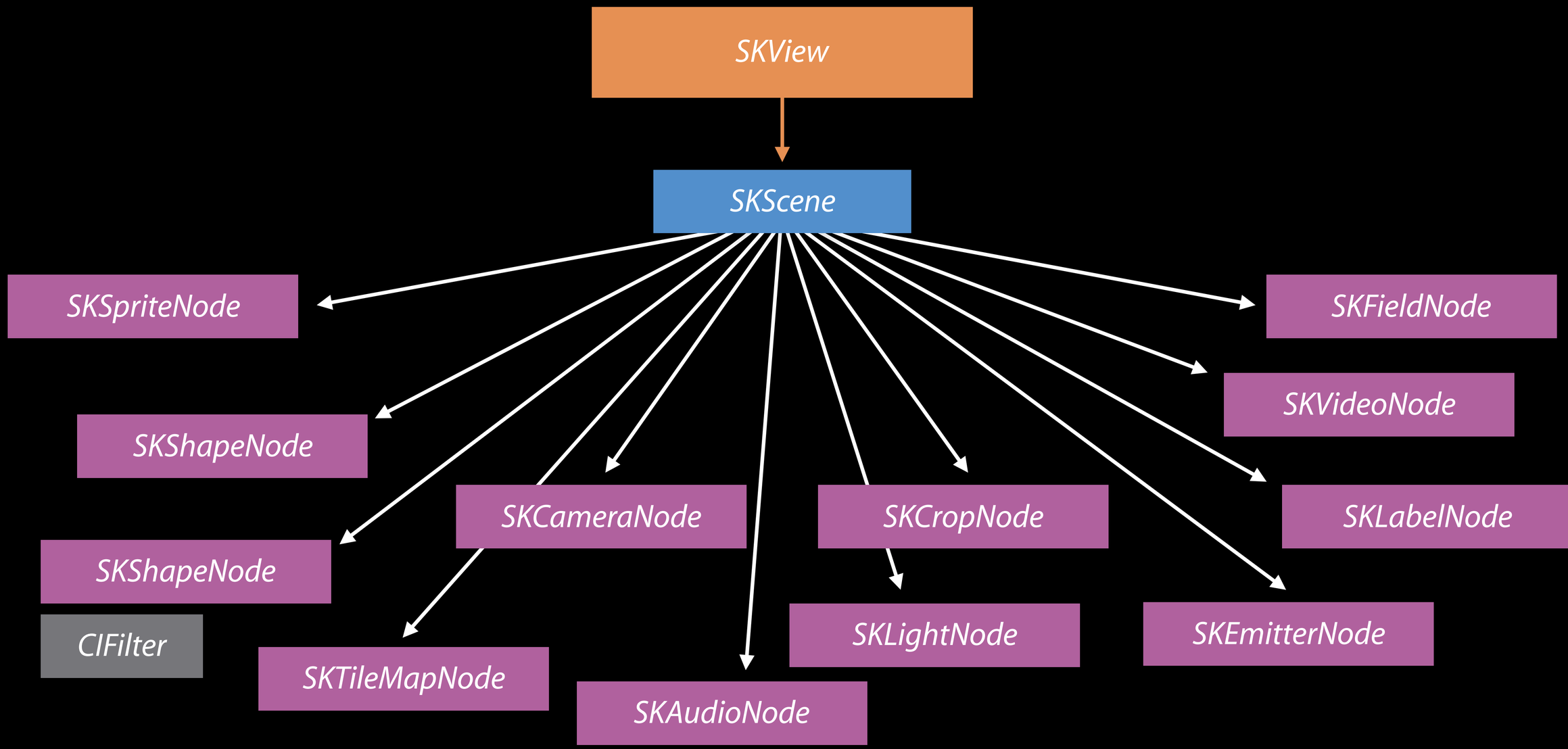
SpriteKit Scene Graph

SKView

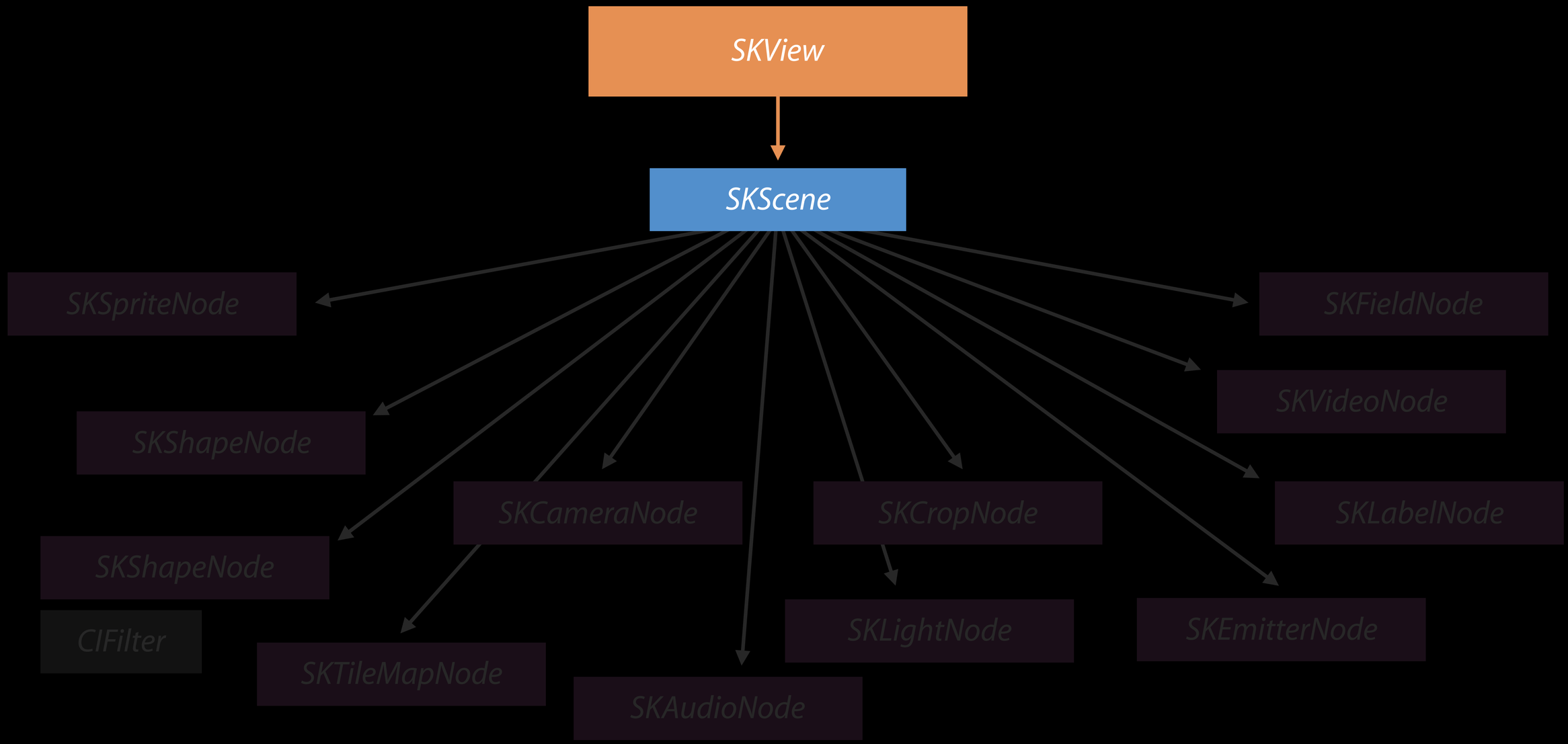
SpriteKit Scene Graph



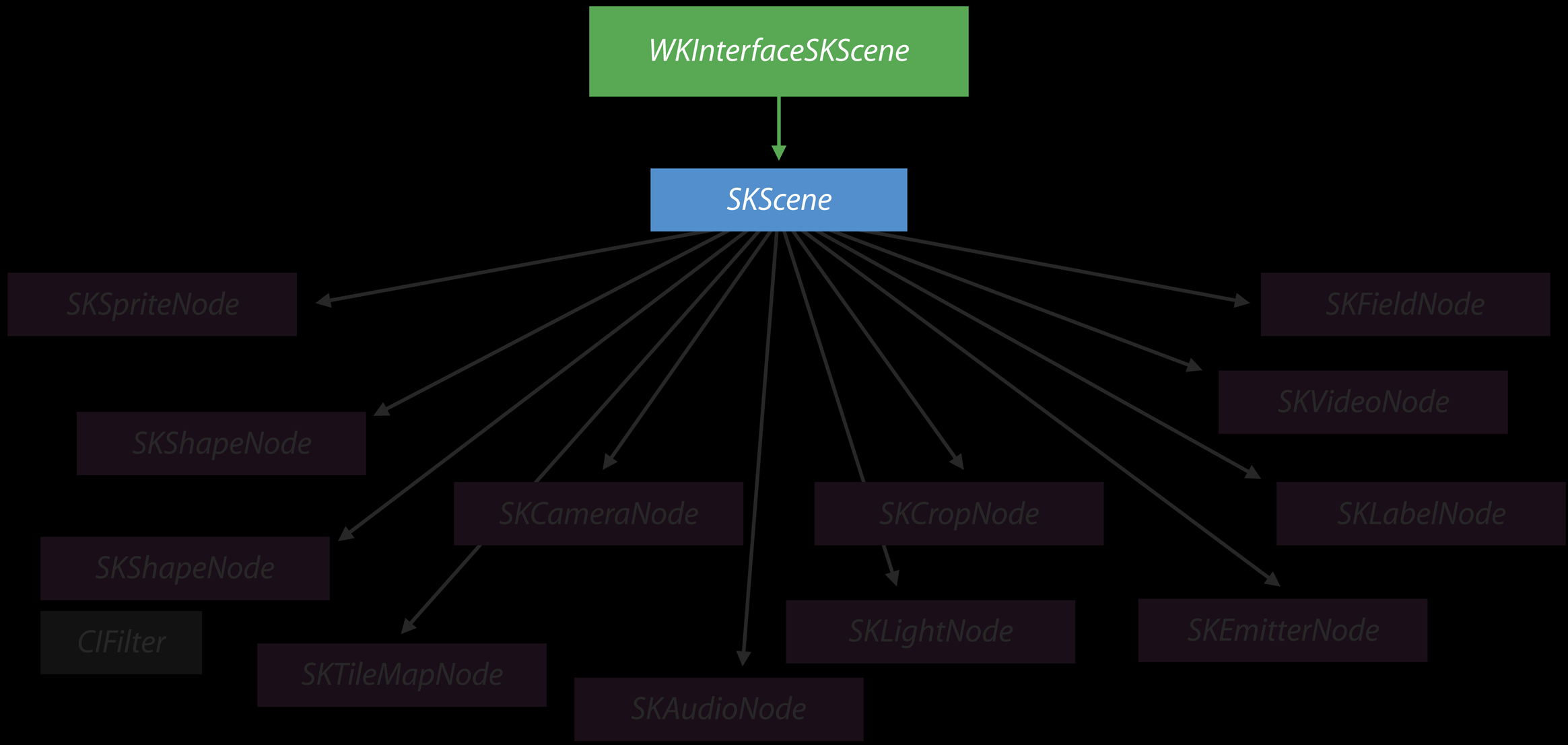
SpriteKit Scene Graph



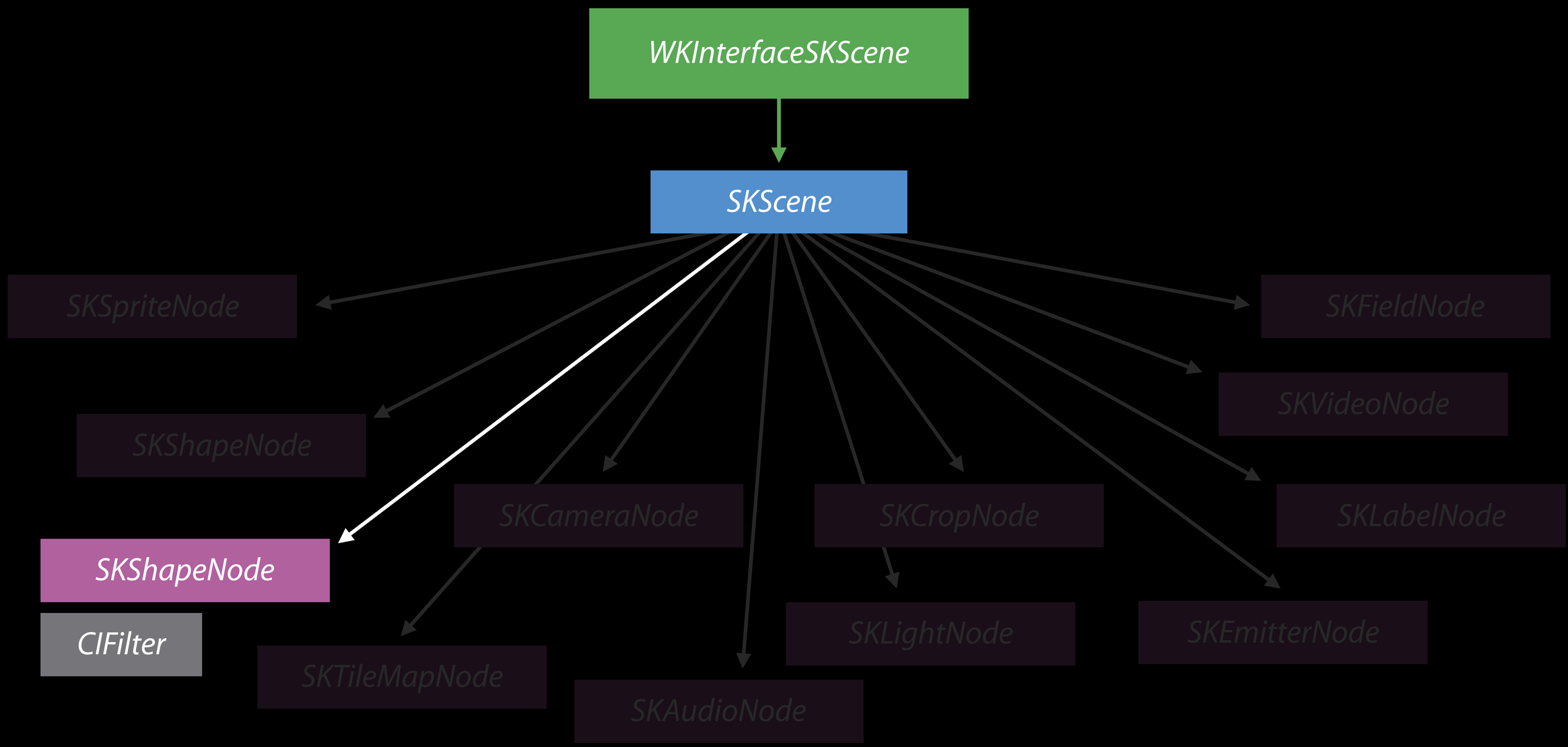
SpriteKit Scene Graph



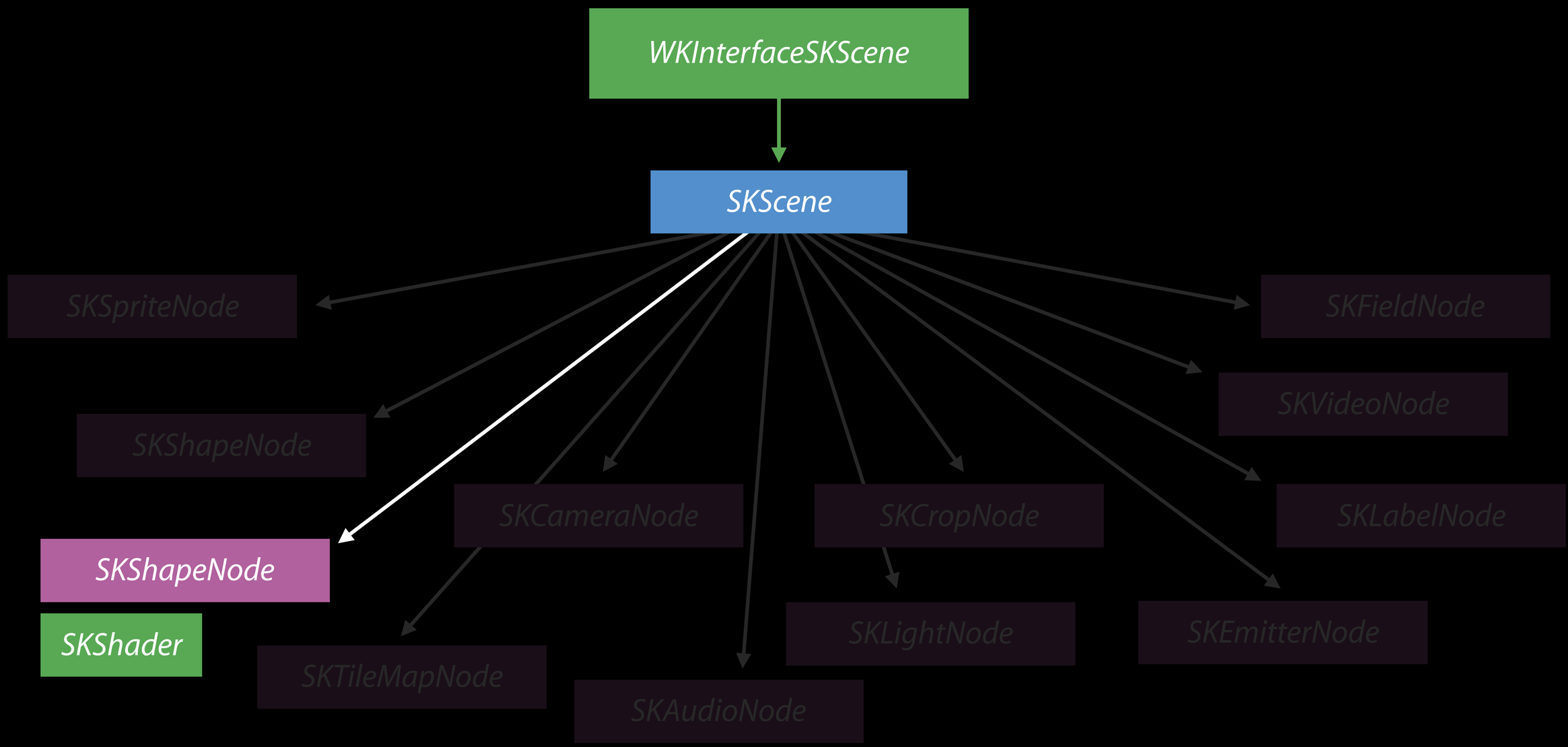
SpriteKit Scene Graph



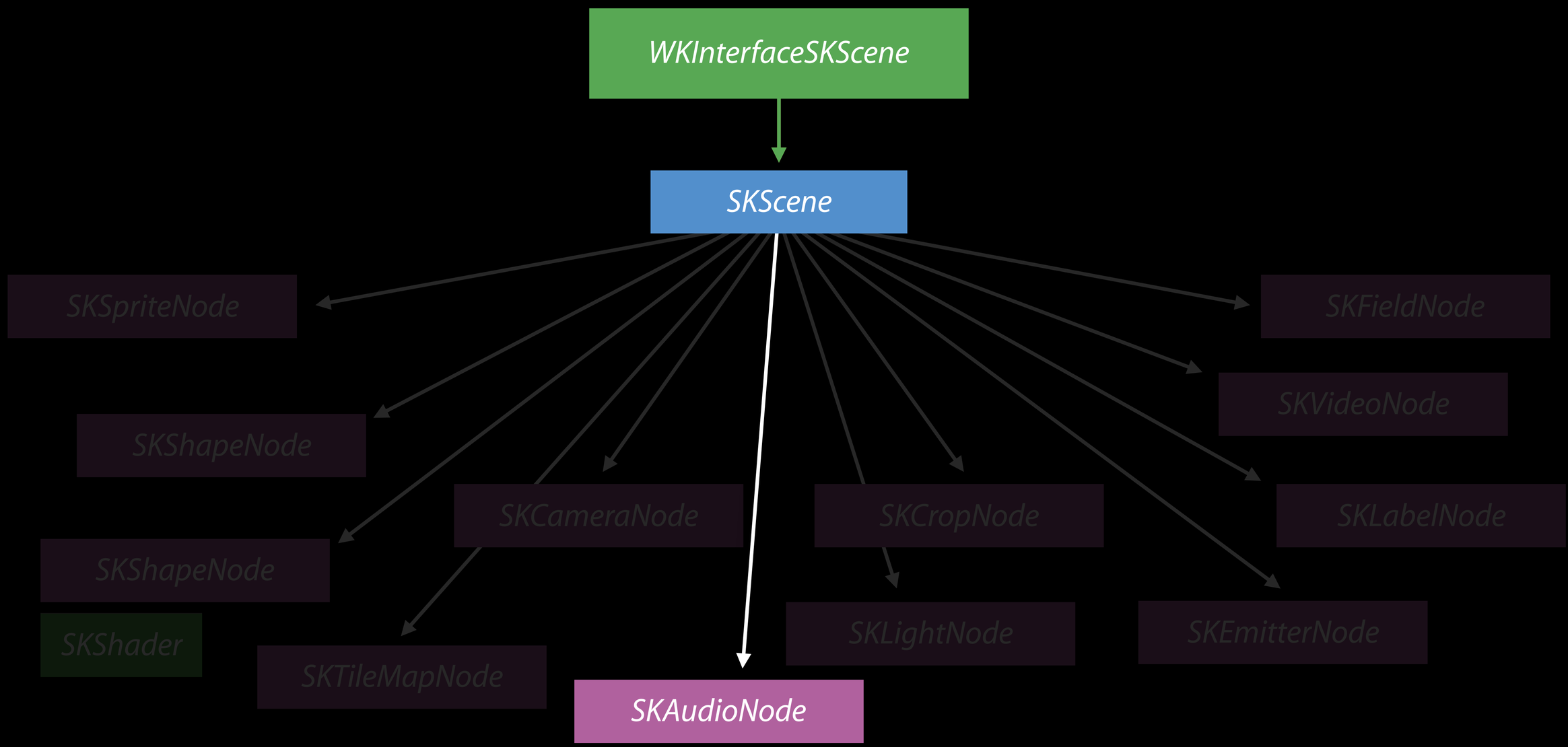
SpriteKit Scene Graph



SpriteKit Scene Graph

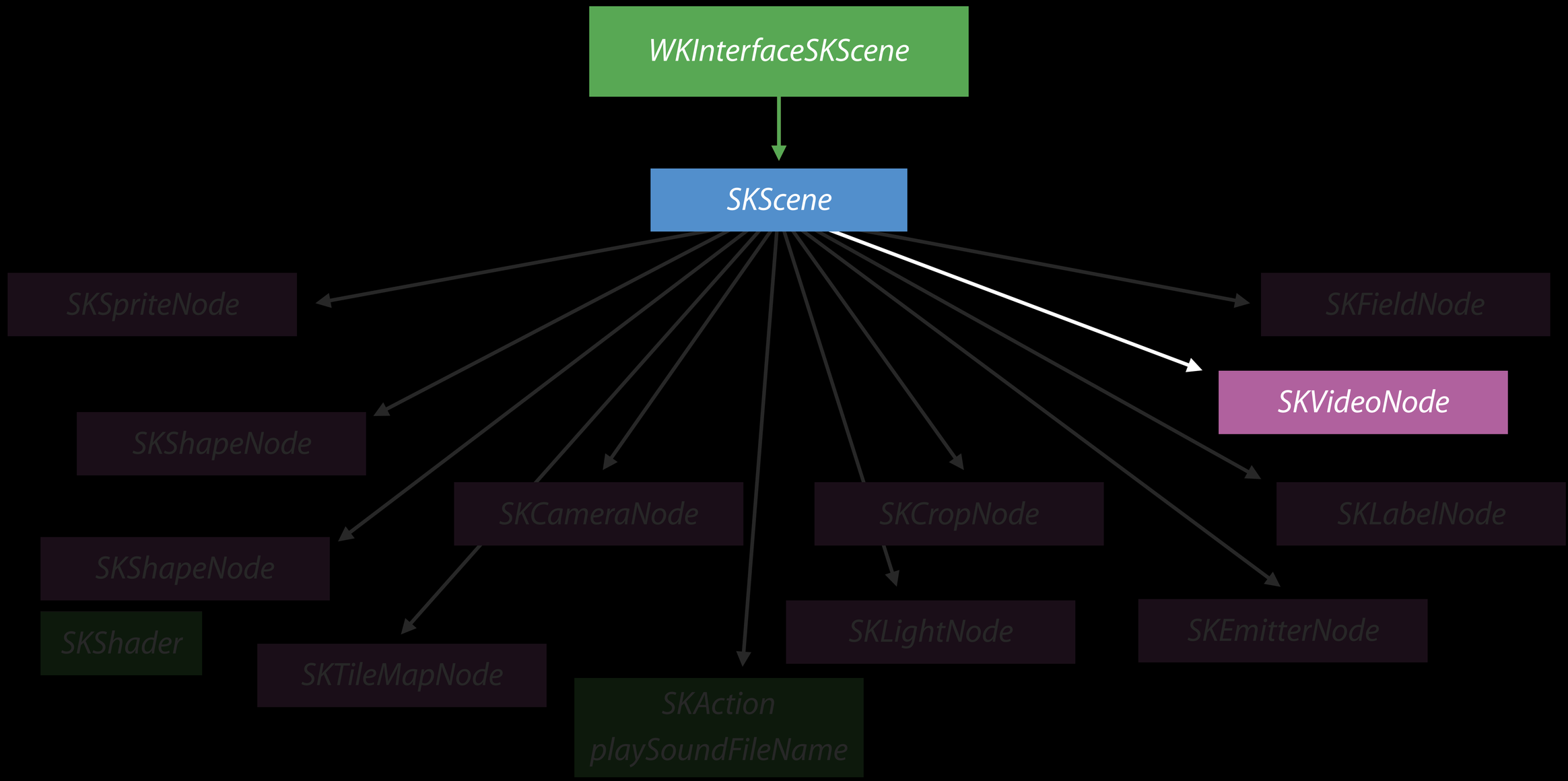


SpriteKit Scene Graph

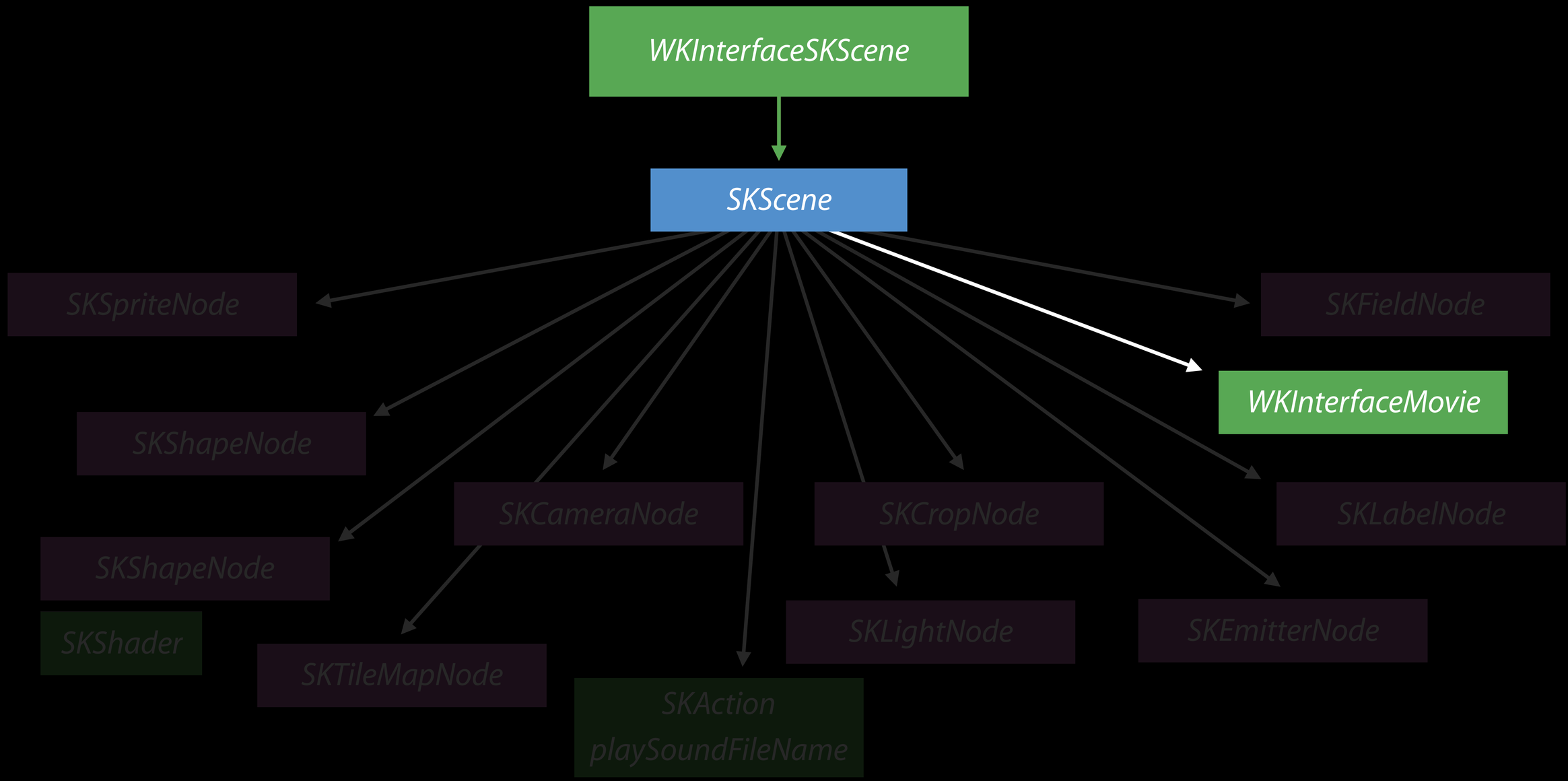


SpriteKit Scene Graph

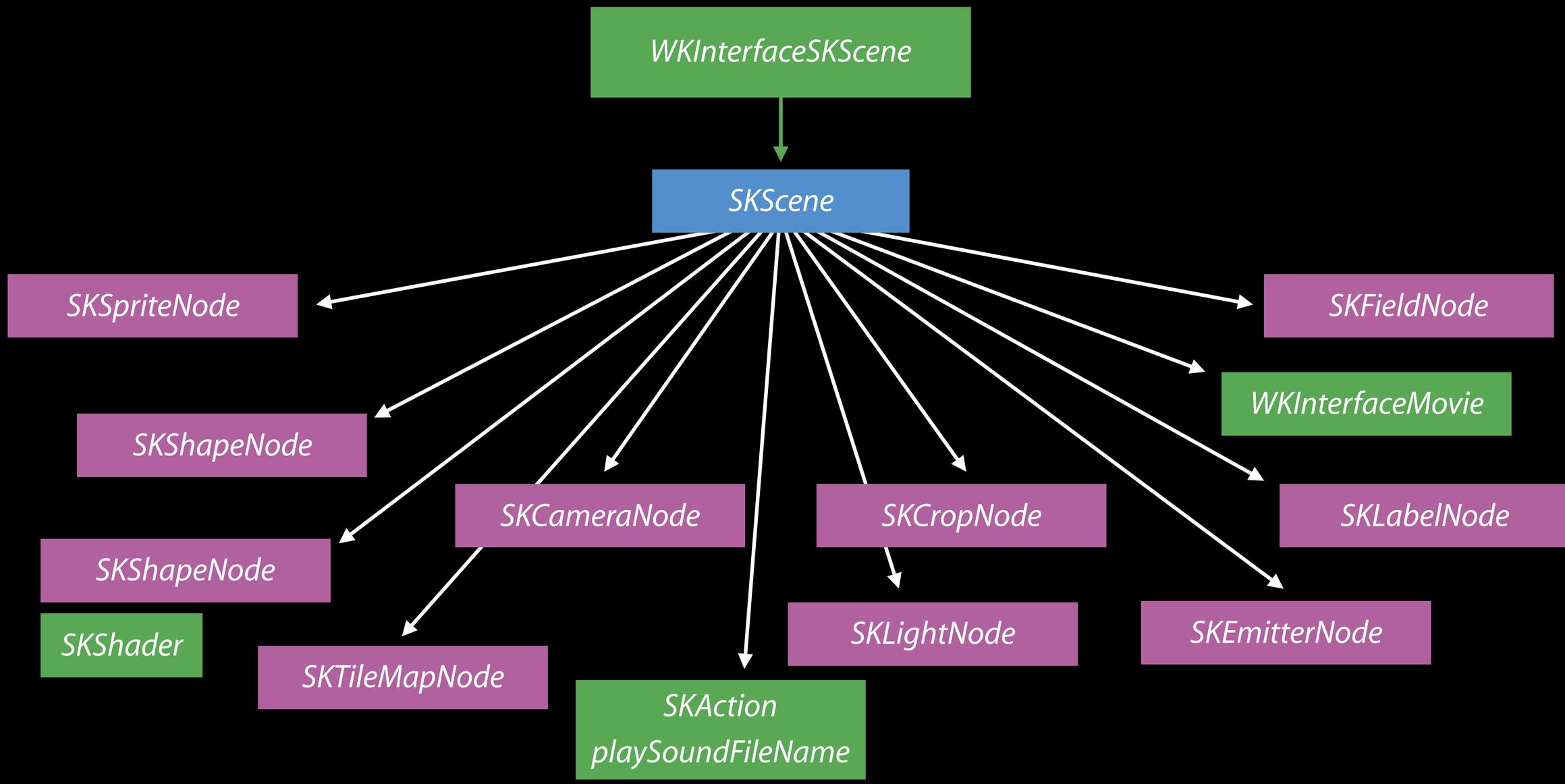
SpriteKit Scene Graph



SpriteKit Scene Graph



SpriteKit Scene Graph



Summary

	iOS	watchOS
SpriteKit	SKView	WKInterfaceSKScene
	SKAudioNode	SKAction playSoundFileNamed
	CIFilter	SKShader
	SKVideoNode/AVPlayer	WKInterfaceMovie, WKInterfaceInlineMovie

Summary

	iOS	watchOS
SpriteKit	SKView	WKInterfaceSKScene
	SKAudioNode	SKAction playSoundFileNamed
	CIFilter	SKShader
	SKVideoNode/AVPlayer	WKInterfaceMovie, WKInterfaceInlineMovie
SceneKit	SCNView	WKInterfaceSCNScene
	SCNAudioSource (Positional)	SCNAudioSource (Standard)
	CIFilter	SCNShadable (Shader Modifiers)

Getting Started

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Game 〉 iPhone 6s | Running Game on iPhone 6s

Game 〉 Game 〉 GameScene.swift 〉 didMove(to:)

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - Products

```
8
9   import SpriteKit
10  import GameplayKit
11
12  class GameScene: SKScene {
13
14      private var label : SKLabelNode?
15      private var spinnyNode : SKShapeNode?
16
17      override func didMove(to view: SKView) {
18
19          // Get label node from scene and store it for use later
20          self.label = self.childNode(withName: "//helloLabel") as?
                SKLabelNode
21          if let label = self.label {
22              label.alpha = 0.0
23              label.run(SKAction.fadeIn(withDuration: 2.0))
24          }
25
26          // Create shape node to use during mouse interaction
27          let w = (self.size.width + self.size.height) * 0.05
28          self.spinnyNode = SKShapeNode.init(rectOf:
                CGSize.init(width: w, height: w), cornerRadius: w * 0.3)
29
30          if let spinnyNode = self.spinnyNode {
31              spinnyNode.lineWidth = 2.5
32
33              spinnyNode.run(SKAction.repeatForever(SKAction.rotate
                    (byAngle: CGFloat(M_PI), duration: 1)))
34              spinnyNode.run(SKAction.sequence([SKAction.wait
                    (forDuration: 0.5),
35                                                SKAction.fadeOut
                                                  (withDuration: 0.5),
36                                                SKAction.
                                                  removeFromParent()]))
37          }
38      }
39
40
41      func touchDown(atPoint pos : CGPoint) {
42          if let n = self.spinnyNode?.copy() as! SKShapeNode? {
43              n.position = pos
44              n.strokeColor = SKColor.green()
```

Identity and Type

Name: GameScene.swift

Type: Default - Swift Source

Location: Relative to Group

GameScene.swift

