What's New in SpriteKit

Session 610

Ross Dexter Games Technologies Engineer
Clément Boissière Games Technologies Engineer

What Is SpriteKit?

What Is SpriteKit?

Framework features

2D graphics framework for games

Flexible, easy to use, high-performance

Supported on iOS, macOS, tvOS & watchOS

Automatic access to the latest updates

Natural integration with Swift

What Is SpriteKit?

Xcode-integrated live editor

Visually lay out your game scenes

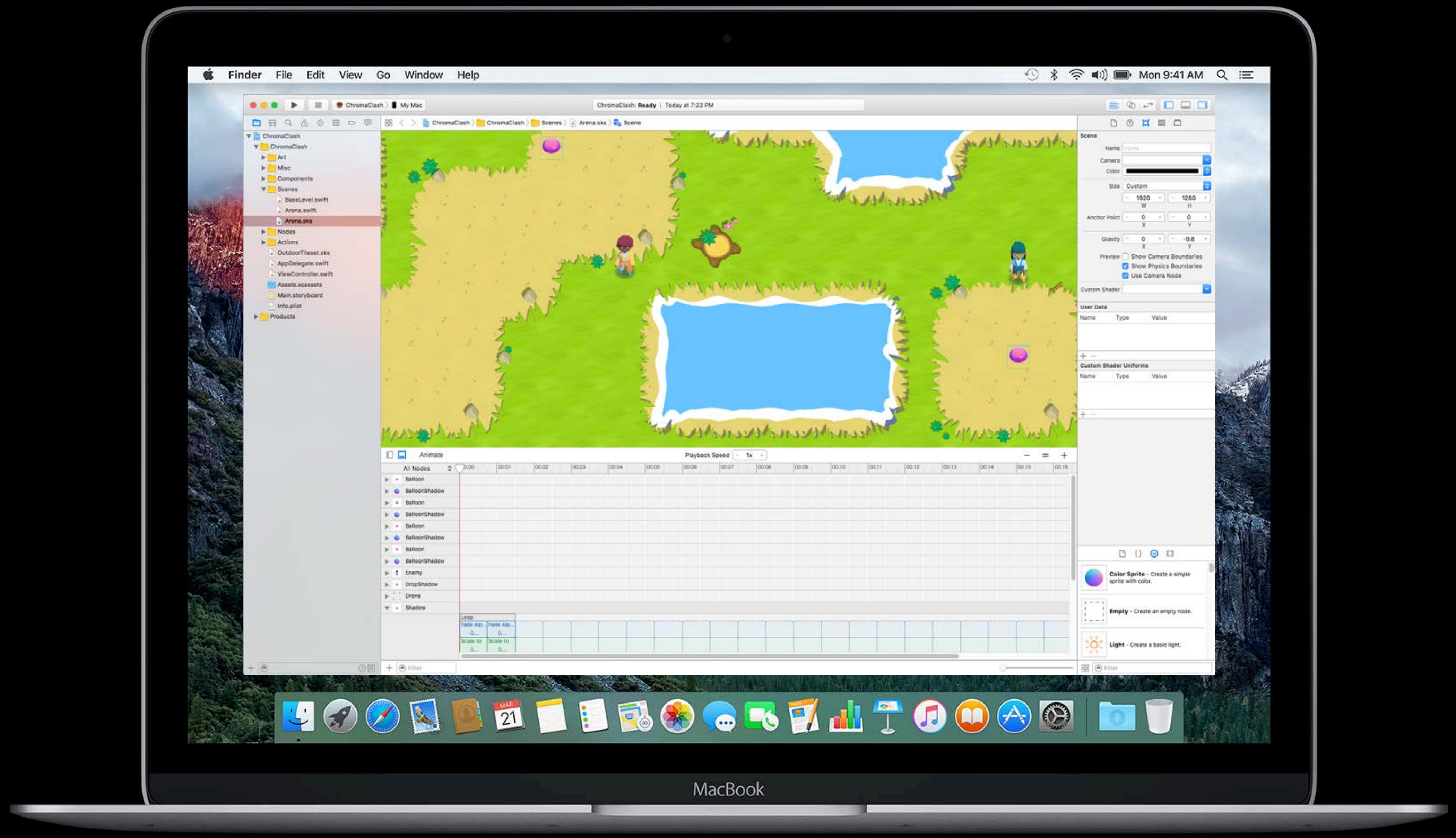
Timeline-based animation

Particle editor

Asset Catalog integration

Tile map editing

GameplayKit integration



What Is SpriteKit?

Built-in Metal support

Automatically Metal-backed

Zero action required for developers

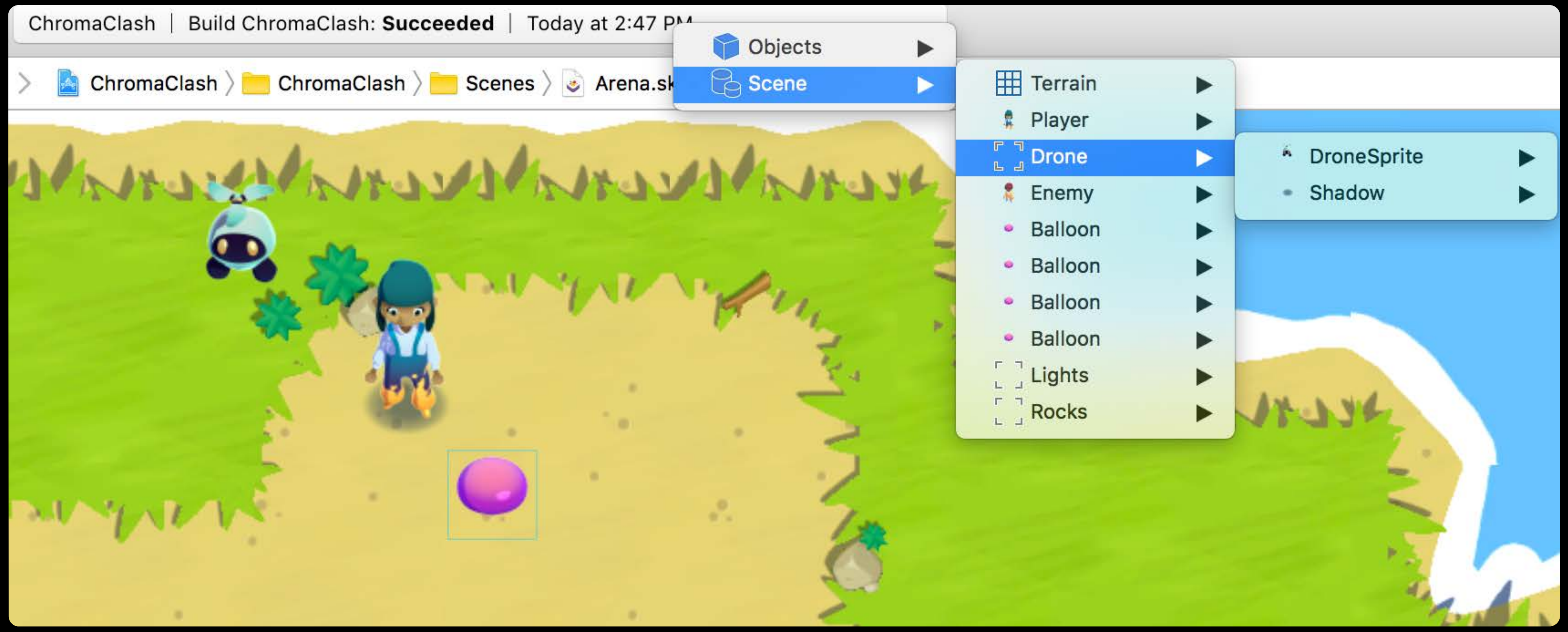
Scene Outline View

Scene Outline View

Scene hierarchy at-a-glance

The Jump Bar contains the scene hierarchy

- Only allows for selection
- Shows one branch at a time



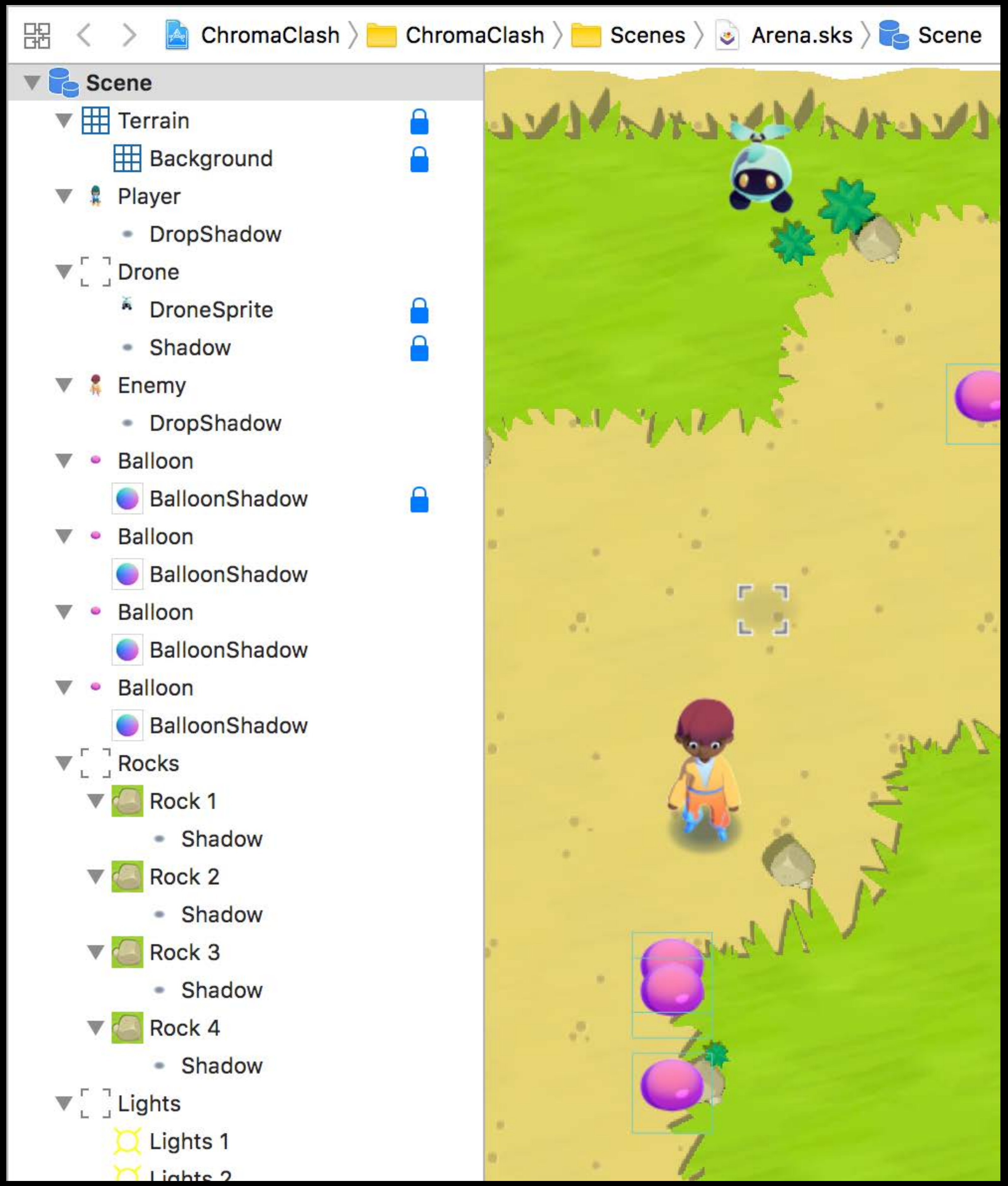
Scene Outline View

Scene hierarchy at-a-glance

NEW

Scene Outline View shows the entire hierarchy

- Select, rename, remove
- Drag to parent/unparent
- Can lock and/or hide nodes



GameplayKit Integration

GameplayKit Integration

Entities and components

Design pattern focused on modularity

Components encapsulate behavior

- Health
- Collision
- Player input

Write it once, assign to multiple objects

GameplayKit Integration

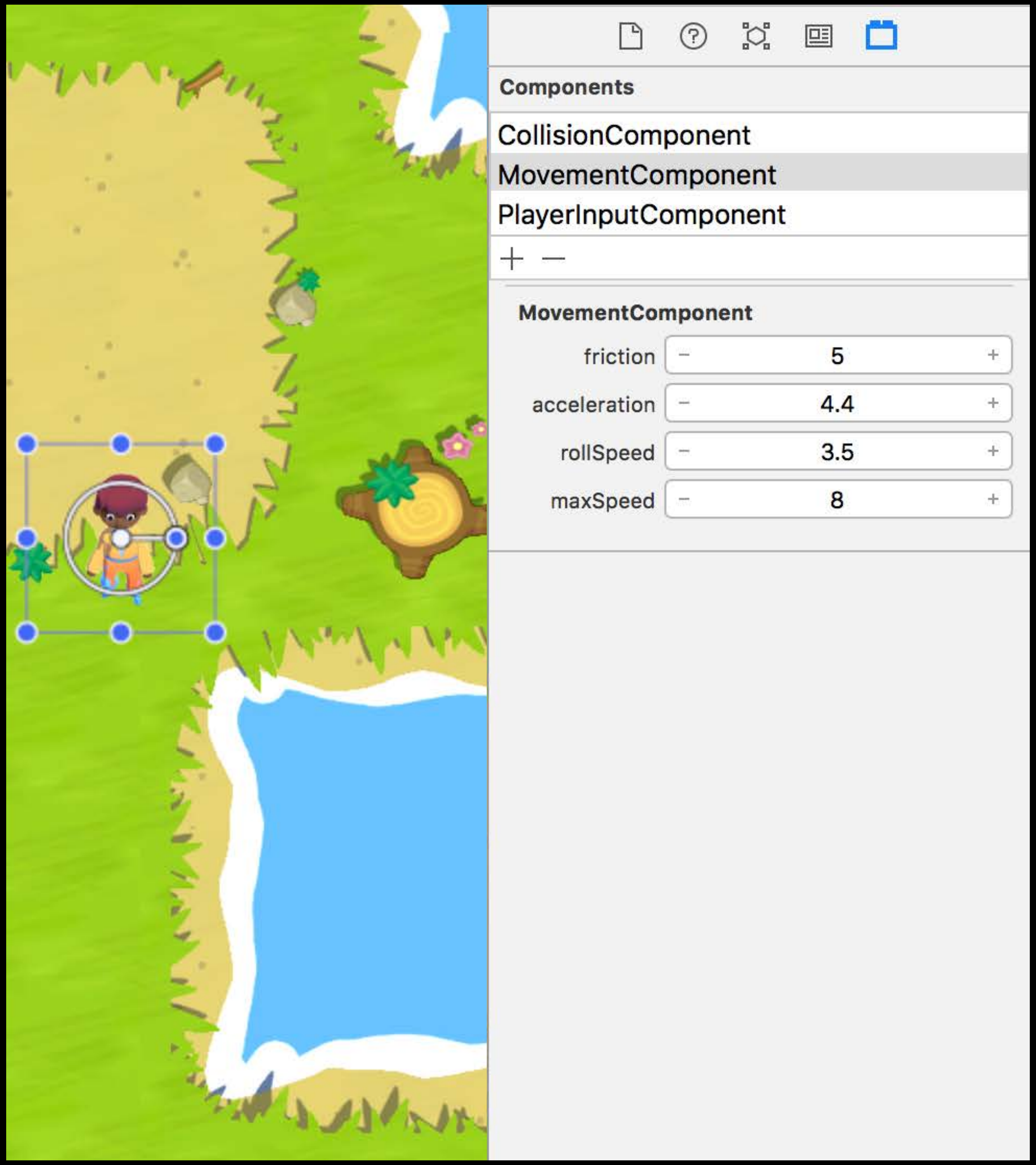
Entities and components

NEW

Assign components directly from the editor

Properties can be tweaked via the inspector

We take care of the hard stuff for you



Components

CollisionComponent
MovementComponent
PlayerInputComponent

MovementComponent

friction	5
acceleration	4.4
rollSpeed	3.5
maxSpeed	8

GameplayKit Integration

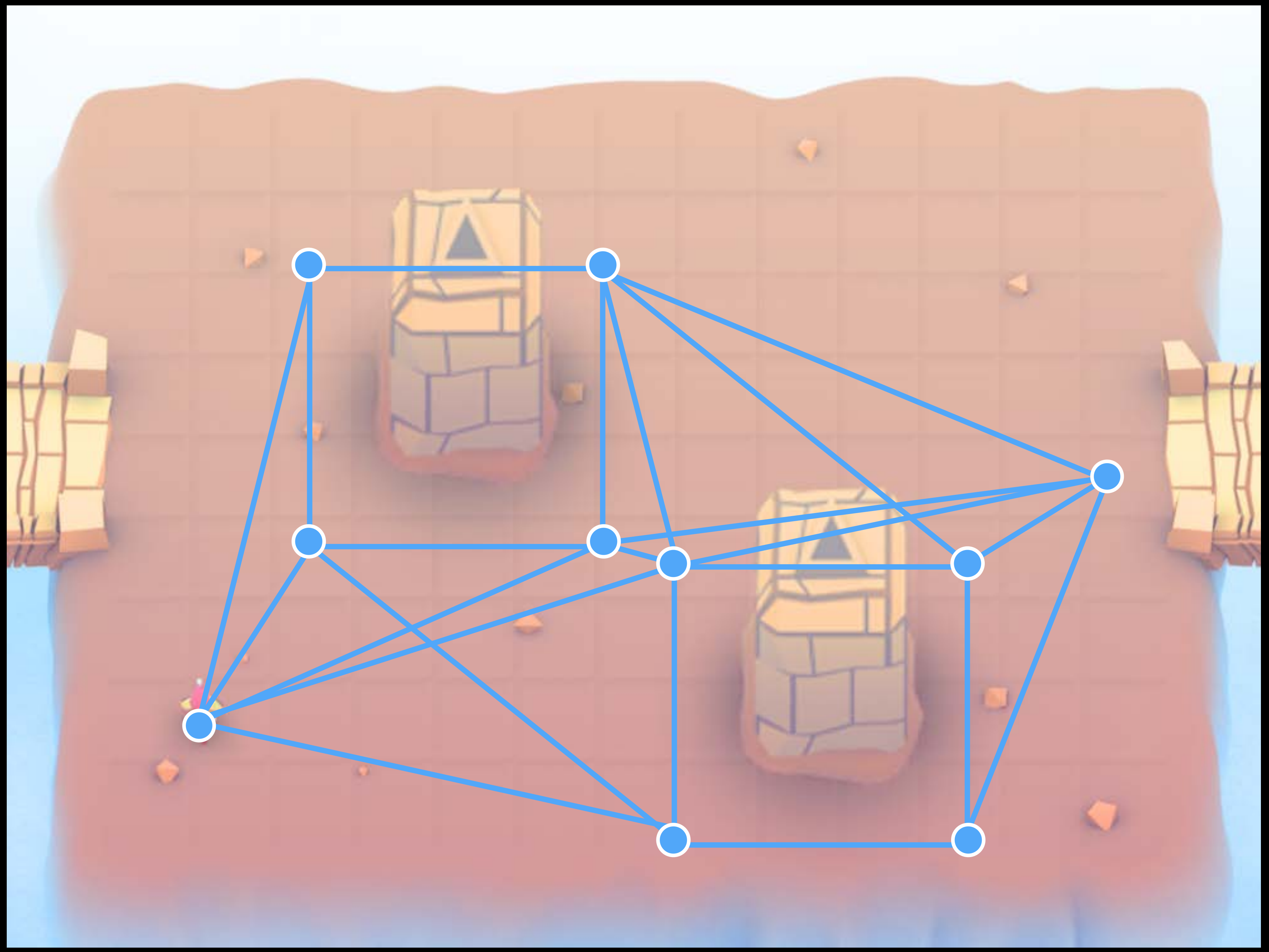
Pathfinding

Pathfinding uses navigation graphs

Graphs are collections of nodes

Nodes are joined by connections

Describes how to move through scene



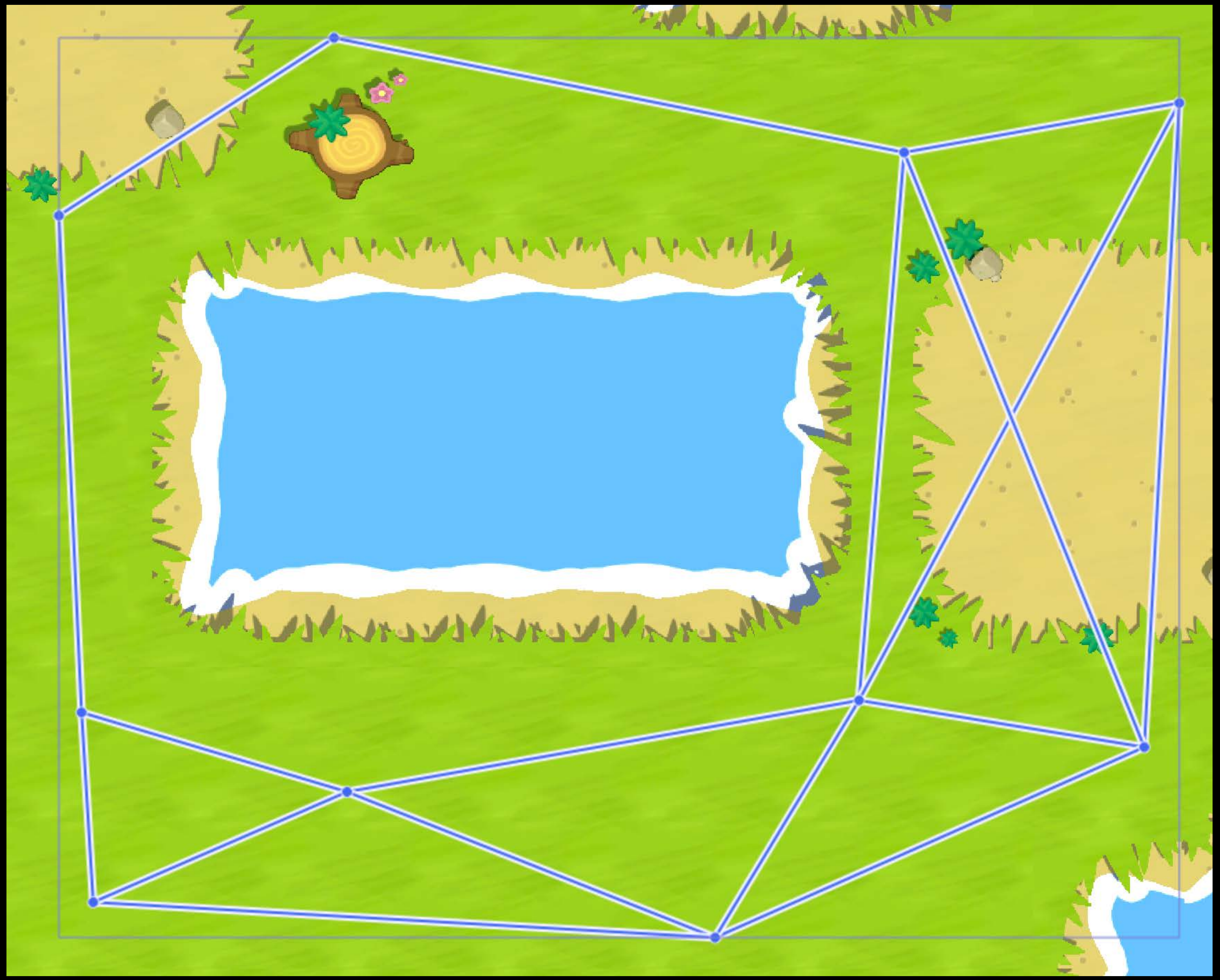
GameplayKit Integration

NEW

Navigation graph editor

Create and edit navigation graphs

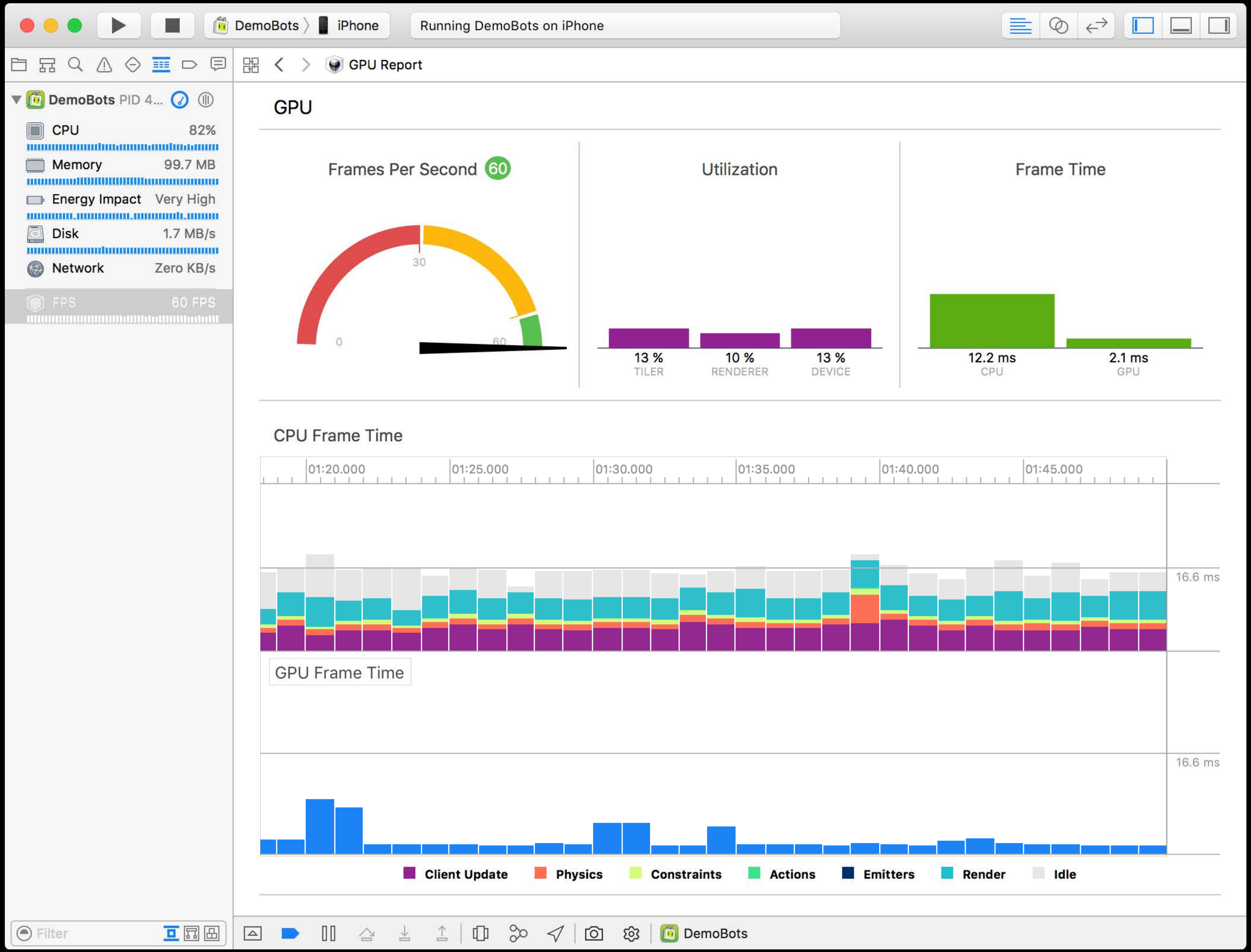
- Add and remove nodes
- Create or adjust connections