Full Path: /Users/fbroom/Desktop/Game/Game/GameScene.swift

On Demand Resource Tags Show

Target Membership

Game

Text Settings

Text Encoding: Default - Unicode (UTF-8)

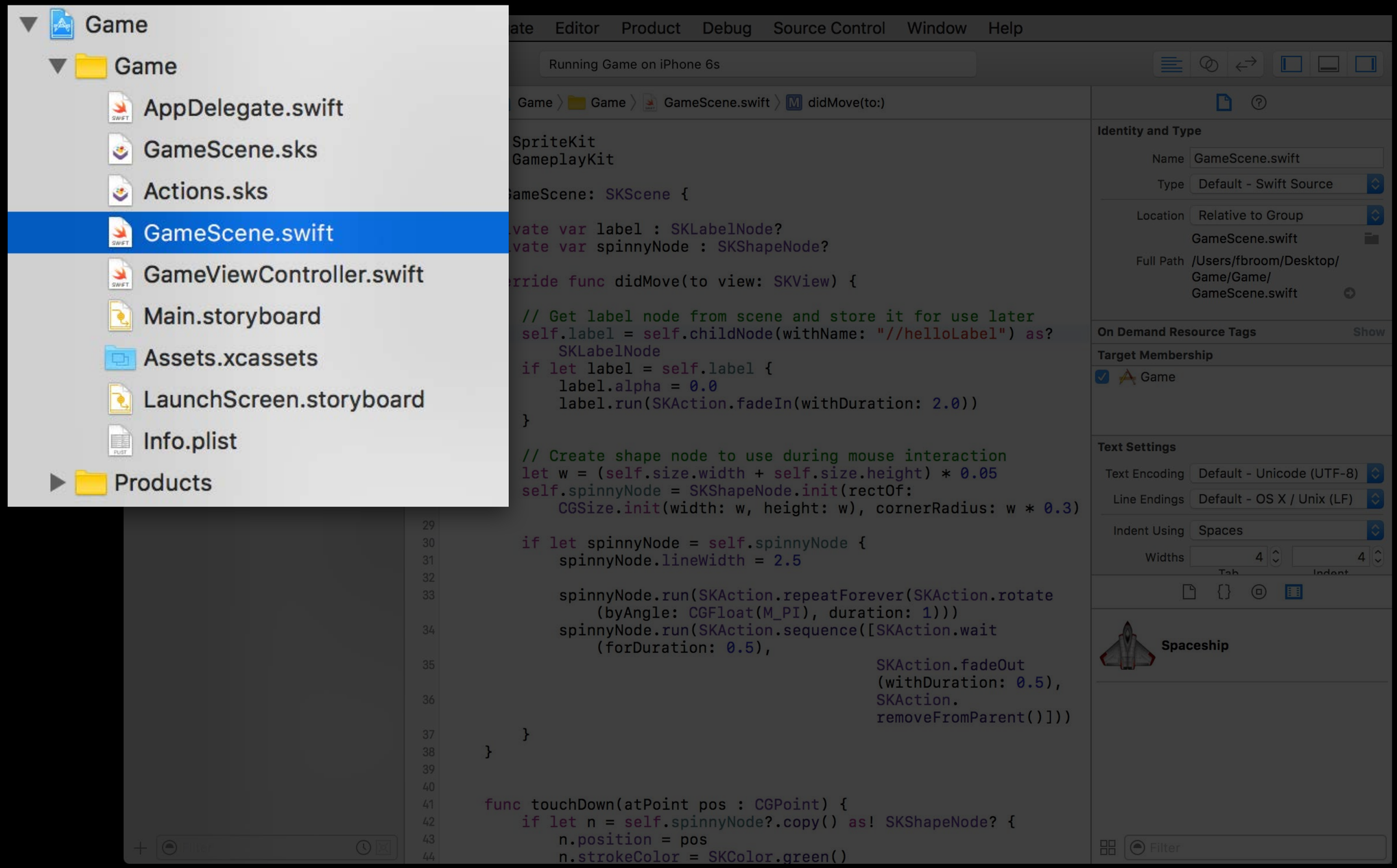
Line Endings: Default - OS X / Unix (LF)

Indent Using: Spaces

Widths: 4 (Tab) 4 (Indent)

Spaceship

Filter


Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
ate Editor Product Debug Source Control Window Help
Running Game on iPhone 6s
Game 〉 Game 〉 GameScene.swift 〉 didMove(to:)
SpriteKit
GameplayKit
ameScene: SKScene {
vate var label : SKLabelNode?
vate var spinnyNode : SKShapeNode?
rride func didMove(to view: SKView) {
// Get label node from scene and store it for use later
self.label = self.childNode(withName: "//helloLabel") as?
SKLabelNode
if let label = self.label {
label.alpha = 0.0
label.run(SKAction.fadeIn(withDuration: 2.0))
}
// Create shape node to use during mouse interaction
let w = (self.size.width + self.size.height) * 0.05
self.spinnyNode = SKShapeNode.init(rectOf:
CGSize.init(width: w, height: w), cornerRadius: w * 0.3)
if let spinnyNode = self.spinnyNode {
spinnyNode.lineWidth = 2.5
spinnyNode.run(SKAction.repeatForever(SKAction.rotate
(byAngle: CGFloat(M_PI), duration: 1)))
spinnyNode.run(SKAction.sequence([SKAction.wait
(forDuration: 0.5),
SKAction.fadeOut
(withDuration: 0.5),
SKAction.
removeFromParent()]))
}
}
func touchDown(atPoint pos : CGPoint) {
if let n = self.spinnyNode?.copy() as! SKShapeNode? {
n.position = pos
n.strokeColor = SKColor.green()
Identity and Type
Name GameScene.swift
Type Default - Swift Source
Location Relative to Group
GameScene.swift
Full Path /Users/fbroom/Desktop/Game/Game/GameScene.swift
On Demand Resource Tags Show
Target Membership
Game
Text Settings
Text Encoding Default - Unicode (UTF-8)
Line Endings Default - OS X / Unix (LF)
Indent Using Spaces
Widths 4 4
Tab Indent
Spaceship
Filter

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Game 〉 iPhone 6s | Finished running Game on iPhone 6s

Game 〉 Game 〉 GameScene.sks 〉 Scene

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - Products

Scene
- helloLabel

Filter

Hello, World!

Animate | Playback Speed 1x

All Nodes | 0:00

helloLabel

Filter

Scene

Name: name

Camera

Color

Size: Custom

W: 750 H: 1334

Anchor Point X: 0.5 Y: 0.5

Gravity X: 0 Y: -9.8

Preview
- Show Camera Boundaries
- Show Physics Boundaries
- Use Camera Node

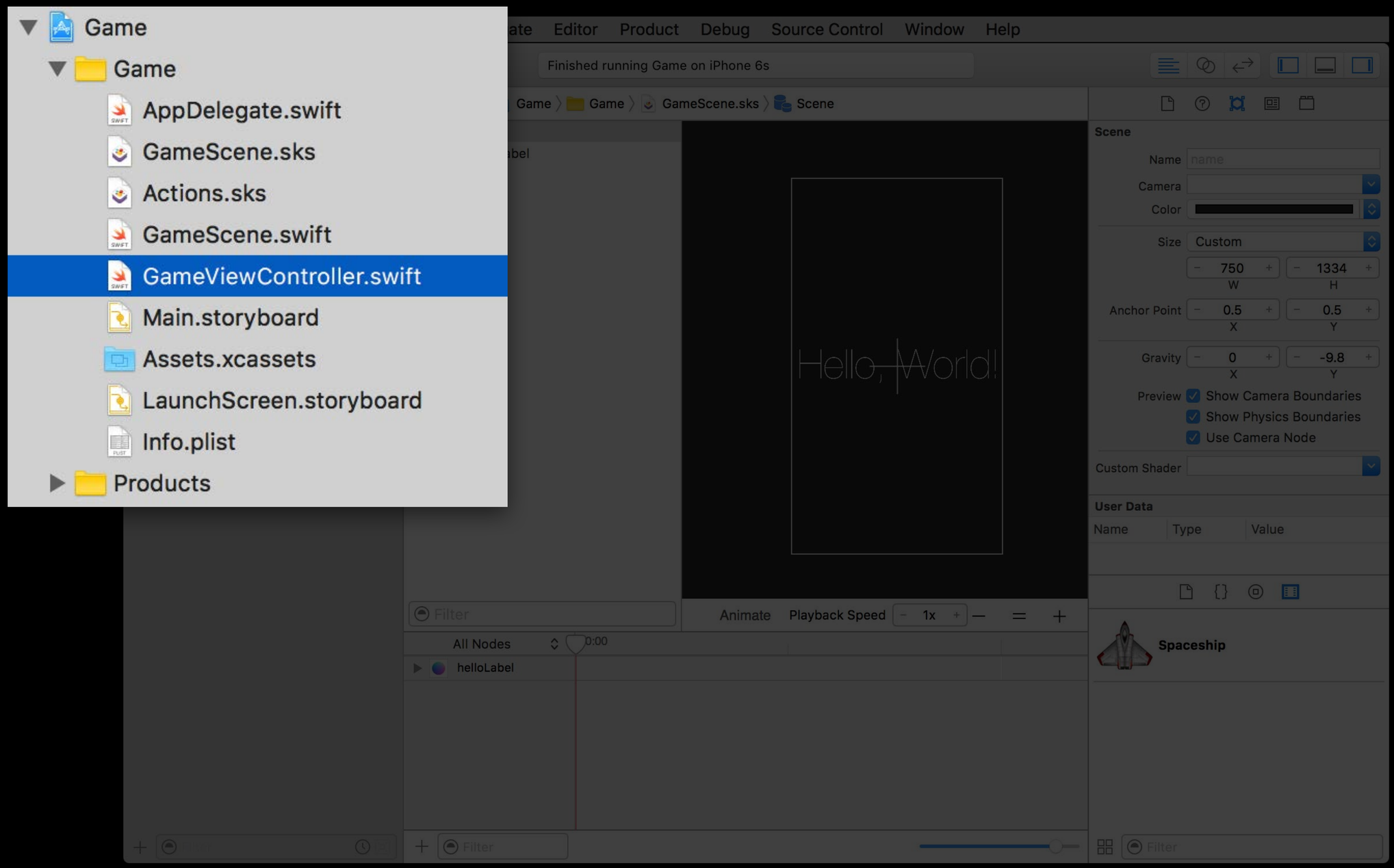
Custom Shader

User Data

Name	Type	Value

Spaceship

Filter


Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Editor
Product
Debug
Source Control
Window
Help
Finished running Game on iPhone 6s
Game
Game
GameScene.sks
Scene
Hello, World!
Scene
Name
name
Camera
Color
Size
Custom
750
W
1334
H
Anchor Point
0.5
X
0.5
Y
Gravity
0
X
-9.8
Y
Preview
Show Camera Boundaries
Show Physics Boundaries
Use Camera Node
Custom Shader
User Data
Name
Type
Value
Spaceship
Filter
Animate
Playback Speed
1x
All Nodes
0:00
helloLabel
Filter
Filter
Filter

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Game 〉 iPhone 6s | Finished running Game on iPhone 6s

Game 〉 Game 〉 GameViewController.swift 〉 viewDidLoad()

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - Products

```
8
9   import UIKit
10  import SpriteKit
11  import GameplayKit
12
13  class GameViewController: UIViewController {
14
15      override func viewDidLoad() {
16          super.viewDidLoad()
17
18          if let view = self.view as! SKView? {
19              // Load the SKScene from 'GameScene.sks'
20              if let scene = SKScene(fileNamed: "GameScene") {
21                  // Set the scale mode to scale to fit the window
22                  scene.scaleMode = .aspectFill
23
24                  // Present the scene
25                  view.presentScene(scene)
26              }
27
28              view.ignoresSiblingOrder = true
29
30              view.showsFPS = true
31              view.showsNodeCount = true
32          }
33      }
34
35      override func shouldAutorotate() -> Bool {
36          return true
37      }
38
39      override func supportedInterfaceOrientations() ->
            UIInterfaceOrientationMask {
40          if UIDevice.current().userInterfaceIdiom == .phone {
41              return .allButUpsideDown
42          } else {
43              return .all
44          }
45      }
46
47      override func didReceiveMemoryWarning() {
48          super.didReceiveMemoryWarning()
49          // Release any cached data, images, etc that aren't in use
```

Identity and Type

Name: GameViewController.swift

Type: Default - Swift Source

Location: Relative to Group

GameViewController.swift

Full Path: /Users/fbroom/Desktop/Game/Game/GameViewController.swift

On Demand Resource Tags Show

Target Membership

Game

Text Settings

Text Encoding: Default - Unicode (UTF-8)

Line Endings: Default - OS X / Unix (LF)

Indent Using: Spaces

Widths: 4 (Tab) 4 (Indent)

Spaceship

Filter

```
    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)
            }

            view.ignoresSiblingOrder = true

            view.showsFPS = true
            view.showsNodeCount = true
        }
    }
```

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Game 〉 iPhone 6s | Finished running Game on iPhone 6s

Game 〉 Game 〉 GameViewController.swift 〉 viewDidLoad()

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - Products

```
8
9   import UIKit
10  import SpriteKit
11  import GameplayKit
12
13  class GameViewController: UIViewController {
14
15      override func viewDidLoad() {
16          super.viewDidLoad()
17
18          if let view = self.view as! SKView? {
19              // Load the SKScene from 'GameScene.sks'
20              if let scene = SKScene(fileNamed: "GameScene") {
21                  // Set the scale mode to scale to fit the window
22                  scene.scaleMode = .aspectFill
23
24                  // Present the scene
25                  view.presentScene(scene)
26              }
27
28              view.ignoresSiblingOrder = true
29
30              view.showsFPS = true
31              view.showsNodeCount = true
32          }
33      }
34
35      override func shouldAutorotate() -> Bool {
36          return true
37      }
38
39      override func supportedInterfaceOrientations() ->
            UIInterfaceOrientationMask {
40          if UIDevice.current().userInterfaceIdiom == .phone {
41              return .allButUpsideDown
42          } else {
43              return .all
44          }
45      }
46
47      override func didReceiveMemoryWarning() {
48          super.didReceiveMemoryWarning()
49          // Release any cached data, images, etc that aren't in use
```

Identity and Type

Name: GameViewController.swift

Type: Default - Swift Source

Location: Relative to Group

GameViewController.swift

Full Path: /Users/fbroom/Desktop/Game/Game/GameViewController.swift

On Demand Resource Tags Show

Target Membership

Game

Text Settings

Text Encoding: Default - Unicode (UTF-8)

Line Endings: Default - OS X / Unix (LF)

Indent Using: Spaces

Widths: 4 (Tab) 4 (Indent)

Spaceship

Filter

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

New ▶
Add Files to “Game”... ⌥⌘A
Open... ⌘O
Open Recent ▶
Open Quickly... ⇧⌘O
Close Window ⌘W
Close Tab
Close “GameViewController.swift” ^⌘W
Close Project ⌥⌘W
Save ⌘S
Duplicate... ⇧⌘S
Revert to Saved...
Unlock...
Export...
Show in Finder
Open with External Editor
Save As Workspace...
Project Settings...
Page Setup... ⇧⌘P
Print... ⌘P

Tab ⌘T
Window ⇧⌘T
File... ⌘N
Playground... ⌥⇧⌘N
Target...
Project... ⇧⌘N
Workspace... ^⌘N
Group ⌥⌘N
Group from Selection

Game
Game
AppDel
GameS
Actions
GameS
GameV
Main.st
Assets.
Launch
Info.pli
Products

viewDidLoad()

```
er {

iew = self.view as! SKView? {
ad the SKScene from 'GameScene.sks'
t scene = SKScene(fileNamed: "GameScene") {
// Set the scale mode to scale to fit the window
cene.scaleMode = .aspectFill

// Present the scene
iew.presentScene(scene)

ignoresSiblingOrder = true

showsFPS = true
showsNodeCount = true
```

```
35      override func shouldAutorotate() -> Bool {
36          return true
37      }
38
39      override func supportedInterfaceOrientations() ->
            UIInterfaceOrientationMask {
40          if UIDevice.current().userInterfaceIdiom == .phone {
41              return .allButUpsideDown
42          } else {
43              return .all
44          }
45      }
46
47      override func didReceiveMemoryWarning() {
48          super.didReceiveMemoryWarning()
49          // Release any cached data, images, etc that aren't in use
```

Identity and Type
Name GameViewController.swift
Type Default - Swift Source
Location Relative to Group
GameViewController.swift
Full Path /Users/fbroom/Desktop/Game/Game/GameViewController.swift
On Demand Resource Tags Show
Target Membership
Game
Text Settings
Text Encoding Default - Unicode (UTF-8)
Line Endings Default - OS X / Unix (LF)
Indent Using Spaces
Widths 4 Tab 4 Indent

Spaceship

Filter

Choose a template for your new target:

- iOS
  - Application
  - Framework & Library
  - Application Extension
  - Test
- watchOS
  - Application
  - Framework & Library
- tvOS
  - Application
  - Framework & Library
  - Application Extension
  - Test
- OS X
  - Application
  - Framework & Library
  - Application Extension
  - Test

WatchKit App

Game App

Game App

This template provides a starting point for a WatchKit game app with an associated app extension.

Cancel | Previous | Next

Choose options for your new target:

Product Name: AppleWatchGame

Team: Apple Inc. - GameKit

Organization Name: Apple

Organization Identifier: com.apple.games.Game

Bundle Identifier: com.apple.games.Game.AppleWatch...

Language: Swift

Game Technology: SpriteKit

Include Notification Scene

Include Complication

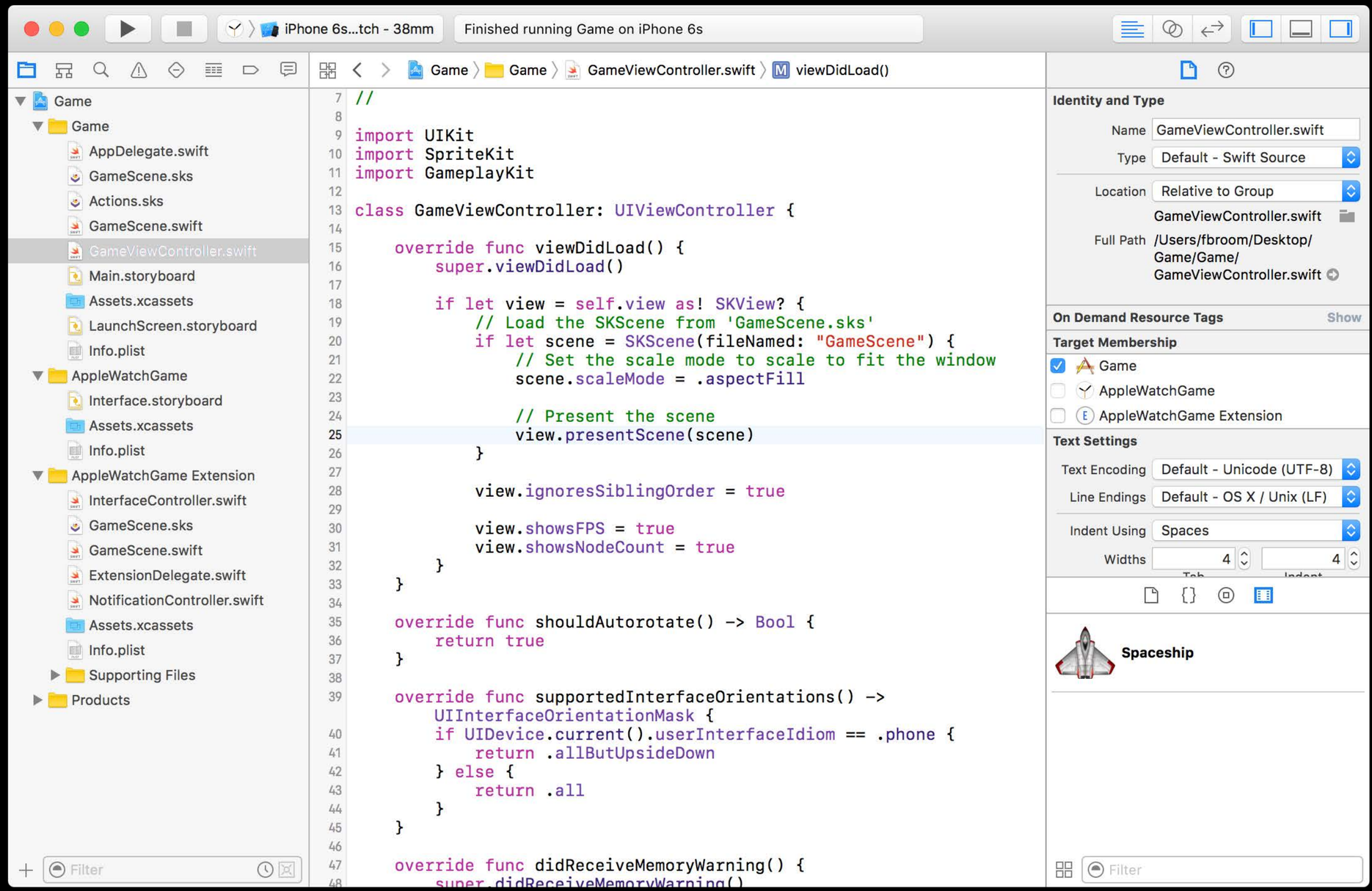
Project: Game

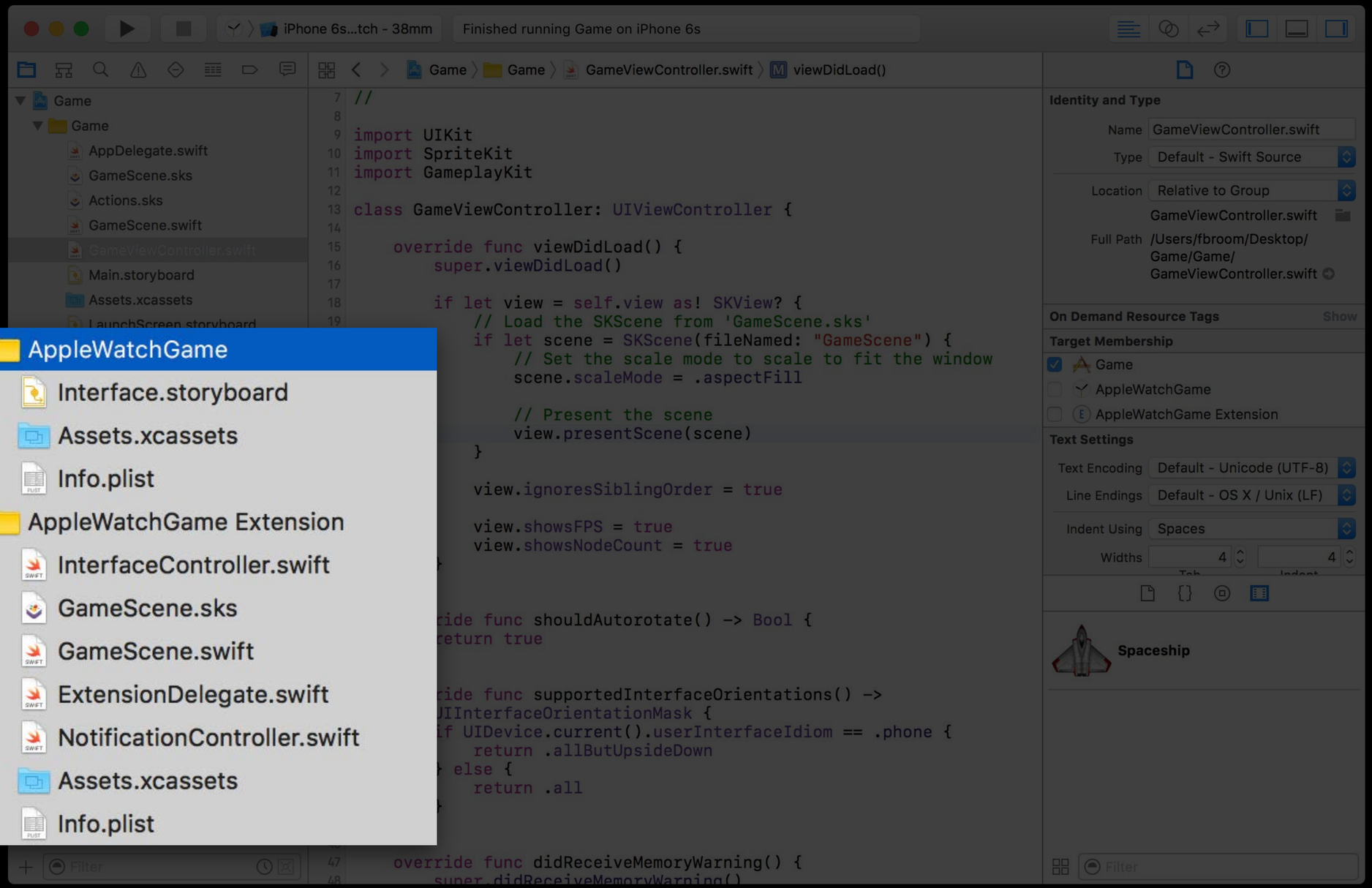
Embed in Companion Application: Game

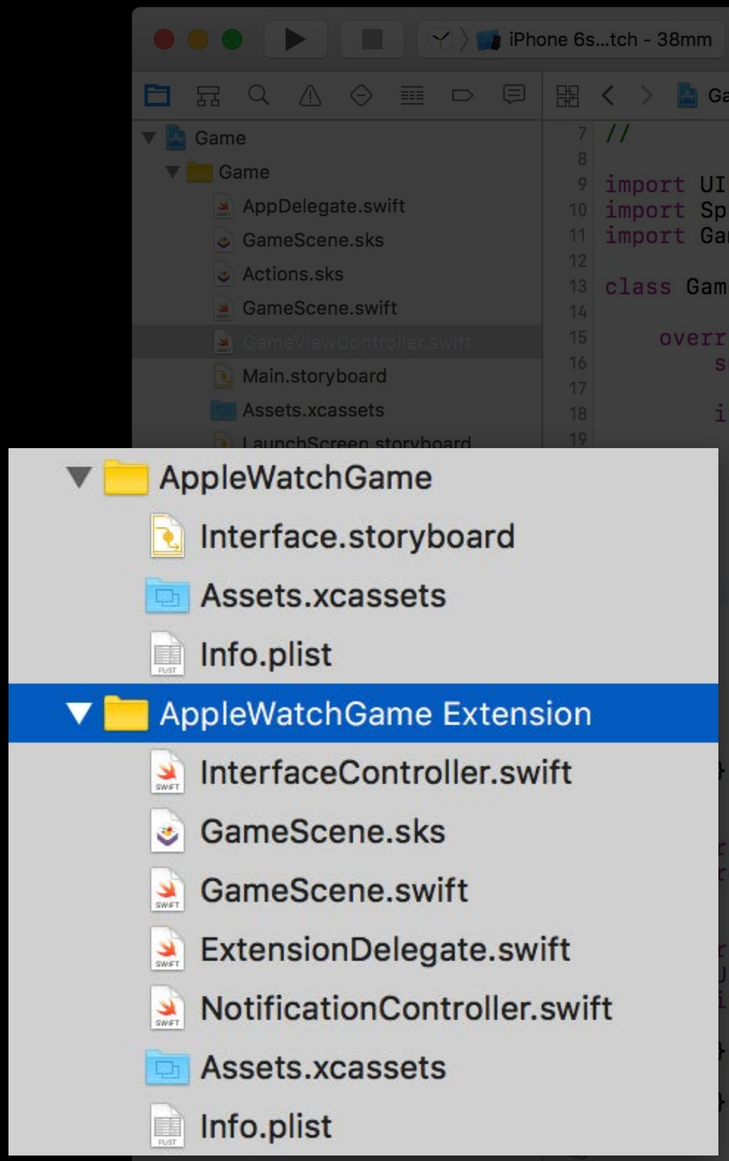
Cancel

Previous

Finish


iPhone 6s...tch - 38mm
Finished running Game on iPhone 6s
Game 〉 Game 〉 GameViewController.swift 〉 viewDidLoad()
Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist
Supporting Files
Products
Filter
7 //
8
9 import UIKit
10 import SpriteKit
11 import GameplayKit
12
13 class GameViewController: UIViewController {
14
15 override func viewDidLoad() {
16 super.viewDidLoad()
17
18 if let view = self.view as! SKView? {
19 // Load the SKScene from 'GameScene.sks'
20 if let scene = SKScene(fileNamed: "GameScene") {
21 // Set the scale mode to scale to fit the window
22 scene.scaleMode = .aspectFill
23
24 // Present the scene
25 view.presentScene(scene)
26 }
27
28 view.ignoresSiblingOrder = true
29
30 view.showsFPS = true
31 view.showsNodeCount = true
32 }
33 }
34
35 override func shouldAutorotate() -> Bool {
36 return true
37 }
38
39 override func supportedInterfaceOrientations() ->
UIInterfaceOrientationMask {
40 if UIDevice.current().userInterfaceIdiom == .phone {
41 return .allButUpsideDown
42 } else {
43 return .all
44 }
45 }
46
47 override func didReceiveMemoryWarning() {
48 super.didReceiveMemoryWarning()
Identity and Type
Name GameViewController.swift
Type Default - Swift Source
Location Relative to Group
GameViewController.swift
Full Path /Users/fbroom/Desktop/Game/Game/GameViewController.swift
On Demand Resource Tags Show
Target Membership
Game
AppleWatchGame
AppleWatchGame Extension
Text Settings
Text Encoding Default - Unicode (UTF-8)
Line Endings Default - OS X / Unix (LF)
Indent Using Spaces
Widths 4 4
Tab Indent
Spaceship
Filter

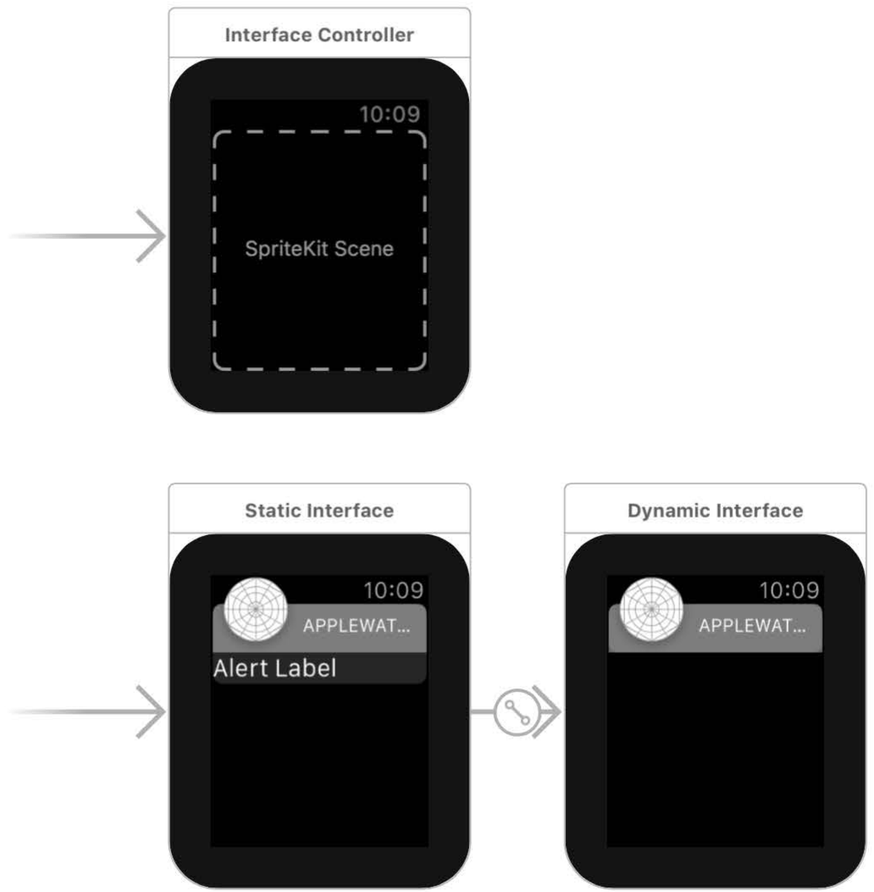

iPhone 6s...tch - 38mm
Finished running Game on iPhone 6s
Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
Game
Game
GameViewController.swift
viewDidLoad()
//
import UIKit
import SpriteKit
import GameplayKit
class GameViewController: UIViewController {
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)
}
view.ignoresSiblingOrder = true
view.showsFPS = true
view.showsNodeCount = true
ride func shouldAutorotate() -> Bool {
return true
ride func supportedInterfaceOrientations() ->
UIInterfaceOrientationMask {
if UIDevice.current().userInterfaceIdiom == .phone {
return .allButUpsideDown
} else {
return .all
override func didReceiveMemoryWarning() {
super.didReceiveMemoryWarning()
Identity and Type
Name
GameViewController.swift
Type
Default - Swift Source
Location
Relative to Group
GameViewController.swift
Full Path
/Users/fbroom/Desktop/Game/Game/GameViewController.swift
On Demand Resource Tags
Show
Target Membership
Game
AppleWatchGame
AppleWatchGame Extension
Text Settings
Text Encoding
Default - Unicode (UTF-8)
Line Endings
Default - OS X / Unix (LF)
Indent Using
Spaces
Widths
4
4
Spaceship
Filter
Filter
AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist


AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist

iPhone 6s...tch - 38mm | Finished running Game on iPhone 6s

Game 〉 Apple...Game 〉 Interf...board 〉 Interface.storyboard (Base) 〉 No Selection

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - AppleWatchGame
    - Interface.storyboard
    - Assets.xcassets
    - Info.plist
  - AppleWatchGame Extension
    - InterfaceController.swift
    - GameScene.sks
    - GameScene.swift
    - ExtensionDelegate.swift
    - NotificationController.swift
    - Assets.xcassets
    - Info.plist
  - Supporting Files
  - Products



Identity and Type

Name: Interface.storyboard
Type: Default - Interface Builder...
Location: Relative to Group
Choose Containing Folder
Full Path: /Users/fbroom/Desktop/Game/AppleWatchGame
Dev Region: /Users/fbroom/Desktop/Game/AppleWatchGame/Base.lproj/Interface.storyboard

Localization

- [x] Base
- [ ] English — Localizable Strings

Target Membership

- [ ] Game
- [x] AppleWatchGame
- [ ] AppleWatchGame Extension

Text Settings

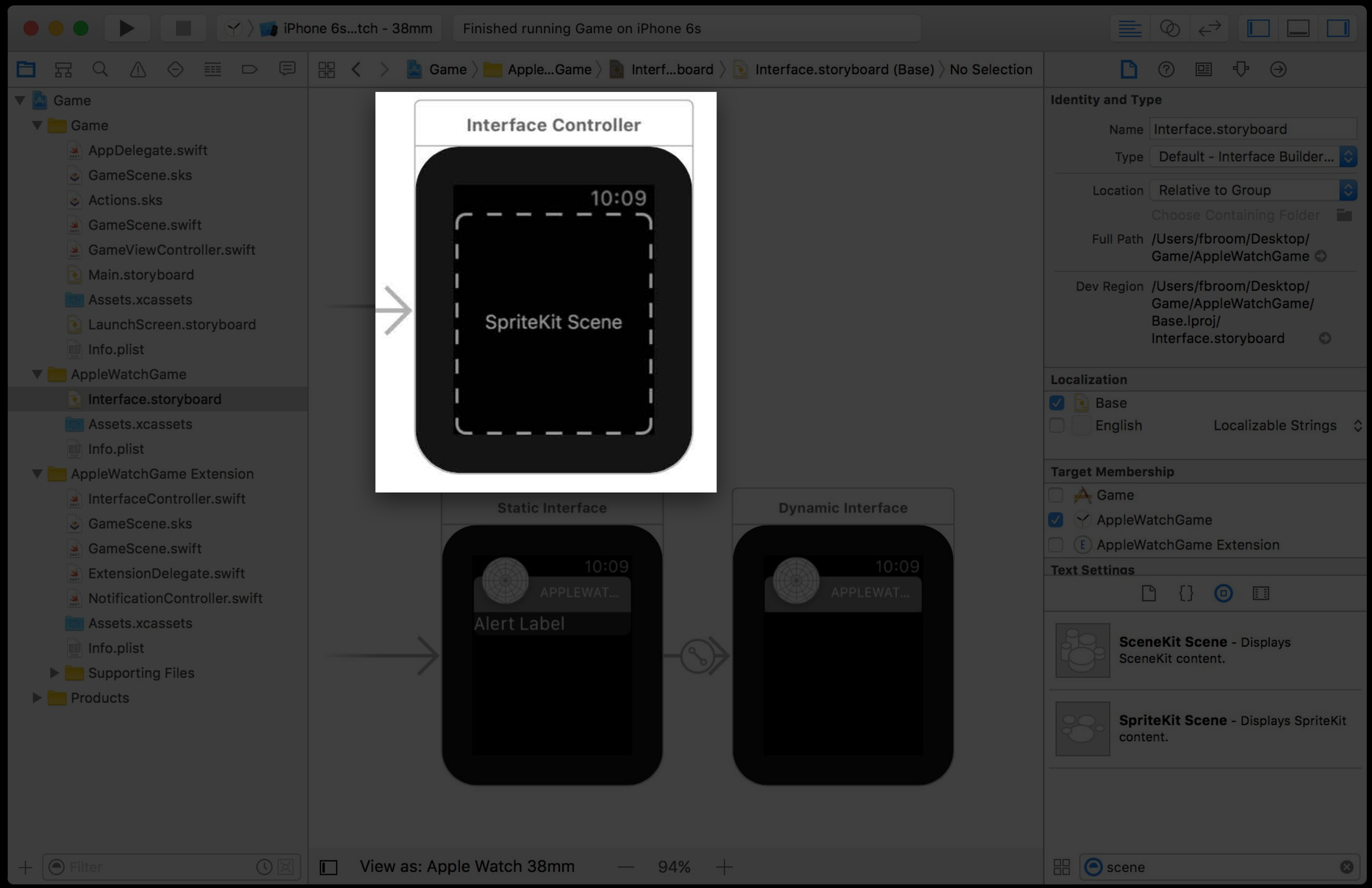
SceneKit Scene - Displays SceneKit content.