FPS Performance Gauge

FPS Performance Gauge

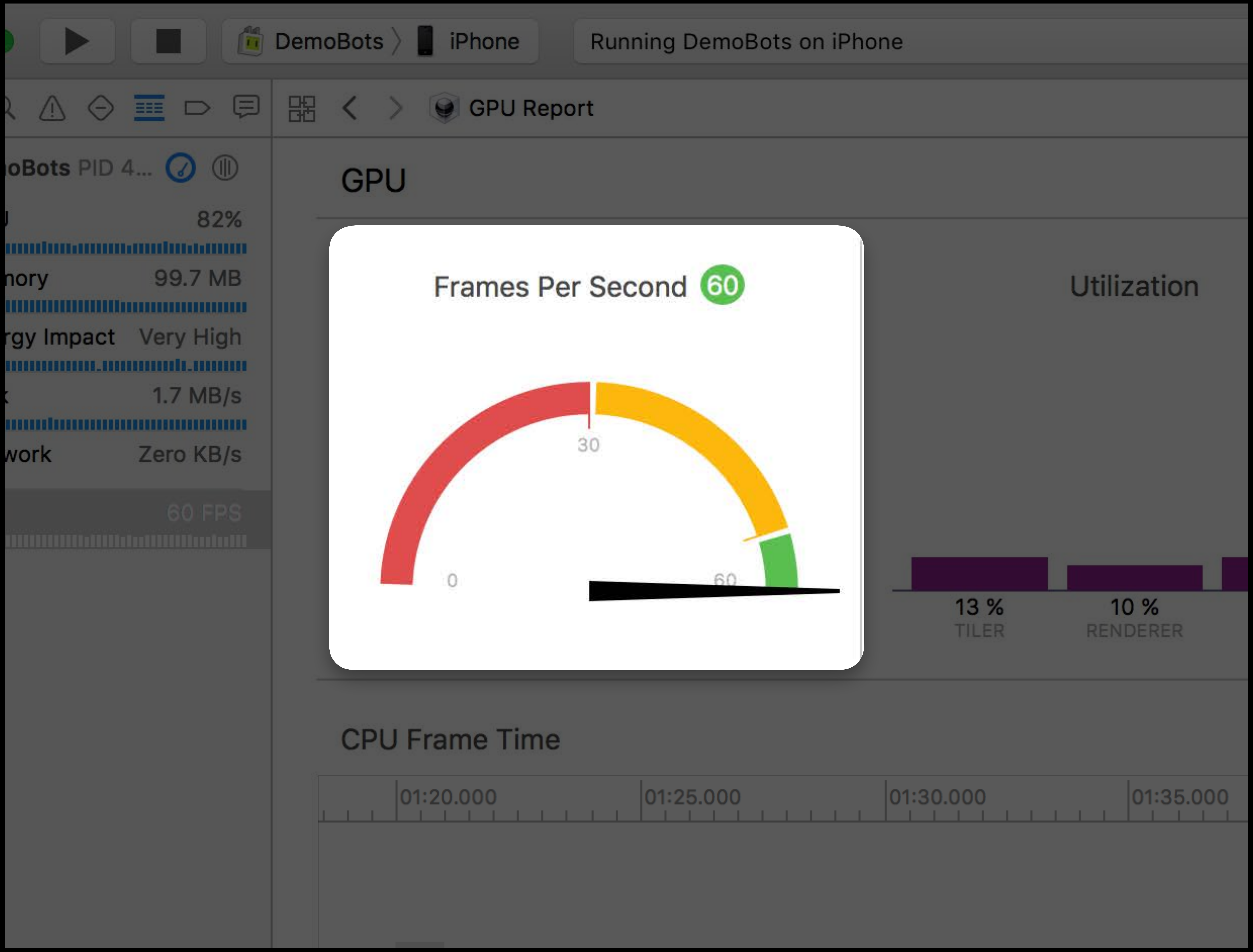
Real-time performance breakdown



FPS Performance Gauge

Real-time performance breakdown

Frame rate

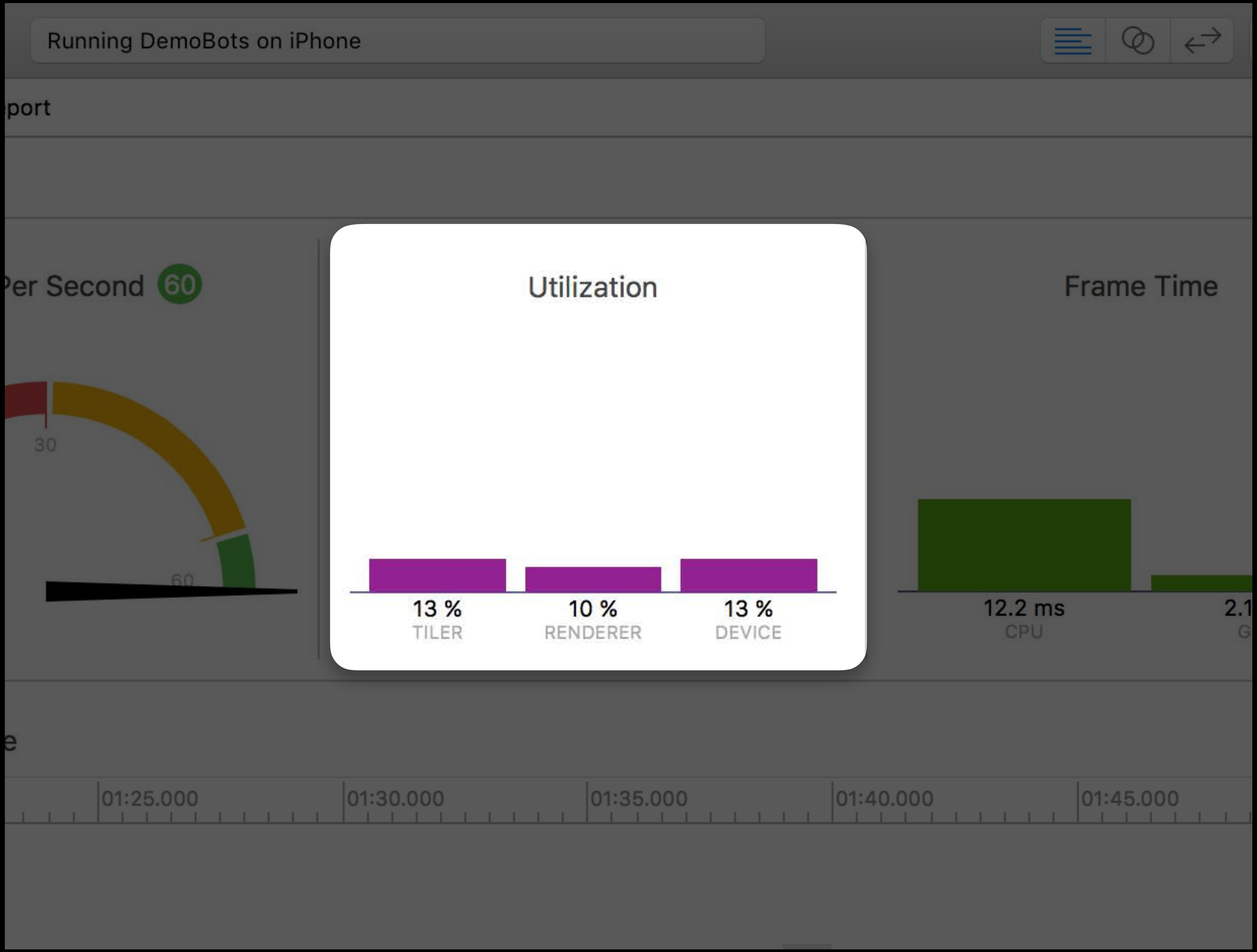


FPS Performance Gauge

Real-time performance breakdown

Frame rate

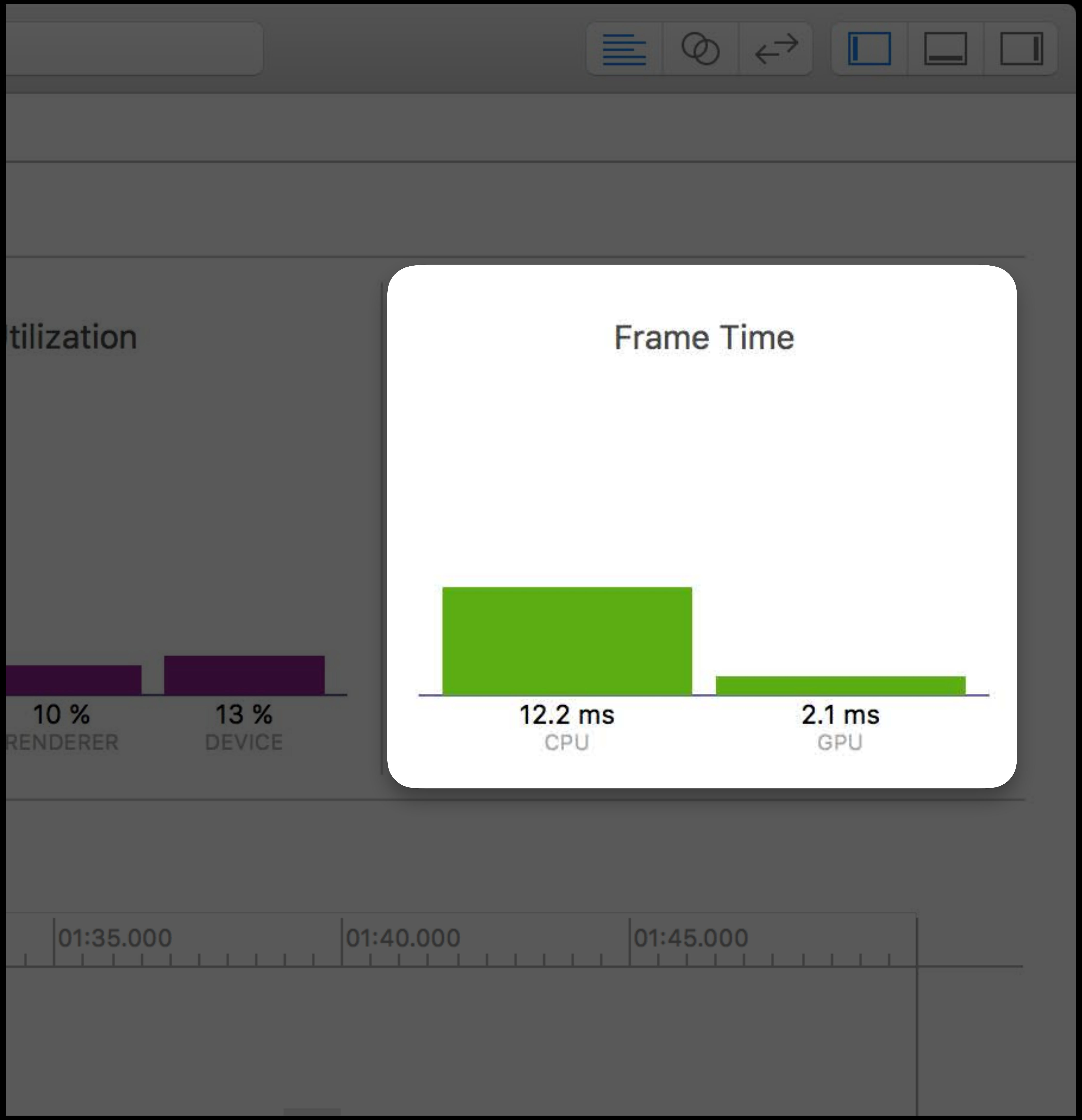
GPU utilization



FPS Performance Gauge

Real-time performance breakdown

- Frame rate
- GPU utilization
- CPU/GPU frame time



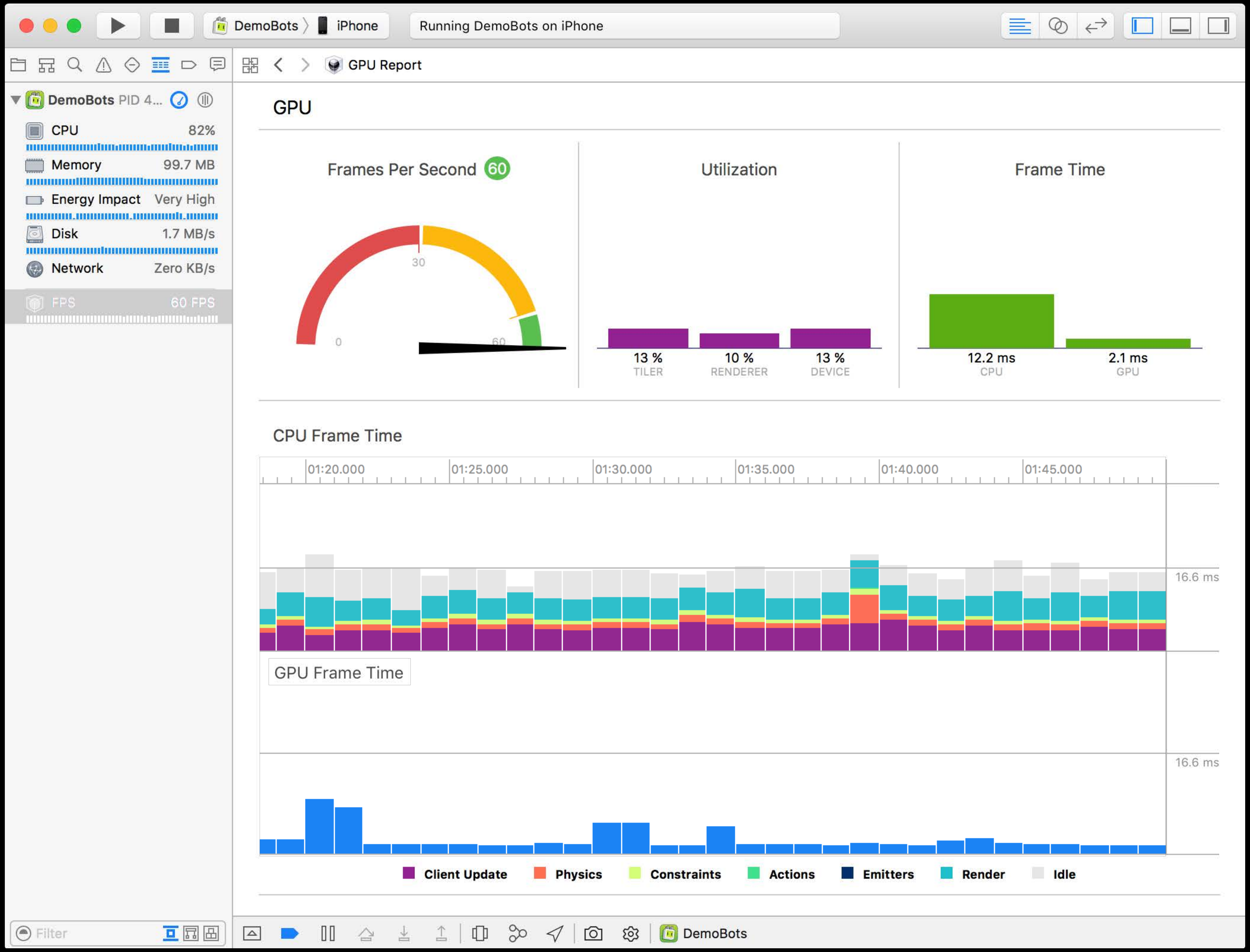
FPS Performance Gauge

Real-time performance breakdown

Frame rate

GPU utilization

CPU/GPU frame time



FPS Performance Gauge

Real-time performance breakdown

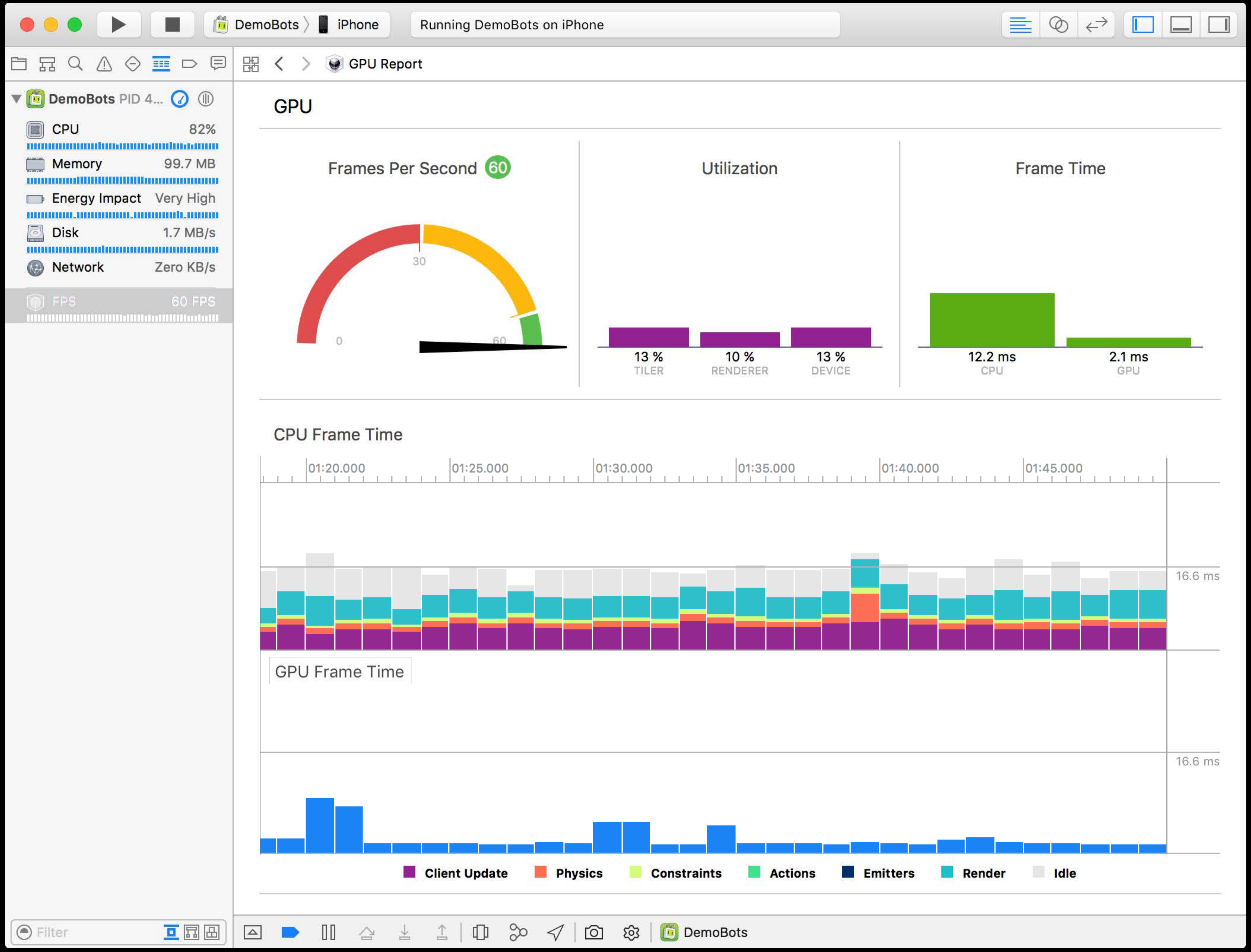
NEW

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks

Available on iOS and watchOS



FPS Performance Gauge

NEW

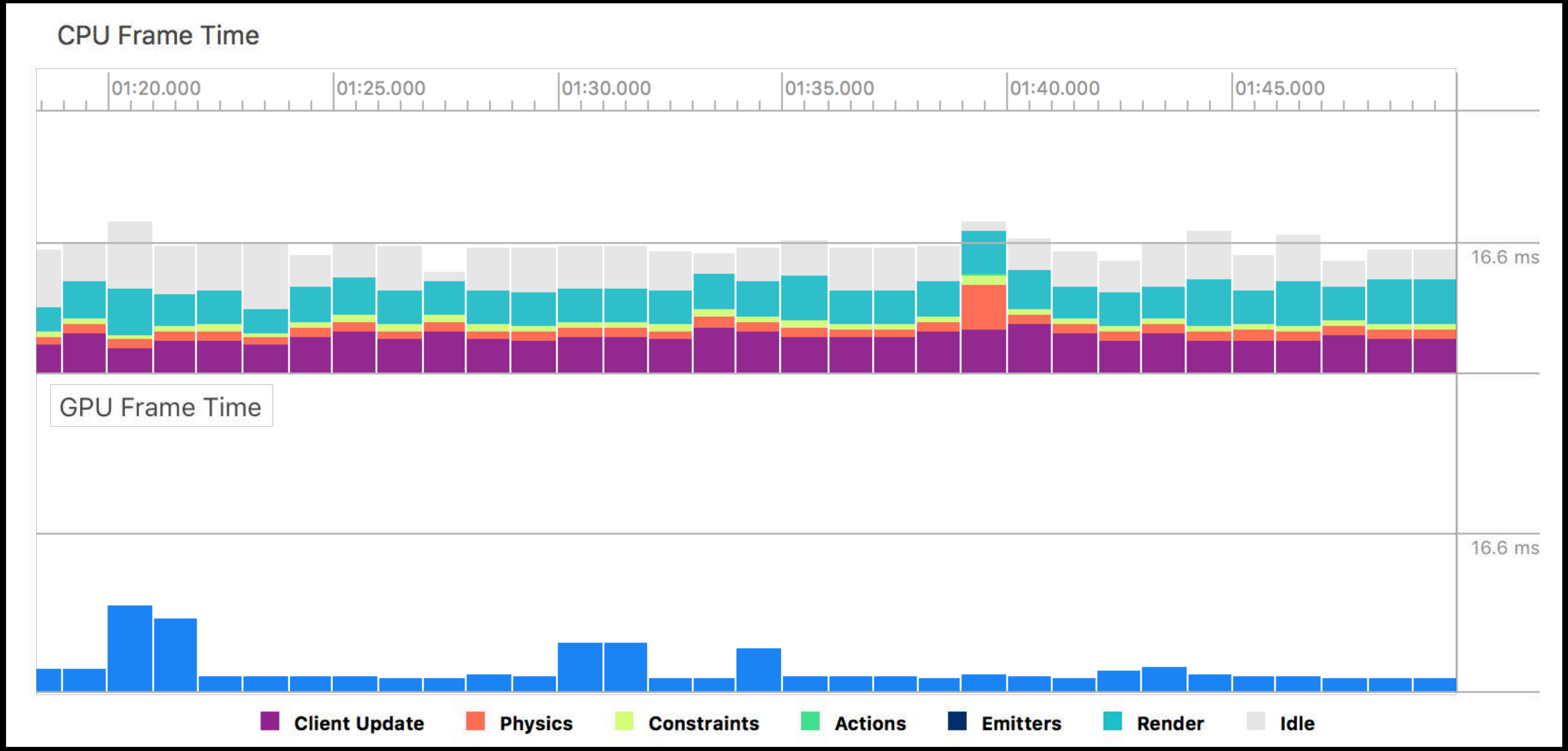
Real-time performance breakdown

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks

Available on iOS and watchOS



FPS Performance Gauge

Real-time performance breakdown

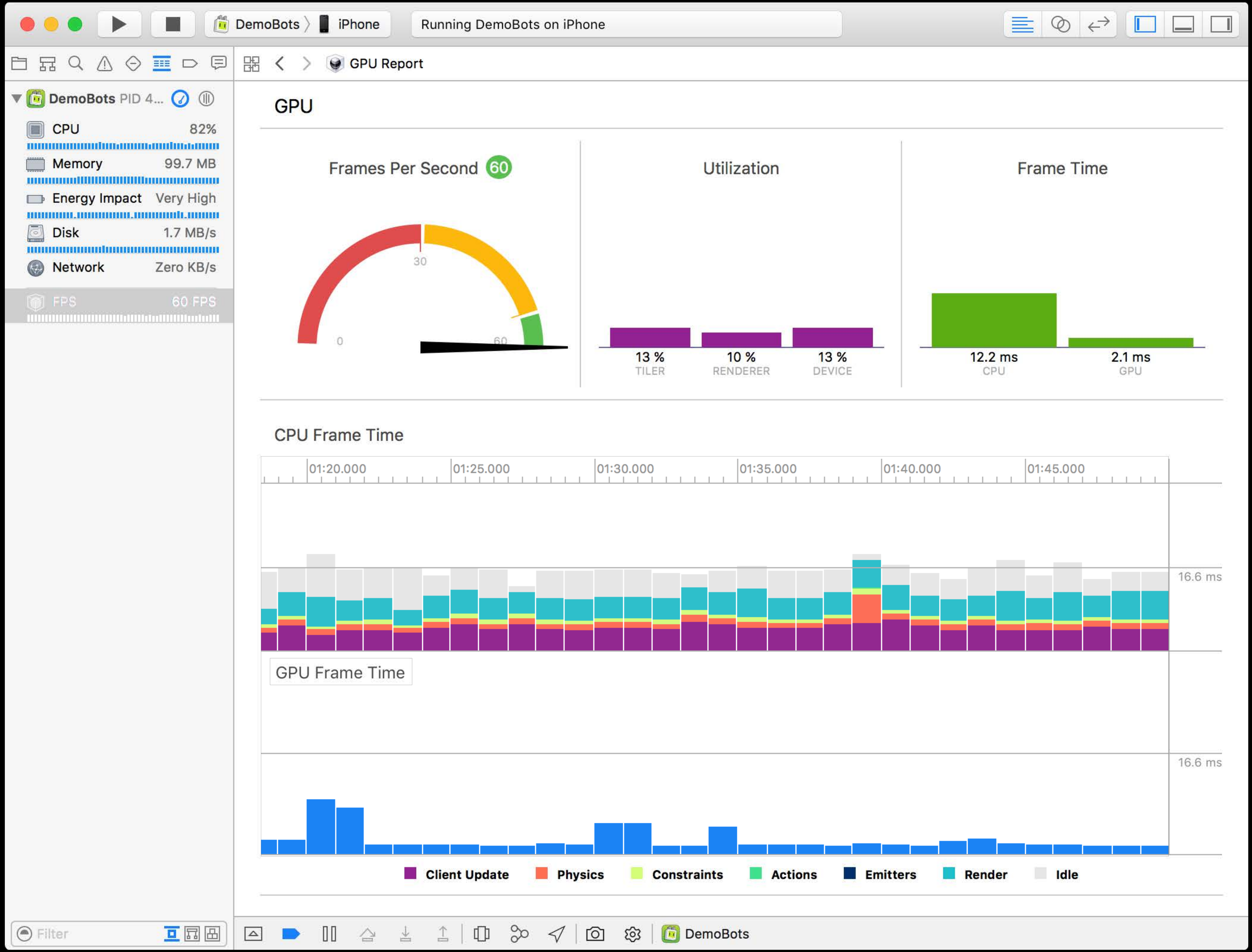
NEW

Breakdown of update loop

- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks

Available on iOS and watchOS



Tile Maps

Tile Maps

NEW

What are tile maps?

Tile maps are a grid of evenly spaced images

Used to build scenes from repeating images

Quickly create large, detailed scenes

Tile Maps

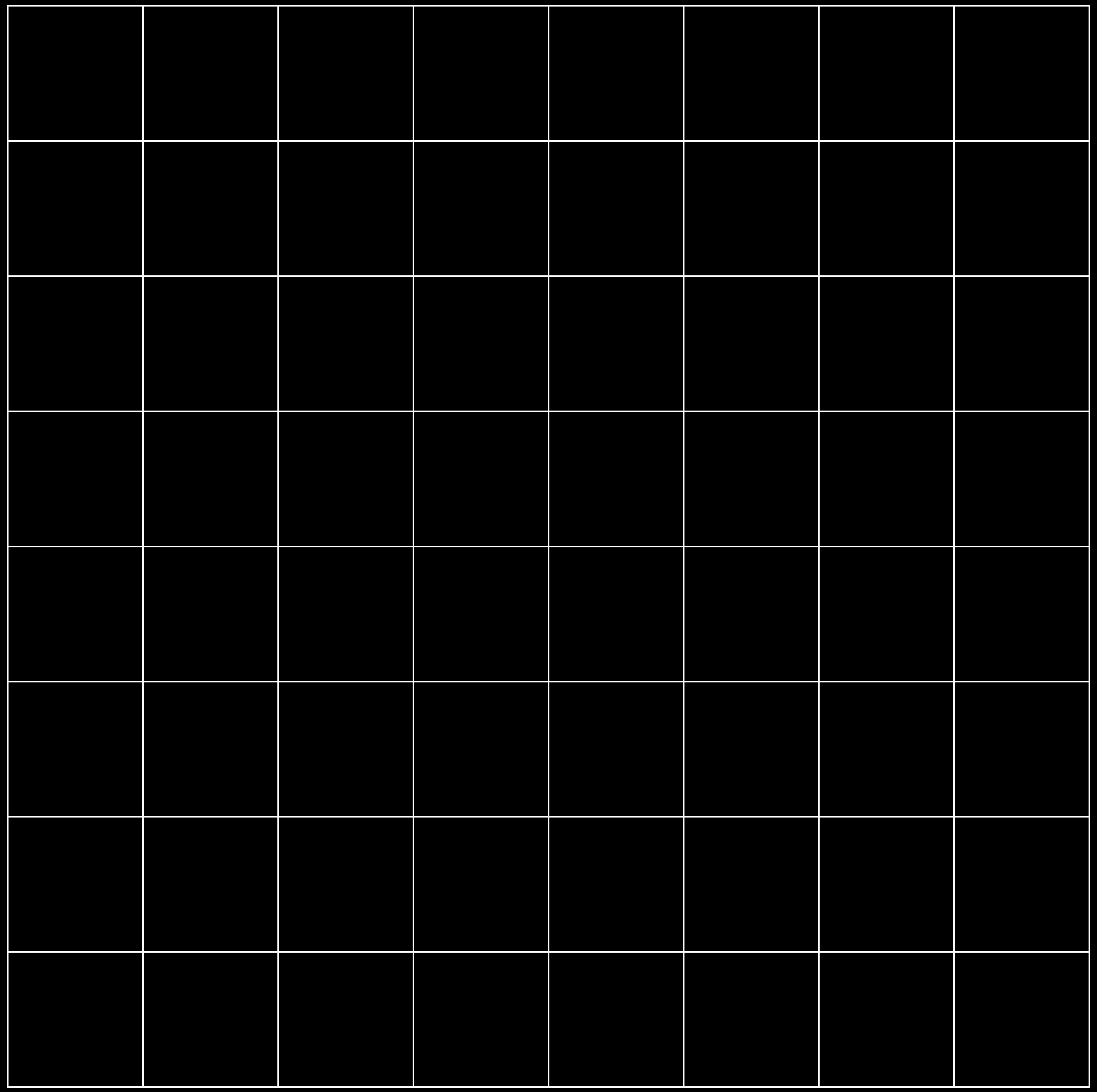
NEW

What are tile maps?

Tile maps are a grid of evenly spaced images

Used to build scenes from repeating images

Quickly create large, detailed scenes



Tile Maps

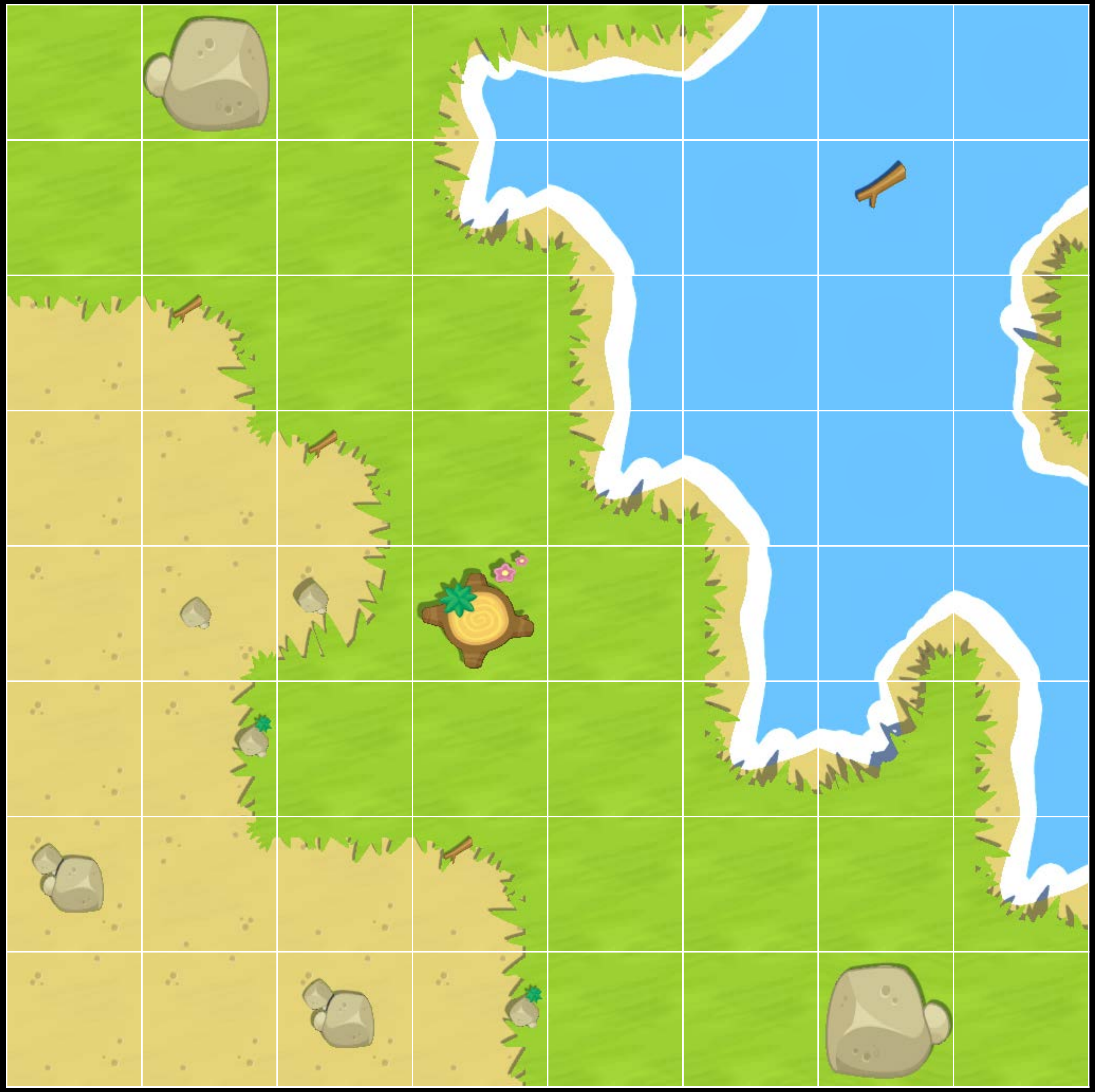
What are tile maps?

NEW

Tile maps are a grid of evenly spaced images

Used to build scenes from repeating images

Quickly create large, detailed scenes



Tile Maps

Why use tile maps?

Could place individual images by hand

Pros

- Small images help keep overhead low
- Can be rearranged

Cons

- Tedious and time consuming
- Clutters the scene with lots of nodes
- Quickly becomes difficult to manage

Tile Maps

Why use tile maps?

Could place individual images by hand

Pros

- Small images help keep overhead low
- Can be rearranged

Cons

- Tedious and time consuming
- Clutters the scene with lots of nodes
- Quickly becomes difficult to manage

Tile Maps

Why use tile maps?

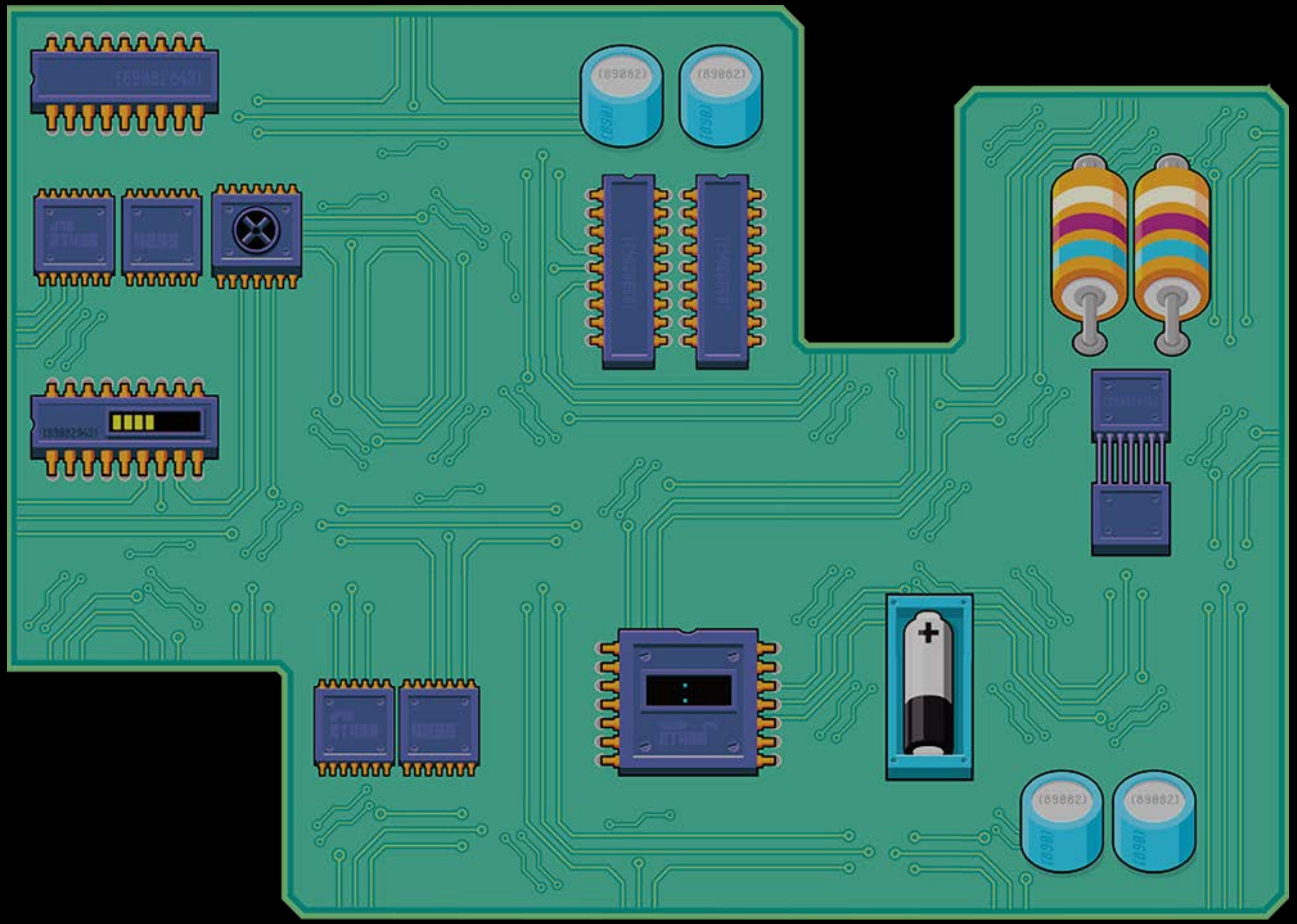
Could use static images for your scenes

Pros

- Easy to place and manage
- Doesn't clutter the scene

Cons

- Tweaks require changing your assets
- Large images require more memory
- Variety requires additional large assets



Tile Maps

NEW

Why use tile maps?

Tile maps get you the best of both solutions

- Easy to manage
- Can be quickly modified
- Large scenes with low overhead

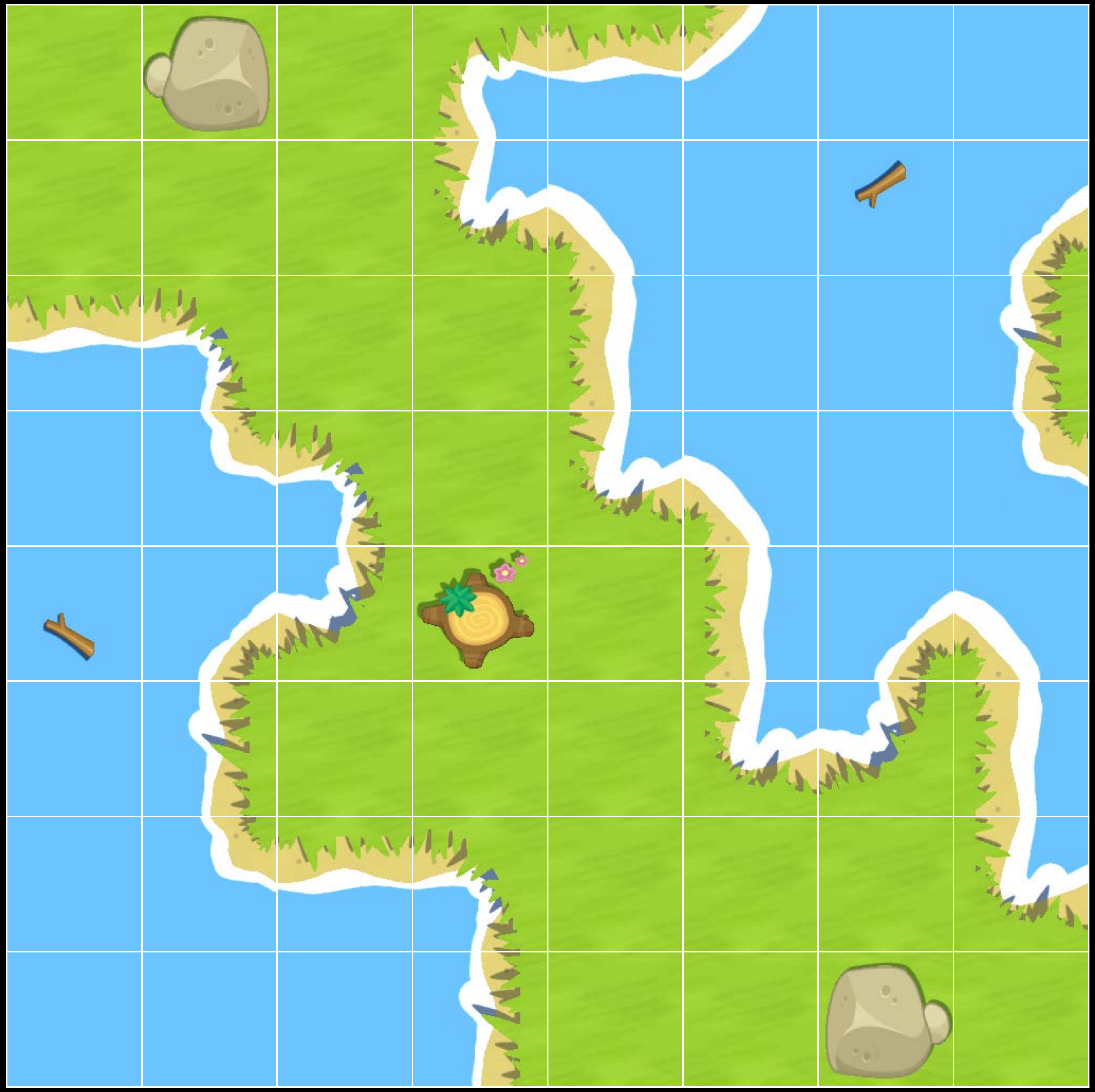
Tile Maps

NEW

Why use tile maps?

Tile maps get you the best of both solutions

- Easy to manage
- Can be quickly modified
- Large scenes with low overhead



Tile Maps

NEW

Why use tile maps?

Great for lots of different games and art styles

Tile Maps

NEW

Why use tile maps?

Great for lots of different games and art styles

- Top-down RPGs

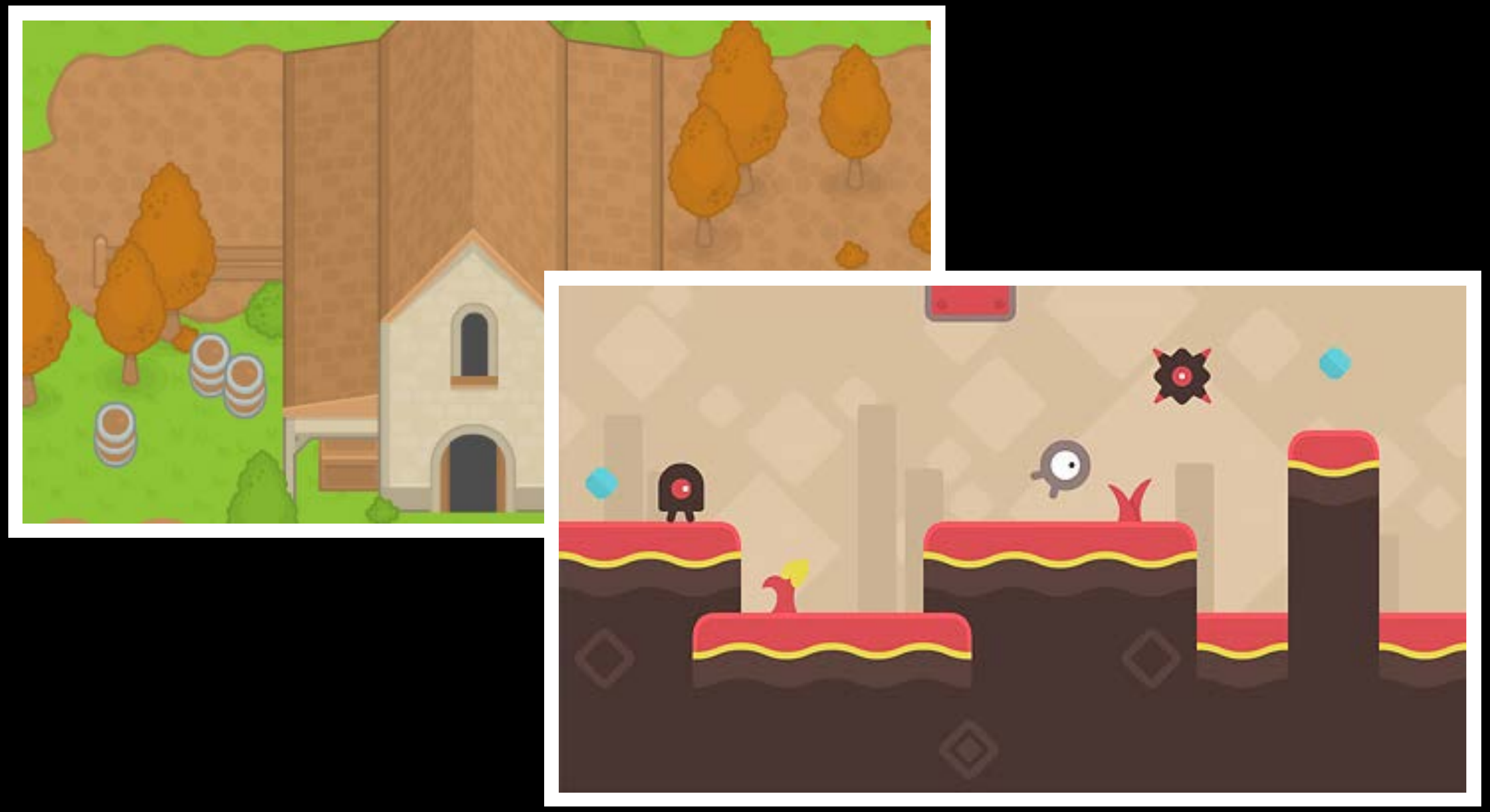
Tile Maps

NEW

Why use tile maps?

Great for lots of different games and art styles

- Top-down RPGs
- Side-scrolling platformers



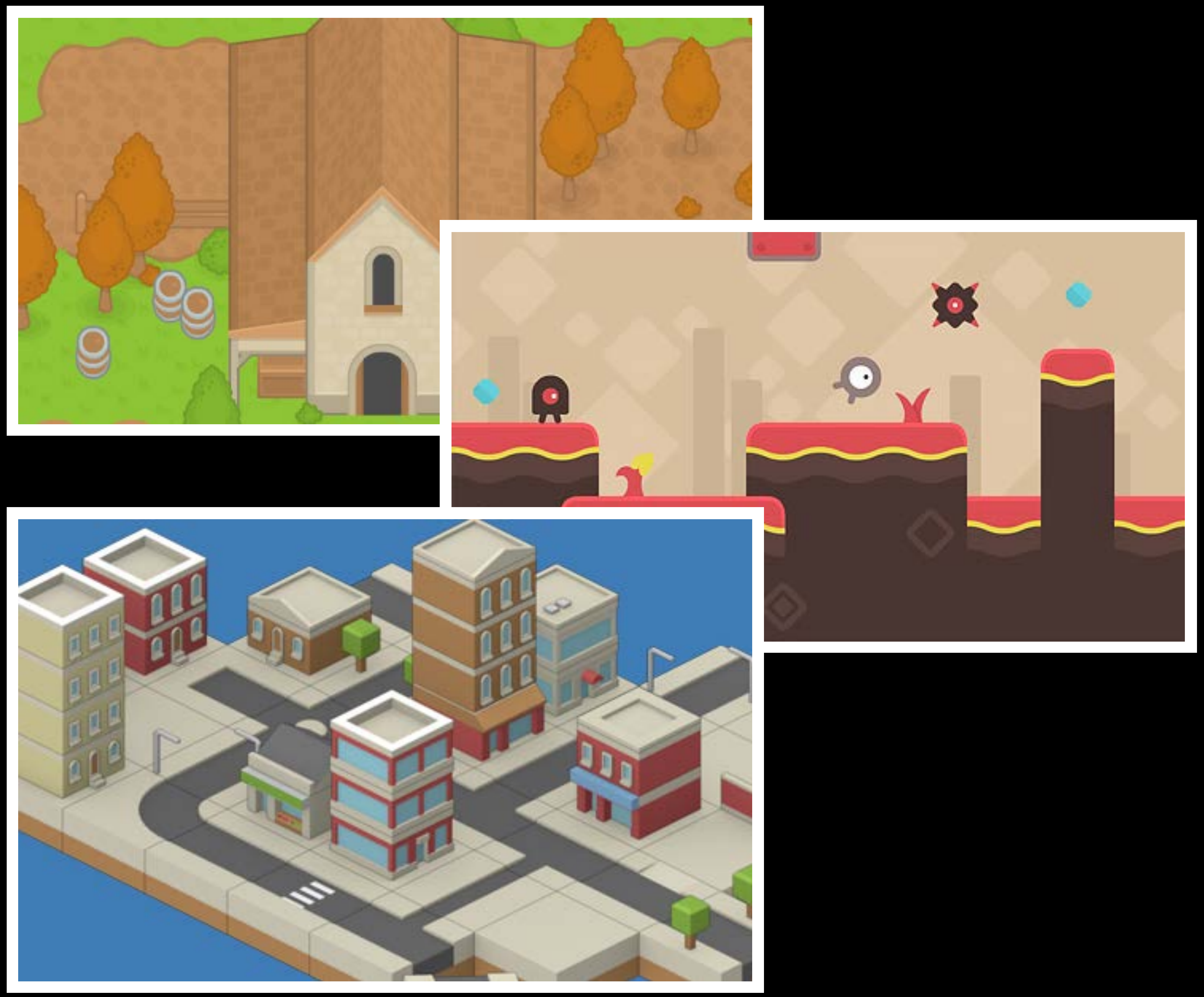
Tile Maps

NEW

Why use tile maps?

Great for lots of different games and art styles

- Top-down RPGs
- Side-scrolling platformers
- Isometric city builders



Tile Maps

NEW

Why use tile maps?

Great for lots of different games and art styles

- Top-down RPGs
- Side-scrolling platformers
- Isometric city builders
- Hex-based board games

Demo

Tile maps in action

Tile Maps

Class overview

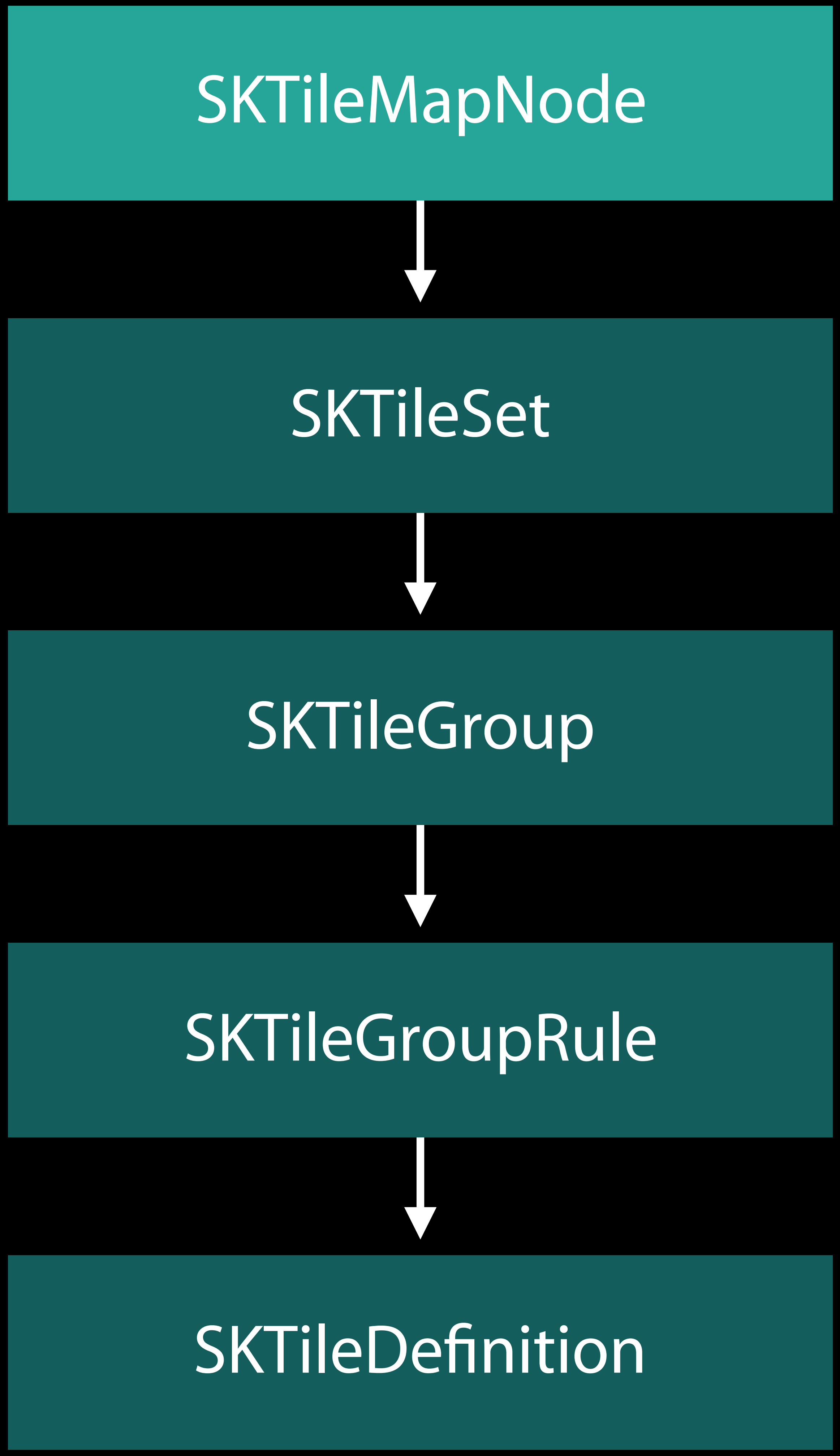
NEW

SKTileMapNode is the tile map

Derived from SKNode

Contains all of the placed tiles

Needs a tile set to be able to place tiles



Tile Maps

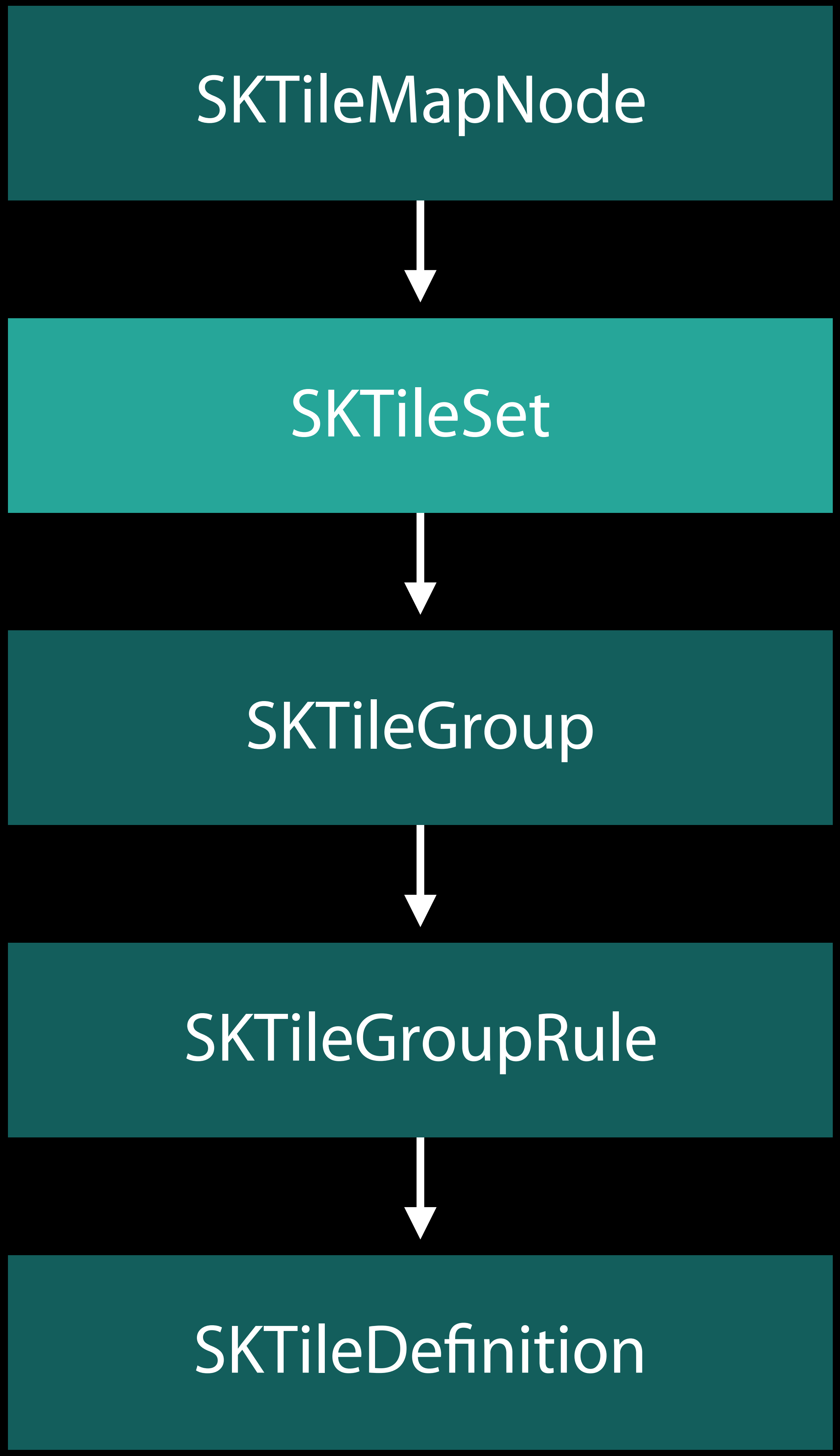
Class overview

NEW

SKTileSet contains all placeable tile groups

Also defines the type of tiles it contains

- Grid
- Isometric
- Hexagonal



Tile Maps

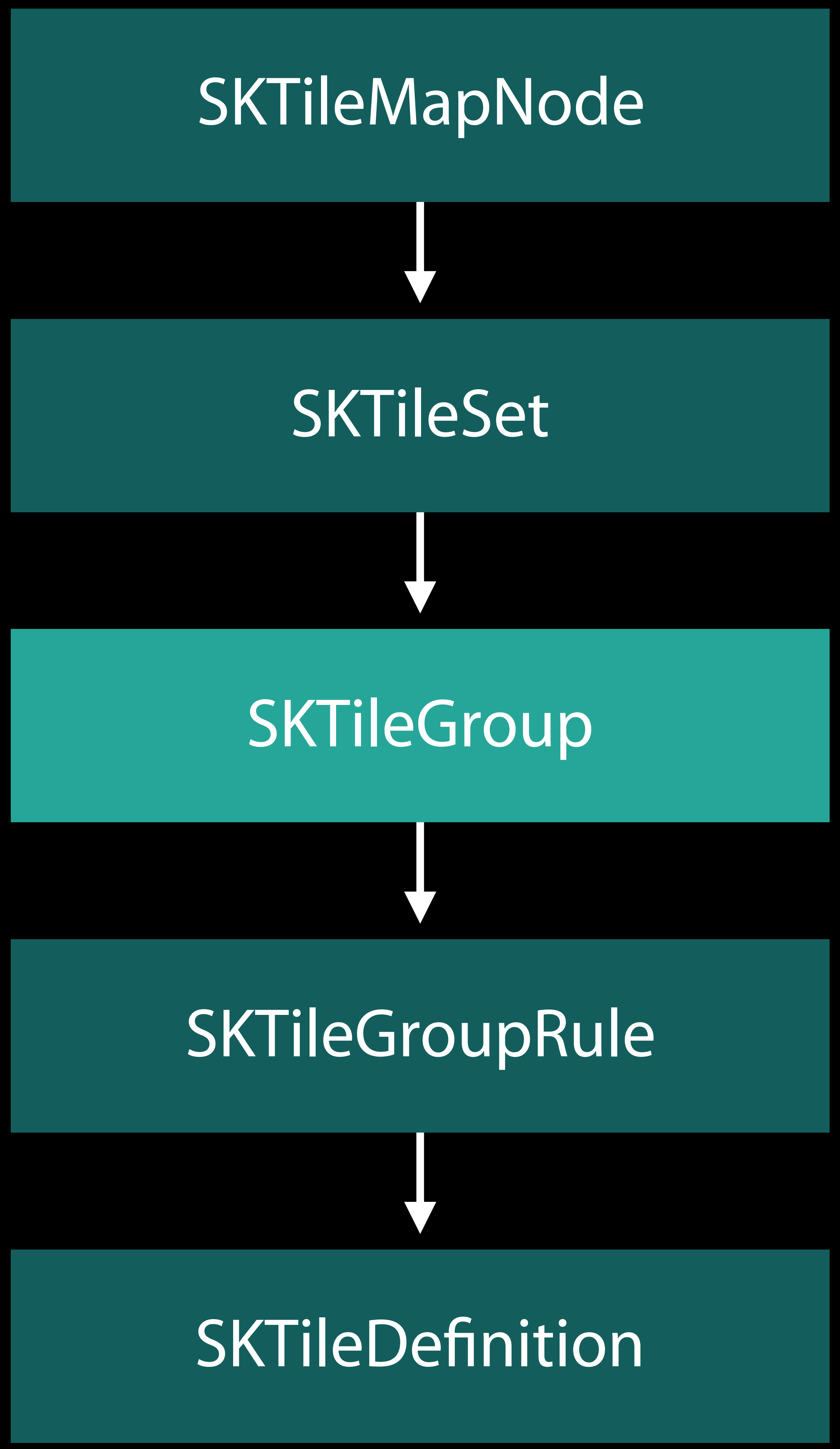
Class overview

NEW

SKTileGroup contains a set of related tiles

- Grass
- Water
- Stone

Has rules that govern tile placement



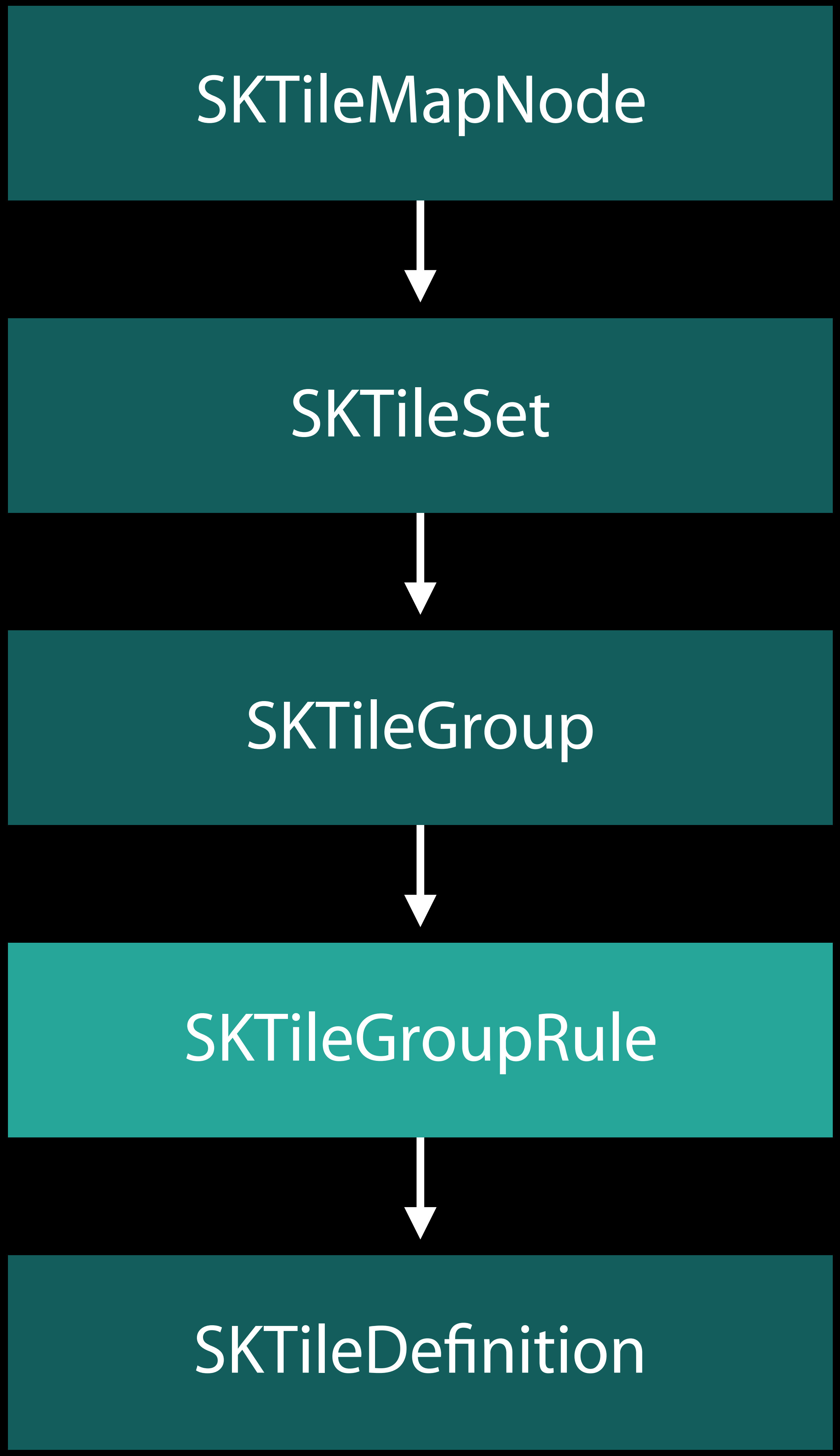
Tile Maps

Class overview

NEW

SKTileGroupRule controls how to interact

Contains tile variants



Tile Maps

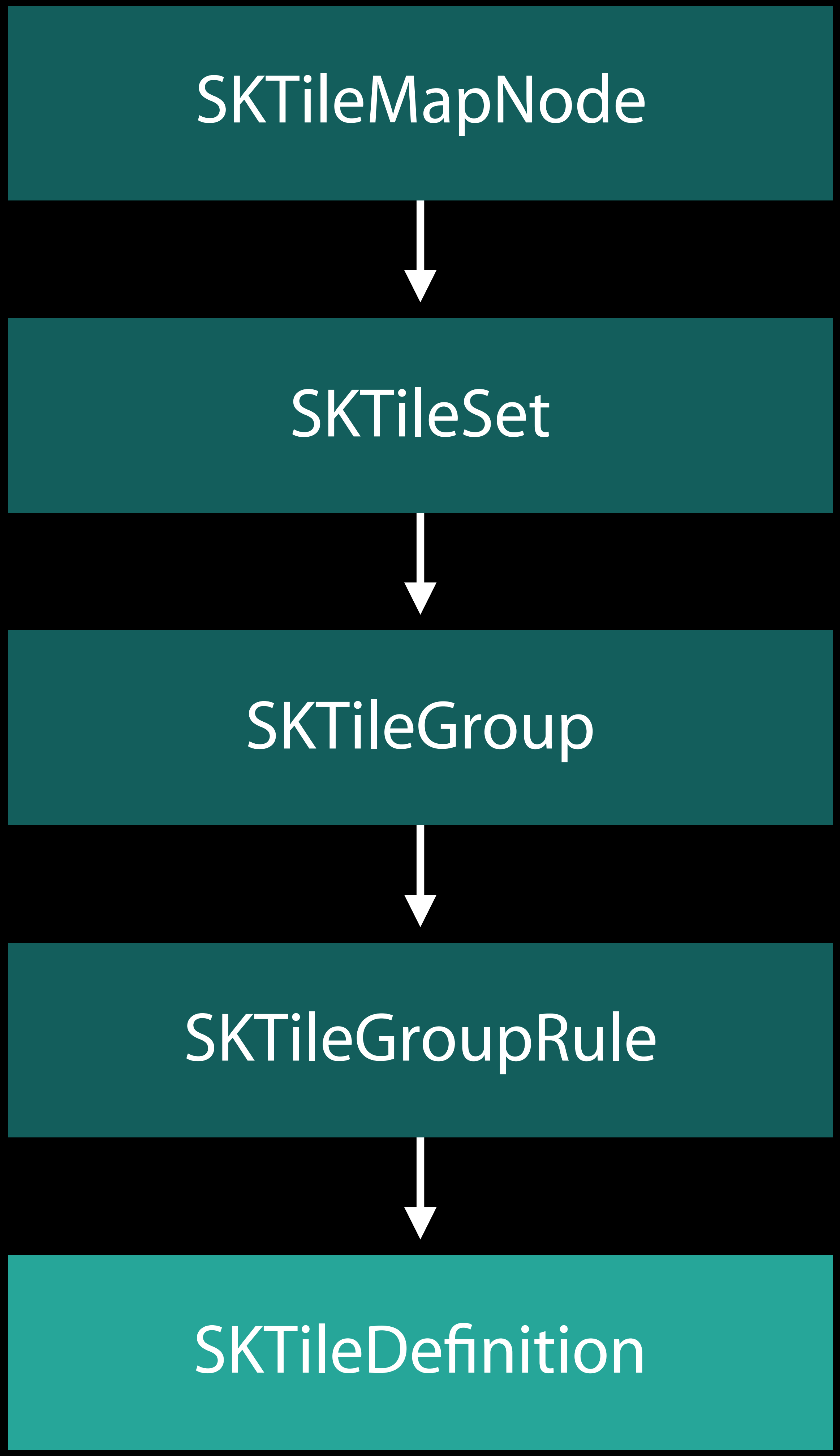
Class overview

NEW

SKTileDefinition defines tile appearance

Allows for animation

Images can be flipped and/or rotated



```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Creating Tile Maps and Setting Tiles

// Get the tile set
guard let tileSet = SKTileSet(named: ”MyTileSet”) else { return }

// Create a tile map
let tileSize = CGSize(width: 32.0, height: 32.0)
let tileMap = SKTileMapNode(tileSet: tileSet, columns: 16, rows: 16, tileSize: tileSize)

// Get a tile group from the tile set
let tileGroup = tileSet.tileGroups.first

// Set tile group for a specific tile
tileMap.setTileGroup(tileGroup, forColumn: 4, row: 7)

// Fill the entire map with a tile group
tileMap.fill(with: tileGroup)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player's sprite

// Convert the player's position into the tile map's frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player's sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: "MyKey")
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

```swift
// Check user data in the tile under the player’s sprite

// Convert the player’s position into the tile map’s frame of reference
let position = tileMap.convert(playerSprite.position, from: playerSprite)

// Get the column and row of the tile that contains the position
let column = tileMap.tileColumnIndex(fromPosition: position)
let row = tileMap.tileRowIndex(fromPosition: position)

// Get the tile definition in the tile the player’s sprite is over
guard let definition = tileMap.tileDefinition(atColumn: column, row: row) else { return }