SpriteKit Scene - Displays SpriteKit content.

Filter

View as: Apple Watch 38mm — 94% +

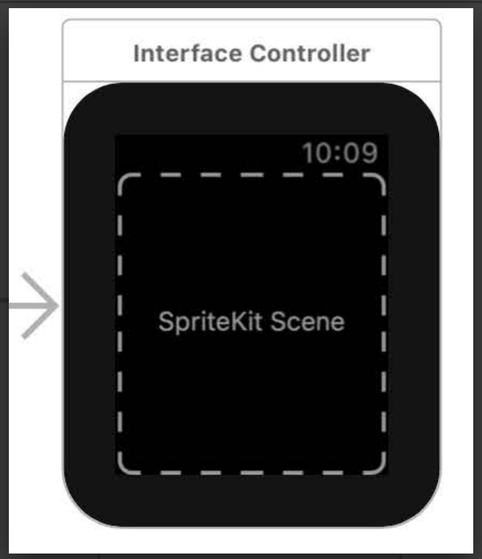
scene


iPhone 6s...tch - 38mm
Finished running Game on iPhone 6s
Game
Apple...Game
Interf...board
Interface.storyboard (Base)
No Selection
Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist
Supporting Files
Products
Interface Controller
10:09
SpriteKit Scene
Static Interface
Dynamic Interface
10:09
APPLEWAT...
Alert Label
10:09
APPLEWAT...
Identity and Type
Name
Interface.storyboard
Type
Default - Interface Builder...
Location
Relative to Group
Choose Containing Folder
Full Path
/Users/fbroom/Desktop/Game/AppleWatchGame
Dev Region
/Users/fbroom/Desktop/Game/AppleWatchGame/Base.lproj/Interface.storyboard
Localization
Base
English
Localizable Strings
Target Membership
Game
AppleWatchGame
AppleWatchGame Extension
Text Settings
SceneKit Scene - Displays SceneKit content.
SpriteKit Scene - Displays SpriteKit content.
Filter
View as: Apple Watch 38mm
94%
scene

iPhone 6s...tch - 38mm | Finished running Game on iPhone 6s

Game 〉 Apple...Game 〉 Interf...board 〉 Interface.storyboard (Base) 〉 No Selection

- Game
  - Game
    - AppDelegate.swift
    - GameScene.sks
    - Actions.sks
    - GameScene.swift
    - GameViewController.swift
    - Main.storyboard
    - Assets.xcassets
    - LaunchScreen.storyboard
    - Info.plist
  - AppleWatchGame
    - Interface.storyboard
    - Assets.xcassets
    - Info.plist
  - AppleWatchGame Extension
    - InterfaceController.swift
    - GameScene.sks
    - GameScene.swift
    - ExtensionDelegate.swift
    - NotificationController.swift
    - Assets.xcassets
    - Info.plist
  - Supporting Files
  - Products



Identity and Type

Name: Interface.storyboard

Type: Default - Interface Builder...

Location: Relative to Group

Choose Containing Folder

Full Path: /Users/fbroom/Desktop/Game/AppleWatchGame

Dev Region: /Users/fbroom/Desktop/Game/AppleWatchGame/Base.lproj/Interface.storyboard

Localization

Base

English — Localizable Strings

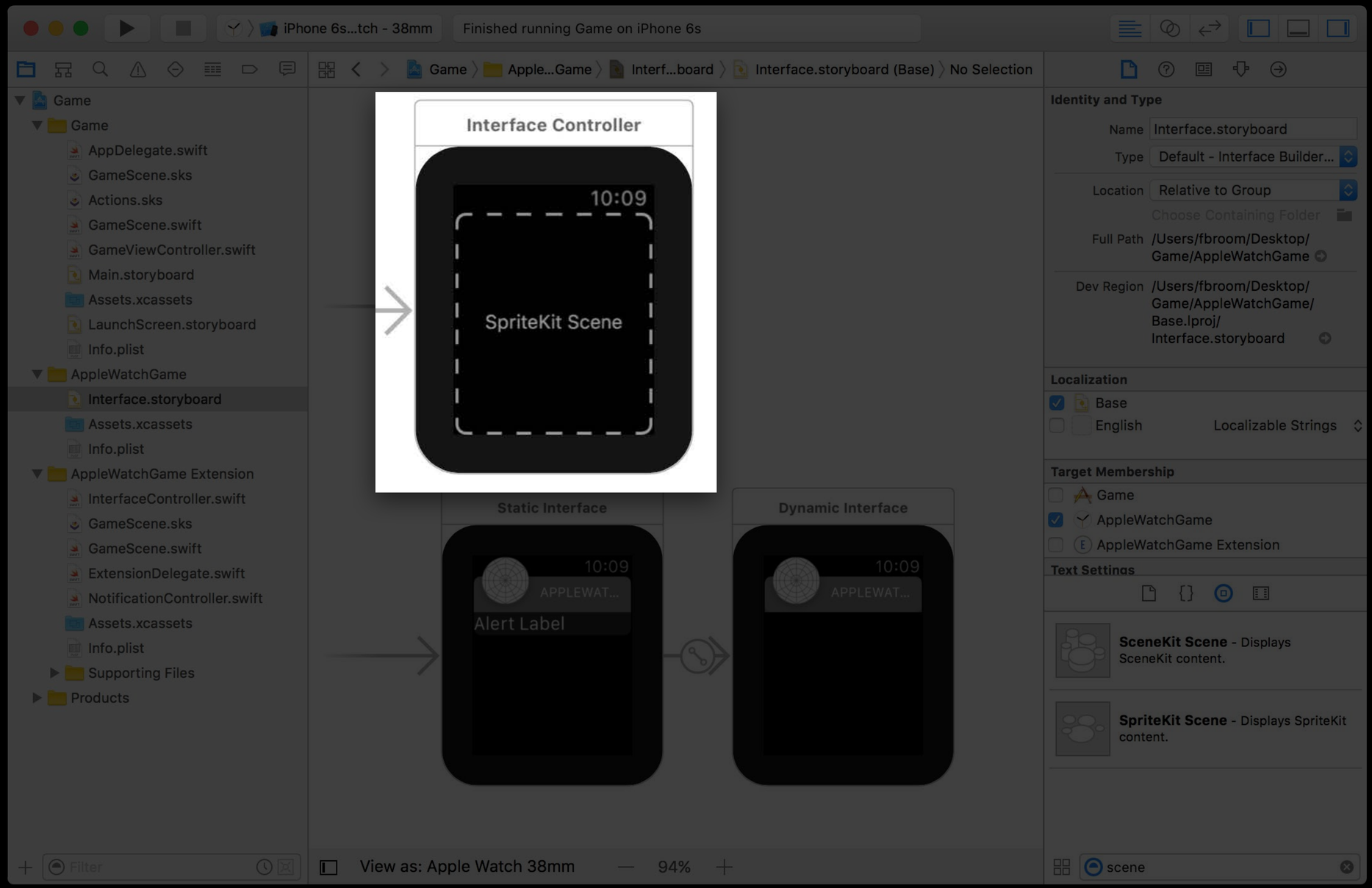
SceneKit Scene - Displays SceneKit content.

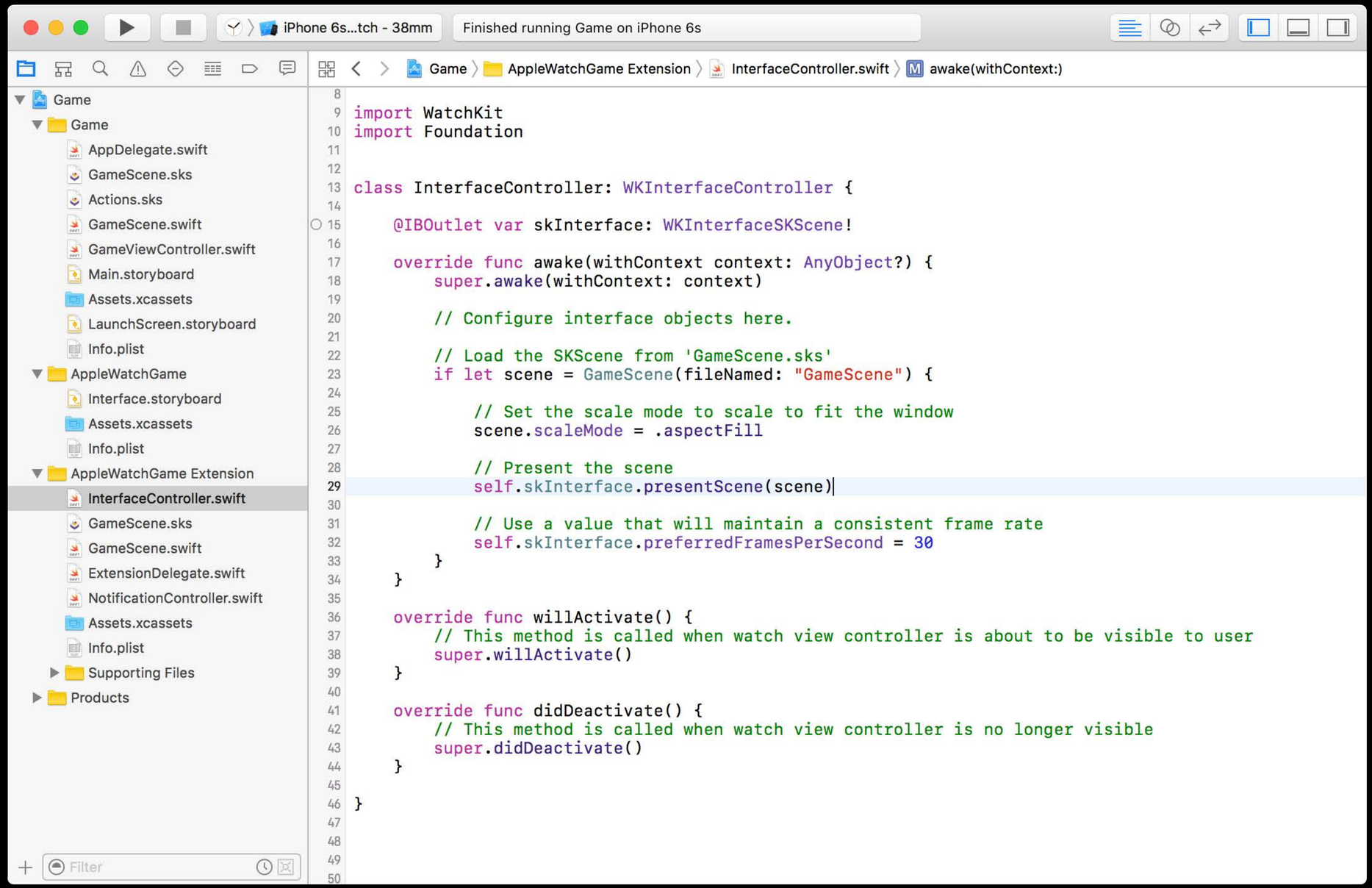
SpriteKit Scene - Displays SpriteKit content.

scene

Filter

View as: Apple Watch 38mm — 94% +


iPhone 6s...tch - 38mm
Finished running Game on iPhone 6s
Game
Apple...Game
Interf...board
Interface.storyboard (Base)
No Selection
Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist
Supporting Files
Products
Interface Controller
10:09
SpriteKit Scene
Static Interface
Dynamic Interface
10:09
APPLEWAT...
Alert Label
10:09
APPLEWAT...
Identity and Type
Name
Interface.storyboard
Type
Default - Interface Builder...
Location
Relative to Group
Choose Containing Folder
Full Path
/Users/fbroom/Desktop/Game/AppleWatchGame
Dev Region
/Users/fbroom/Desktop/Game/AppleWatchGame/Base.lproj/Interface.storyboard
Localization
Base
English
Localizable Strings
Target Membership
Game
AppleWatchGame
AppleWatchGame Extension
Text Settings
SceneKit Scene - Displays SceneKit content.
SpriteKit Scene - Displays SpriteKit content.
Filter
View as: Apple Watch 38mm
94%
scene


iPhone 6s...tch - 38mm
Finished running Game on iPhone 6s
Game
AppleWatchGame Extension
InterfaceController.swift
awake(withContext:)
Game
Game
AppDelegate.swift
GameScene.sks
Actions.sks
GameScene.swift
GameViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
AppleWatchGame
Interface.storyboard
Assets.xcassets
Info.plist
AppleWatchGame Extension
InterfaceController.swift
GameScene.sks
GameScene.swift
ExtensionDelegate.swift
NotificationController.swift
Assets.xcassets
Info.plist
Supporting Files
Products
Filter
8
9 import WatchKit
10 import Foundation
11
12
13 class InterfaceController: WKInterfaceController {
14
15 @IBOutlet var skInterface: WKInterfaceSKScene!
16
17 override func awake(withContext context: AnyObject?) {
18 super.awake(withContext: context)
19
20 // Configure interface objects here.
21
22 // Load the SKScene from 'GameScene.sks'
23 if let scene = GameScene(fileNamed: "GameScene") {
24
25 // Set the scale mode to scale to fit the window
26 scene.scaleMode = .aspectFill
27
28 // Present the scene
29 self.skInterface.presentScene(scene)
30
31 // Use a value that will maintain a consistent frame rate
32 self.skInterface.preferredFramesPerSecond = 30
33 }
34 }
35
36 override func willActivate() {
37 // This method is called when watch view controller is about to be visible to user
38 super.willActivate()
39 }
40
41 override func didDeactivate() {
42 // This method is called when watch view controller is no longer visible
43 super.didDeactivate()
44 }
45
46 }
47
48
49
50

```swift
// Interface Controller


class InterfaceController: WKInterfaceController {

    @IBOutlet var skInterface: WKInterfaceSKScene!

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)

        // Configure interface objects here.

        // Load the SKScene from 'GameScene.sks'
        if let scene = GameScene(fileNamed: "GameScene") {

            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill

            // Present the scene
            self.skInterface.presentScene(scene)

            // Use a value that will maintain a consistent frame rate
            self.skInterface.preferredFramesPerSecond = 30
        }
    }
}
```

```swift
// Interface Controller


class InterfaceController: WKInterfaceController {

    @IBOutlet var skInterface: WKInterfaceSKScene!

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)

        // Configure interface objects here.

        // Load the SKScene from 'GameScene.sks'
        if let scene = GameScene(fileNamed: "GameScene") {

            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill

            // Present the scene
            self.skInterface.presentScene(scene)

            // Use a value that will maintain a consistent frame rate
            self.skInterface.preferredFramesPerSecond = 30
        }
    }
}
```

```swift
// Interface Controller


class InterfaceController: WKInterfaceController {

    @IBOutlet var skInterface: WKInterfaceSKScene!

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)

        // Configure interface objects here.

        // Load the SKScene from 'GameScene.sks'
        if let scene = GameScene(fileNamed: "GameScene") {

            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill

            // Present the scene
            self.skInterface.presentScene(scene)

            // Use a value that will maintain a consistent frame rate
            self.skInterface.preferredFramesPerSecond = 30
        }
    }
}
```

```swift
// Interface Controller


class InterfaceController: WKInterfaceController {

    @IBOutlet var skInterface: WKInterfaceSKScene!

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)

        // Configure interface objects here.

        // Load the SKScene from 'GameScene.sks'
        if let scene = GameScene(fileNamed: "GameScene") {

            // Set the scale mode to scale to fit the window
            scene.scaleMode = .aspectFill

            // Present the scene
            self.skInterface.presentScene(scene)

            // Use a value that will maintain a consistent frame rate
            self.skInterface.preferredFramesPerSecond = 30
        }
    }
}
```

Game Center

Christy Warren Game Technologies Engineer

Game Center

Introduction

Social gaming

- Achievements
- Leaderboards
- Turn-based Multiplayer

Turn Based Multiplayer

A good match

Brief interaction

Asynchronous

Details

- Documentation and past sessions

Getting Started

Authentication

Adoption is similar to iOS

Streamlined

- Sign in on companion
- No view controller

```swift
// Authentication Implementation

class InterfaceController: WKInterfaceController {

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)
        // setup Game Center Authenticate Handler
            GKLocalPlayer.localPlayer().authenticateHandler = {(error) in
                if (error == nil) {
                    print("success!")
                    // give feedback user that they are signed in
                }
                else {
                    print("failed \(error)")
                    // give feedback that game center failed to sign in
                }
            }
    }
}
```

```
// Authentication Implementation

class InterfaceController: WKInterfaceController {

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)
        // setup Game Center Authenticate Handler
            GKLocalPlayer.localPlayer().authenticateHandler = {(error) in
                if (error == nil) {
                    print("success!")
                    // give feedback user that they are signed in
                }
                else {
                    print("failed \(error)")
                    // give feedback that game center failed to sign in
                }
            }
    }
}
```

Turn-Based Multiplayer

Finding players

Automatch

Invite recent players

Turn-Based Multiplayer

Automatch

Pick Players

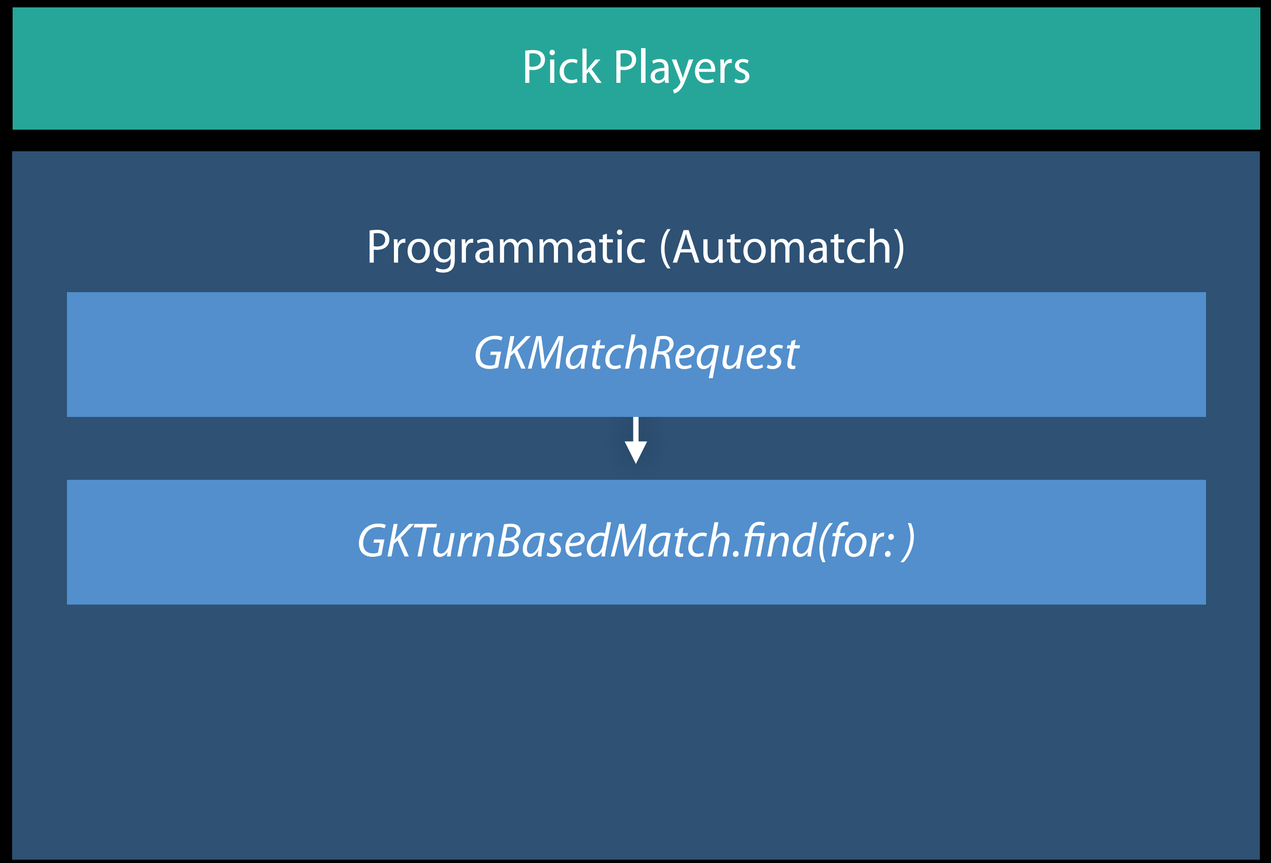
Programmatic (Automatch)

GKMatchRequest

GKTurnBasedMatch.find(for:)