// Access custom user data on the tile definition
let customUserData = definition.userData?.value(forKey: “MyKey”)
```

Tile Maps

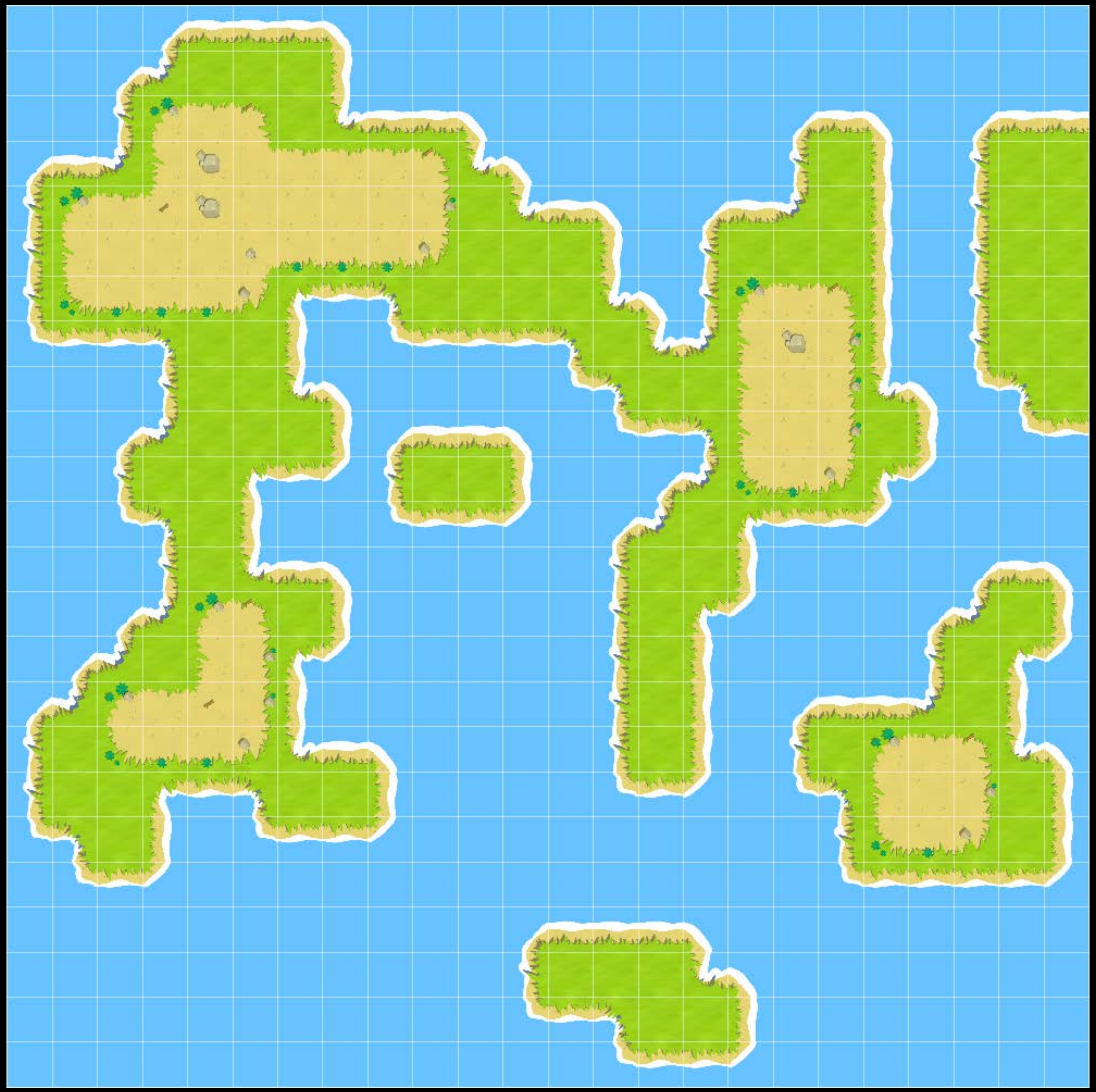
Framework feature recap

NEW

Tile maps get more out of your art budget

- Fewer assets needed
- Reduced memory overhead

Supports animation



Tile Maps

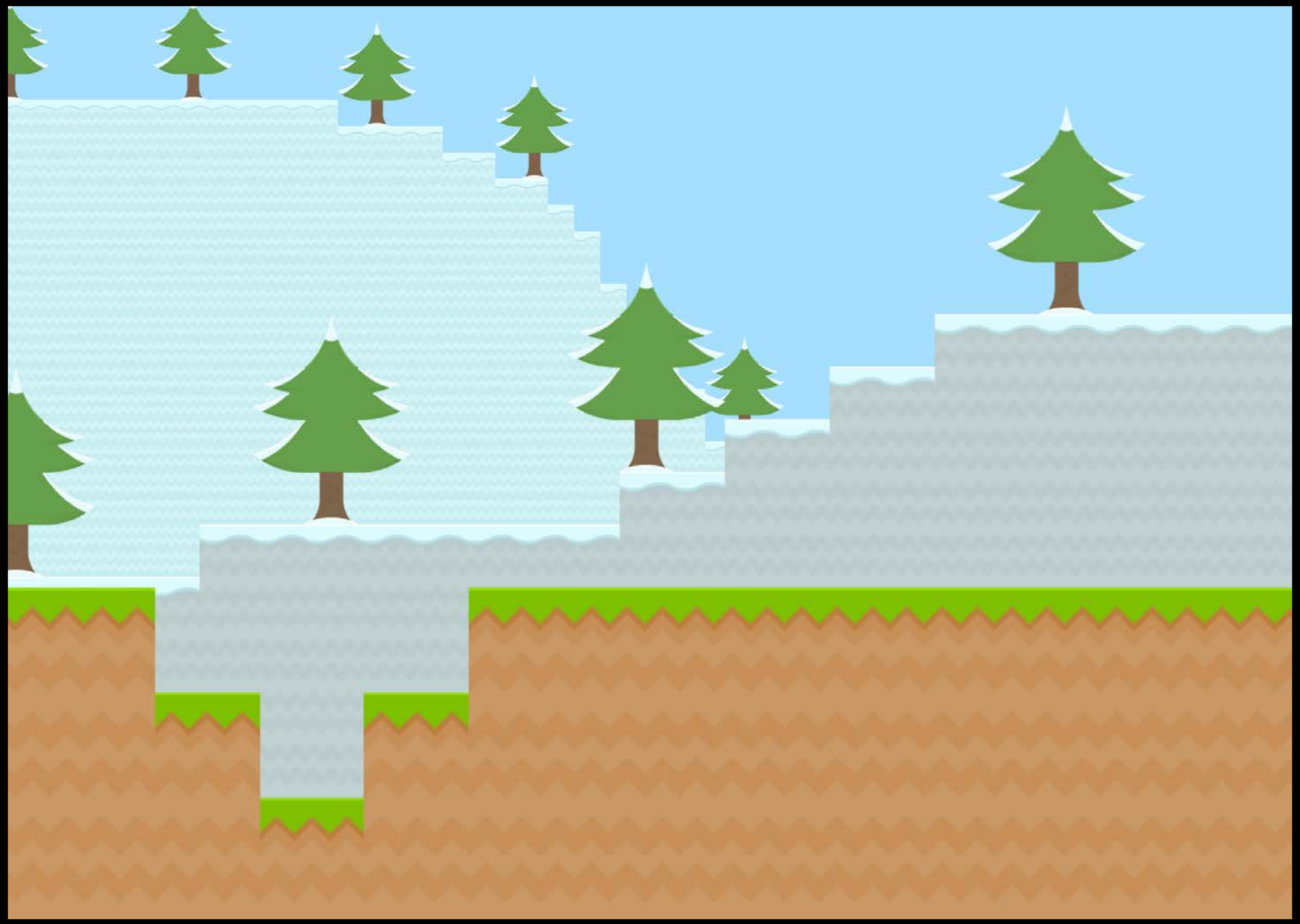
Framework feature recap

NEW

Designed to be layered

- Increased asset versatility
- Enables effects

Great for different art styles and games



Tile Maps

Framework feature recap

NEW

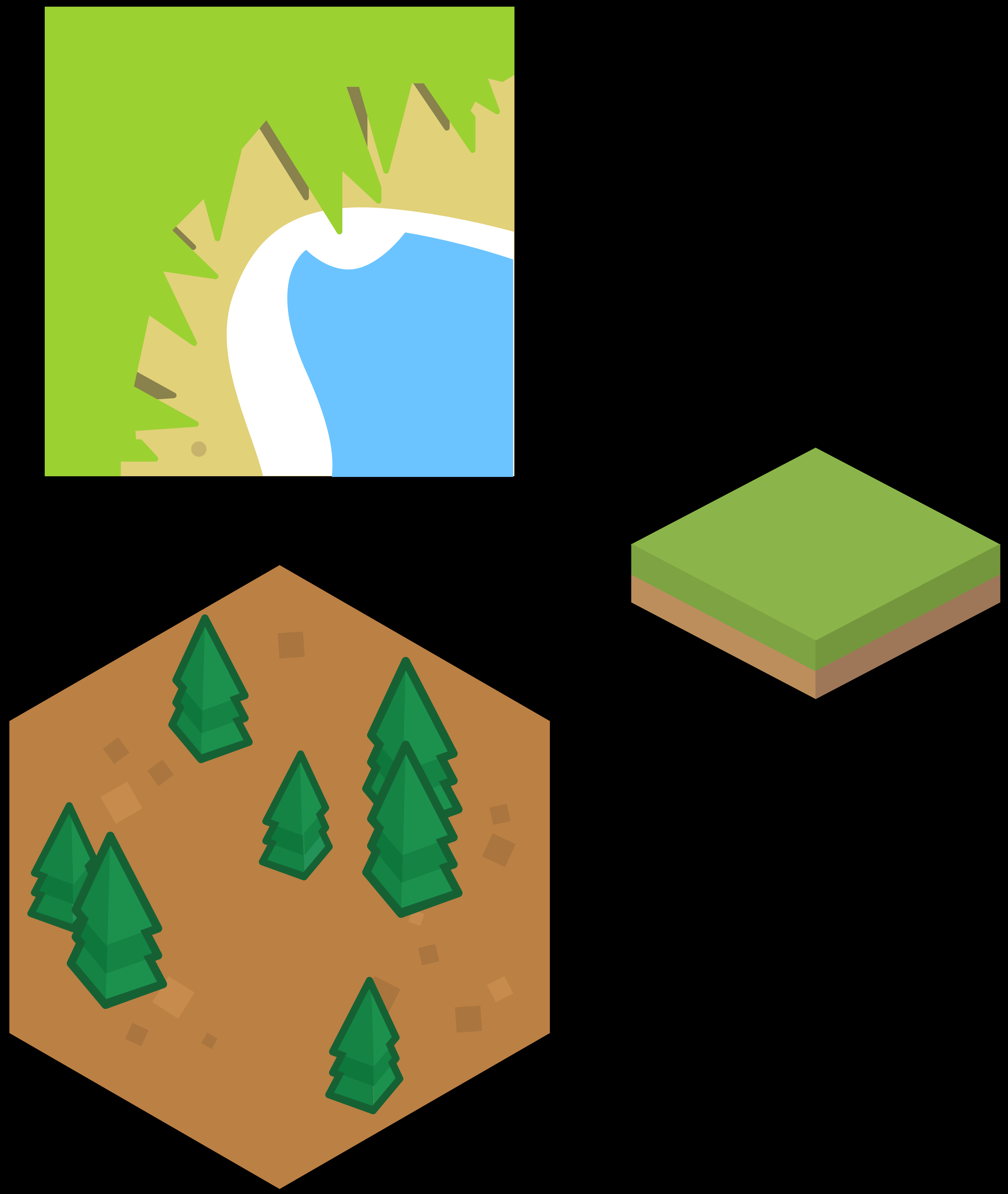
Automatic subdivision

- Only visible chunks are drawn

Batch rendering

Multiple tile types

- Grid
- Isometric
- Hexagonal



Tile Maps

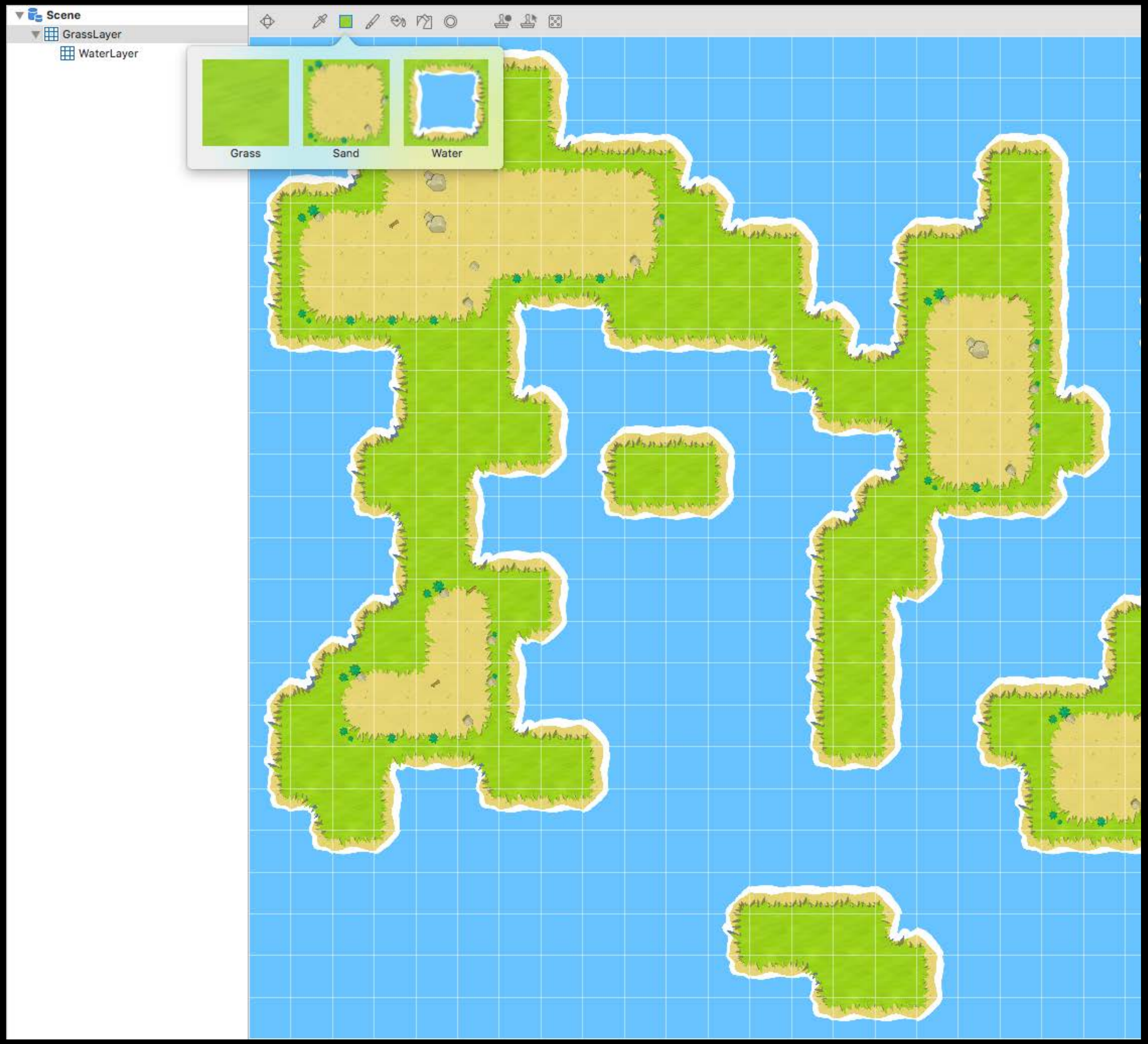
Editor feature recap

NEW

Editing tile maps is simple and easy

Automapping does the hard work for you

Create new tile sets visually



Warp Transformation

Clément Boissière Games Technologies Engineer

Warp Transformation

Introduction

Warp Transformation

Introduction

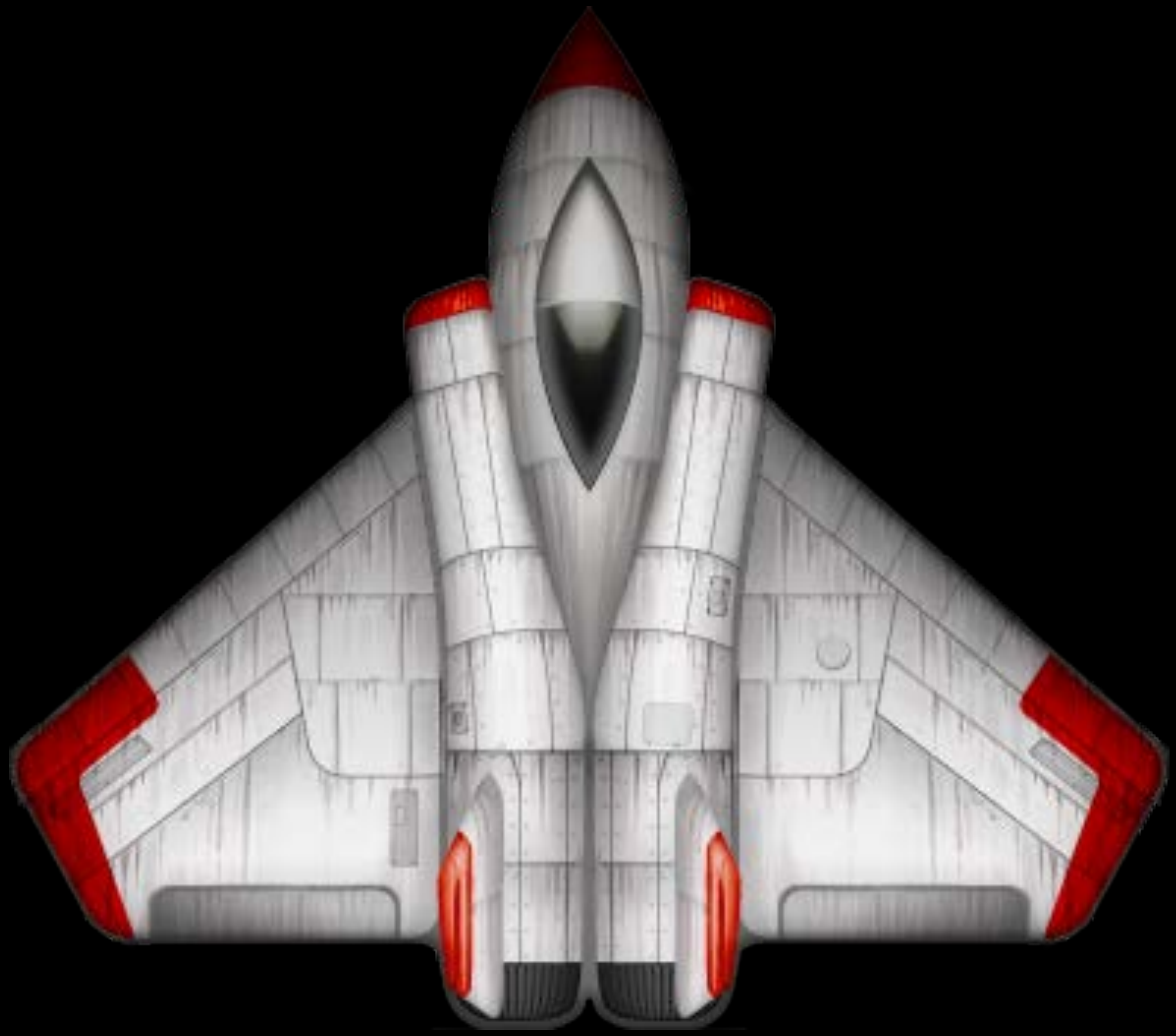
Available transforms in SpriteKit

Warp Transformation

Introduction

Available transforms in SpriteKit

- Scale

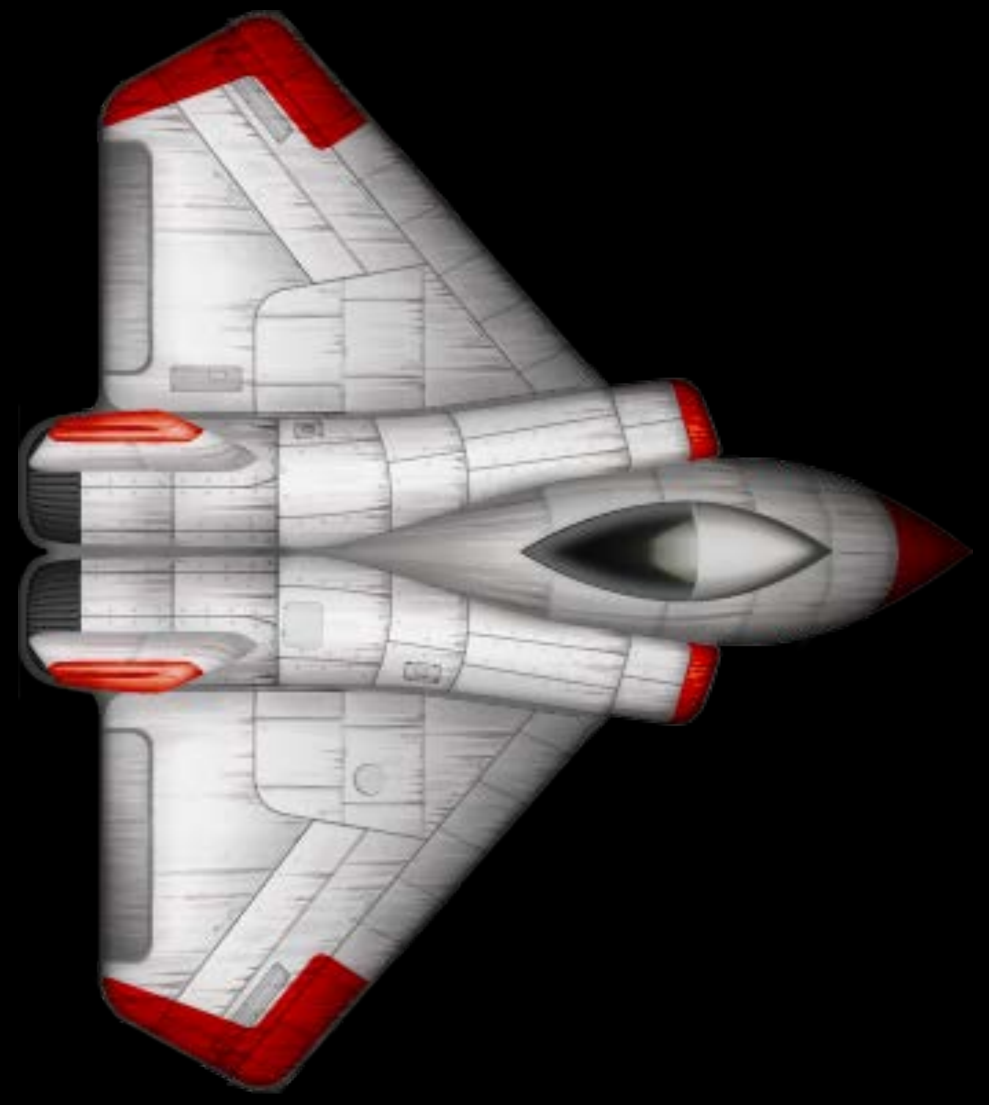


Warp Transformation

Introduction

Available transforms in SpriteKit

- Scale
- Rotation



Warp Transformation

Introduction

Available transforms in SpriteKit

- Scale
- Rotation
- Custom shader

Warp Transformation

NEW

Introduction

SKWarpGeometry

- Two grids of points defining the distortion
- Source positions
- Destination positions

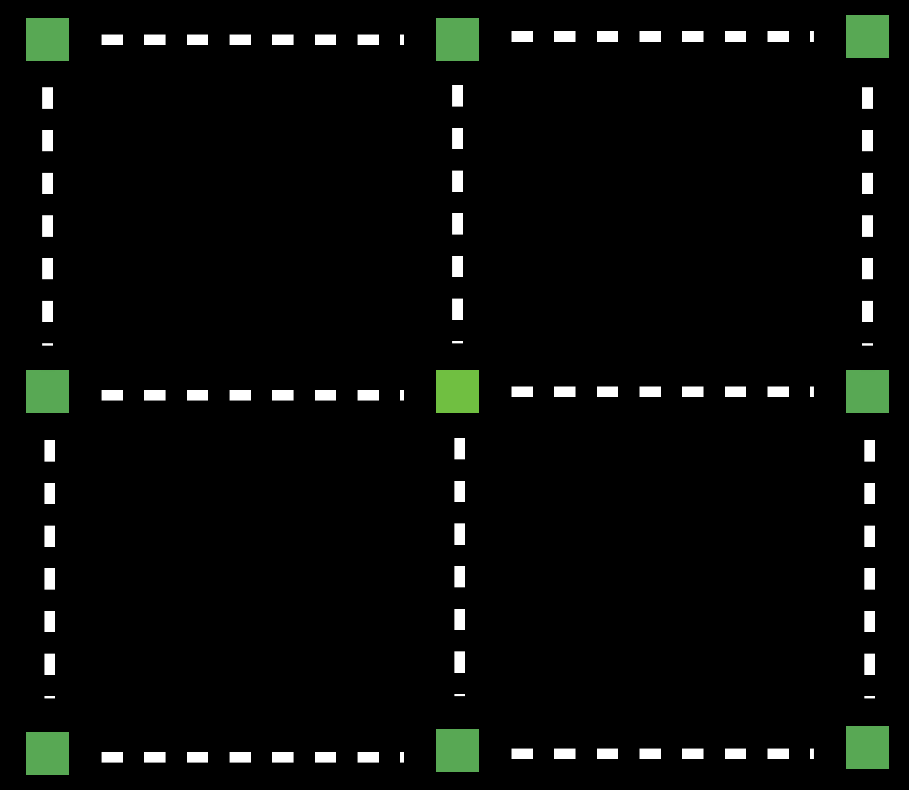
Warp Transformation

Introduction

NEW

SKWarpGeometry

- Two grids of points defining the distortion
- Source positions
- Destination positions



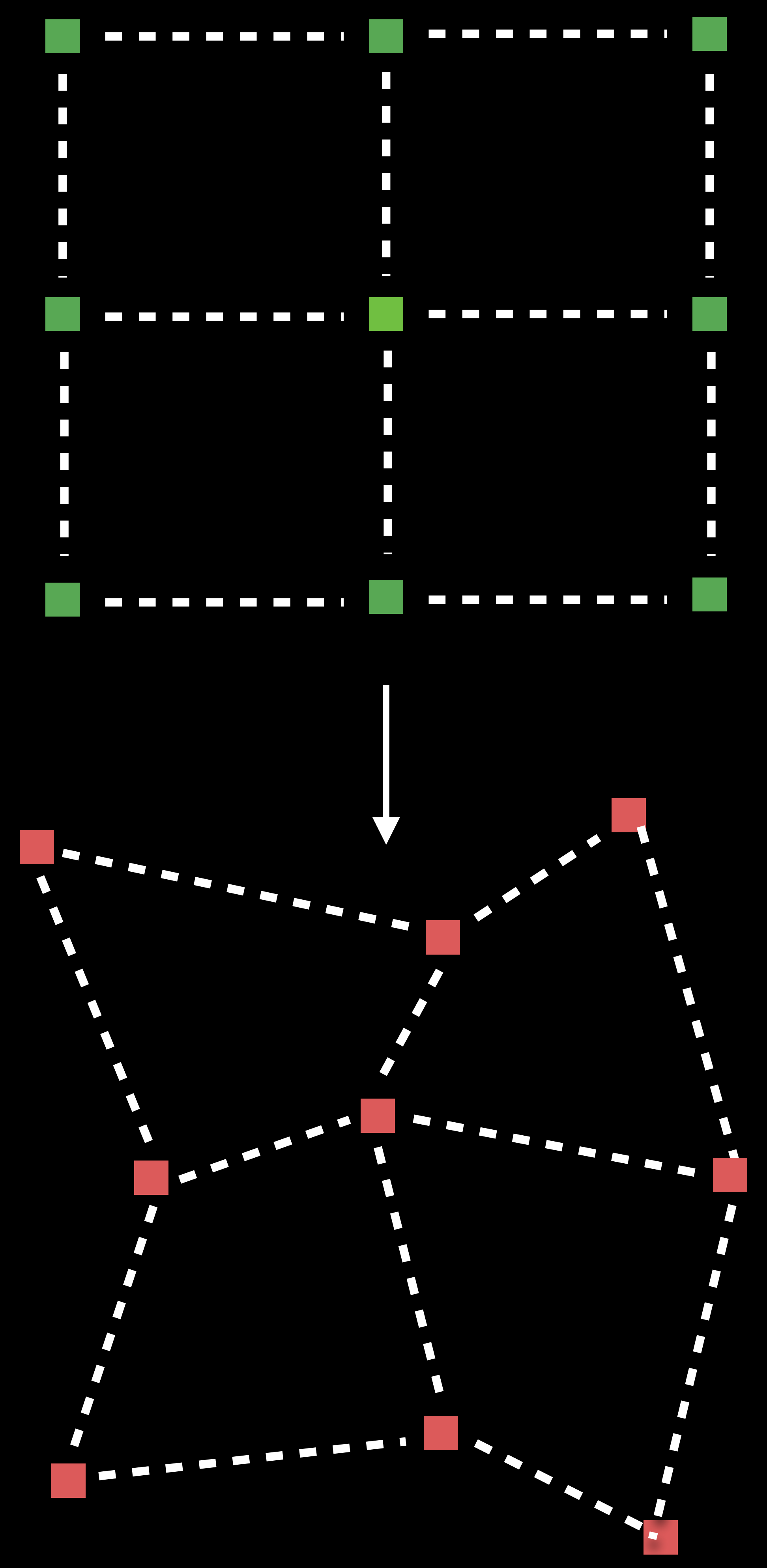
Warp Transformation

Introduction

NEW

SKWarpGeometry

- Two grids of points defining the distortion
- Source positions
- Destination positions



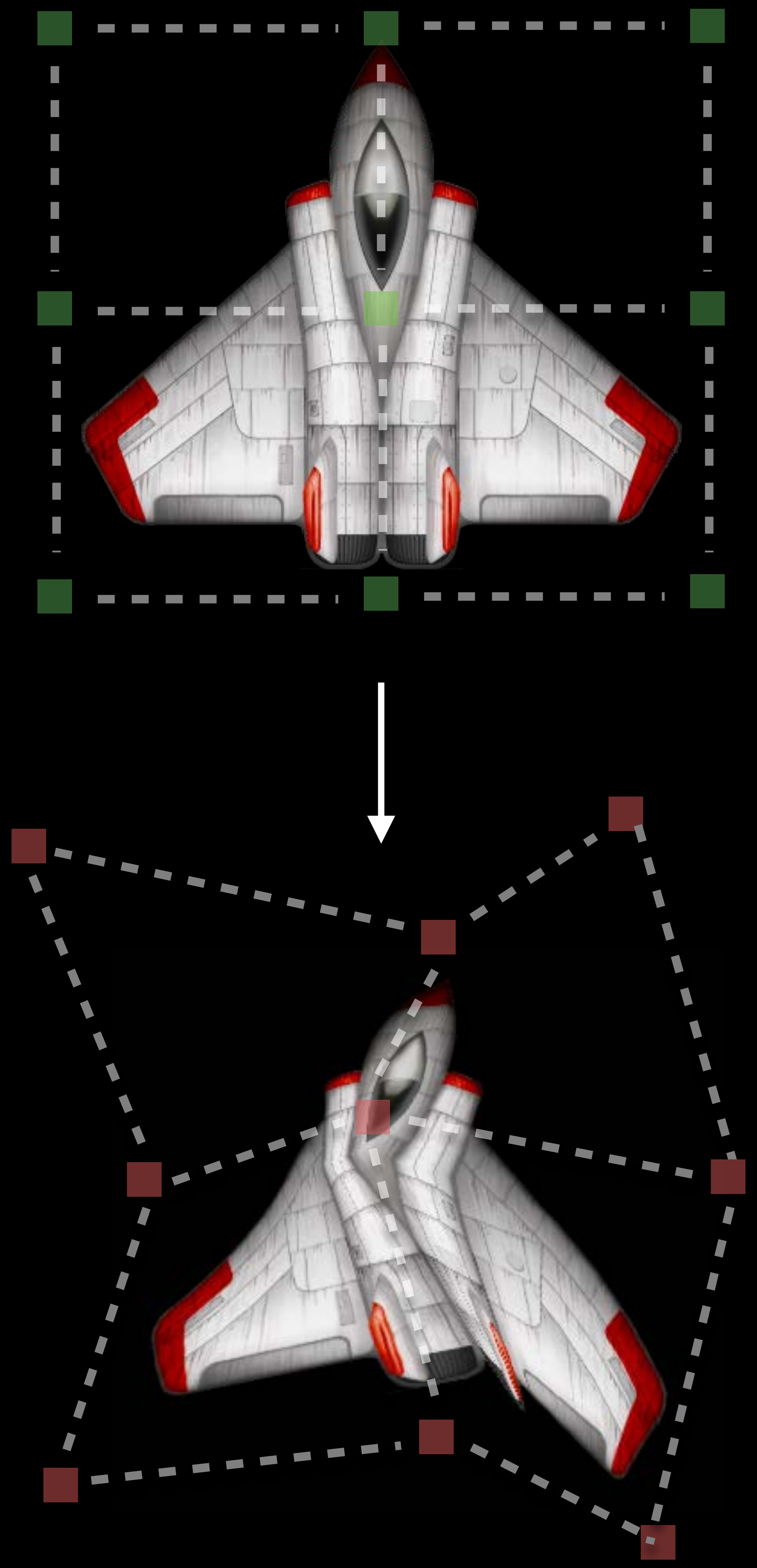
Warp Transformation

Introduction

NEW

SKWarpGeometry

- Two grids of points defining the distortion
- Source positions
- Destination positions



Warp Transformation

Examples

NEW

Warp Transformation

Examples

NEW

A few examples

Warp Transformation

Examples

NEW

A few examples

- Squash

Warp Transformation

Examples

NEW

A few examples

- Squash
- Stretch

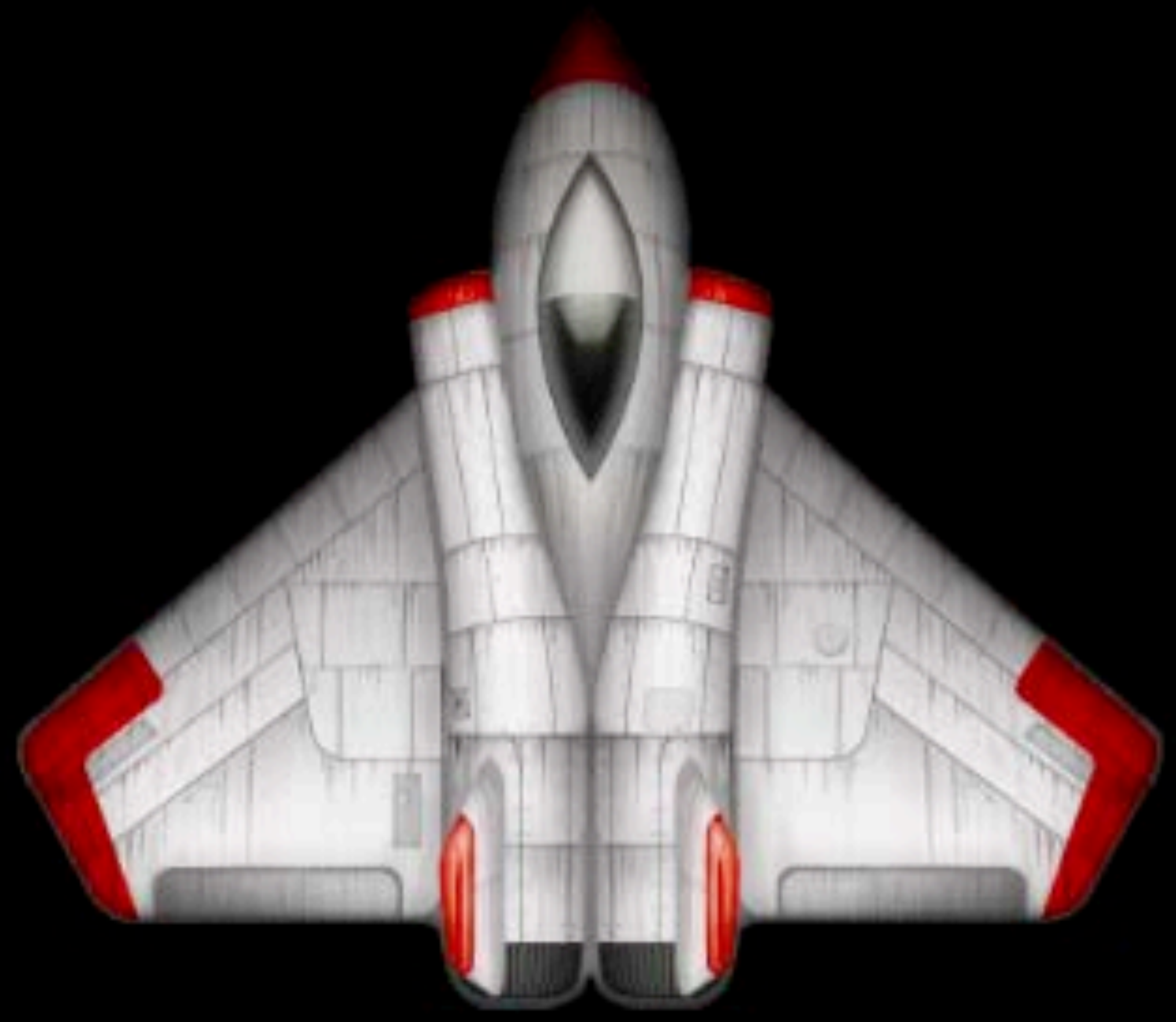
Warp Transformation

Examples

NEW

A few examples

- Squash
- Stretch
- Keyframe-based animations



Warp Transformation

How it works

NEW

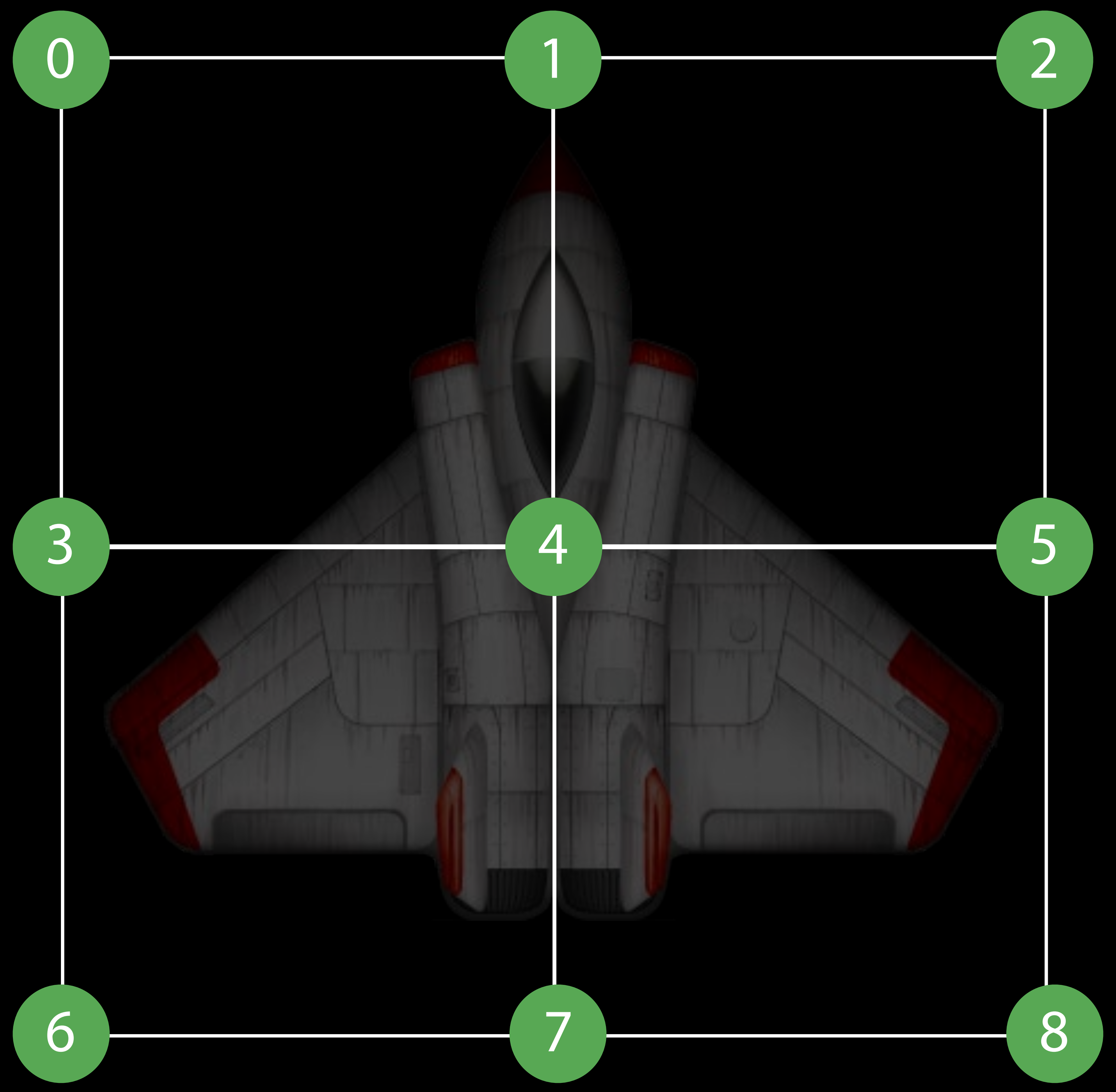
Warp Transformation

How it works

NEW

Concept

- A grid is an indexed set of points



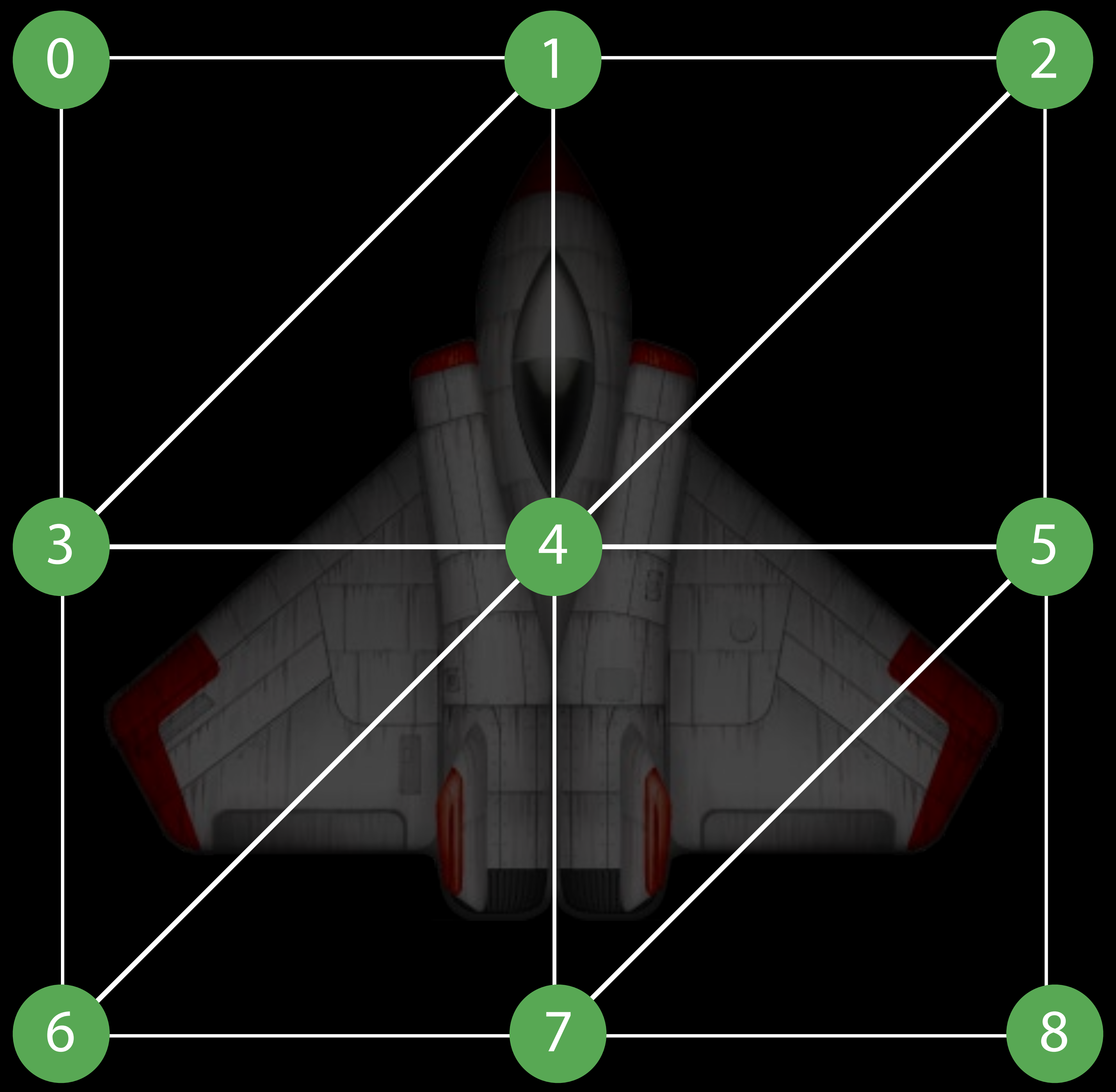
Warp Transformation

How it works

NEW

Concept

- A grid is an indexed set of points
- Each cell is a quad



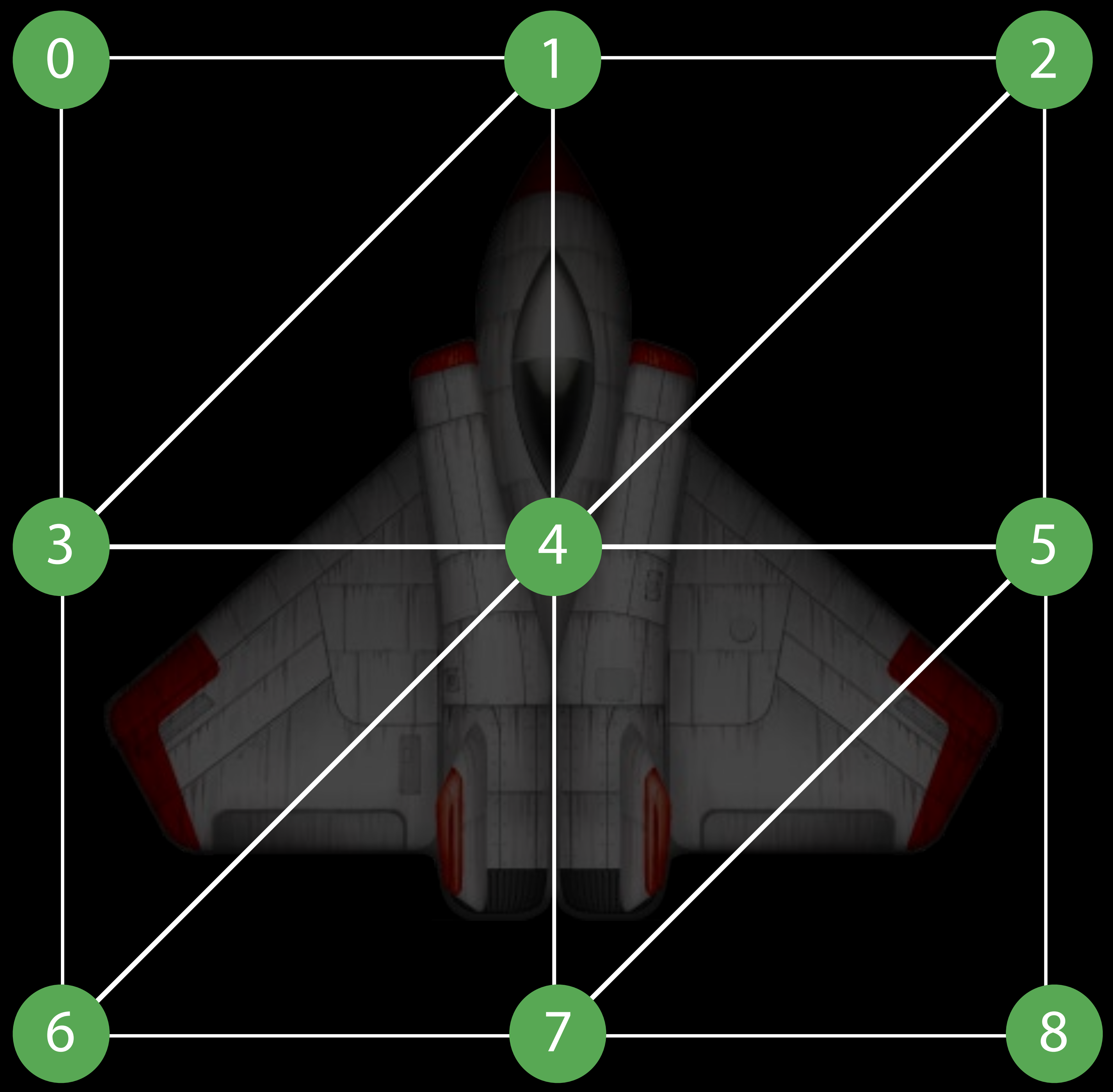
Warp Transformation

How it works

NEW

Concept

- A grid is an indexed set of points
- Each cell is a quad
- Vertices change to create distortion



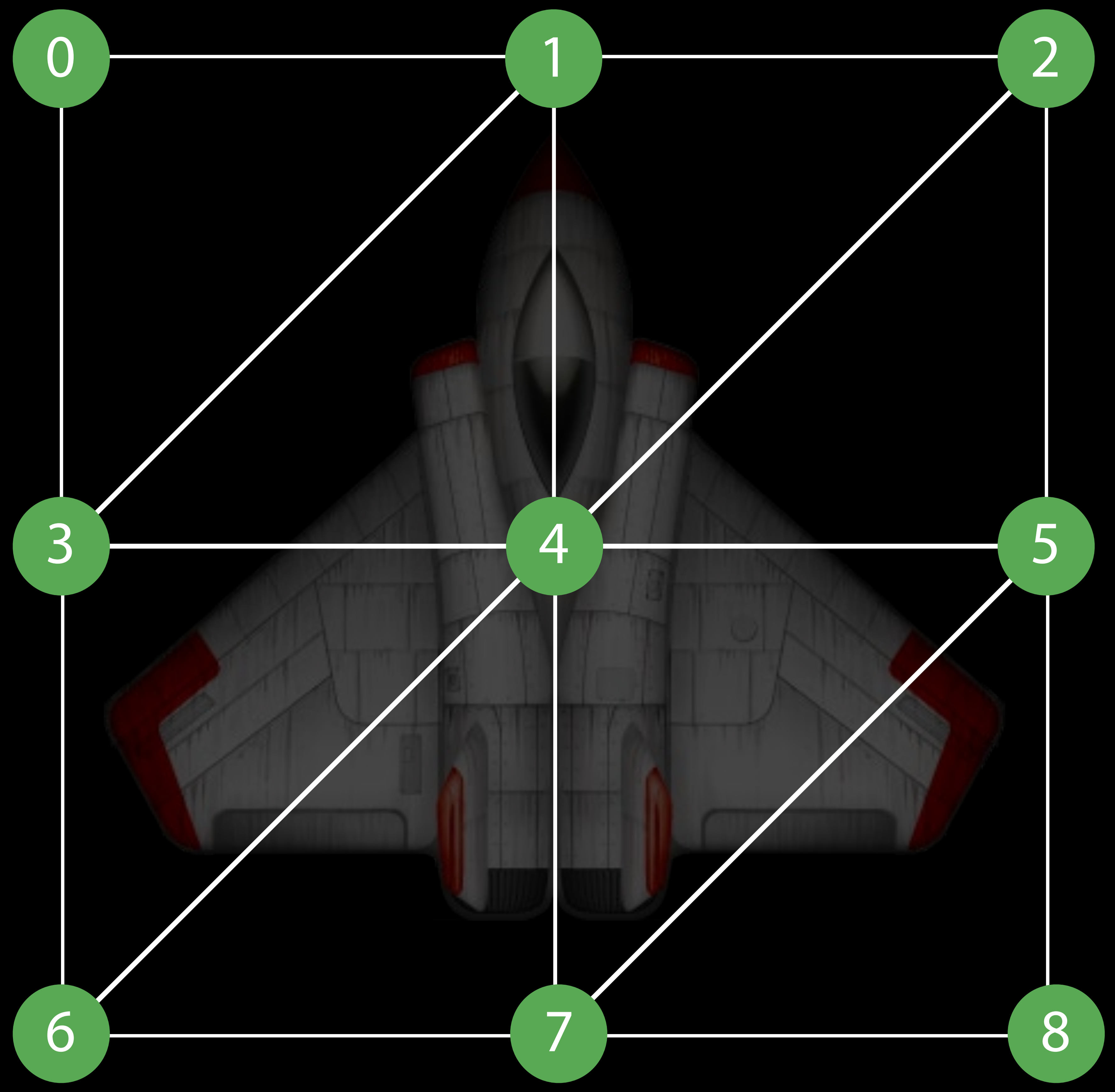
Warp Transformation

How it works

NEW

Concept

- A grid is an indexed set of points
- Each cell is a quad
- Vertices change to create distortion
- Keep the same texture coordinates



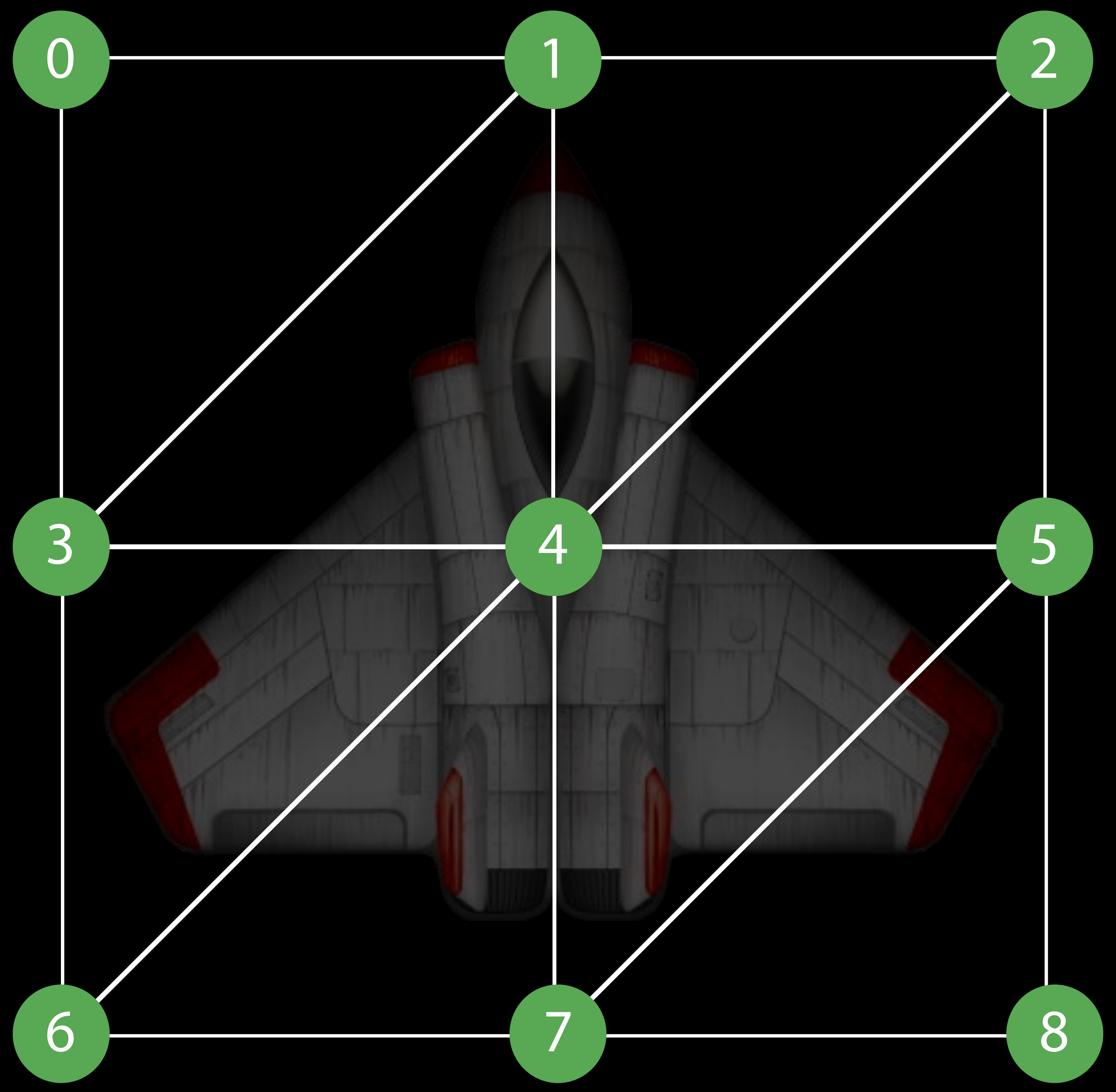
Warp Transformation

How it works

NEW

Concept

- A grid is an indexed set of points
- Each cell is a quad
- Vertices change to create distortion
- Keep the same texture coordinates
- GPU interpolation



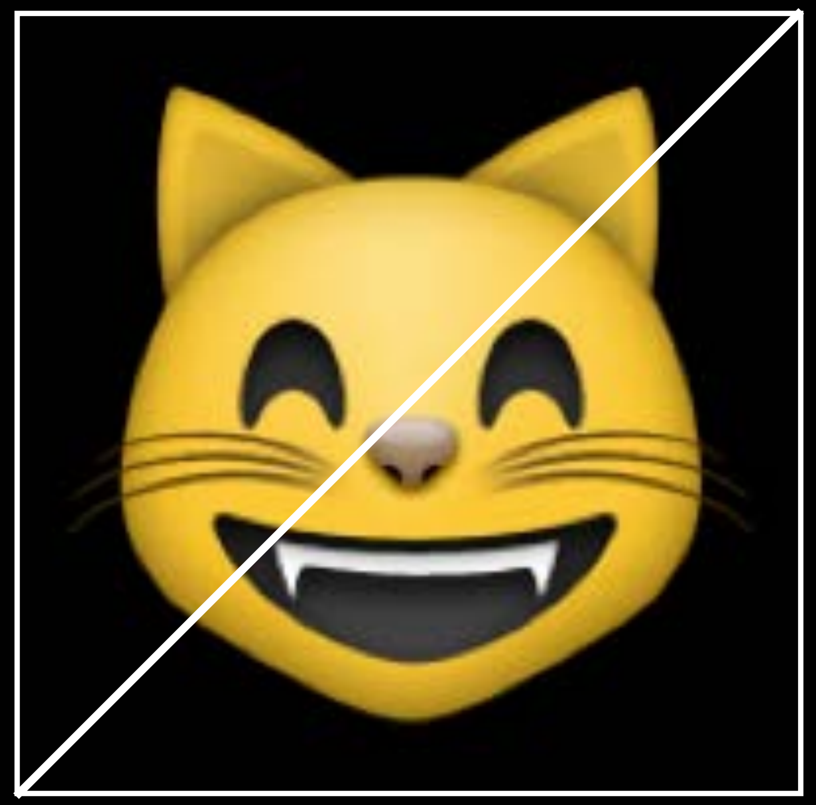
Warp Transformation

How it works

NEW

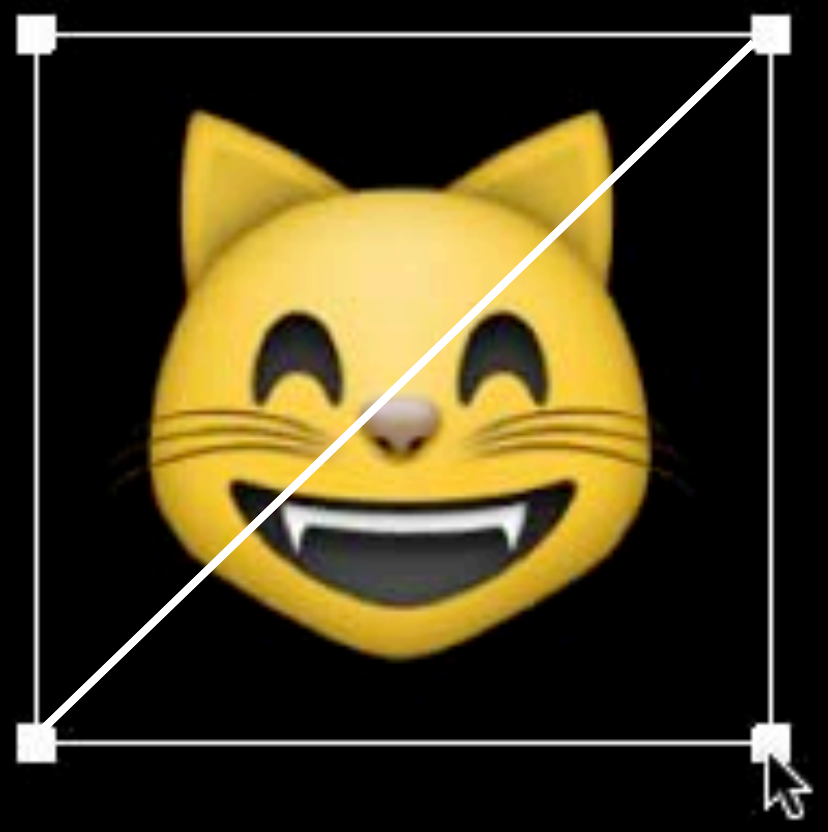
Concept

- A grid is an indexed set of points
- Each cell is a quad
- Vertices change to create distortion
- Keep the same texture coordinates
- GPU interpolation



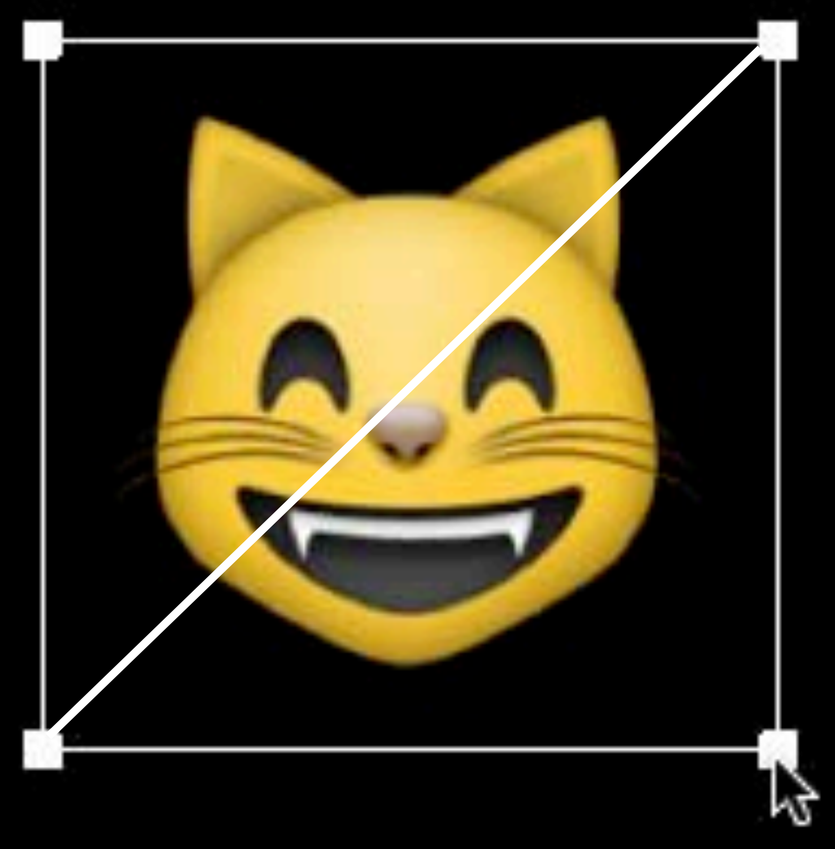
Warp Transformation

NEW



Warp Transformation

NEW



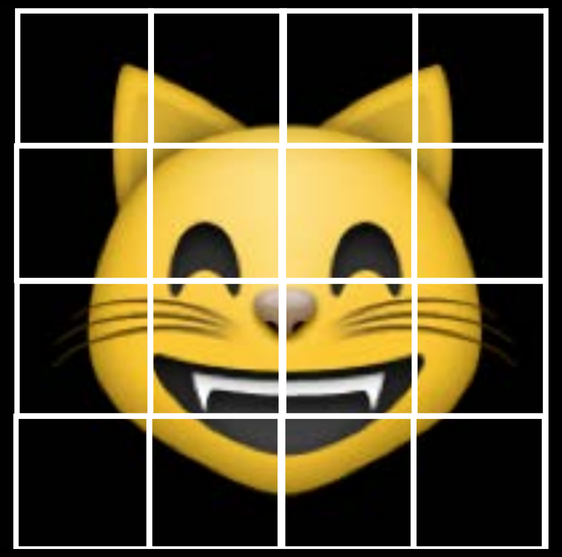
Warp Transformation

How it works

NEW

Higher level of details

- More cells?



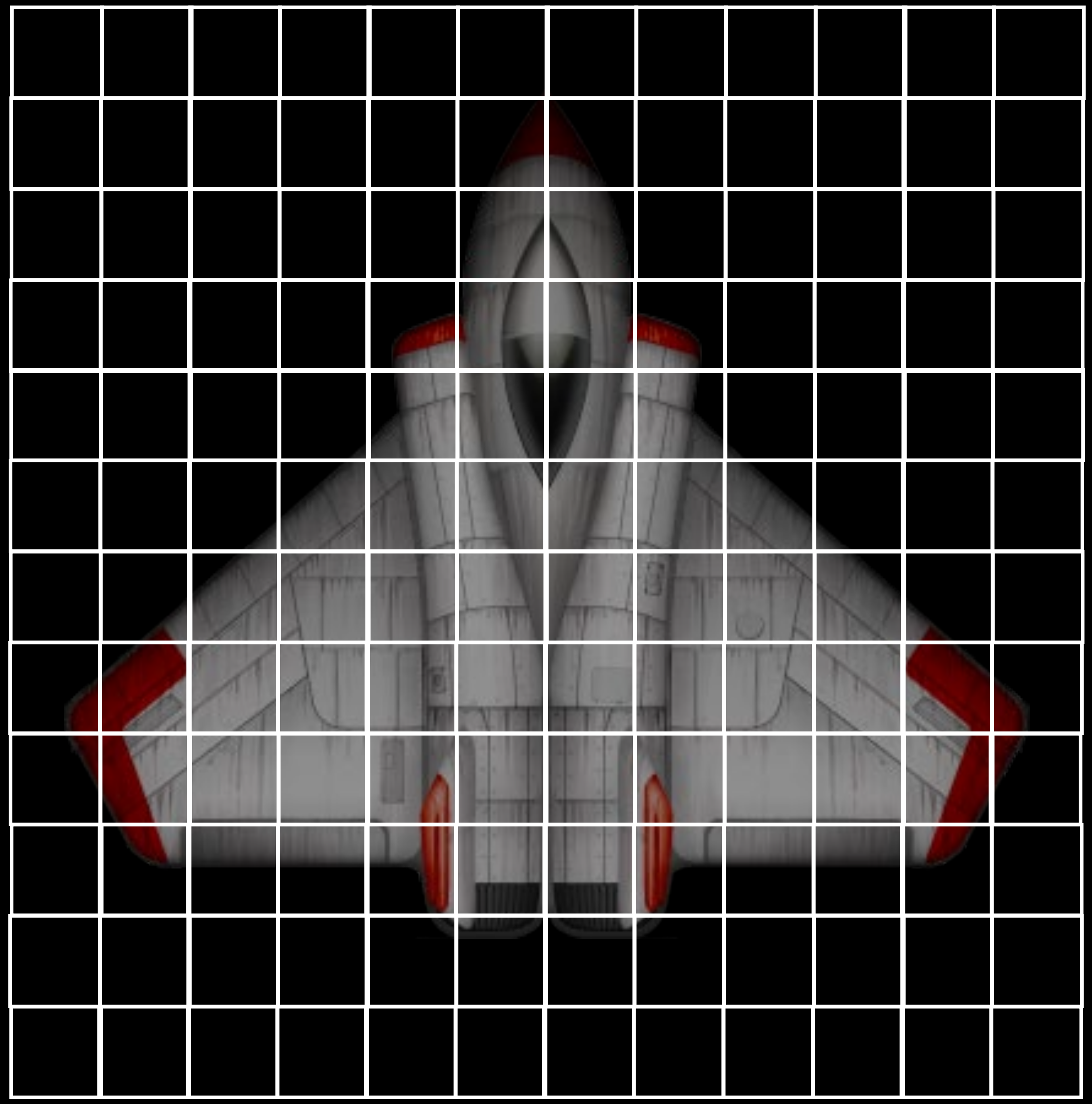
Warp Transformation

How it works

NEW

Higher level of details

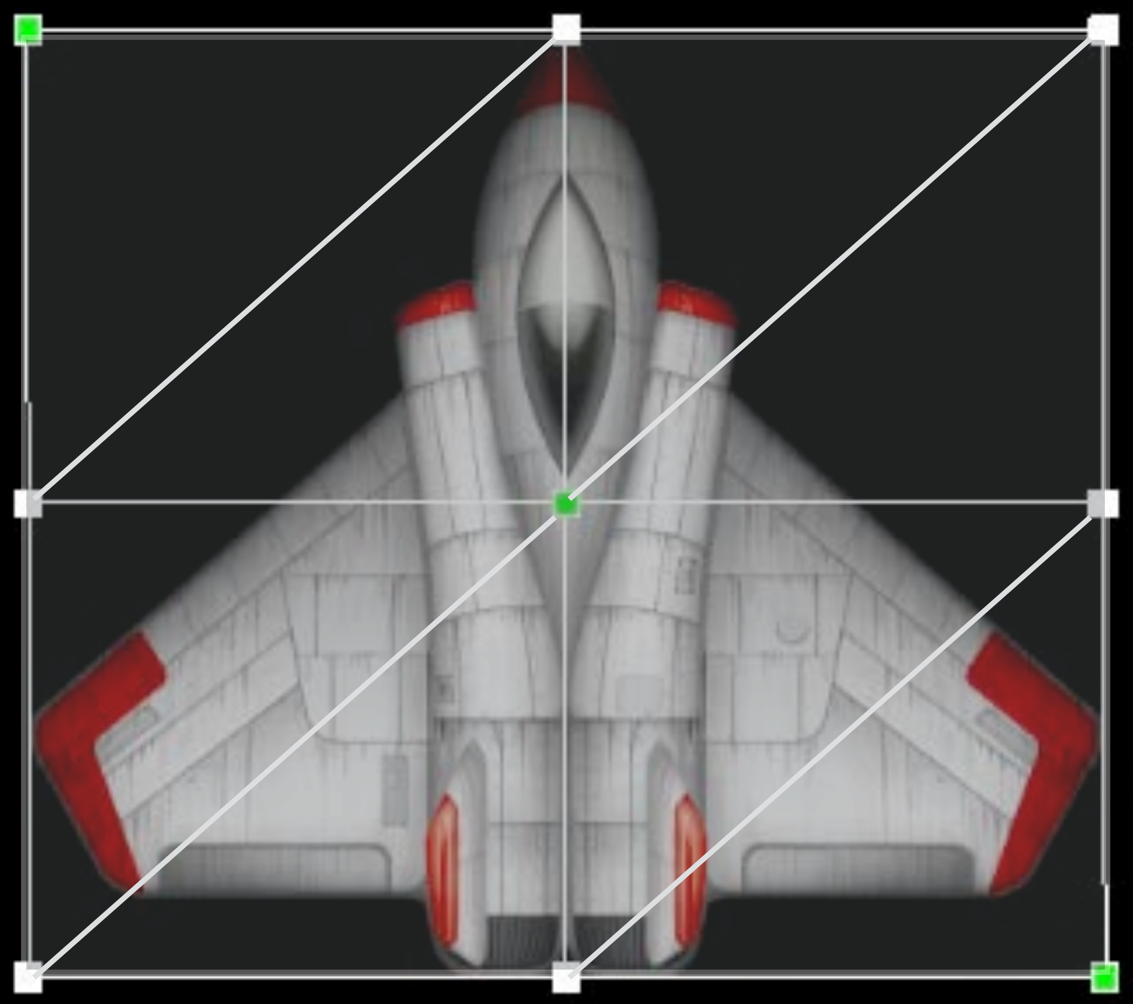
- More cells?



Warp Transformation

How it works

NEW



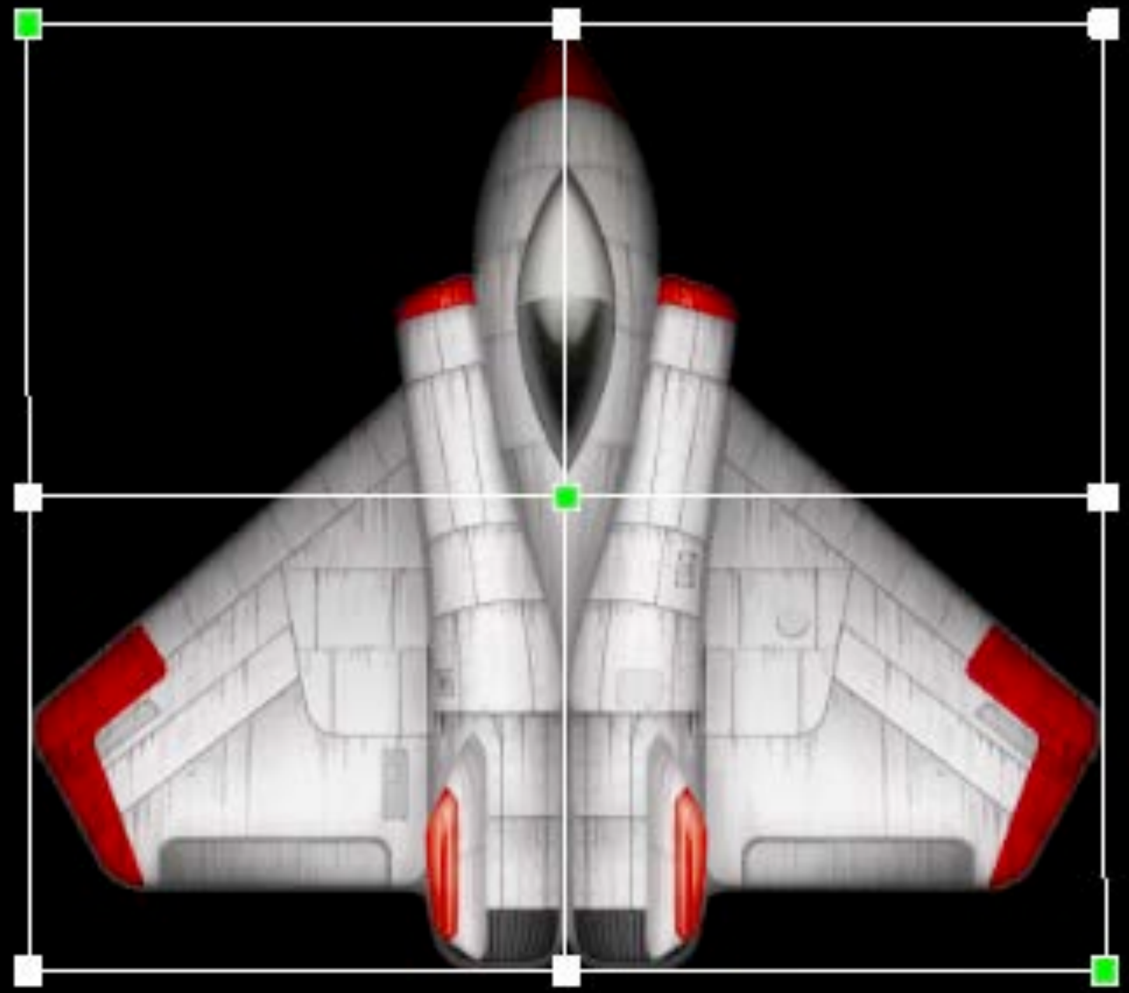
Warp Transformation

How it works

NEW

End result

- Automatic quad subdivisions
- High level of detail
- Minimal quad count



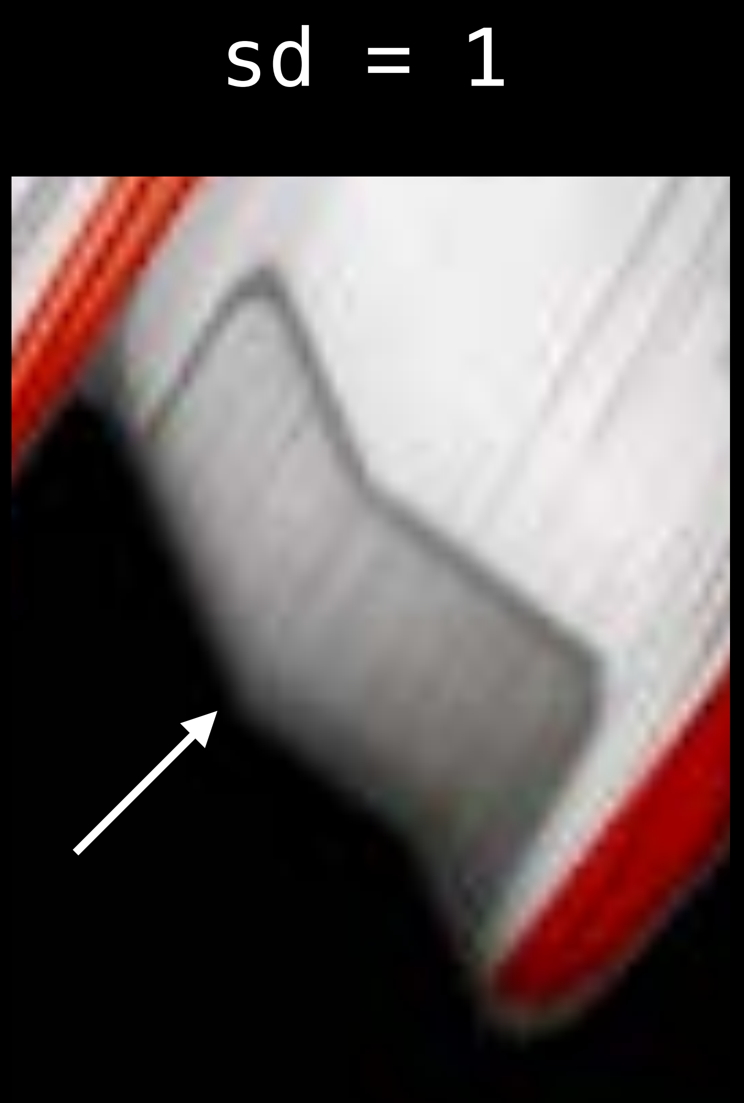
Warp Transformation

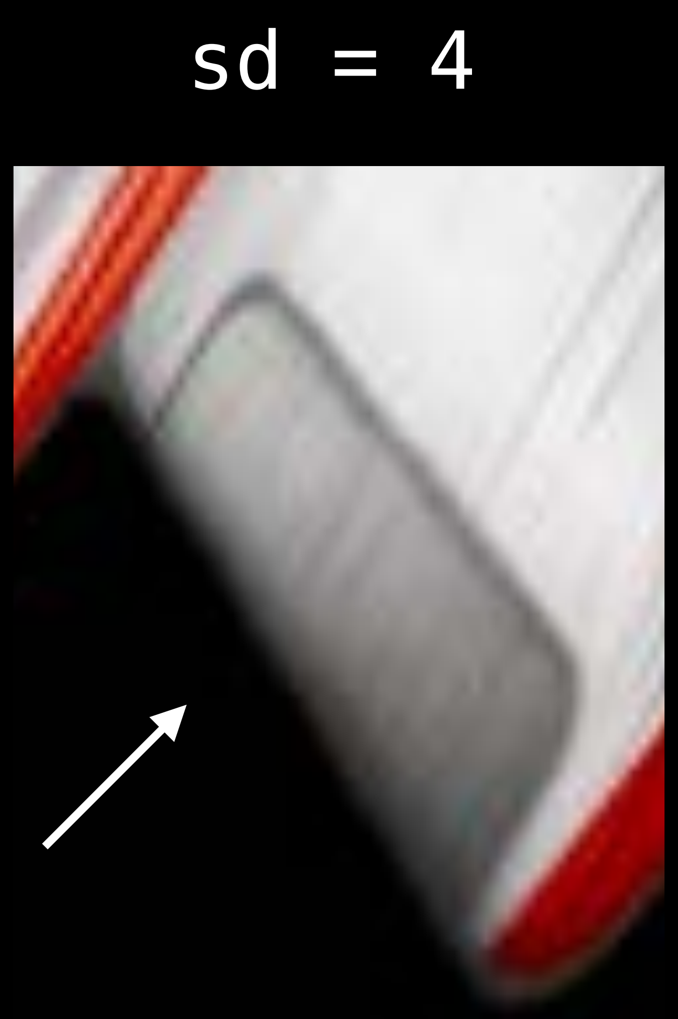
How it works

NEW

You can specify the max subdivision level

- Adjust details level
- Performance tuning





NEW

```
// SKWarpGeometryGrid - 2x2 grid example.
// [0]---[1]---[2]
//  |     |     |
// [3]---[4]---[5]
//  |     |     |
// [6]---[7]---[8]

var src = [float2]()
var dst = [float2]()

let warpGrid = SKWarpGeometryGrid(columns: 2,
                                  rows: 2,
                                  sourcePositions: src,
                                  destPositions: dst)

sprite.warpGeometry = warpGrid
sprite.subdivisionLevels = 3  // Optional, defaults to 2
```

```
// SKWarpGeometryGrid - 2x2 grid example.
// [0]---[1]---[2]
//  |     |     |
// [3]---[4]---[5]
//  |     |     |
// [6]---[7]---[8]

var src = [float2]()
var dst = [float2]()

let warpGrid = SKWarpGeometryGrid(columns: 2,
                                  rows: 2,
                                  sourcePositions: src,
                                  destPositions: dst)

sprite.warpGeometry = warpGrid
sprite.subdivisionLevels = 3  // Optional, defaults to 2
```

NEW

NEW

```swift
// SKWarpGeometryGrid - 2x2 grid example.
// [0]---[1]---[2]
//  |     |     |
// [3]---[4]---[5]
//  |     |     |
// [6]---[7]---[8]

var src = [float2]()
var dst = [float2]()

let warpGrid = SKWarpGeometryGrid(columns: 2,
                                  rows: 2,
                                  sourcePositions: src,
                                  destPositions: dst)

sprite.warpGeometry = warpGrid
sprite.subdivisionLevels = 3  // Optional, defaults to 2
```

NEW

```
// SKWarpGeometryGrid - 2x2 grid example.
// [0]---[1]---[2]
//  |     |     |
// [3]---[4]---[5]
//  |     |     |
// [6]---[7]---[8]

var src = [float2]()
var dst = [float2]()

let warpGrid = SKWarpGeometryGrid(columns: 2,
                                  rows: 2,
                                  sourcePositions: src,
                                  destPositions: dst)

sprite.warpGeometry = warpGrid
sprite.subdivisionLevels = 3  // Optional, defaults to 2
```

NEW

```
// SKWarpGeometryGrid - 2x2 grid example.
// [0]---[1]---[2]
//  |     |     |
// [3]---[4]---[5]
//  |     |     |
// [6]---[7]---[8]

var src = [float2]()
var dst = [float2]()

let warpGrid = SKWarpGeometryGrid(columns: 2,
                                  rows: 2,
                                  sourcePositions: src,
                                  destPositions: dst)

sprite.warpGeometry = warpGrid
sprite.subdivisionLevels = 3  // Optional, defaults to 2
```

```swift
// New SKAction

let a1 = SKAction.warp(to: grid,
                       duration: 5.0)

let a2 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3])

let a3 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3],
                          restore: true)
```

NEW

NEW

```
// New SKAction

let a1 = SKAction.warp(to: grid,
                       duration: 5.0)

let a2 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3])

let a3 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3],
                          restore: true)
```

NEW

```swift
// New SKAction

let a1 = SKAction.warp(to: grid,
                       duration: 5.0)

let a2 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3])