Turn-Based Multiplayer

Automatch

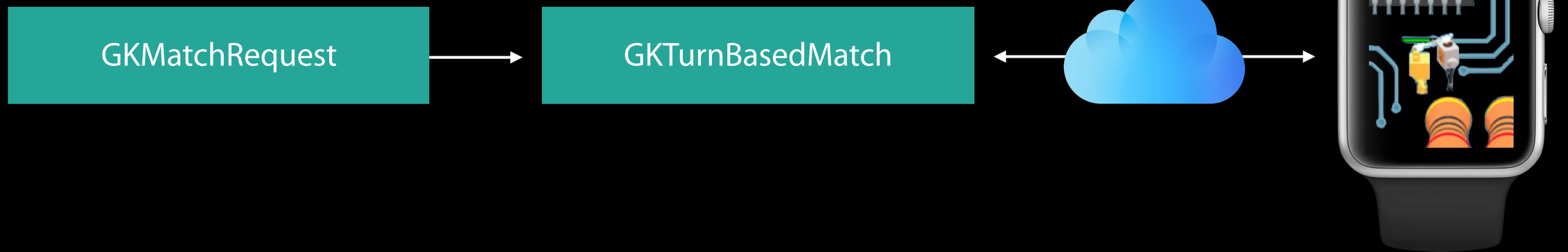


Turn-Based Multiplayer

Automatch

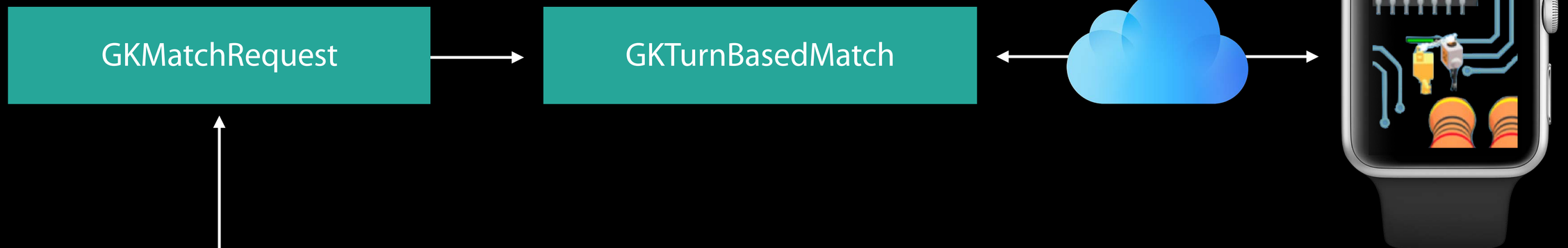
Automatch

Quick and easy



Automatch

Quick and easy

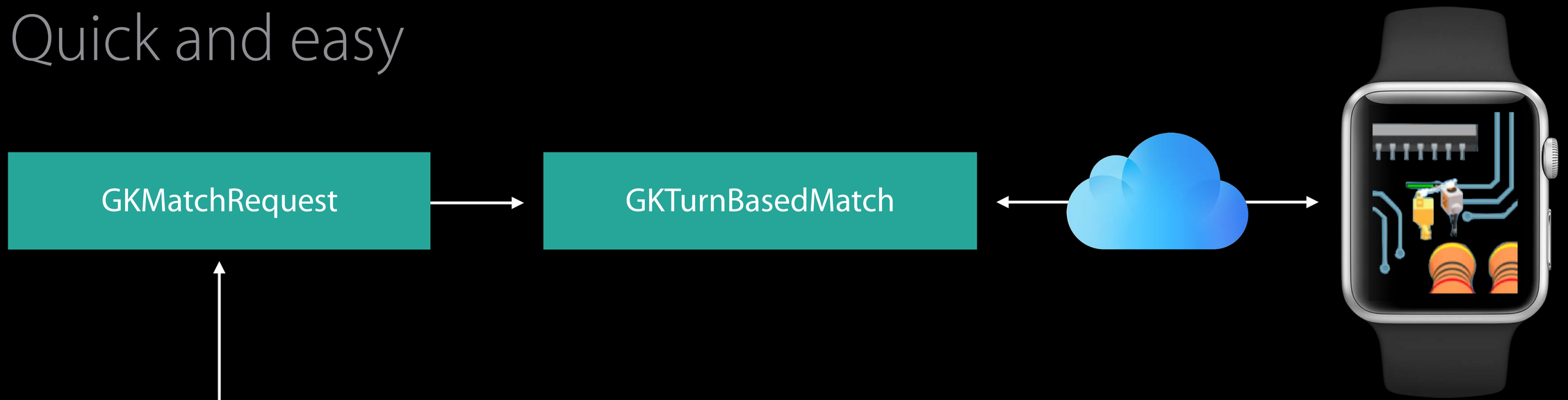



```
let request = GKMatchRequest()
matchRequest.maxPlayers = 2
GKTurnBasedMatch.find(for: request, withCompletionHandler: { (match, error) in
    if (error == nil) {
        self.startGame(withMatch: match)

    }
    else {
        // give feedback that game center failed to find a match
    }
}
```

Automatch

Quick and easy

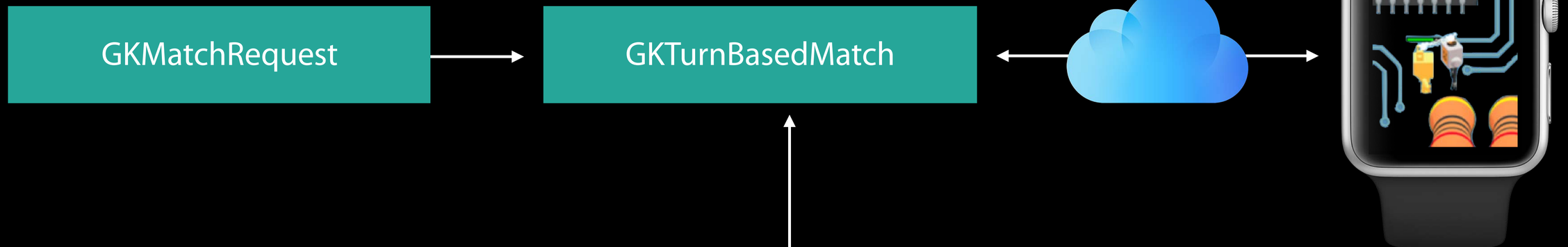



```
let request = GKMatchRequest()
matchRequest.maxPlayers = 2
GKTurnBasedMatch.find(for: request, withCompletionHandler: { (match, error) in
    if (error == nil) {
        self.startGame(withMatch: match)

    }
    else {
        // give feedback that game center failed to find a match
    }
}
```

Automatch

Quick and easy

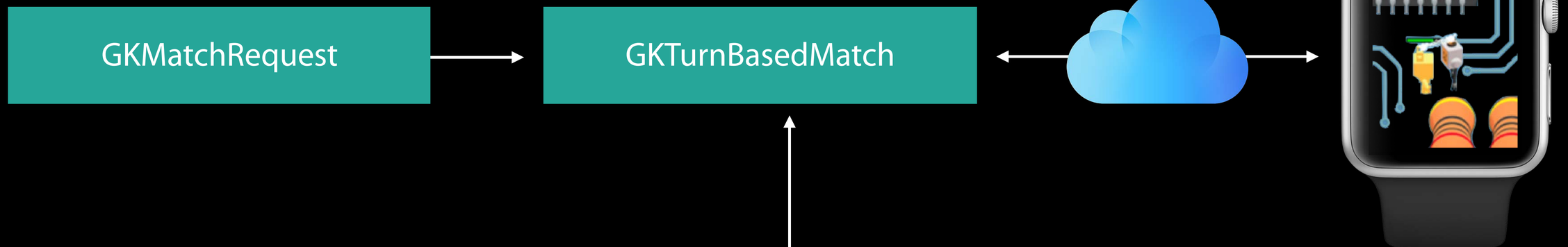



```
let request = GKMatchRequest()
matchRequest.maxPlayers = 2
GKTurnBasedMatch.find(for: request, withCompletionHandler: { (match, error) in
    if (error == nil) {
        self.startGame(withMatch: match)

    }
    else {
        // give feedback that game center failed to find a match
    }
}
```

Programmatic Invites

Sending the invite

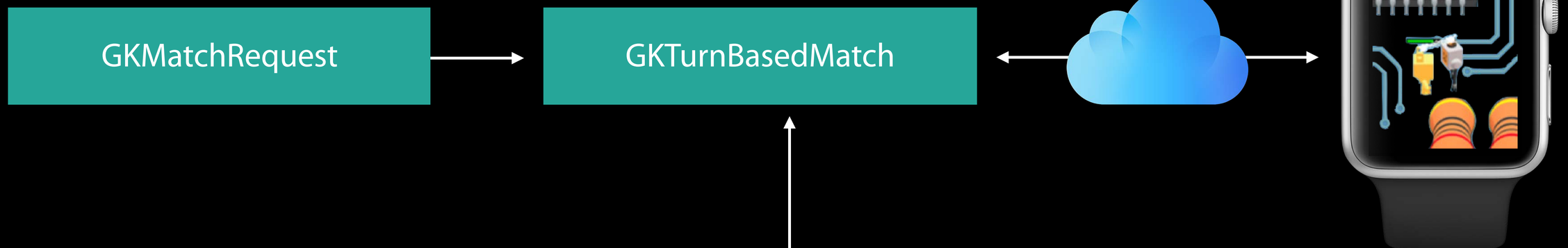



```
let request = GKMatchRequest()
matchRequest.maxPlayers = 2
request.recipients = self.availablePlayers
GKTurnBasedMatch.find(for: request, withCompletionHandler: { (match, error) in
    if (error == nil) {
        self.startGame(withMatch: match)

    }
    else {
        // give feedback that game center failed to find a match
    }
}
```

Programmatic Invites

Sending the invite




```
let request = GKMatchRequest()
matchRequest.maxPlayers = 2
request.recipients = self.availablePlayers
GKTurnBasedMatch.find(for: request, withCompletionHandler: { (match, error) in
    if (error == nil) {
        self.startGame(withMatch: match)

    }
    else {
        // give feedback that game center failed to find a match
    }
}
```

```
// Programmatic Invites
// Loading recent players

class PlayerPickerInterface: WKInterfaceController {

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)
        // if we are not authenticated authenticate first
            GKLocalPlayer.localPlayer().loadRecentPlayers {(players, error) in
                if (error == nil) {
                    self.availablePlayers = players
                }
                 else {
                    print("failed \(error)")
                    // give feedback that game center failed to sign in
                }
            }
    }
}
```

```
// Programmatic Invites
// Loading recent players

class PlayerPickerInterface: WKInterfaceController {

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)
        // if we are not authenticated authenticate first
            GKLocalPlayer.localPlayer().loadRecentPlayers {(players, error) in
                if (error == nil) {
                    self.availablePlayers = players
                }
                 else {
                    print("failed \(error)")
                    // give feedback that game center failed to sign in
                }
            }
    }
}
```

```
// Programmatic Invites
// Loading recent players

class PlayerPickerInterface: WKInterfaceController {

    override func awake(withContext context: AnyObject?) {
        super.awake(withContext: context)
        // if we are not authenticated authenticate first
            GKLocalPlayer.localPlayer().loadRecentPlayers {(players, error) in
                if (error == nil) {
                    self.availablePlayers = players
                }
                 else {
                    print("failed \(error)")
                    // give feedback that game center failed to sign in
                }
            }
    }
}
```

HelloGameKit Sample

Turn-based game

Based on SpriteKit template

Uses gesture recognizers

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

Walkthrough

HelloGameKit

https://developer.apple.com/wwdc16/612

Resources

Turn-Based Gaming

Recap

Streamlined authentication

Programmatic API

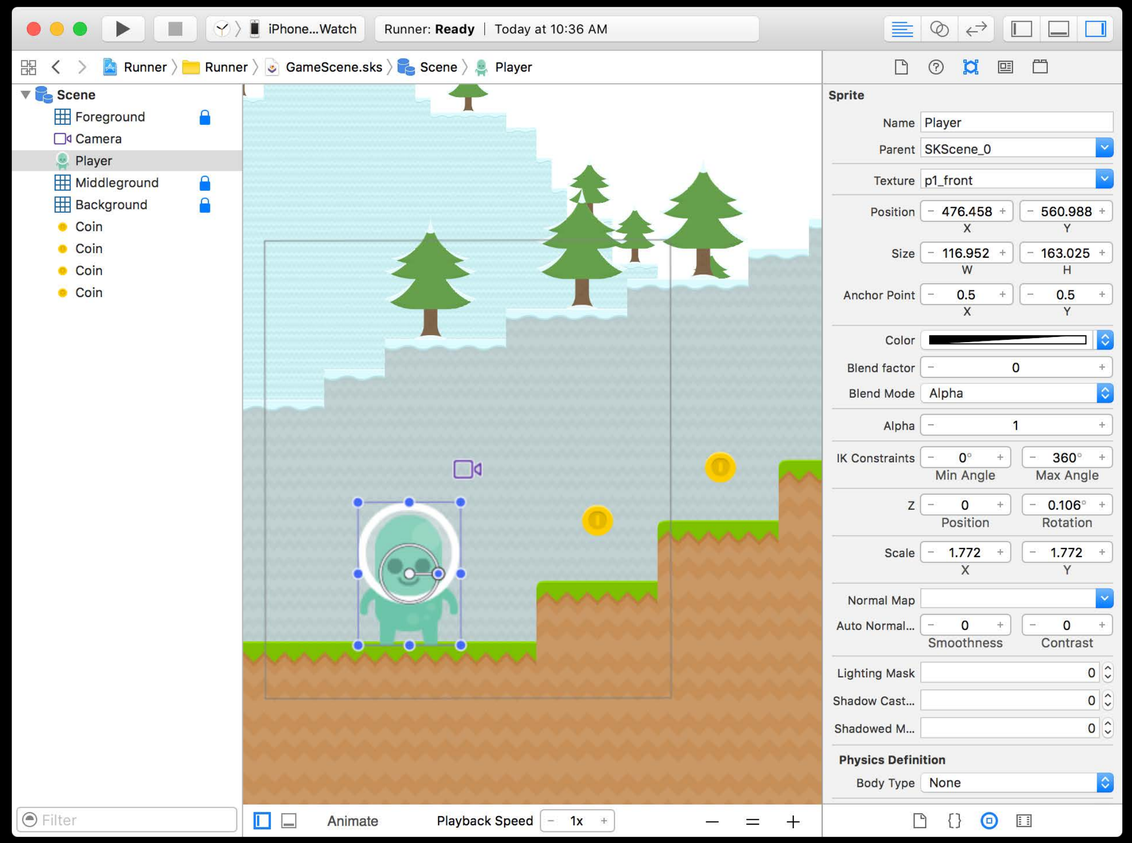
Sample app

Tools and Best Practices

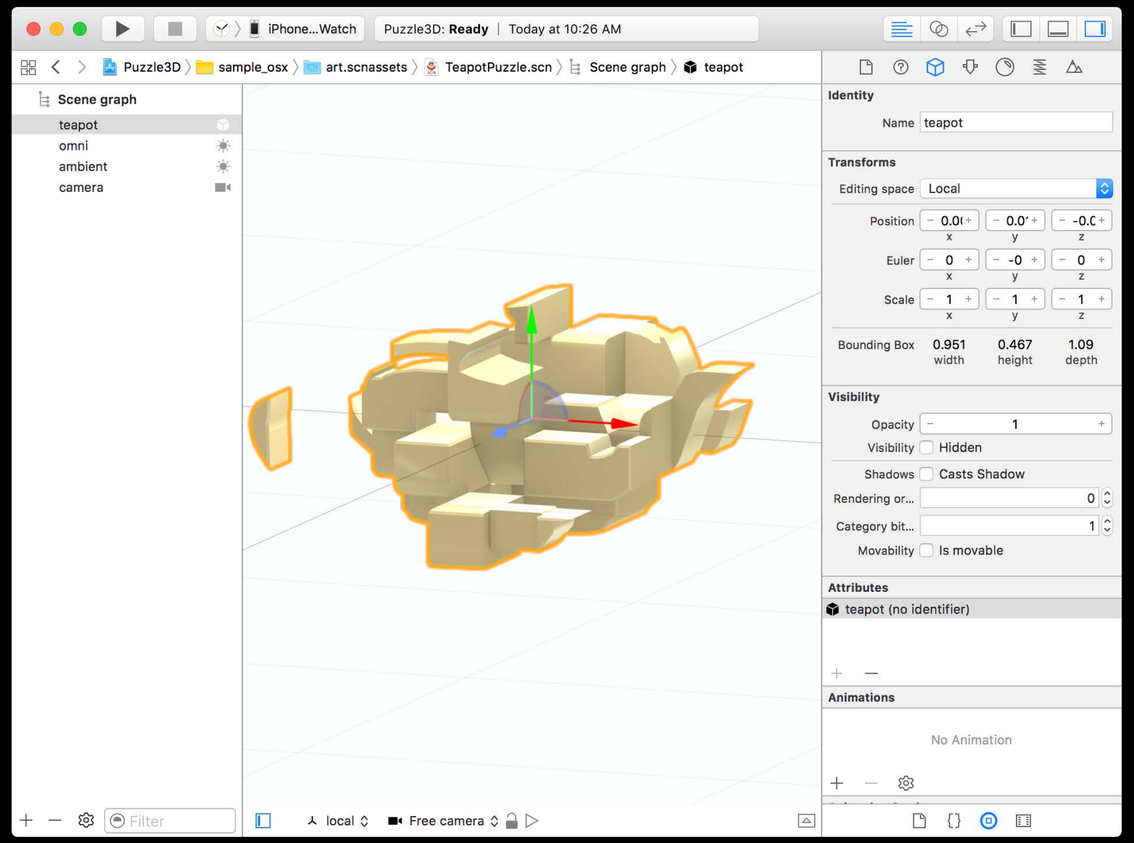
Tyler Casella Game Technologies Engineer