let a3 = SKAction.animate(withWarps: [grid1, grid2, grid3],
                          times: [t1, t2, t3],
                          restore: true)
```

Demo

Warp transformation

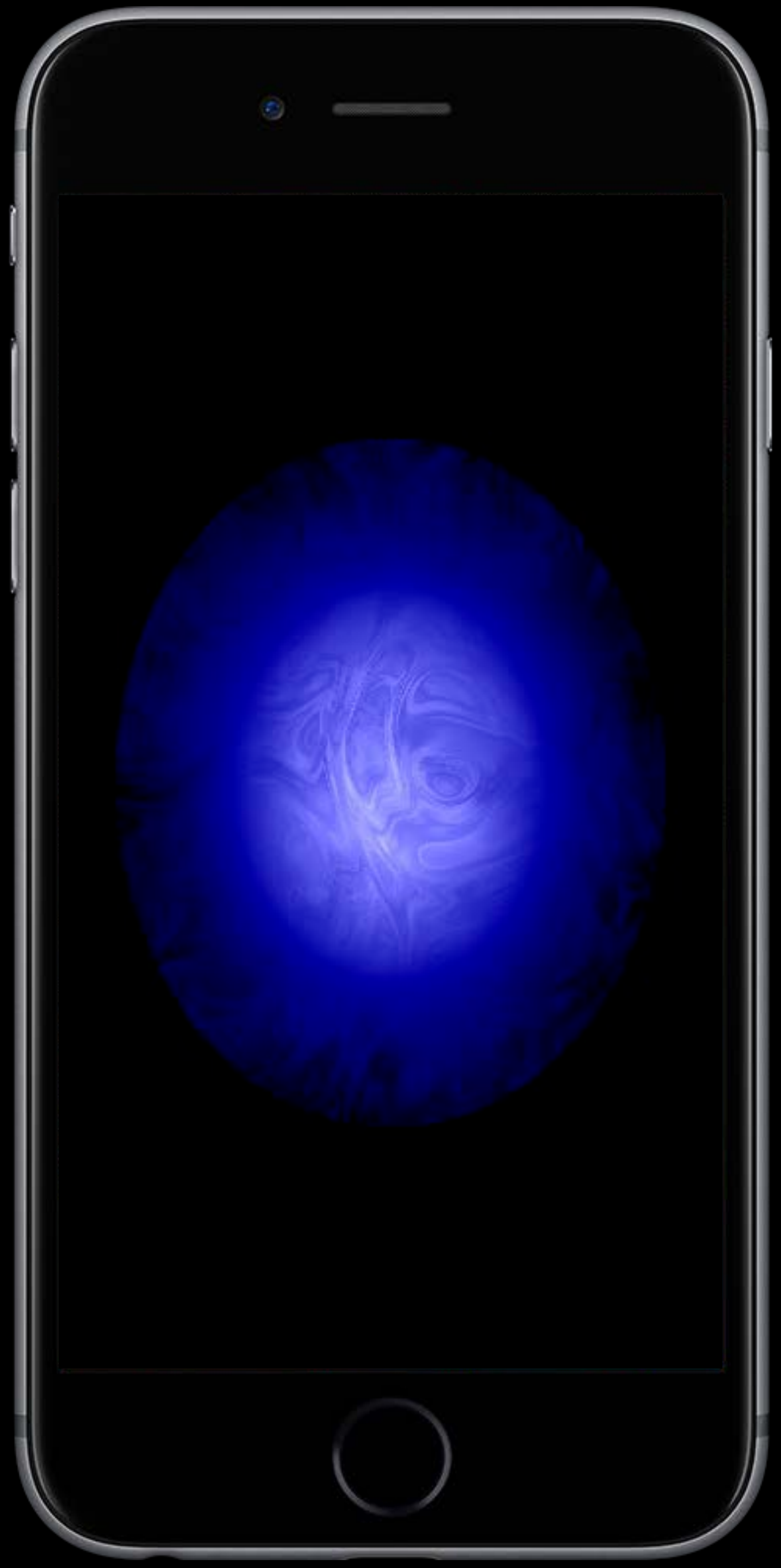
Per-Node Attributes for Custom Shaders

Per-Node Attributes for Custom Shaders

Introduction

Custom shaders in SpriteKit

- SKShader (fragment shader)
- Built-in shader symbols
- SKUniform

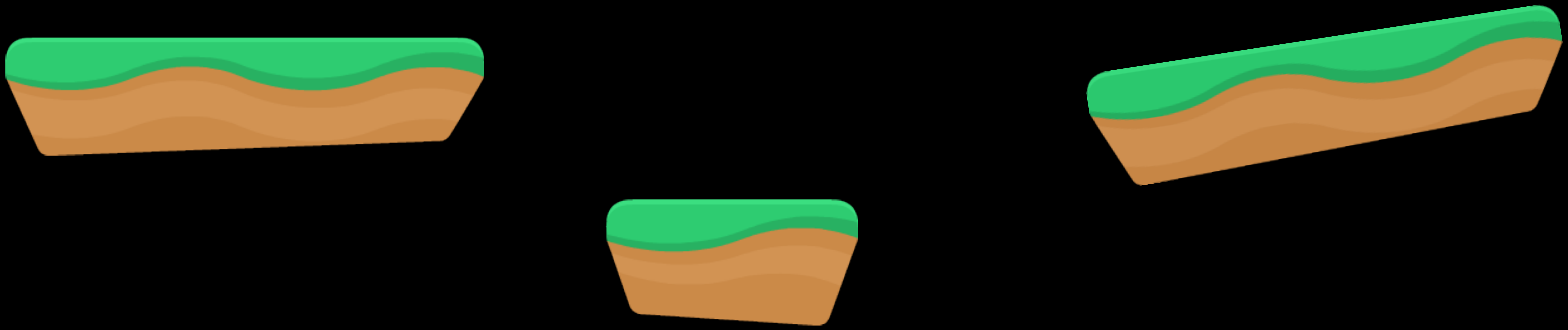


Per-Node Attributes for Custom Shaders

Game idea

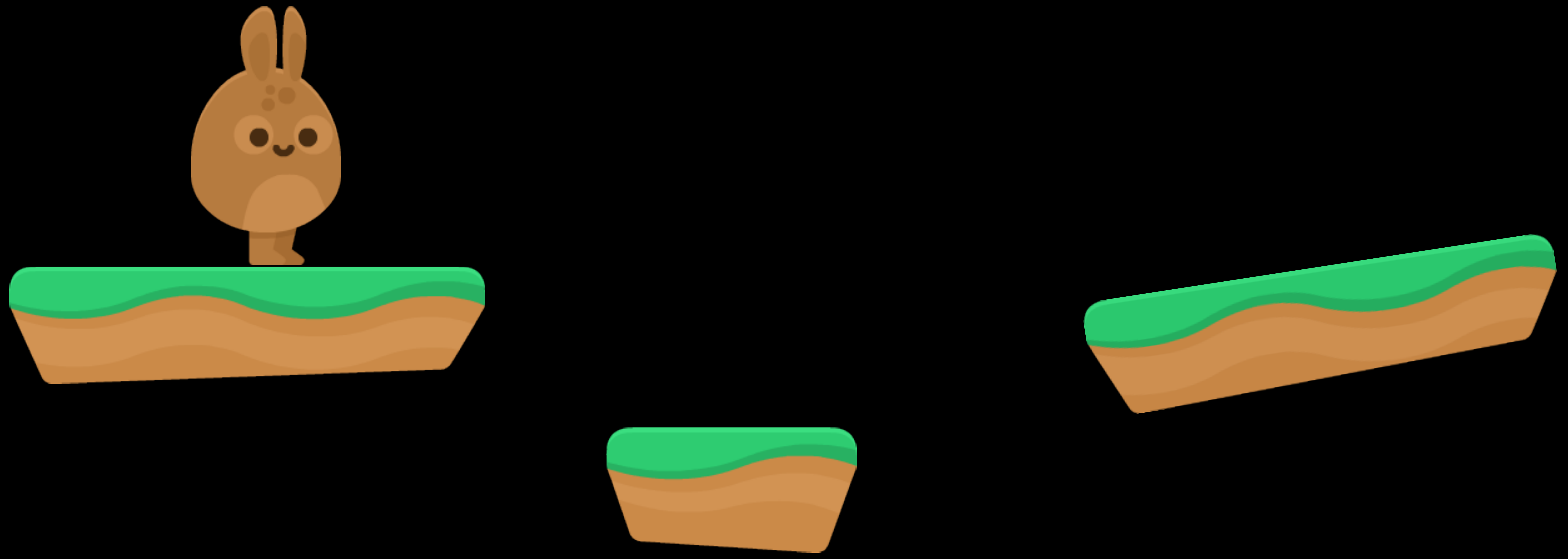
Per-Node Attributes for Custom Shaders

Game idea



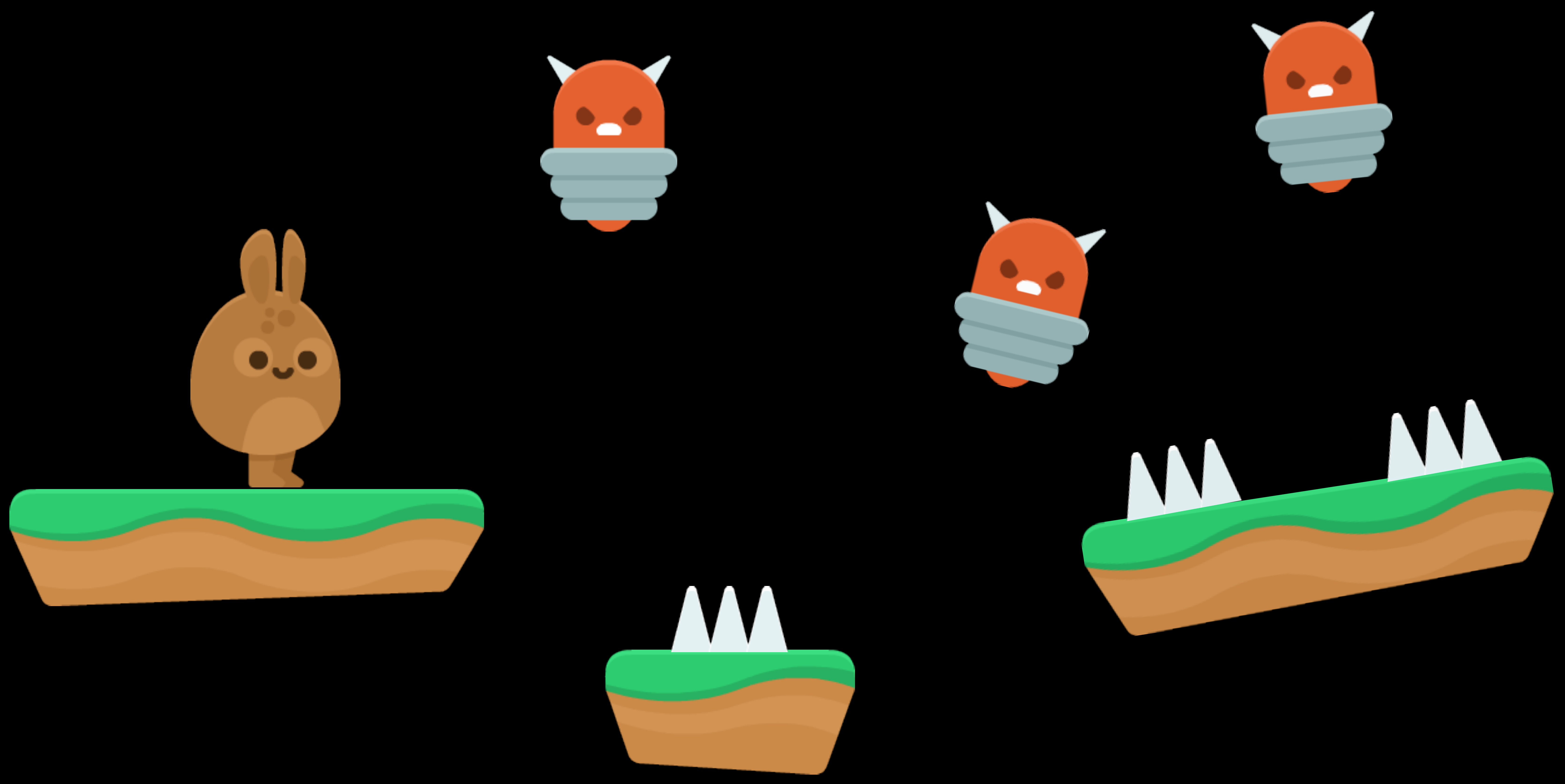
Per-Node Attributes for Custom Shaders

Game idea



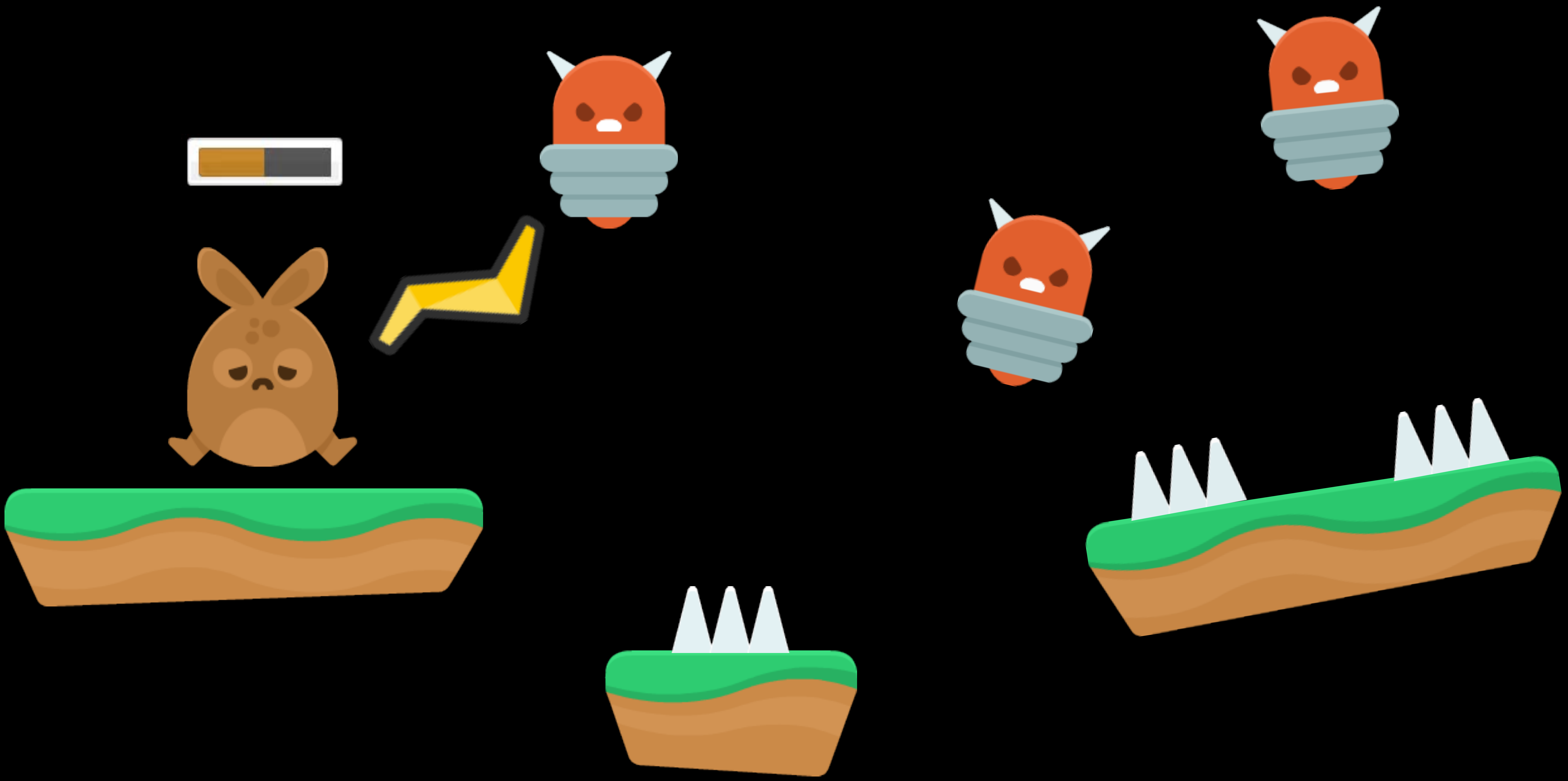
Per-Node Attributes for Custom Shaders

Game idea



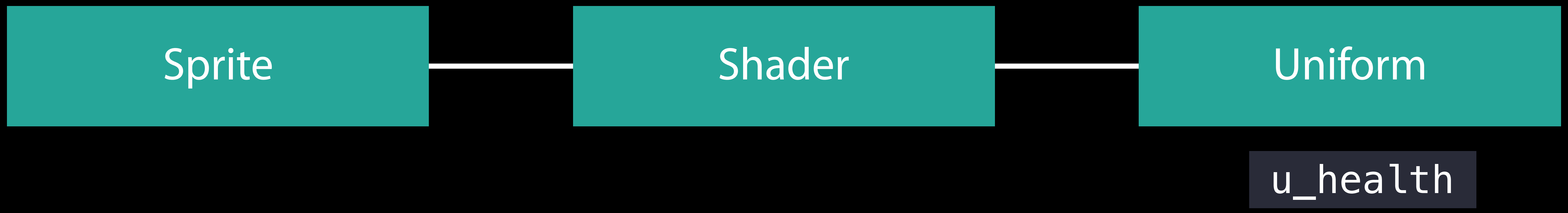
Per-Node Attributes for Custom Shaders

Game idea



Per-Node Attributes for Custom Shaders

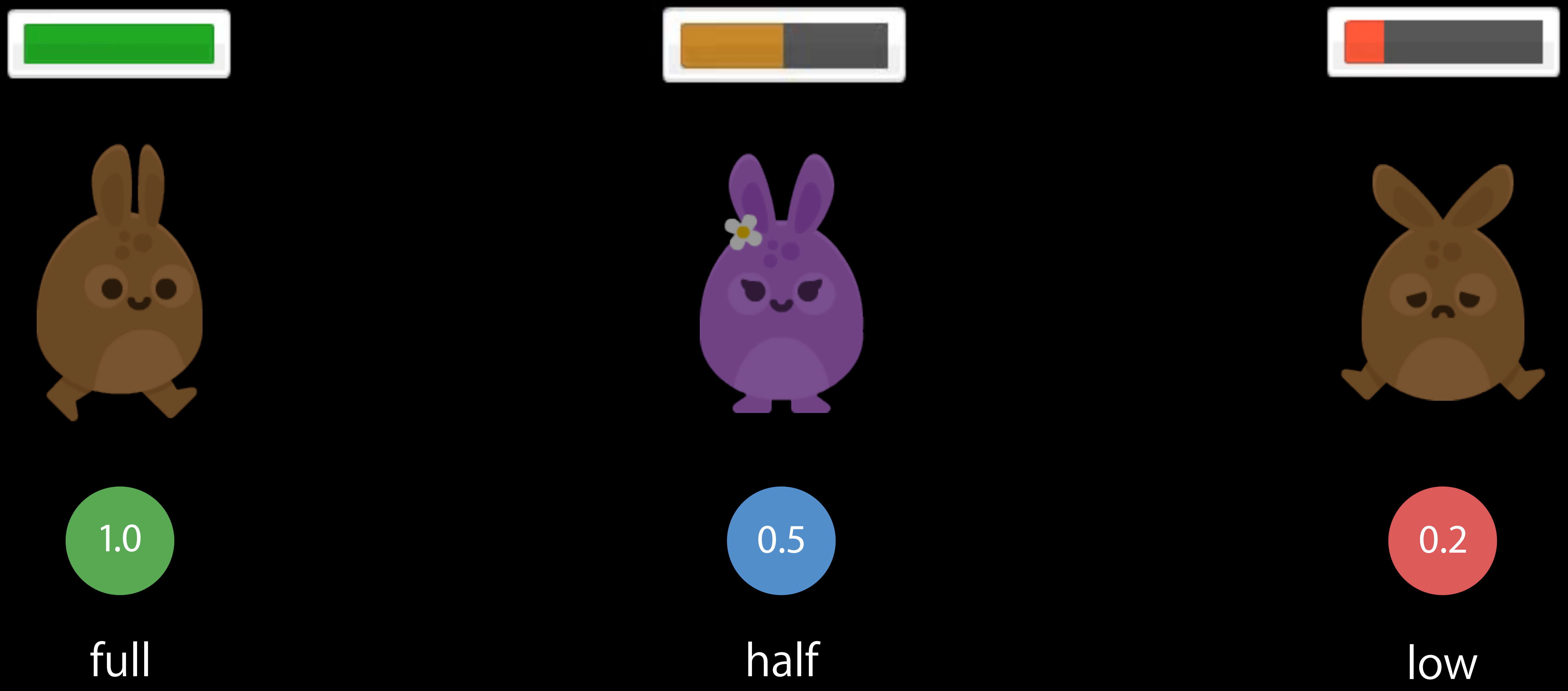
Example



Per-Node Attributes for Custom Shaders

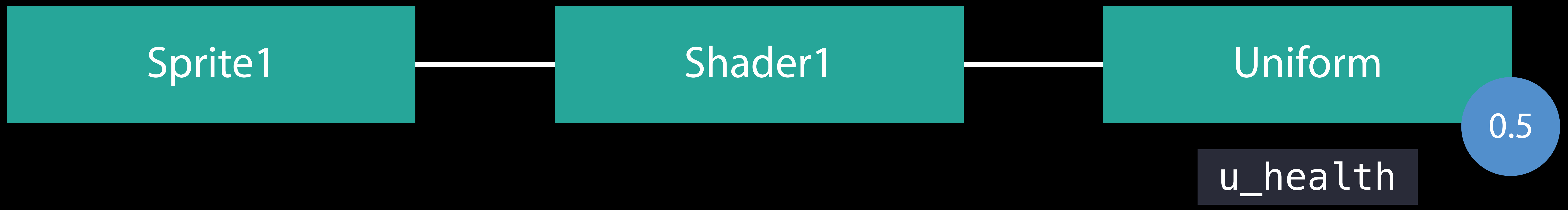
Example

```
u_health : float
```



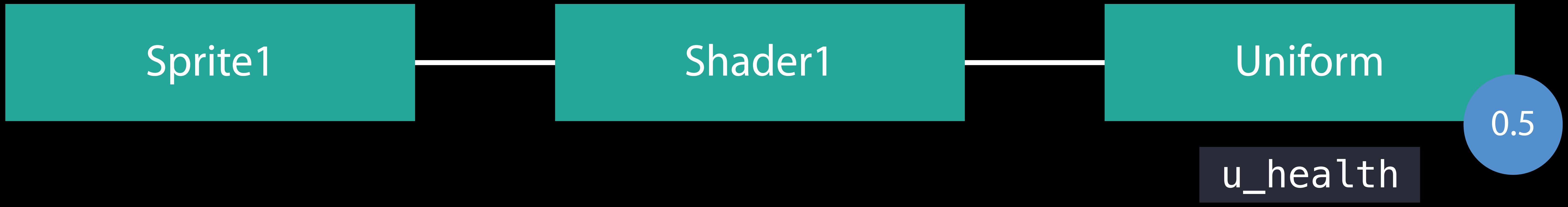
Per-Node Attributes for Custom Shaders

Example



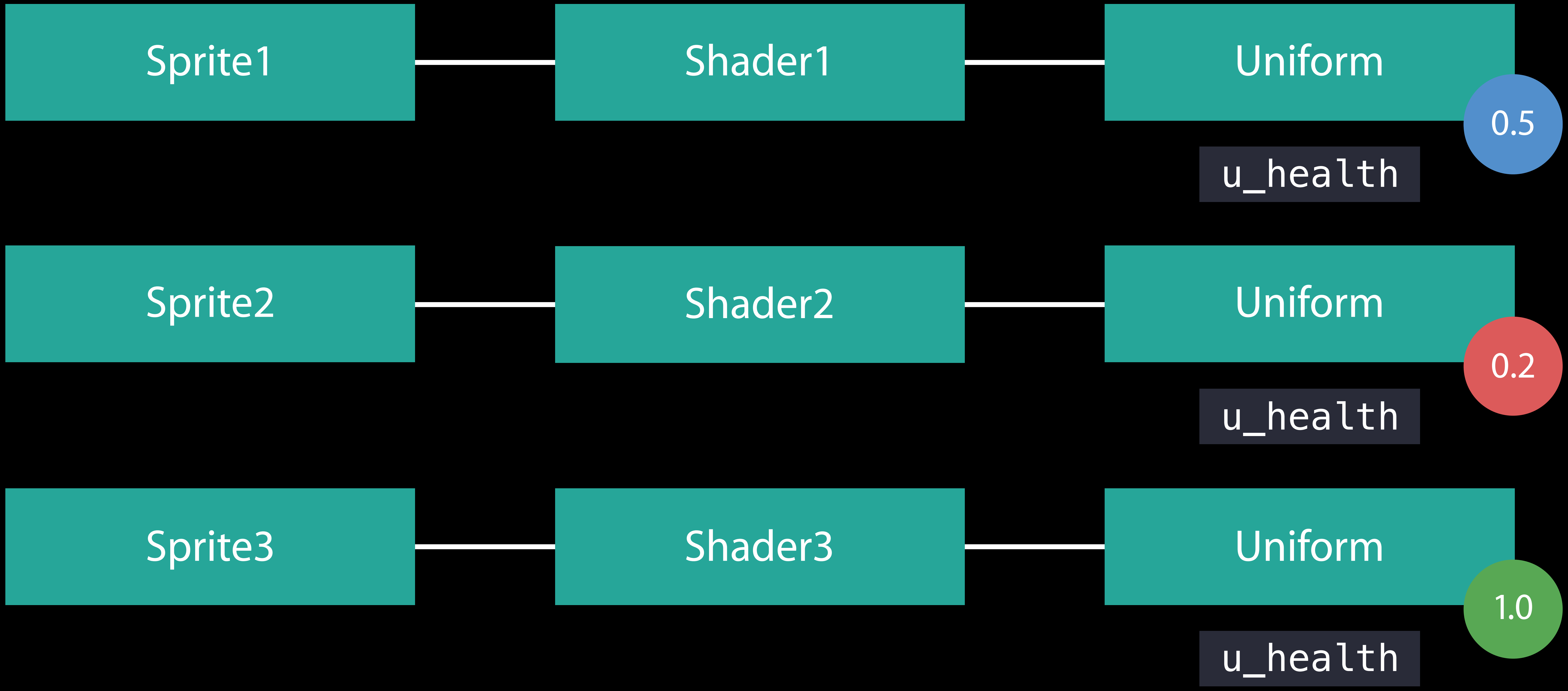
Per-Node Attributes for Custom Shaders

Example



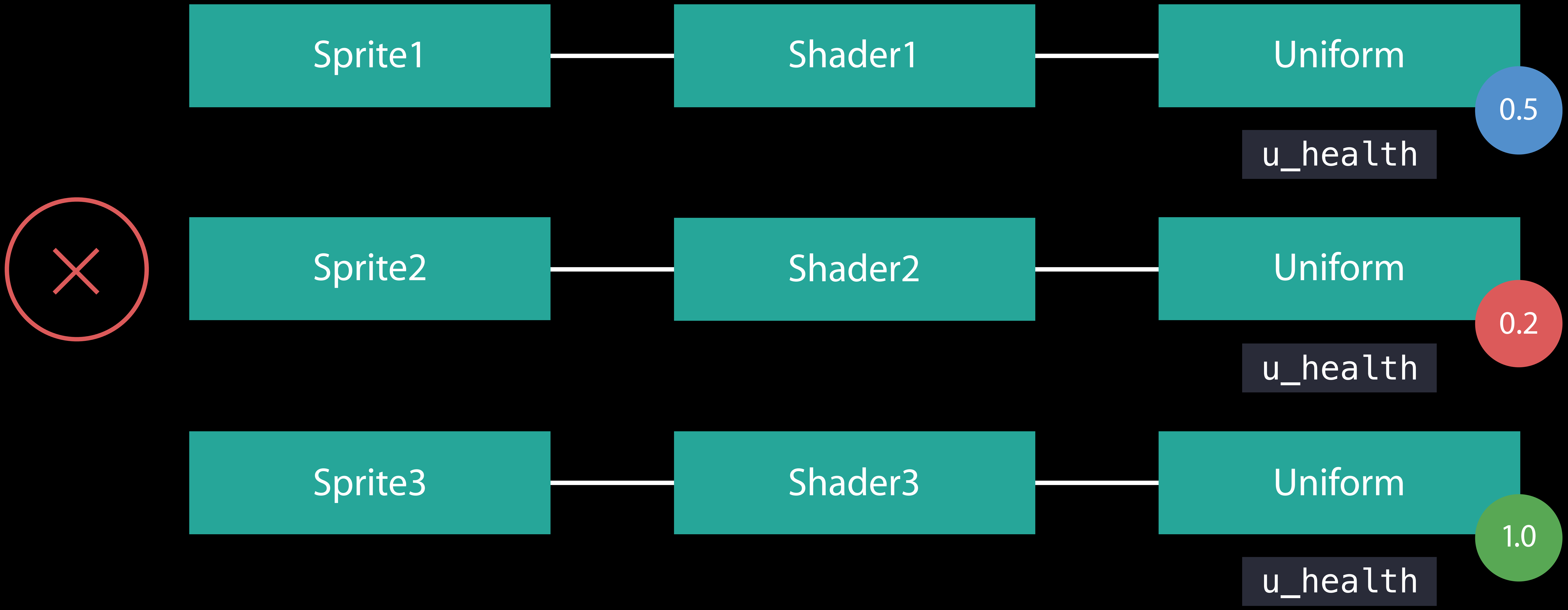
Per-Node Attributes for Custom Shaders

Example



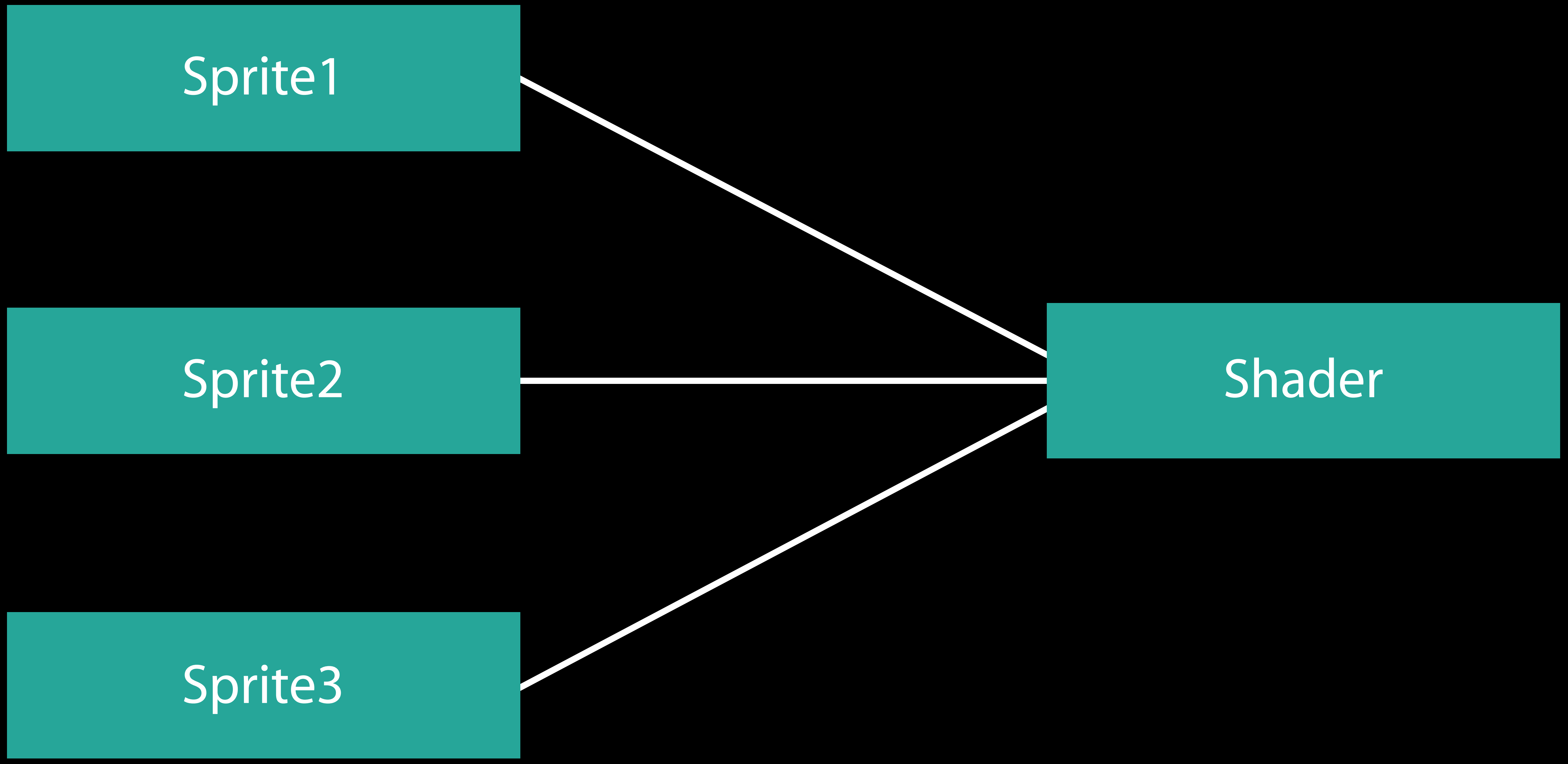
Per-Node Attributes for Custom Shaders

Example



Per-Node Attributes for Custom Shaders

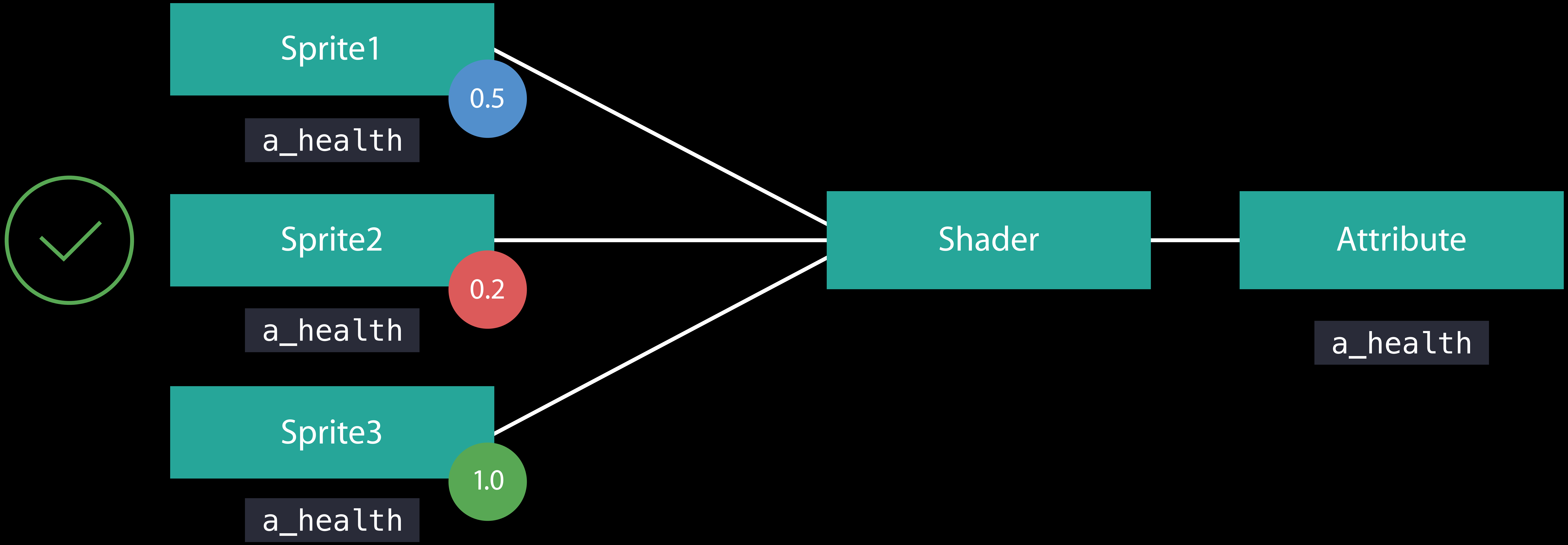
Example



Per-Node Attributes for Custom Shaders

NEW

Example



```swift
// SKAttribute for Per-Node Customization

// 1) Create your attributes:
let attribute = SKAttribute(name: "a_health", type: .float)

// 2) Attach to a shader:
shader.attributes = [attribute]

// 3) Set attributes directly on compatible nodes:
sprite1.setValue(SKAttributeValue(float: 0.2), forAttributeNamed: "a_health")
sprite2.setValue(SKAttributeValue(float: 0.5), forAttributeNamed: "a_health")
sprite3.setValue(SKAttributeValue(float: 1.0), forAttributeNamed: "a_health")
```

NEW

```
// SKAttribute for Per-Node Customization

// 1) Create your attributes:
let attribute = SKAttribute(name: "a_health", type: .float)

// 2) Attach to a shader:
shader.attributes = [attribute]

// 3) Set attributes directly on compatible nodes:
sprite1.setValue(SKAttributeValue(float: 0.2), forAttributeNamed: "a_health")
sprite2.setValue(SKAttributeValue(float: 0.5), forAttributeNamed: "a_health")
sprite3.setValue(SKAttributeValue(float: 1.0), forAttributeNamed: "a_health")
```

NEW

```swift
// SKAttribute for Per-Node Customization

// 1) Create your attributes:
let attribute = SKAttribute(name: "a_health", type: .float)

// 2) Attach to a shader:
shader.attributes = [attribute]

// 3) Set attributes directly on compatible nodes:
sprite1.setValue(SKAttributeValue(float: 0.2), forAttributeNamed: "a_health")
sprite2.setValue(SKAttributeValue(float: 0.5), forAttributeNamed: "a_health")
sprite3.setValue(SKAttributeValue(float: 1.0), forAttributeNamed: "a_health")
```

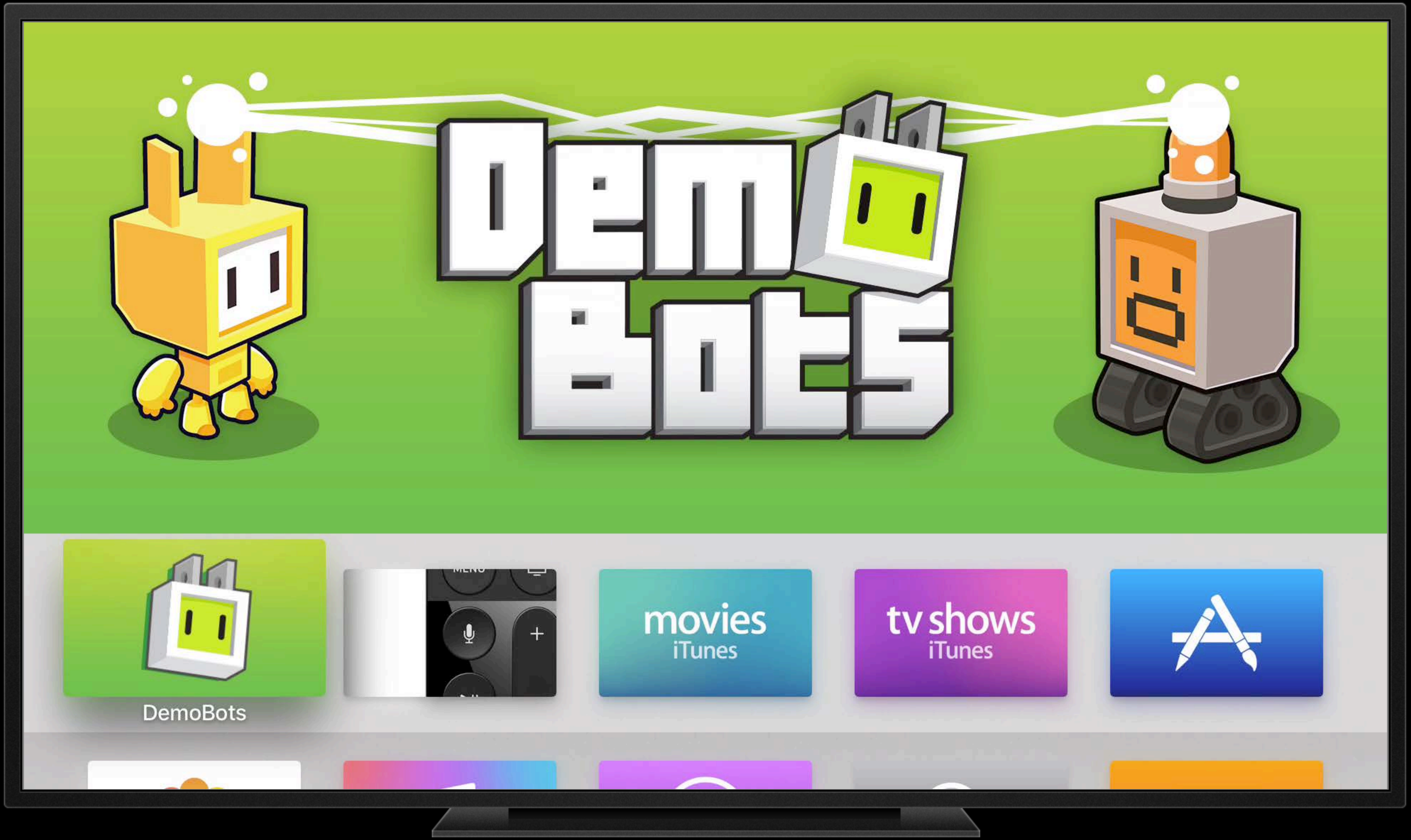
```swift
// SKAttribute for Per-Node Customization

// 1) Create your attributes:
let attribute = SKAttribute(name: "a_health", type: .float)

// 2) Attach to a shader:
shader.attributes = [attribute]

// 3) Set attributes directly on compatible nodes:
sprite1.setValue(SKAttributeValue(float: 0.2), forAttributeNamed: "a_health")
sprite2.setValue(SKAttributeValue(float: 0.5), forAttributeNamed: "a_health")
sprite3.setValue(SKAttributeValue(float: 1.0), forAttributeNamed: "a_health")
```

NEW

Focus Interaction on Apple TV

Focus Interaction on Apple TV

Introduction

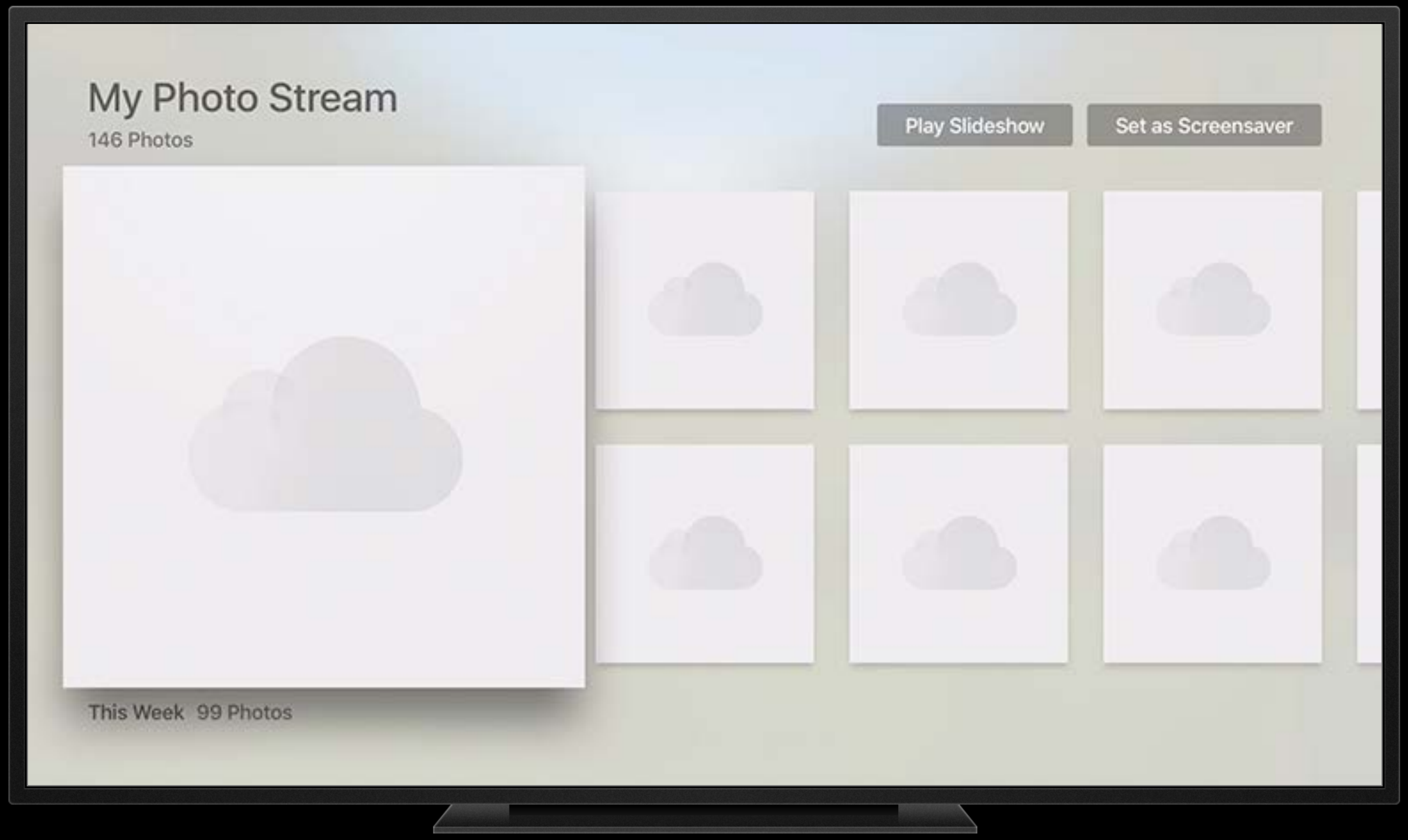


Focus Interaction on Apple TV

Introduction

Interaction on tvOS

- Integrated with UIKit
- Simple to use
- Consistent user experience
- Support a wide range of controllers



Focus Interaction on Apple TV

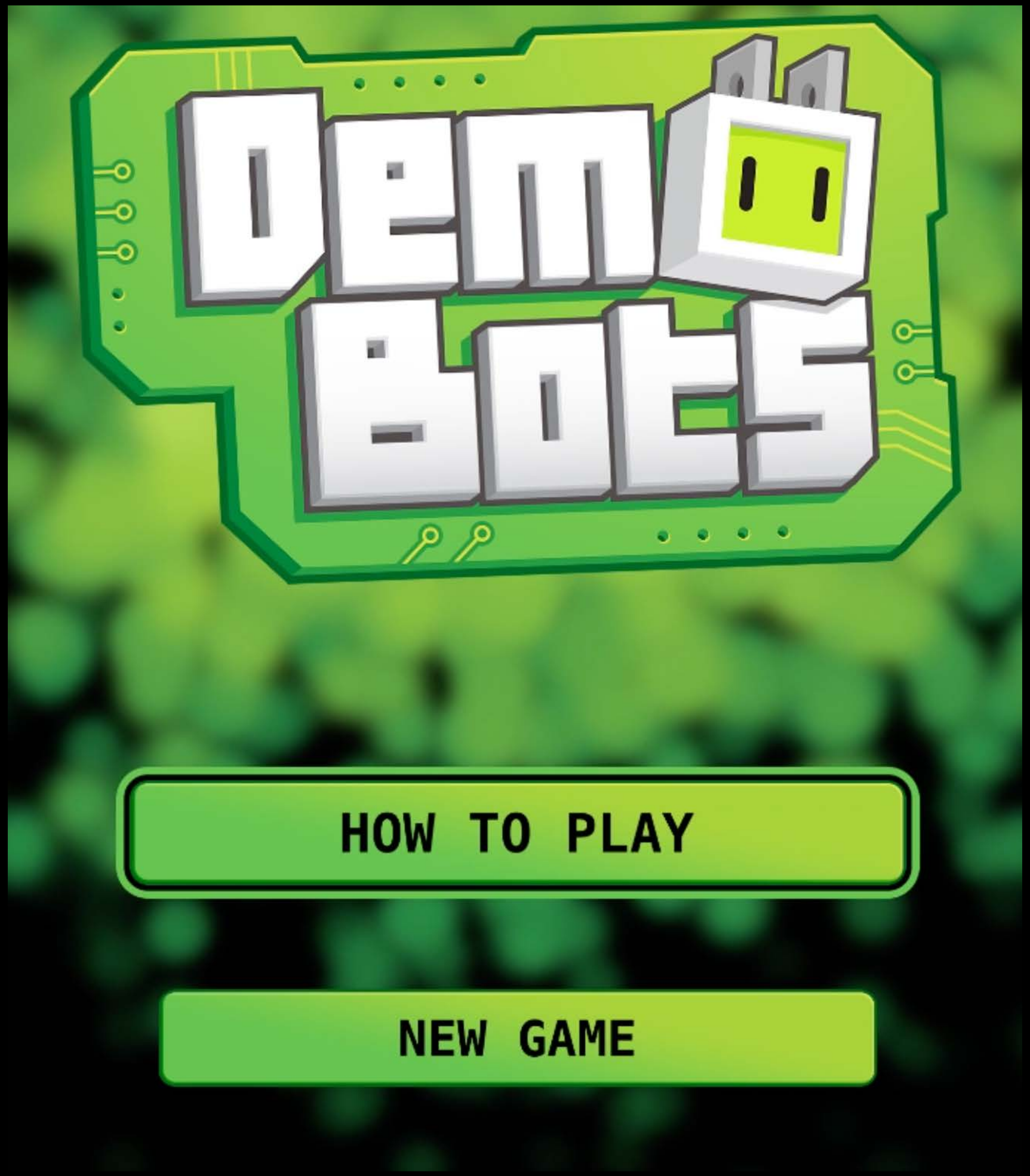
SpriteKit integration

NEW

Now also integrated with SpriteKit!

Use cases

- Game menus
- Entire game interaction
- Less code!



```
// Focus extended support for non-view items
public protocol UIFocusItem : UIFocusEnvironment
```

NEW

```swift
// Focus extended support for non-view items
public protocol UIFocusItem : UIFocusEnvironment
// SKNode now conforms to the UIFocusItem protocol
public class SKNode : UIResponder, NSCopying, NSCoding, UIFocusItem
```

NEW

```swift
// 1) Create a subclass
class MenuElementNode : SKSpriteNode {

    // 2) Override canBecomeFocused
    override func canBecomeFocused() -> Bool {
        return true
    }

}
```

NEW

```swift
// 1) Create a subclass
class MenuElementNode : SKSpriteNode {

    // 2) Override canBecomeFocused
    override func canBecomeFocused() -> Bool {
        return true
    }

}
```

NEW

NEW

```swift
// 1) Create a subclass
class MenuElementNode : SKSpriteNode {

    // 2) Override canBecomeFocused
    override func canBecomeFocused() -> Bool {
        return true
    }

}
```

NEW

```
class GameScene : SKScene {

    let menuItem = MenuElementNode()

    override func sceneDidLoad() {
        // 3) Opt-in the node for focus interaction
        self.menuItem.isUserInteractionEnabled = true;
    }

    // 4) Track focus updates on your SKView, SKScene
    // or any SKNode that would make sense for your app logic.
    override func didUpdateFocus(in context: UIFocusUpdateContext,
                                 with coordinator: UIFocusAnimationCoordinator) {
        let prevItem = context.previouslyFocusedItem
        let nextItem = context.nextFocusedItem

        if nextItem is MenuElementNode {
            // Run some SKAction
        }
    }
}
```

NEW

```swift
class GameScene : SKScene {

    let menuItem = MenuElementNode()

    override func sceneDidLoad() {
        // 3) Opt-in the node for focus interaction
        self.menuItem.isUserInteractionEnabled = true;
    }

    // 4) Track focus updates on your SKView, SKScene
    // or any SKNode that would make sense for your app logic.
    override func didUpdateFocus(in context: UIFocusUpdateContext,
                                 with coordinator: UIFocusAnimationCoordinator) {
        let prevItem = context.previouslyFocusedItem
        let nextItem = context.nextFocusedItem

        if nextItem is MenuElementNode {
            // Run some SKAction
        }
    }
}
```

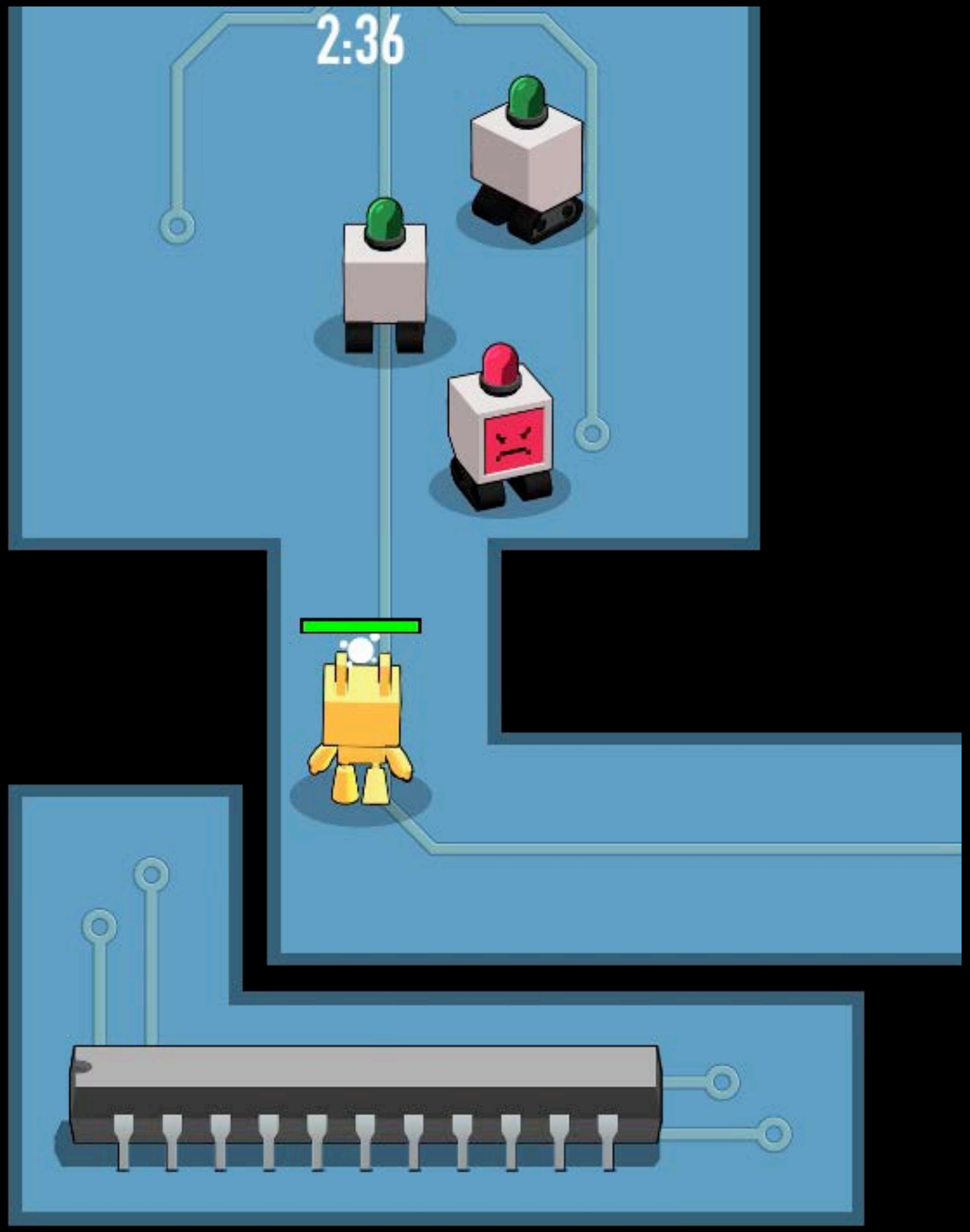
NEW

```swift
class GameScene : SKScene {

    let menuItem = MenuElementNode()

    override func sceneDidLoad() {
        // 3) Opt-in the node for focus interaction
        self.menuItem.isUserInteractionEnabled = true;
    }

    // 4) Track focus updates on your SKView, SKScene
    // or any SKNode that would make sense for your app logic.
    override func didUpdateFocus(in context: UIFocusUpdateContext,
                                 with coordinator: UIFocusAnimationCoordinator) {
        let prevItem = context.previouslyFocusedItem
        let nextItem = context.nextFocusedItem

        if nextItem is MenuElementNode {
            // Run some SKAction
        }
    }
}
```

Focus Interaction on Apple TV

NEW

SpriteKit integration



SpriteKit on Apple Watch

SpriteKit on Apple Watch

NEW

Introduction

SpriteKit now available for Apple Watch!

- High-performance 2D graphics framework
- Particles, actions, physics, animations
- Scene and Particle Editors
- Debugging tools

SpriteKit on Apple Watch

NEW

Introduction

SpriteKit now available for Apple Watch!

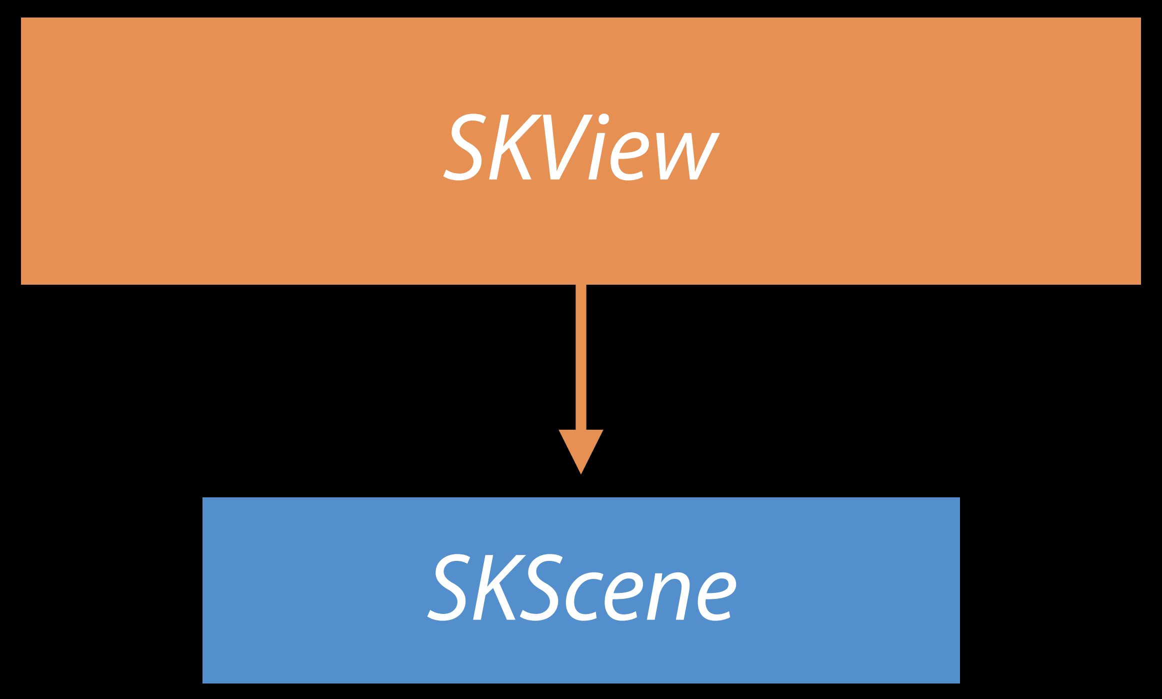
- High-performance 2D graphics framework
- Particles, actions, physics, animations
- Scene and Particle Editors
- Debugging tools

SpriteKit on Apple Watch

Getting started

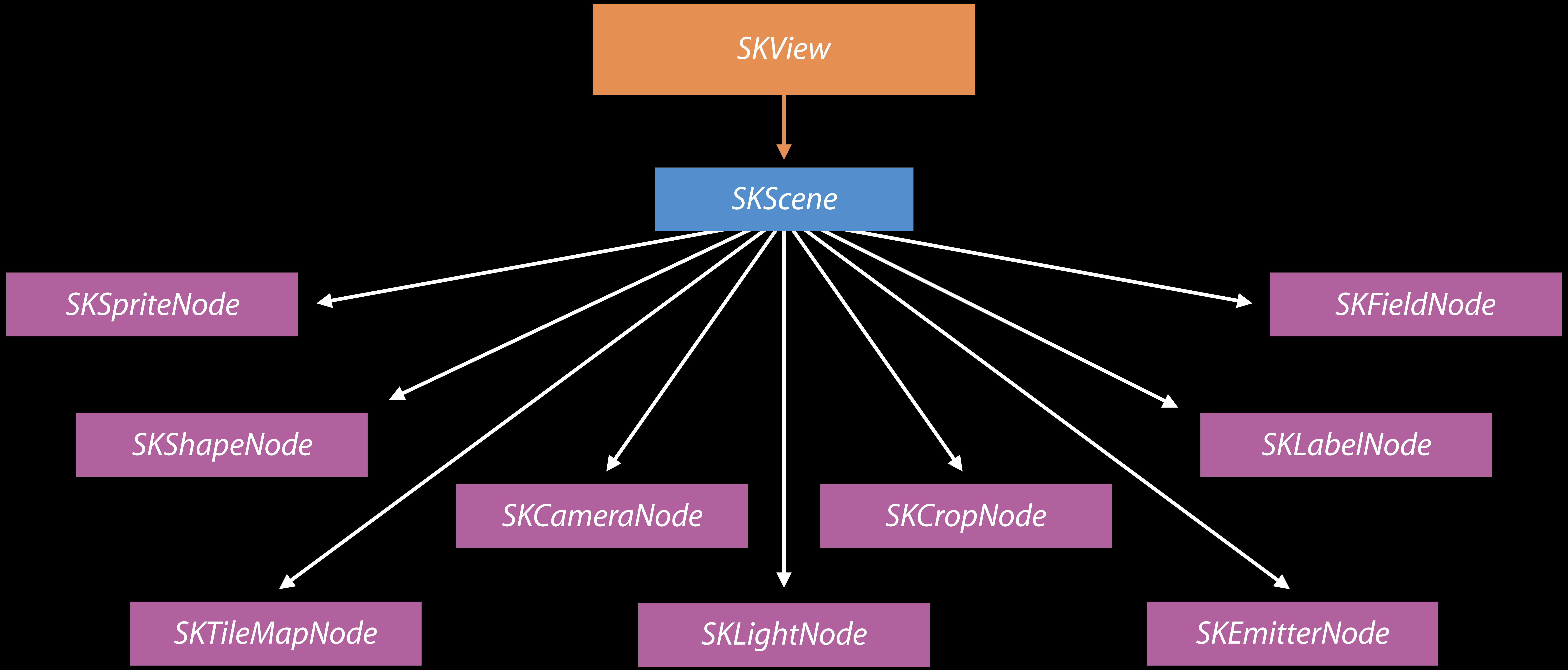
SpriteKit on Apple Watch

Getting started



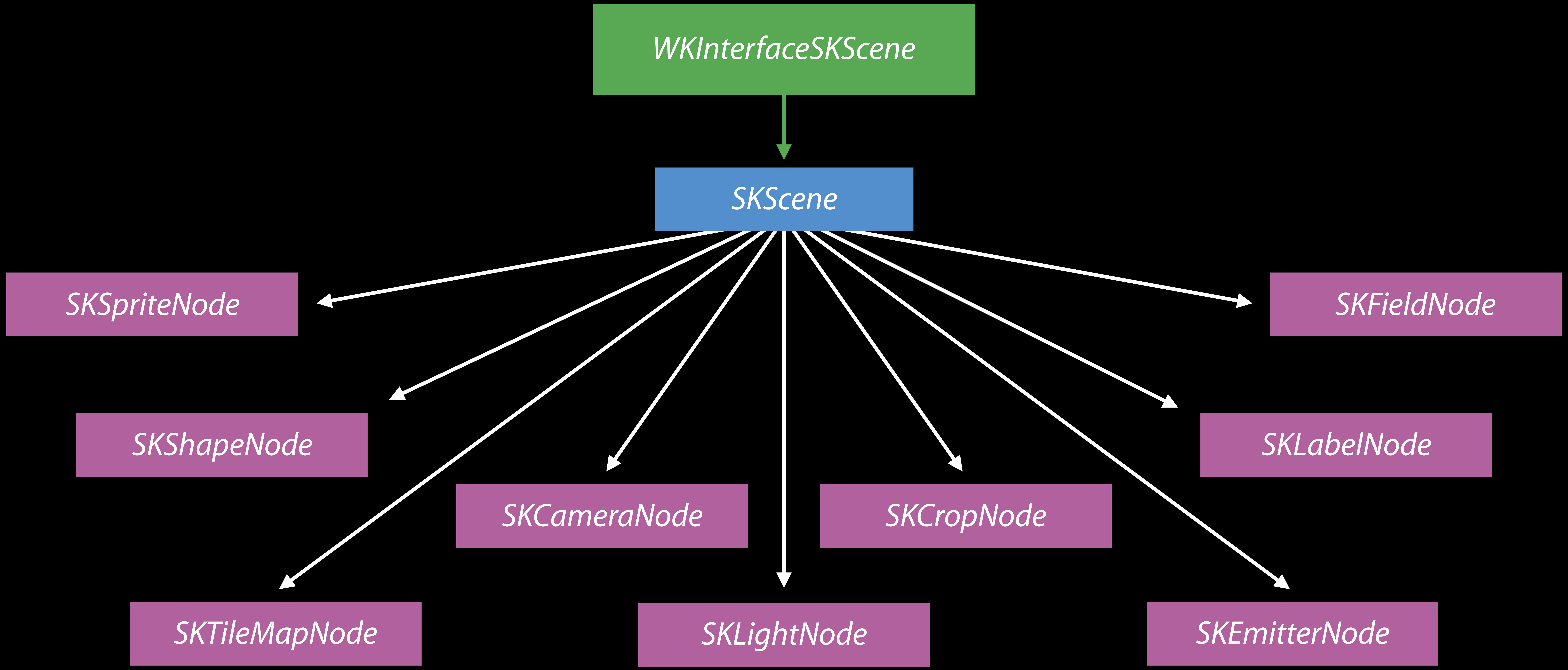
SpriteKit on Apple Watch

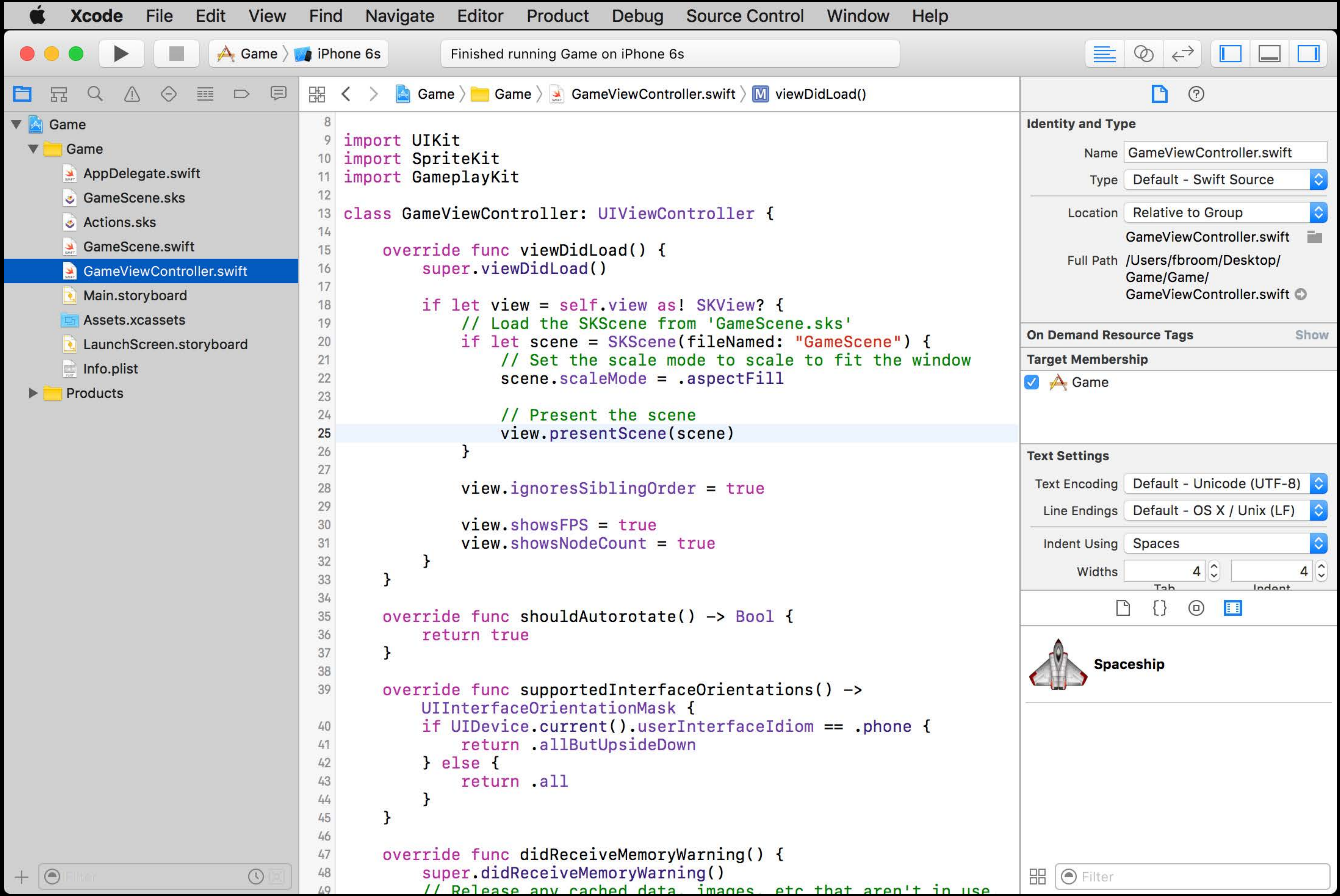
Getting started



SpriteKit on Apple Watch

Getting started





```
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
        }
    }

    override func shouldAutorotate() -> Bool {
        return true
    }

    override func supportedInterfaceOrientations() ->
        UIInterfaceOrientationMask {
        if UIDevice.current().userInterfaceIdiom == .phone {
            return .allButUpsideDown
        } else {
            return .all
        }
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // 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

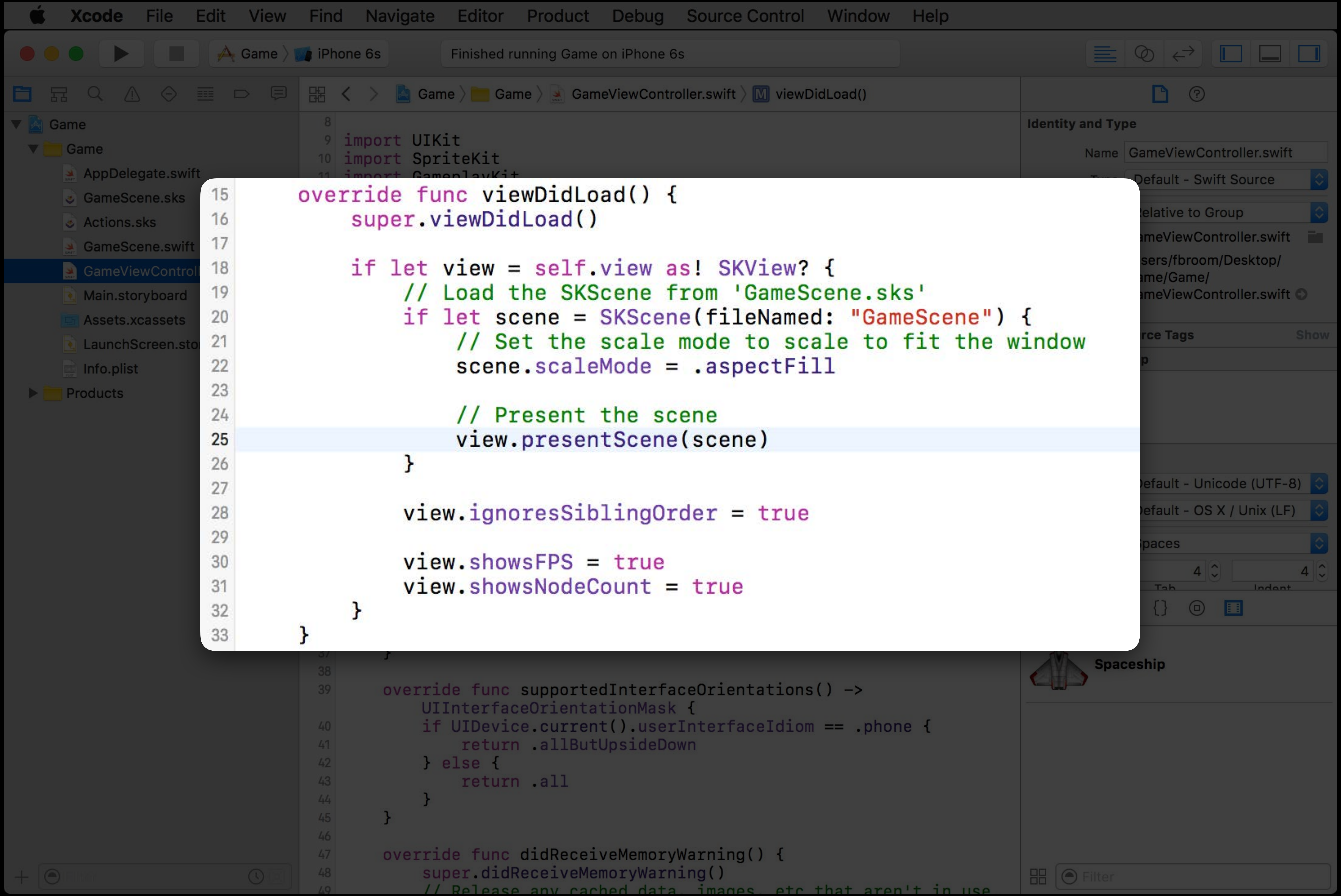
On Demand Resource Tags Show

Target Membership

✓ Game

Text Settings

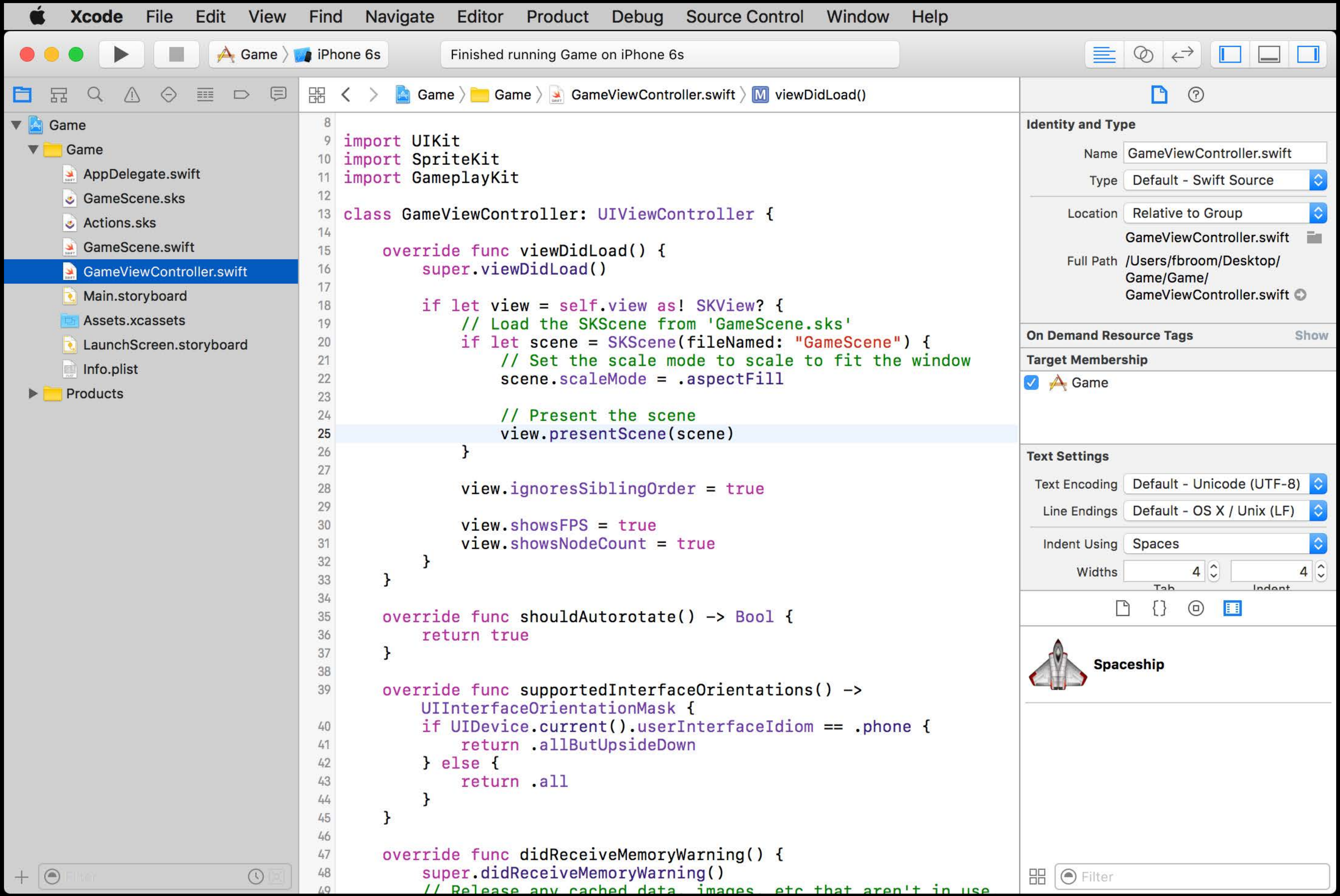
Text Encoding: Default - Unicode (UTF-8)
Line Endings: Default - OS X / Unix (LF)
Indent Using: Spaces
Widths: 4 Tab 4 Indent


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
GameViewControl
Main.storyboard
Assets.xcassets
LaunchScreen.sto
Info.plist
Products
Identity and Type
Name GameViewController.swift
Default - Swift Source
elative to Group
Default - Unicode (UTF-8)
Default - OS X / Unix (LF)
Spaceship

```swift
import UIKit
import SpriteKit
```

```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)
            }

            view.ignoresSiblingOrder = true

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

```swift
    override func supportedInterfaceOrientations() ->
        UIInterfaceOrientationMask {
        if UIDevice.current().userInterfaceIdiom == .phone {
            return .allButUpsideDown
        } else {
            return .all
        }
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
```



```swift
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
        }
    }

    override func shouldAutorotate() -> Bool {
        return true
    }

    override func supportedInterfaceOrientations() ->
        UIInterfaceOrientationMask {
        if UIDevice.current().userInterfaceIdiom == .phone {
            return .allButUpsideDown
        } else {
            return .all
        }
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // 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

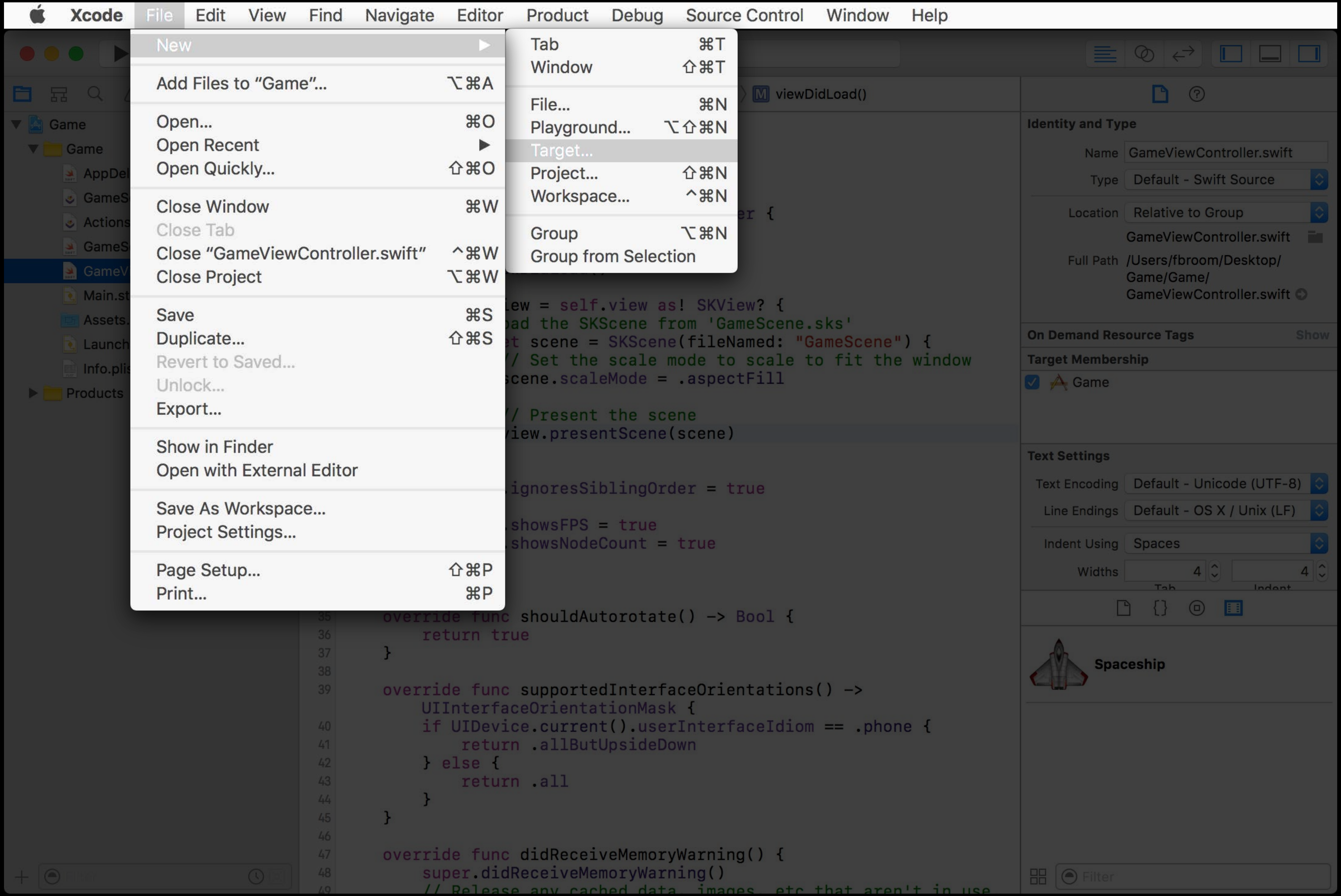
On Demand Resource Tags Show

Target Membership

✓ Game

Text Settings

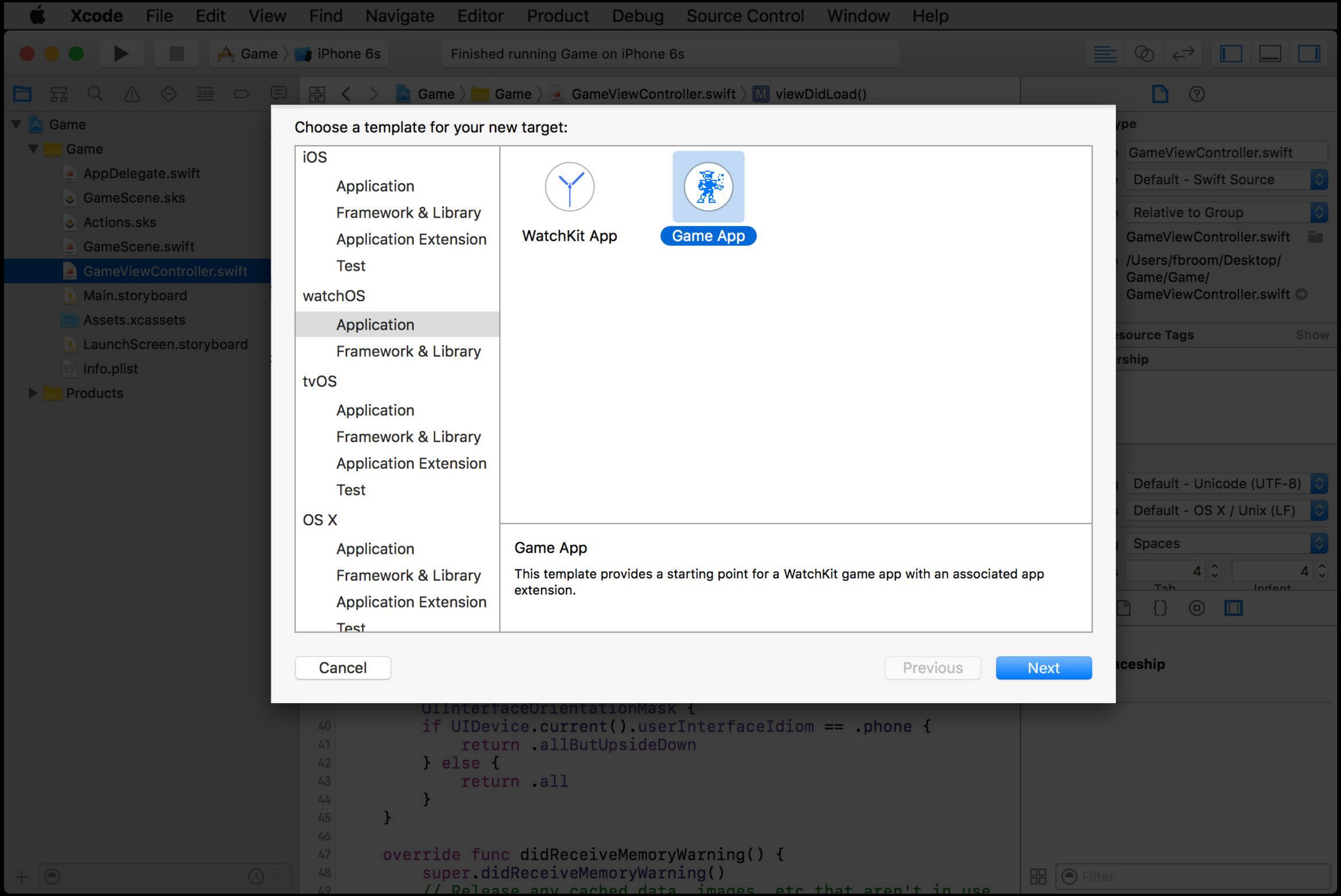
Text Encoding: Default - Unicode (UTF-8)
Line Endings: Default - OS X / Unix (LF)
Indent Using: Spaces
Widths: 4 Tab, 4 Indent



```
er {
ew = self.view as! SKView? {
oad the SKScene from 'GameScene.sks'
et scene = SKScene(fileNamed: "GameScene") {
// Set the scale mode to scale to fit the window
scene.scaleMode = .aspectFill

// Present the scene
view.presentScene(scene)

ignoresSiblingOrder = true

.showsFPS = true
.showsNodeCount = true

override func shouldAutorotate() -> Bool {
    return true
}

override func supportedInterfaceOrientations() ->
    UIInterfaceOrientationMask {
    if UIDevice.current().userInterfaceIdiom == .phone {
        return .allButUpsideDown
    } else {
        return .all
    }
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Release any cached data, images, etc that aren't in use
```