Game Tools for Apple Watch

Scene editor



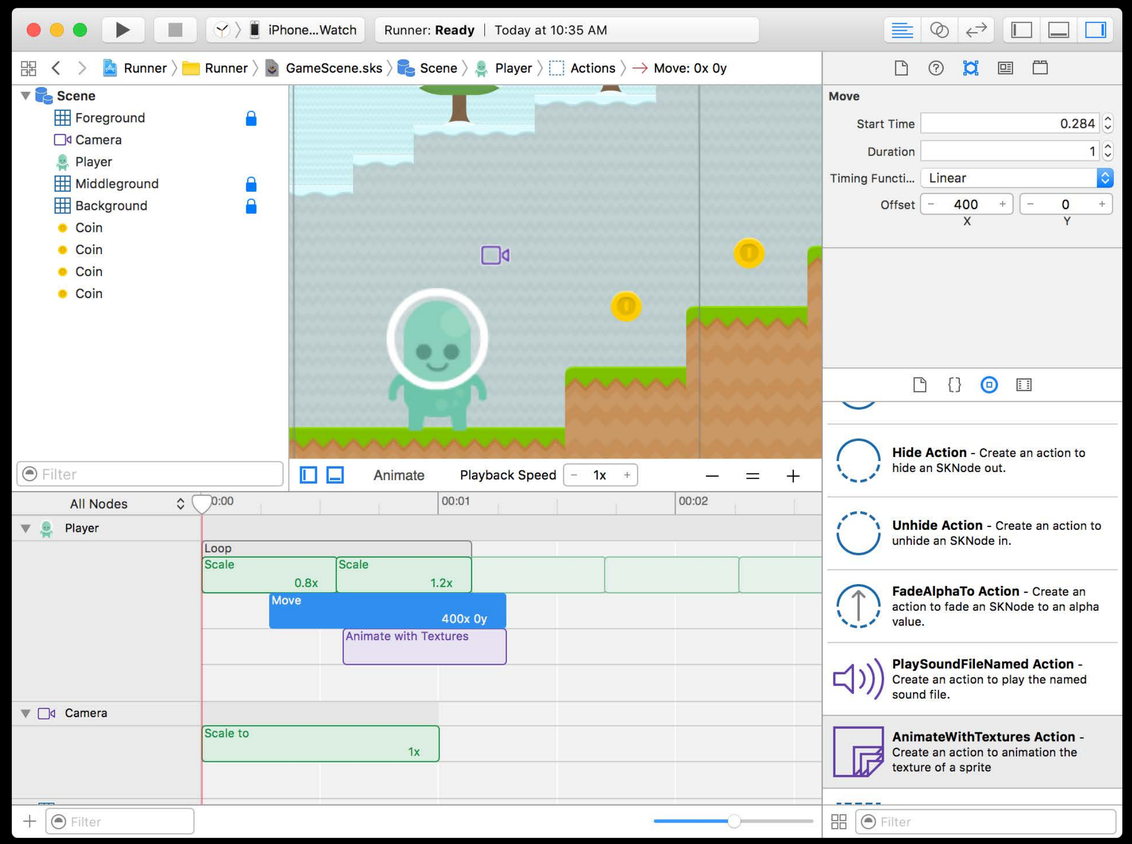
SpriteKit



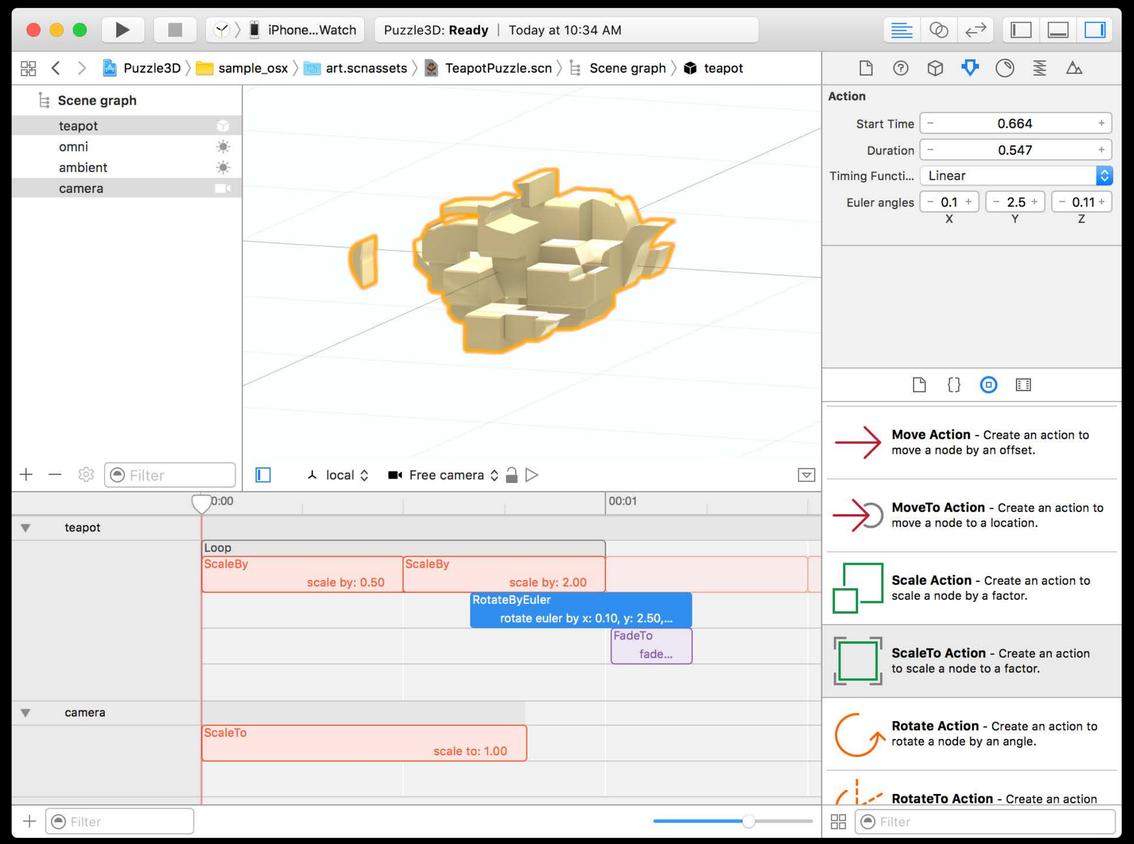
SceneKit

Game Tools for Apple Watch

Action editor



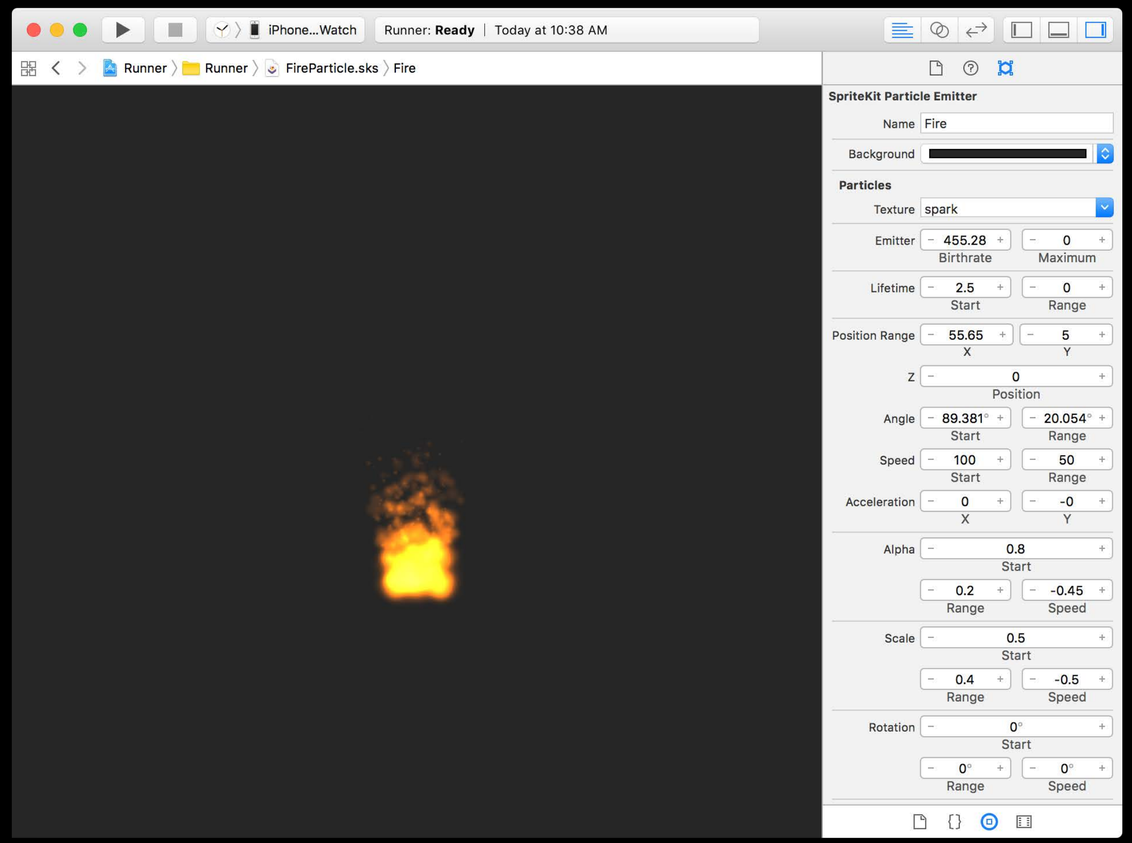
SpriteKit



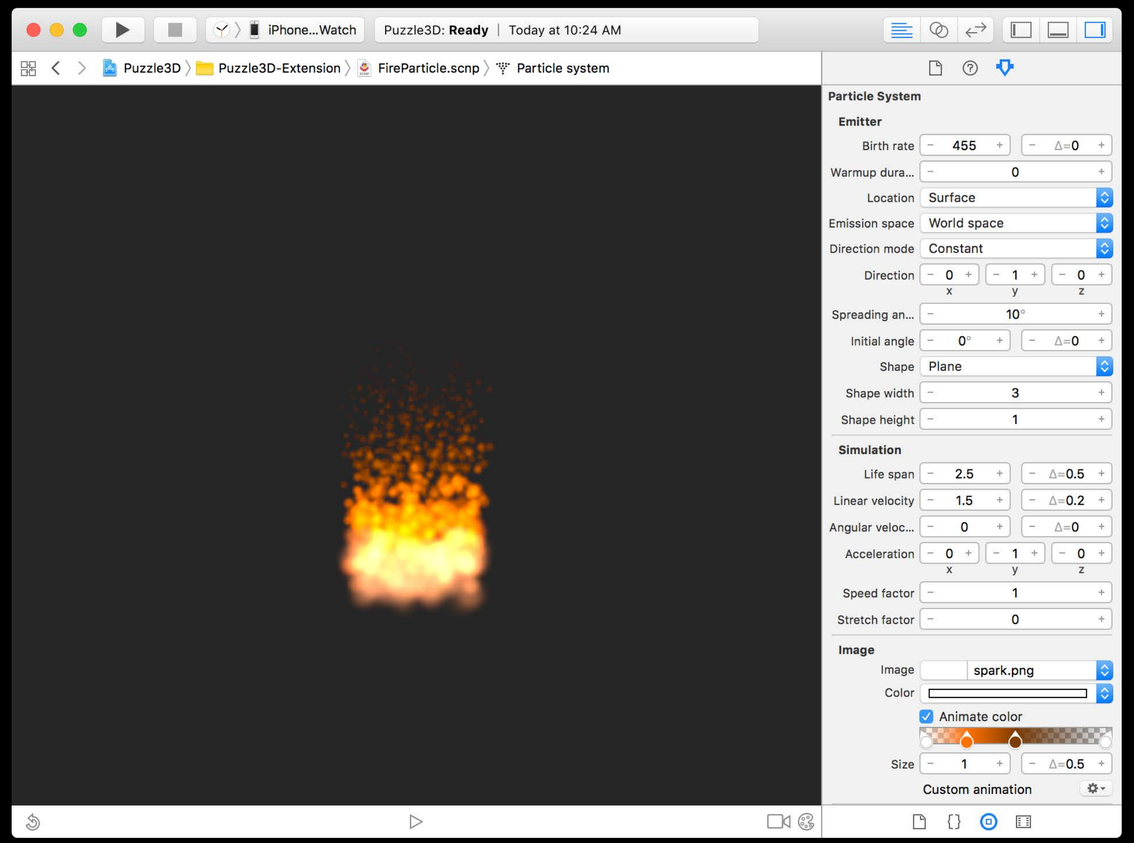
SceneKit

Game Tools for Apple Watch

Particle editor



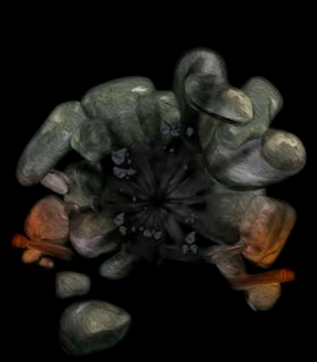
SpriteKit



SceneKit

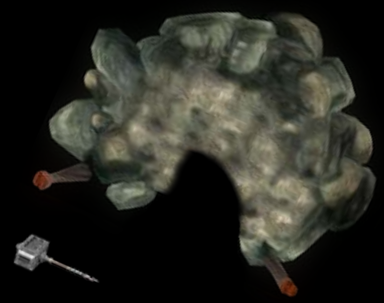
Game Tools for Apple Watch

Texture atlas

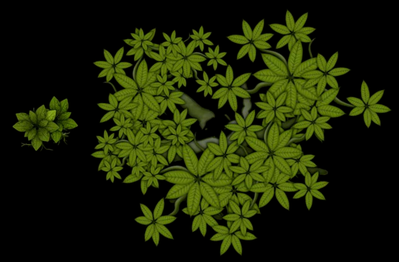


cave_destroyed.png

big_tree_base.png

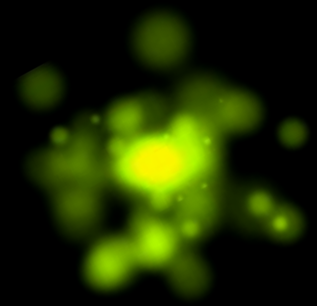


cave_base.png

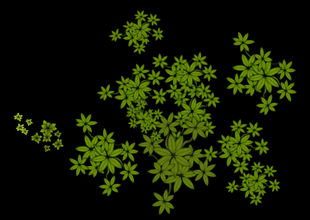


big_tree_middle.png

cave_top.png



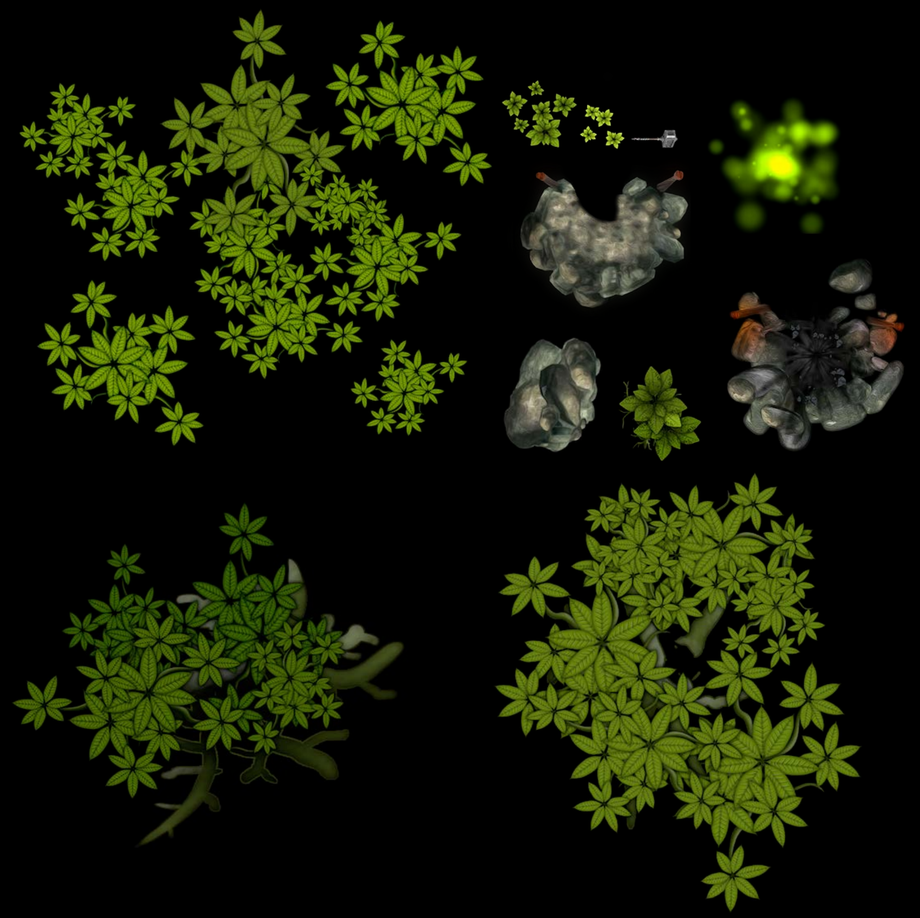
minionSplort.png



big_tree_top.png

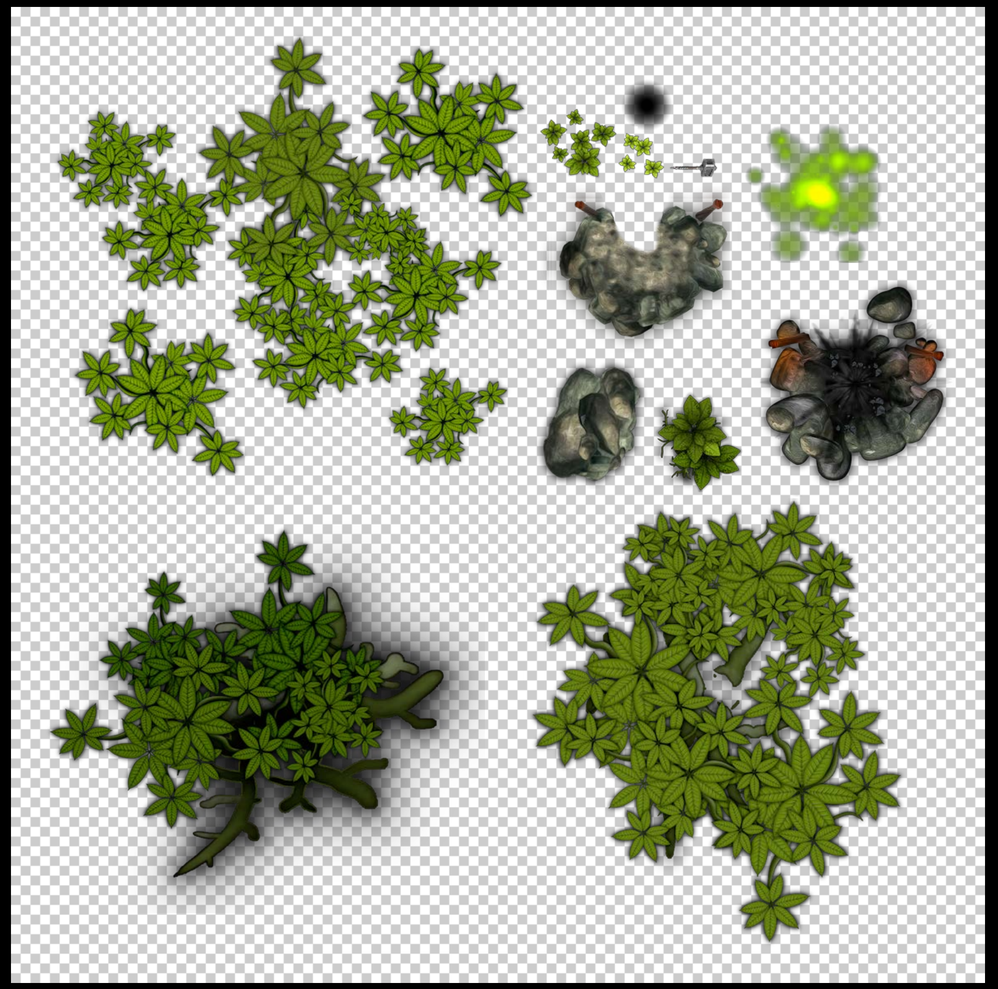
Game Tools for Apple Watch

Texture atlas



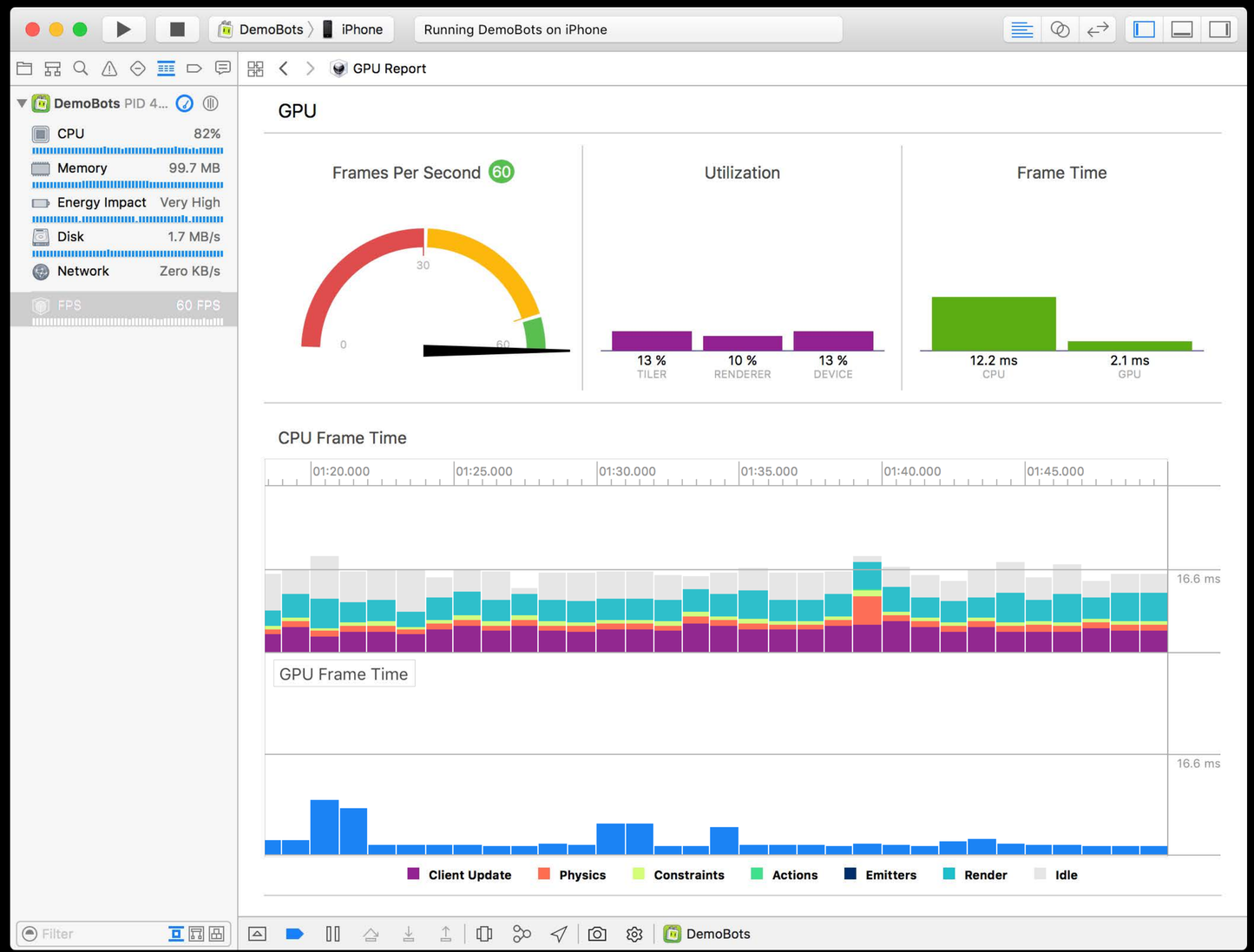
Game Tools for Apple Watch

Texture atlas



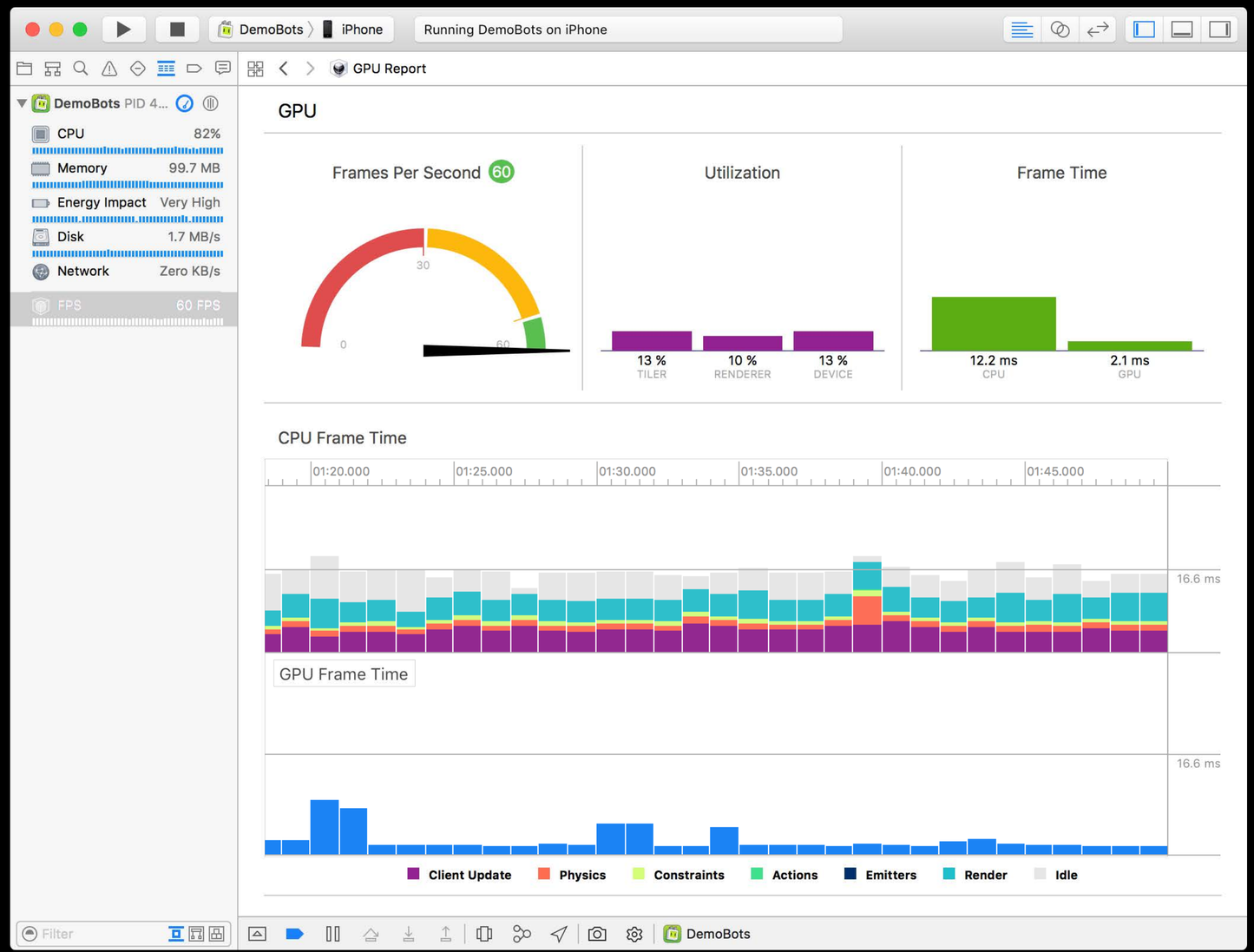
FPS Performance Gauge

Real-time performance



FPS Performance Gauge

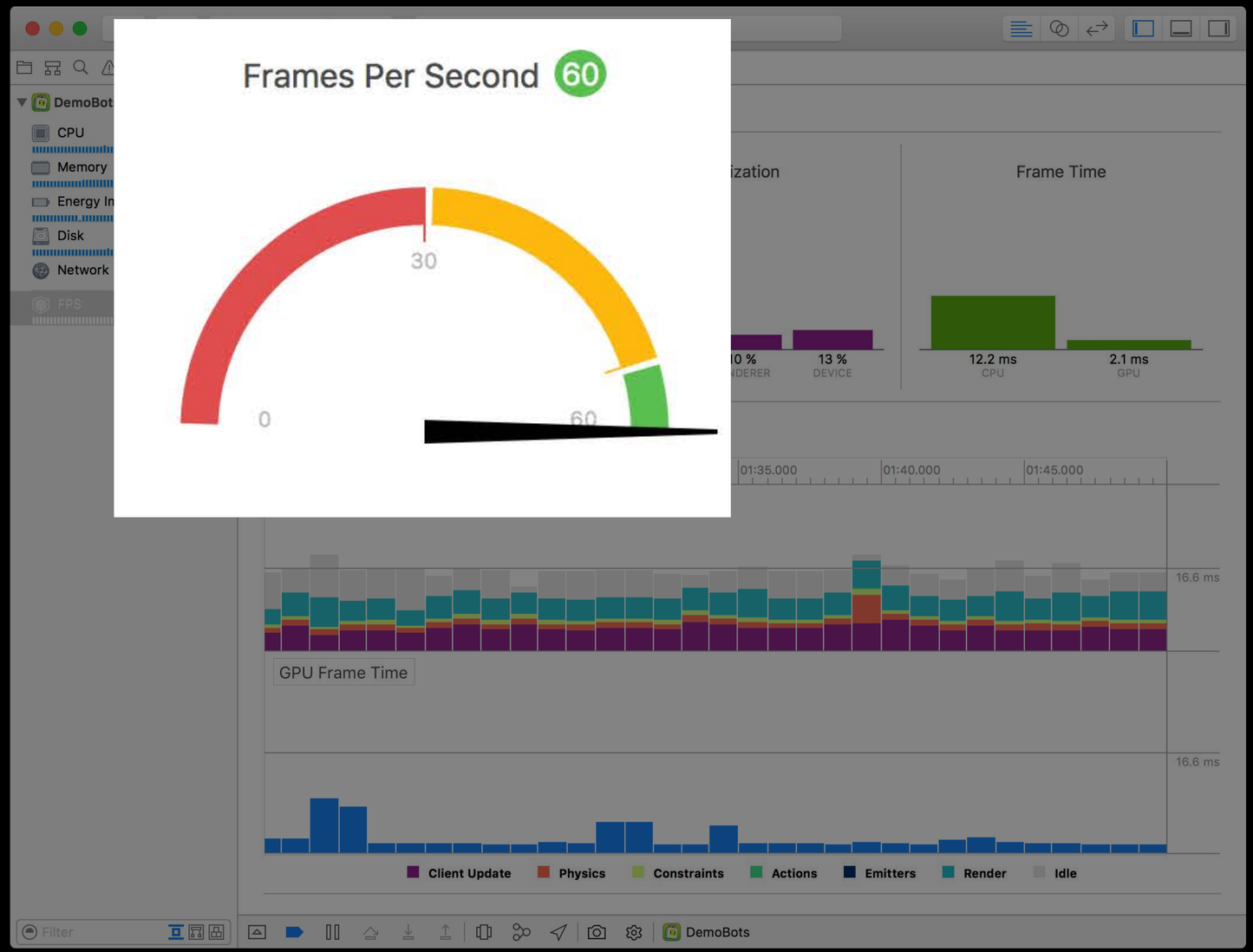
Real-time performance



FPS Performance Gauge

Real-time performance

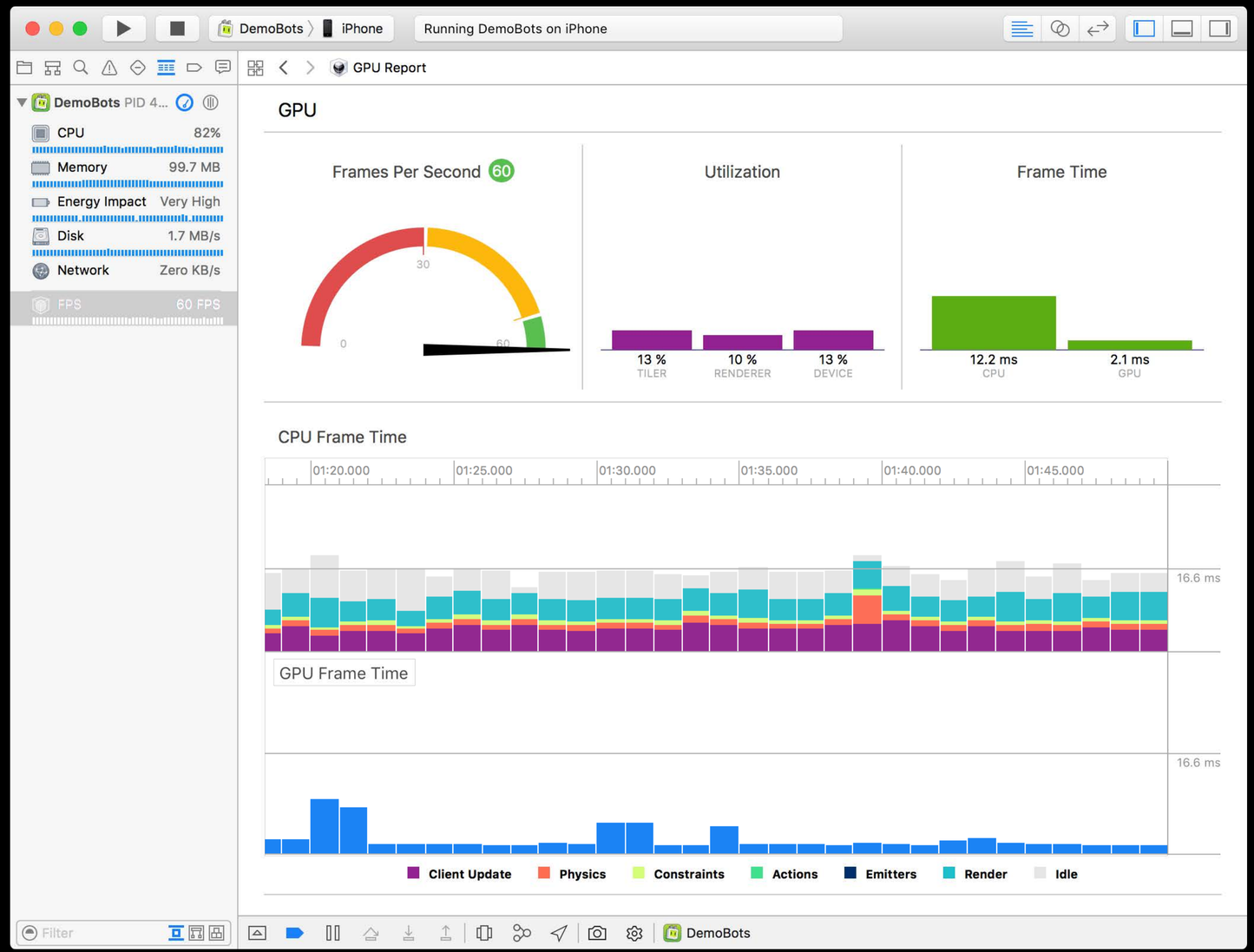
Frame rate



FPS Performance Gauge

Real-time performance

Frame rate

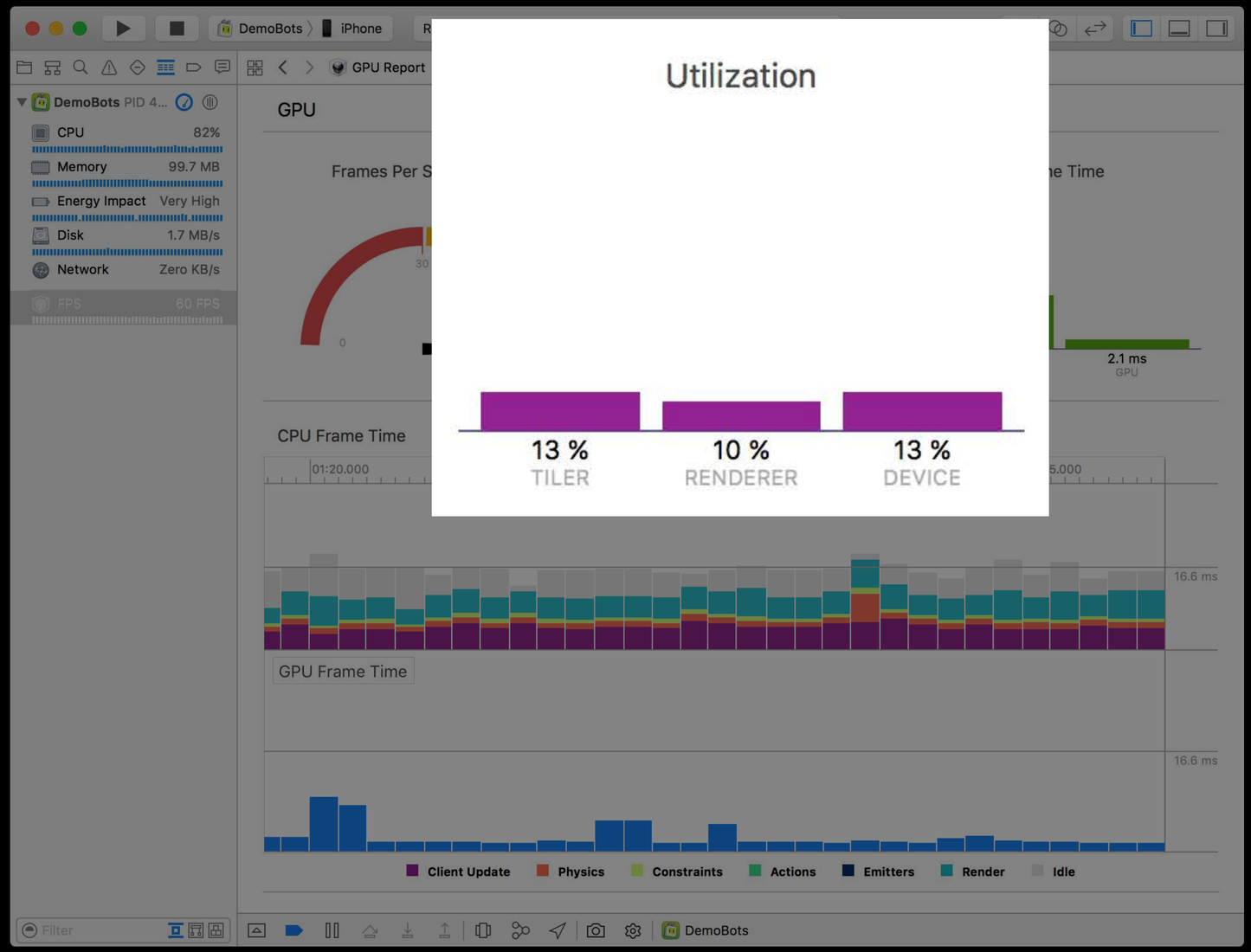


FPS Performance Gauge

Real-time performance

Frame rate

GPU utilization

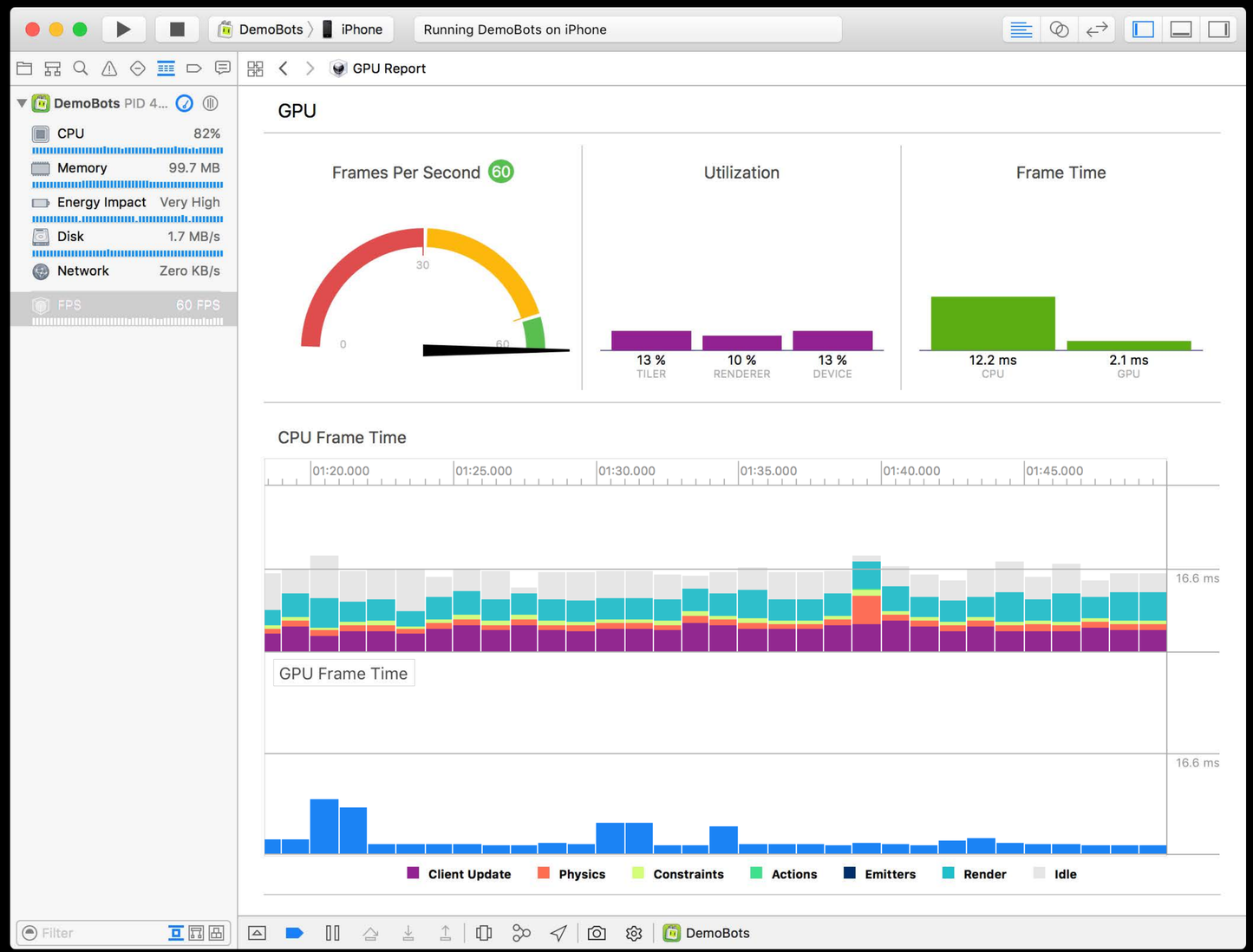


FPS Performance Gauge

Real-time performance

Frame rate

GPU utilization



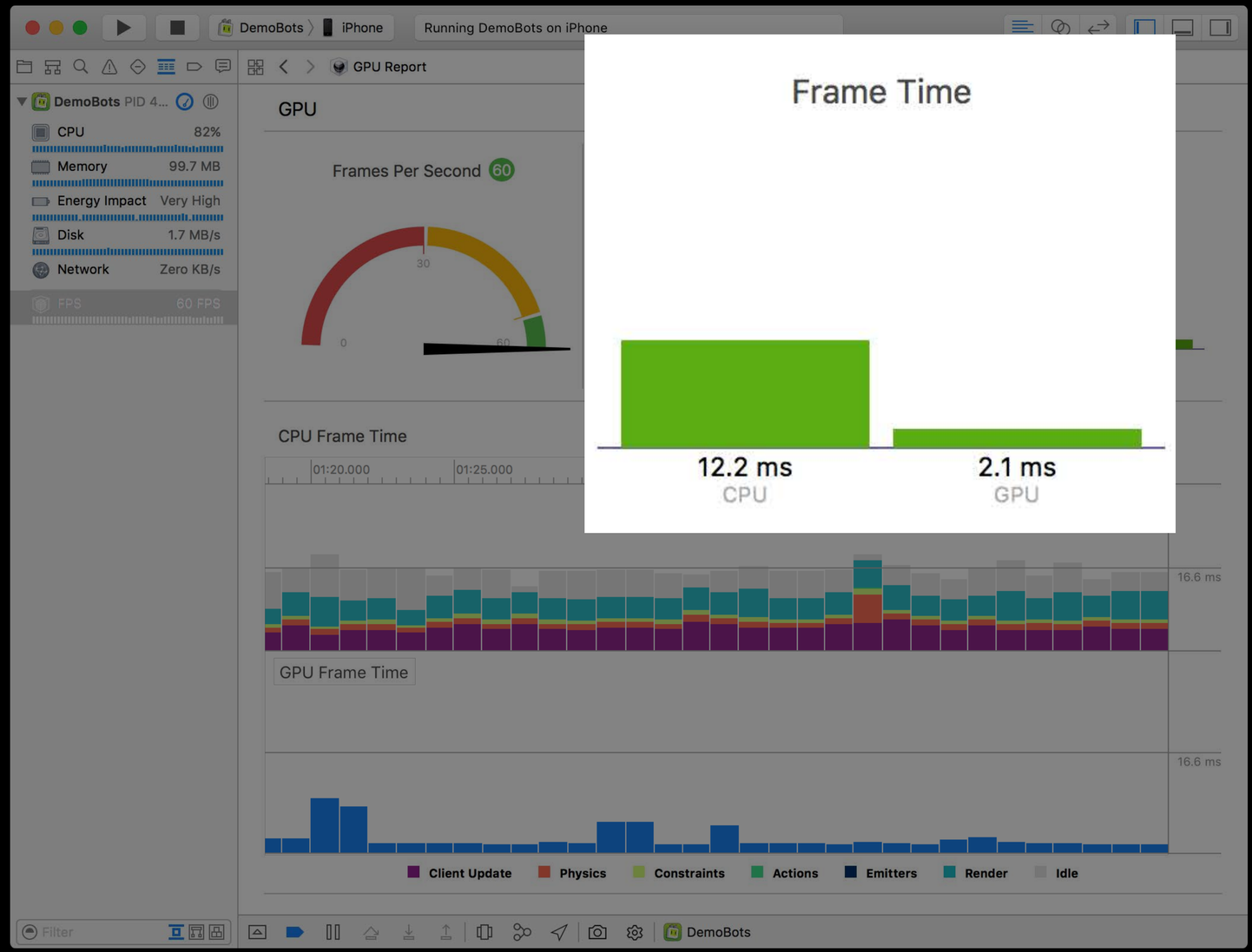
FPS Performance Gauge

Real-time performance

Frame rate

GPU utilization

CPU / GPU frame time



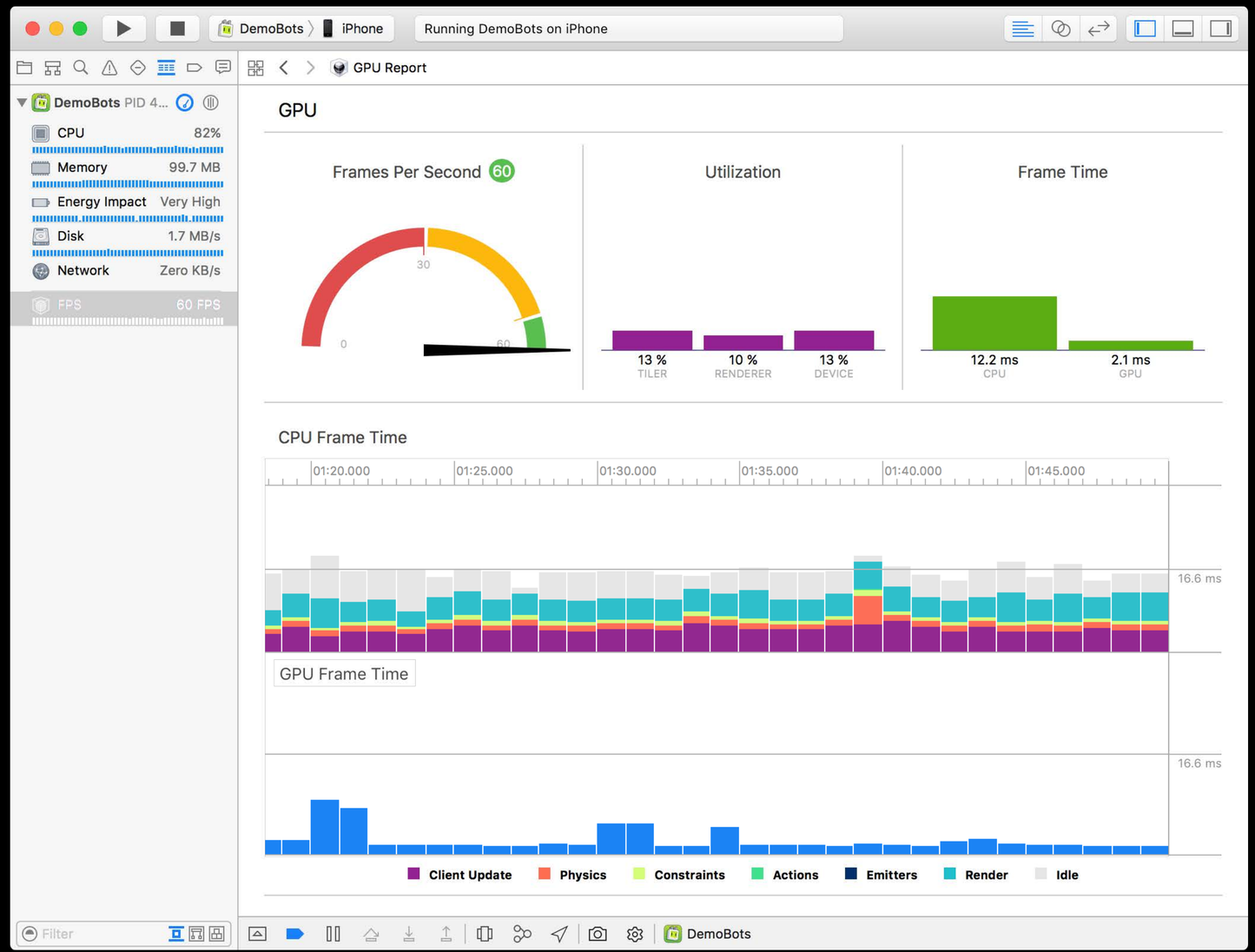
FPS Performance Gauge

Real-time performance

Frame rate

GPU utilization

CPU / GPU frame time



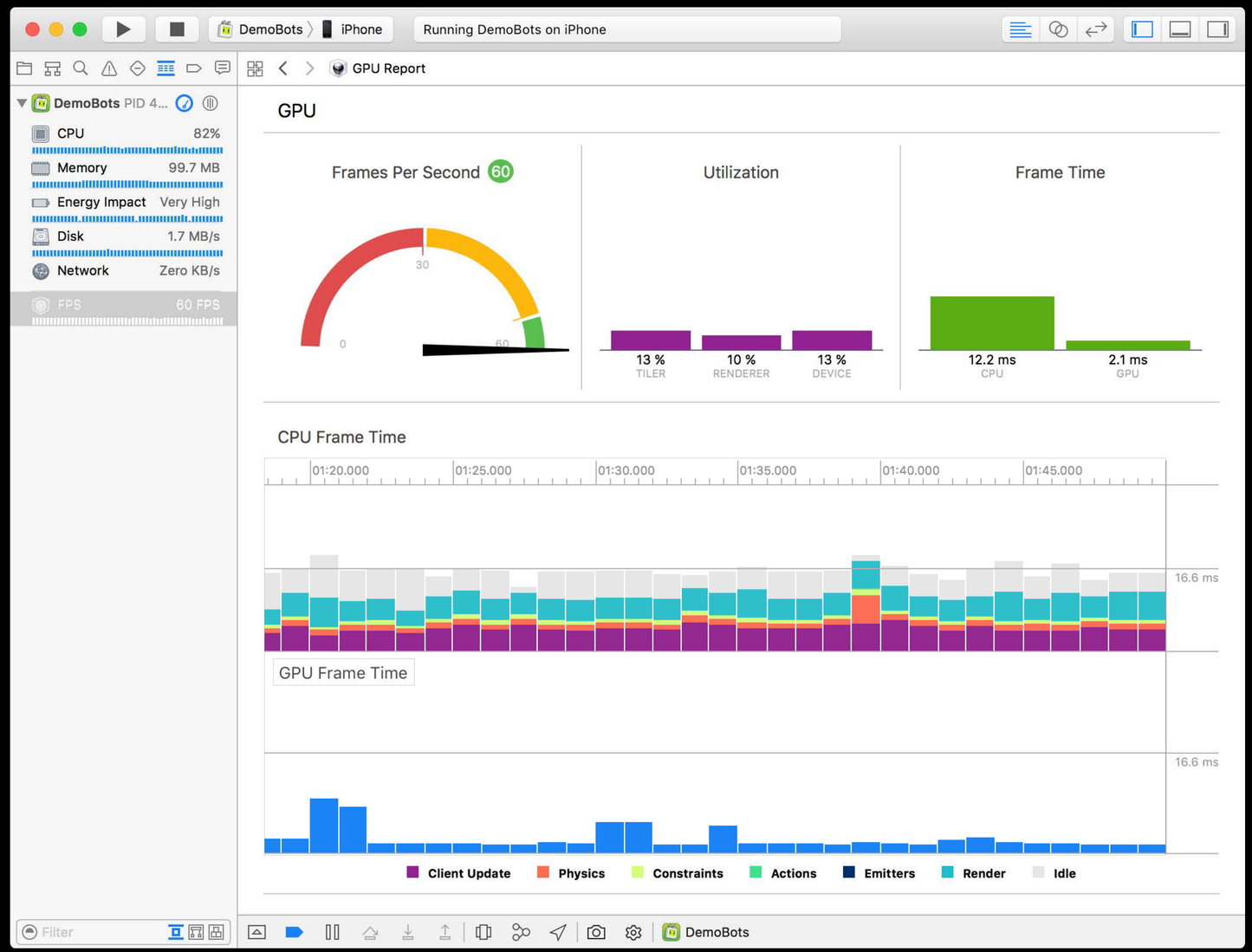
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



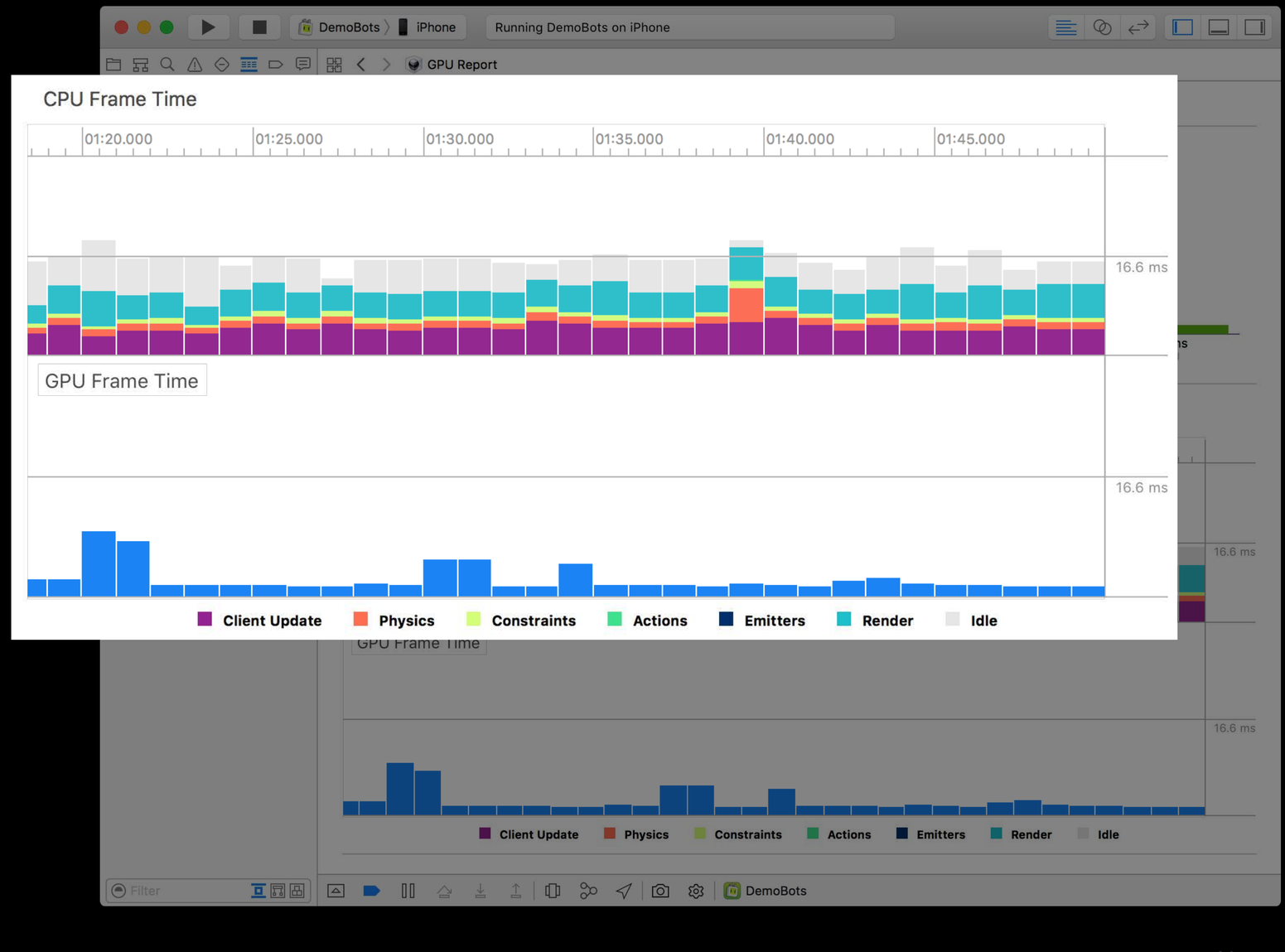
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



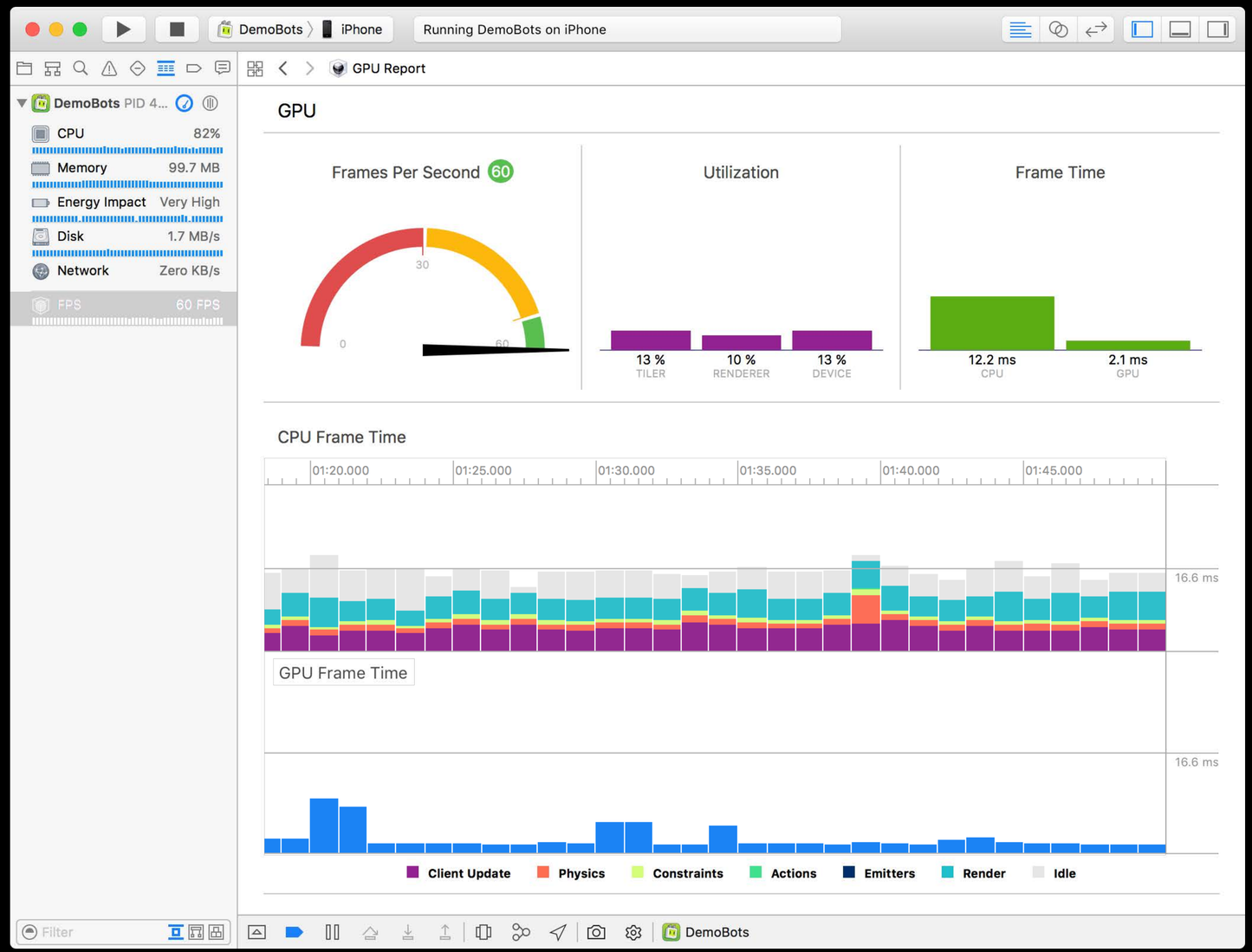
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



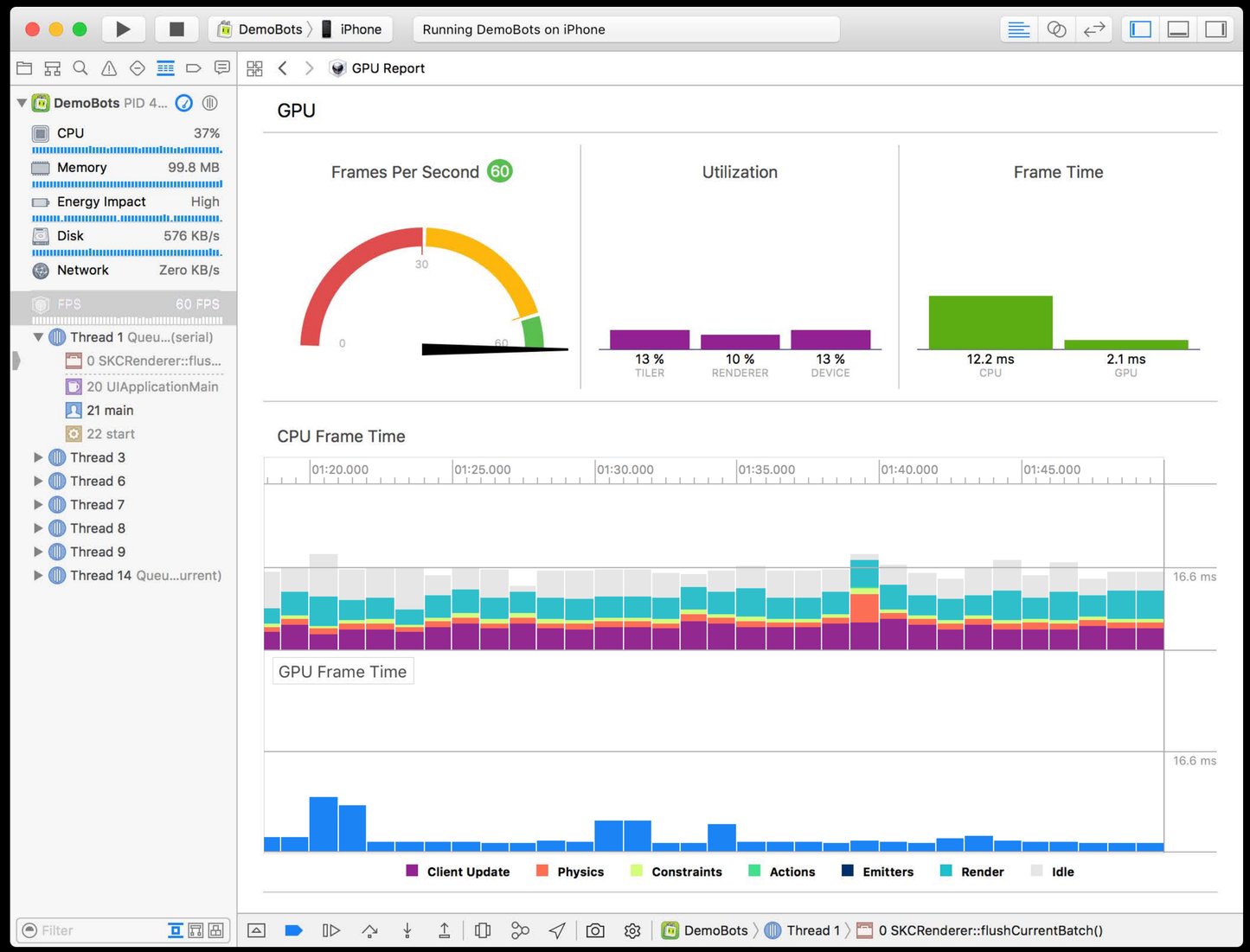
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



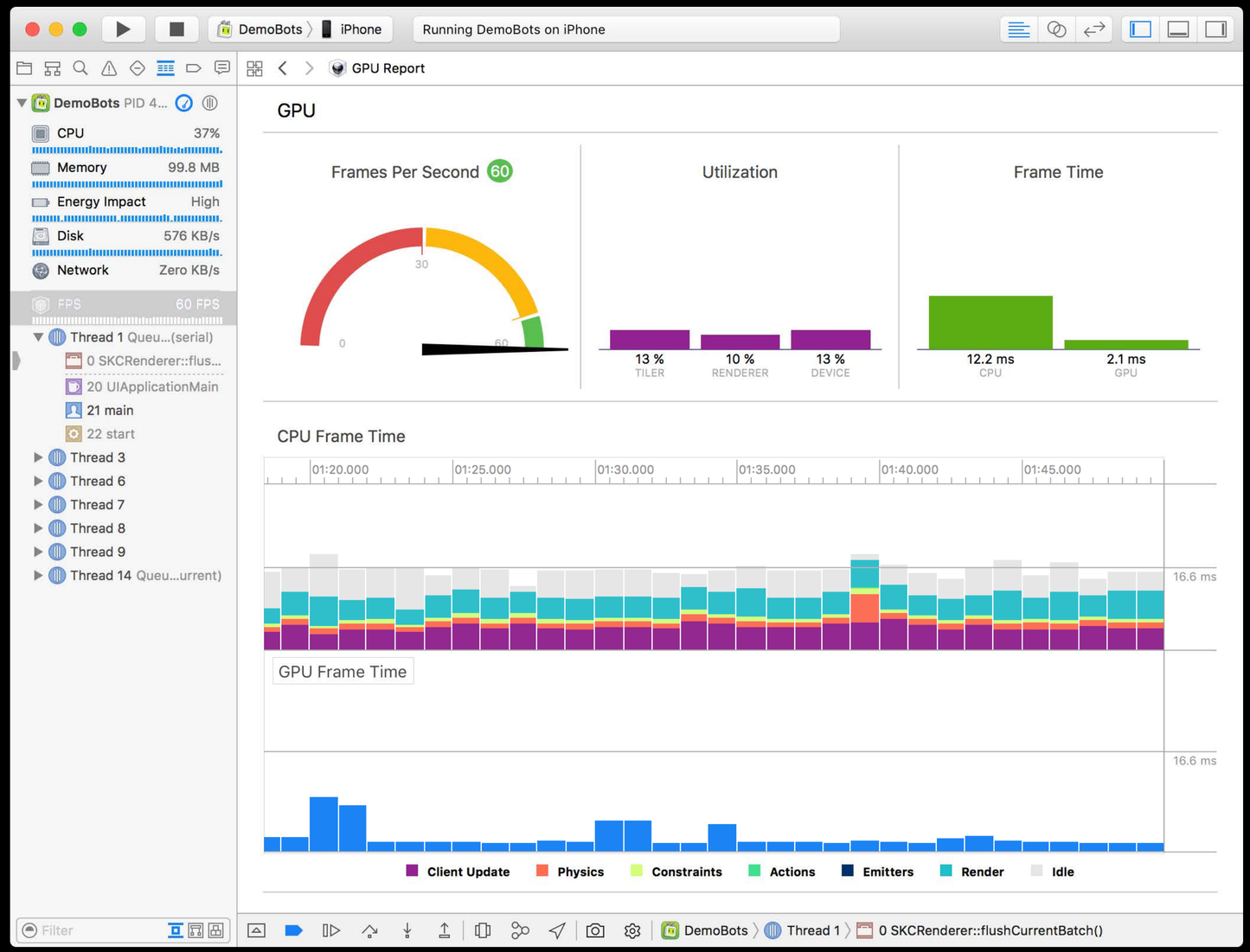
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



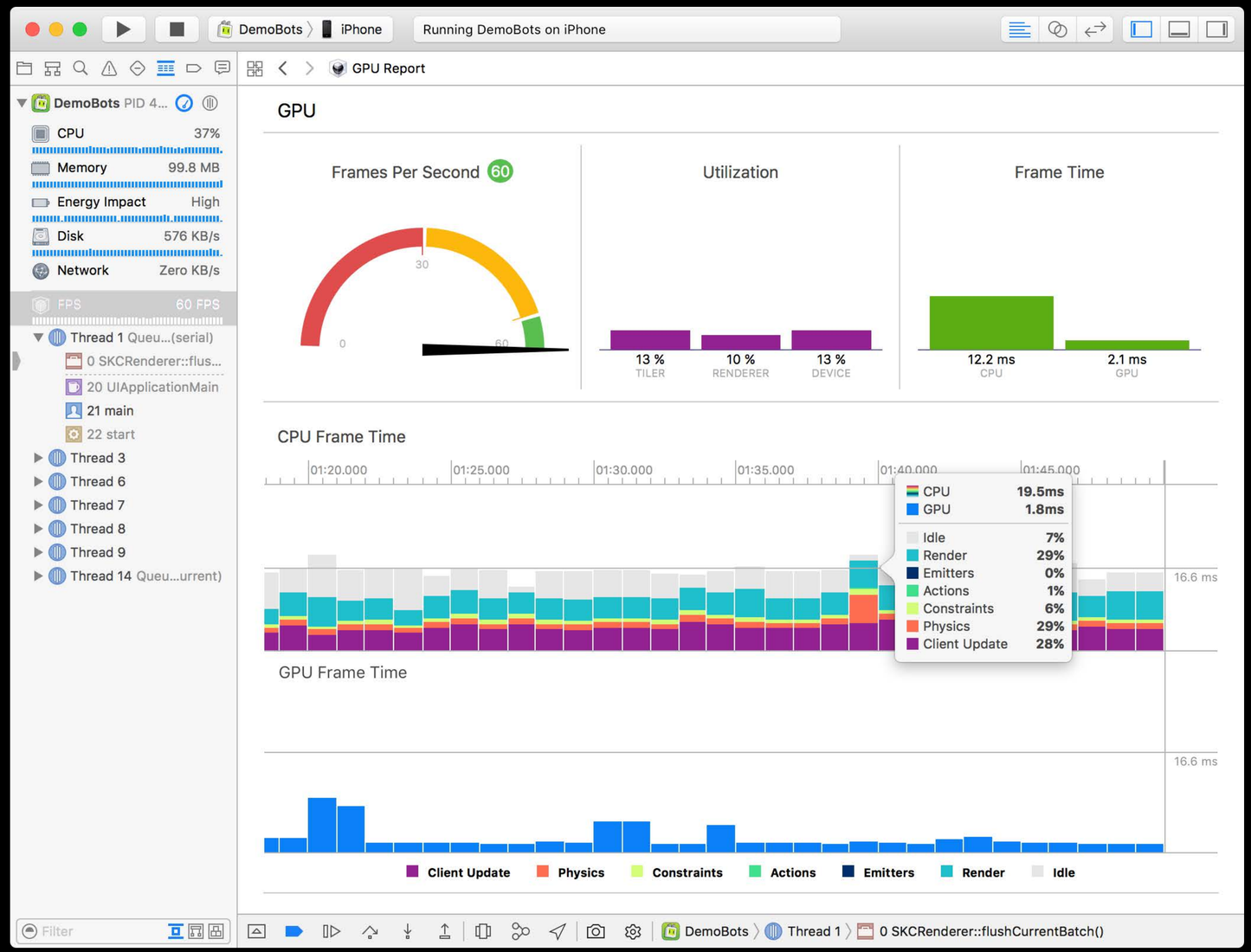
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



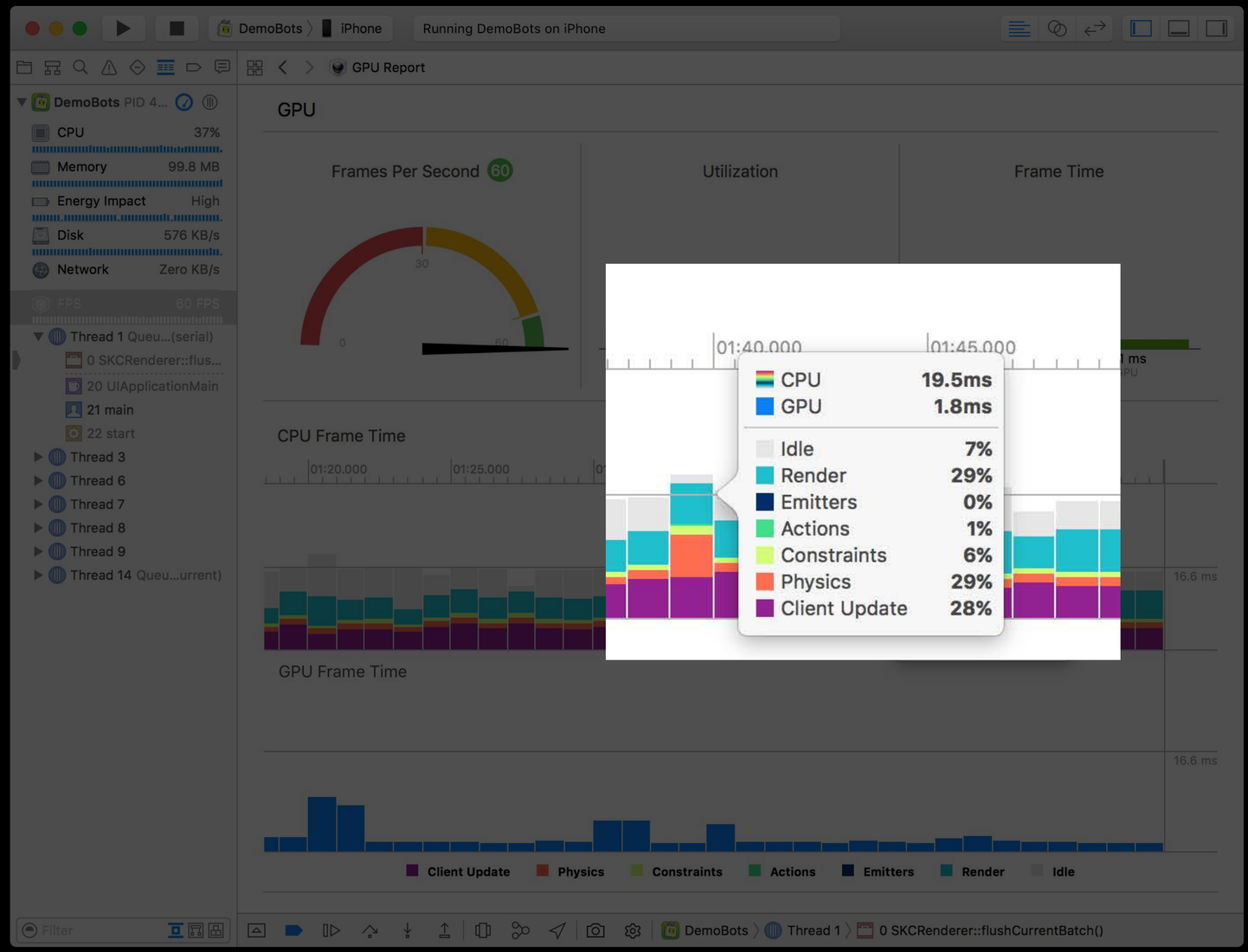
FPS Performance Gauge

Real-time performance

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks



Best Practices

Performance

Avoid continuous updates

Preload assets to avoid bottlenecks

Maintain consistent frame rate

If experiencing performance issues

- Reduce particle count
- Reduce asset details
- Reduce object count

Best Practices

Game considerations

Lightweight interactions

- See progress at a glance
- On the go

Concise interface

- Avoid busy scenes
- Large touch targets

Visual continuity

- Tint color
- Black background

Summary

Exciting year for Apple Watch developers

- New input and feedback
- Powerful graphics frameworks
- Easy social gaming integration
- Full suite of tools

More Information

https://developer.apple.com/wwdc16/612

Related Sessions

Quick Interaction Techniques for watchOS	Presidio	Wednesday 11:00AM
Visual Debugging with Xcode	Presidio	Wednesday 4:00PM
Advances in SceneKit Rendering	Mission	Thursday 11:00AM
What's New in SpriteKit	Presidio	Thursday 5:00PM
What's New in Game Center	Mission	Friday 10:00AM

Labs

watchOS Graphics and Games Lab	Graphics, Games, and Media Lab B	Friday 4:00PM
Xcode Open Hours	Developer Tools Lab B	Friday 3:00PM

WWDC16