Name: GameViewController.swift
Type: Default - Swift Source
Location: Relative to Group
GameViewController.swift
Full Path: /Users/fbroom/Desktop/Game/Game/GameViewController.swift

Text Encoding: Default - Unicode (UTF-8)
Line Endings: Default - OS X / Unix (LF)
Indent Using: Spaces
Widths: 4 (Tab) 4 (Indent)



Choose a template for your new target:

Game App

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

GameViewController.swift

Default - Swift Source

Relative to Group

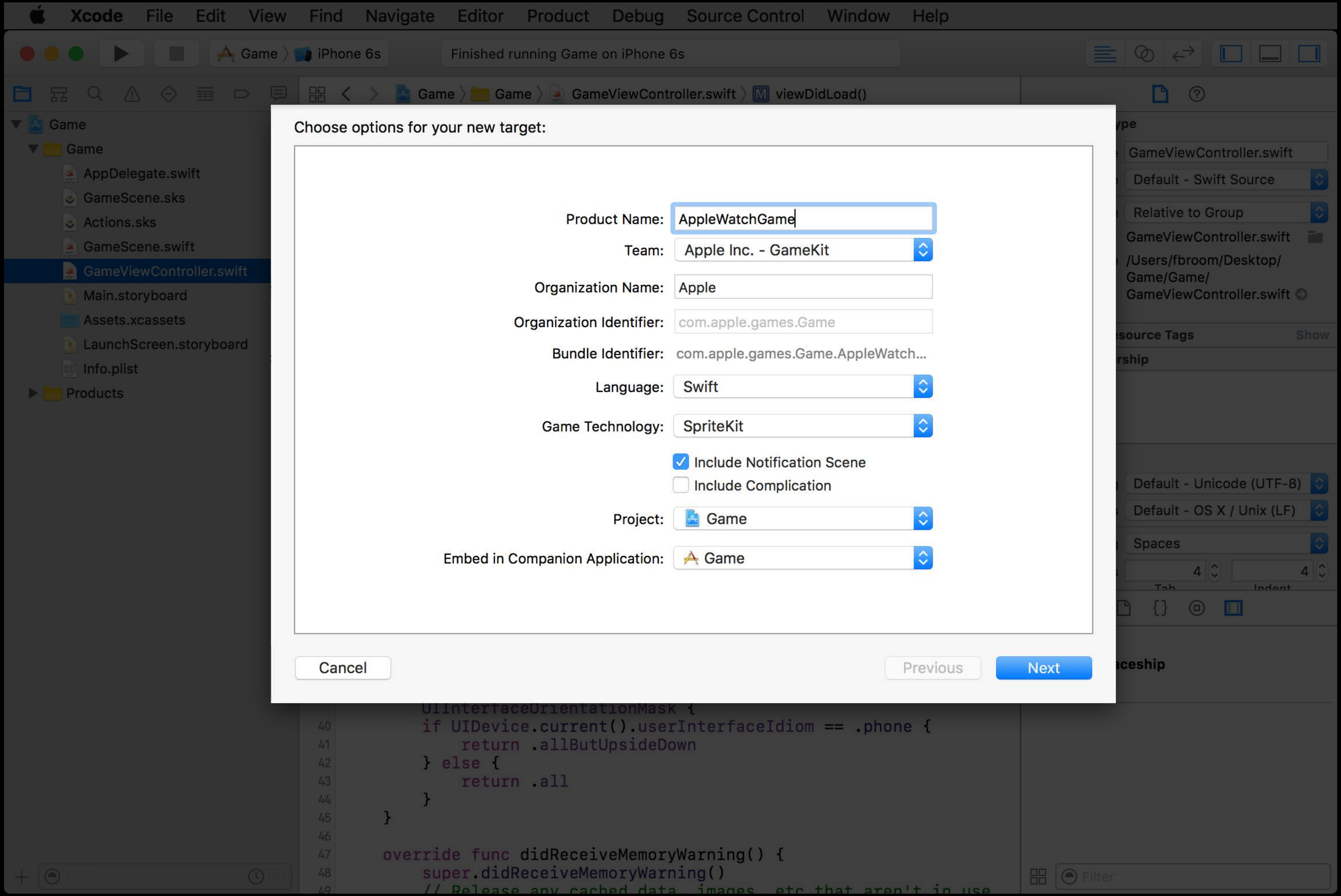
/Users/fbroom/Desktop/Game/Game/GameViewController.swift

Default - Unicode (UTF-8)

Default - OS X / Unix (LF)

Spaces

```
        UIInterfaceOrientationMask {
        if UIDevice.current().userInterfaceIdiom == .phone {
            return .allButUpsideDown
        } else {
            return .all
        }
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Release any cached data, images, etc that aren't in use
```

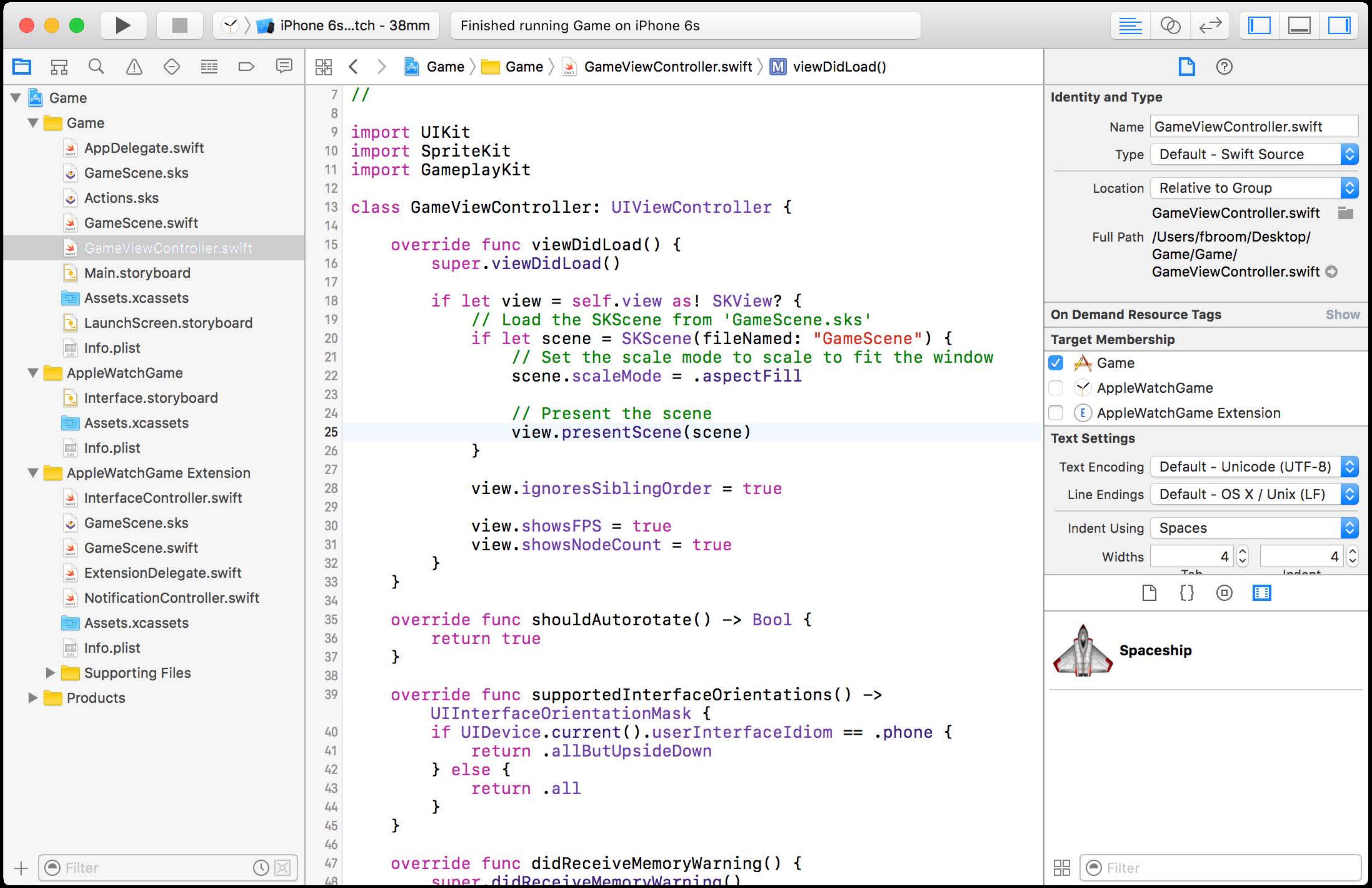


Choose options for your new target:

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

```
        UIInterfaceOrientationMask {
        if UIDevice.current().userInterfaceIdiom == .phone {
            return .allButUpsideDown
        } else {
            return .all
        }
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Release any cached data, images, etc that aren't in use
```

GameViewController.swift
Default - Swift Source
Relative to Group
GameViewController.swift
/Users/fbroom/Desktop/Game/Game/GameViewController.swift
Default - Unicode (UTF-8)
Default - OS X / Unix (LF)
Spaces
4
4



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

```
//

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
        }
    }

    override func shouldAutorotate() -> Bool {
        return true
    }

    override 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)

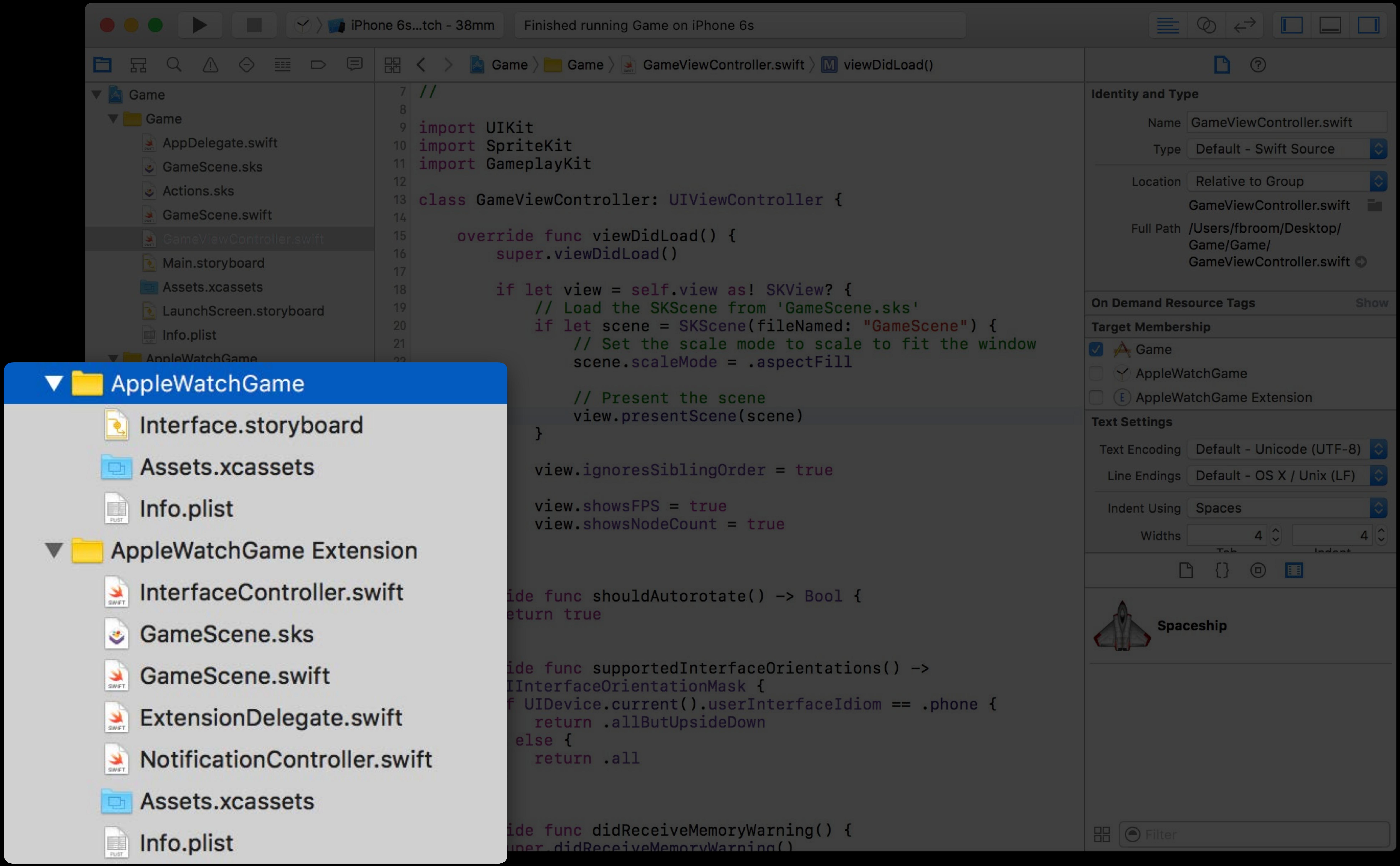
Line Endings: Default - OS X / Unix (LF)

Indent Using: Spaces

Widths: 4 Tab, 4 Indent

Spaceship

Filter


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
```

```
//

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

ide func shouldAutorotate() -> Bool {
eturn true

ide func supportedInterfaceOrientations() ->
IInterfaceOrientationMask {
f UIDevice.current().userInterfaceIdiom == .phone {
    return .allButUpsideDown
 else {
    return .all

ide func didReceiveMemoryWarning() {
uper.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

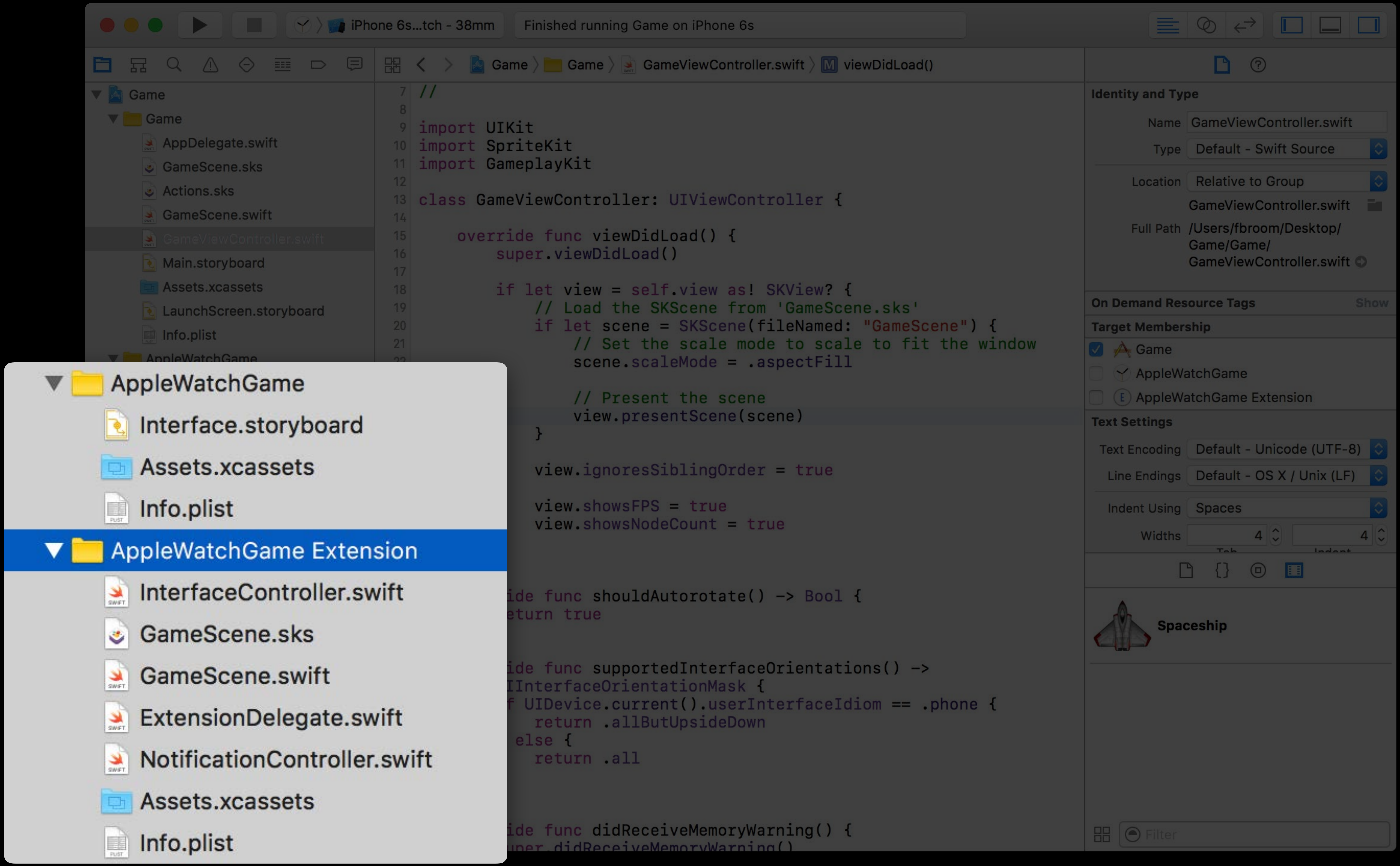
- [✓] Game
- [ ] AppleWatchGame
- [ ] AppleWatchGame Extension

Text Settings

Text Encoding: Default - Unicode (UTF-8)
Line Endings: Default - OS X / Unix (LF)
Indent Using: Spaces
Widths: 4 Tab, 4 Indent

Spaceship

```
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 〉 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

```
//

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
        }
    }

ide func shouldAutorotate() -> Bool {
eturn true

ide func supportedInterfaceOrientations() ->
IInterfaceOrientationMask {
f UIDevice.current().userInterfaceIdiom == .phone {
    return .allButUpsideDown
 else {
    return .all

ide func didReceiveMemoryWarning() {
uper.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

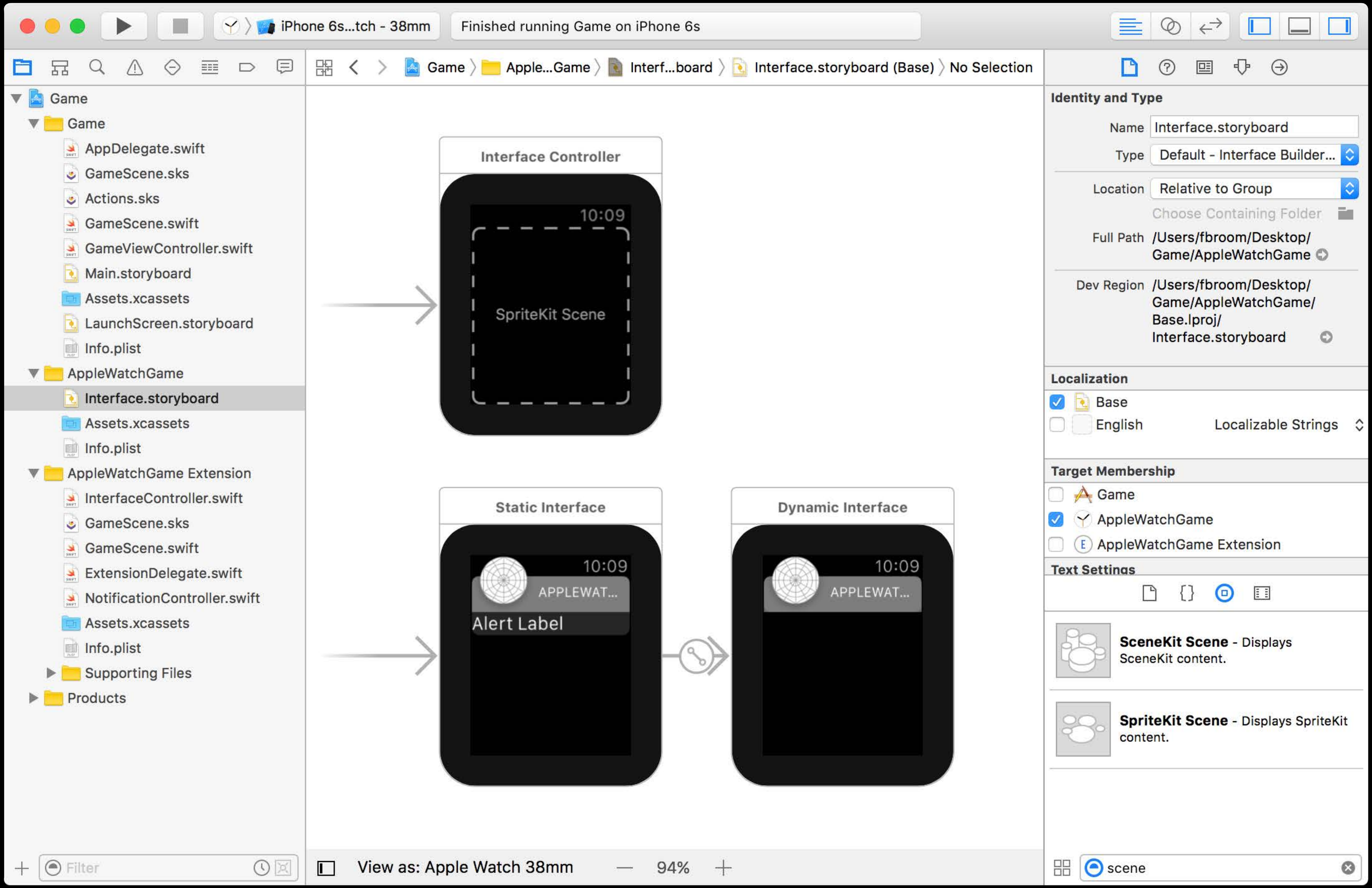
On Demand Resource Tags Show

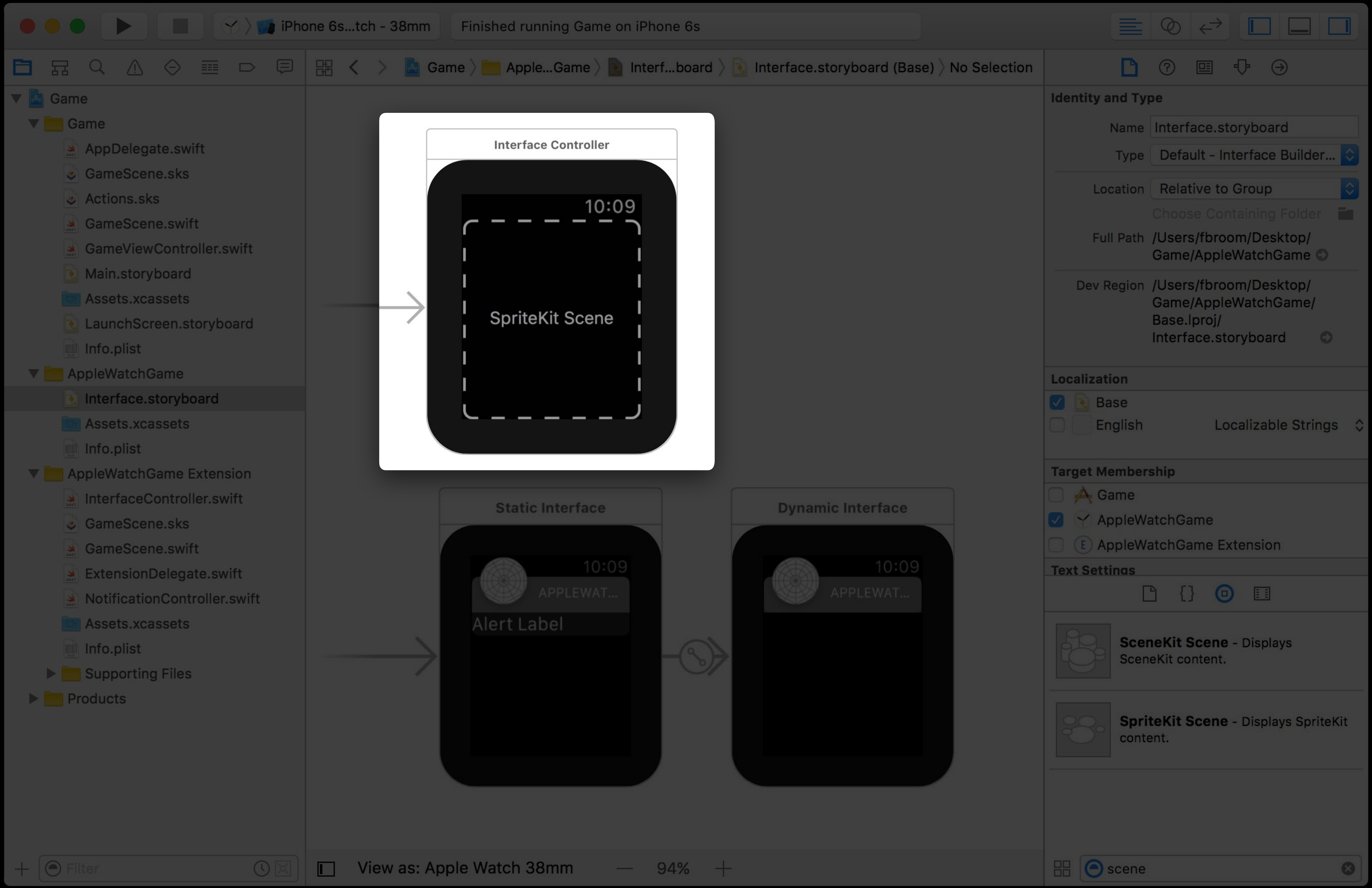
Target Membership

- ✓ Game
- AppleWatchGame
- AppleWatchGame Extension

Text Settings

Text Encoding	Default - Unicode (UTF-8)
Line Endings	Default - OS X / Unix (LF)
Indent Using	Spaces
Widths	4 Tab 4 Indent


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
Filter
Interface Controller
10:09
SpriteKit Scene
Static Interface
10:09
APPLEWAT...
Alert Label
Dynamic Interface
10:09
APPLEWAT...
View as: Apple Watch 38mm
94%
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.
scene



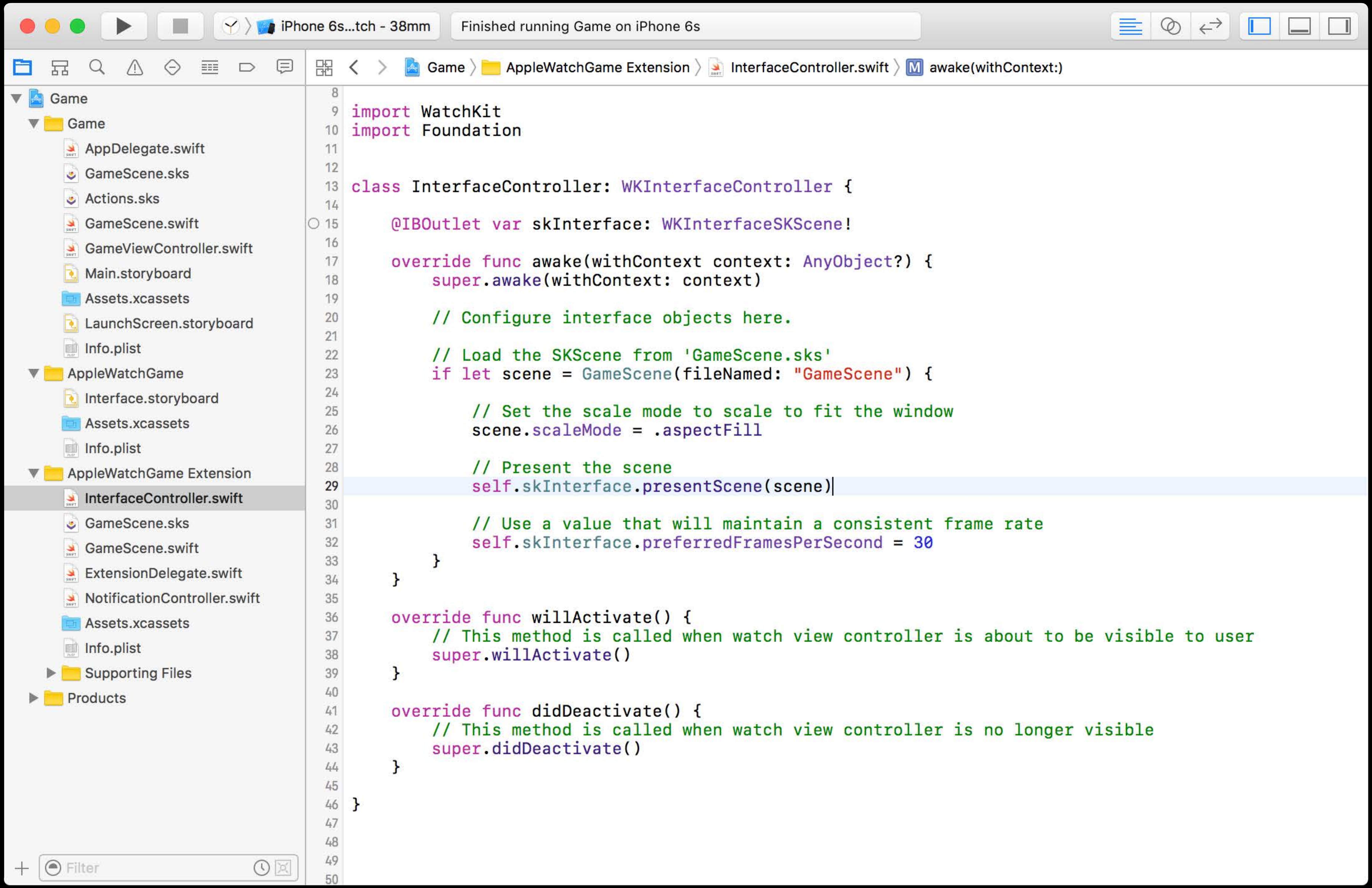
Name: Interface.storyboard
Type: Default - Interface Builder...
Location: Relative to Group
Choose Containing Folder
Full Path: /Users/fbroom/Desktop/Game/AppleWatchGame
Dev Region: /Users/fbroom/Desktop/Game/AppleWatchGame/Base.lproj/Interface.storyboard

Localization:
- [✓] Base
- [ ] English — Localizable Strings

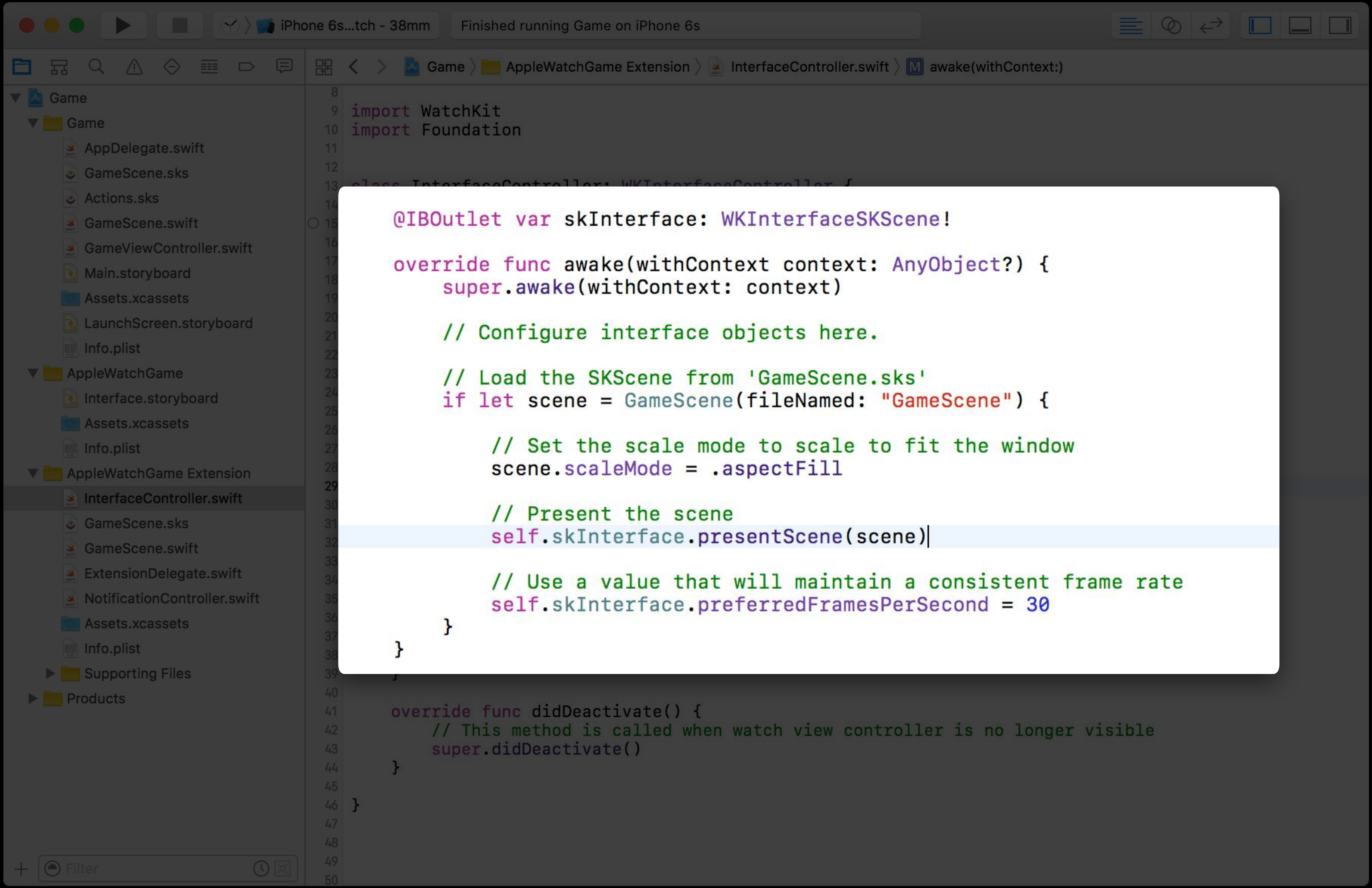
Target Membership:
- [ ] Game
- [✓] AppleWatchGame
- [ ] AppleWatchGame Extension

SceneKit Scene - Displays SceneKit content.

SpriteKit Scene - Displays SpriteKit content.



```swift
import WatchKit
import Foundation


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
        }
    }

    override func willActivate() {
        // This method is called when watch view controller is about to be visible to user
        super.willActivate()
    }

    override func didDeactivate() {
        // This method is called when watch view controller is no longer visible
        super.didDeactivate()
    }

}
```



```swift
import WatchKit
import Foundation

class InterfaceController: WKInterfaceController {
```

```swift
@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
    override func didDeactivate() {
        // This method is called when watch view controller is no longer visible
        super.didDeactivate()
    }

}
```

SpriteKit on Apple Watch

Compatibility

NEW

- Audio playback
    - SKAudioNode not supported
    - SKAction playSoundFileNamed
- Video playback
    - SKVideoNode not supported
    - WKInterfaceMovie
- Visual effects
    - SKEffectNode using CoreImage Filter
    - SKEffectNode using SKShader

SpriteKit on Apple Watch

NEW

Compatibility

Audio playback

ⓧ SKAudioNode not supported

SKAction playSoundFileNamed

Video playback

SKVideoNode not supported

WKInterfaceMovie

Visual effects

SKEffectNode using CoreImage Filter

SKEffectNode using SKShader

SpriteKit on Apple Watch

NEW

Compatibility

Audio playback

✗ SKAudioNode not supported

✓ SKAction playSoundFileNamed

Video playback

SKVideoNode not supported

WKInterfaceMovie

Visual effects

SKEffectNode using CoreImage Filter

SKEffectNode using SKShader

SpriteKit on Apple Watch

Compatibility

NEW

Audio playback

✗ SKAudioNode not supported

✓ SKAction playSoundFileNamed

Video playback

✗ SKVideoNode not supported

WKInterfaceMovie

Visual effects

SKEffectNode using CoreImage Filter

SKEffectNode using SKShader

SpriteKit on Apple Watch

Compatibility

NEW

Audio playback

✗ SKAudioNode not supported

✓ SKAction playSoundFileNamed

Video playback

✗ SKVideoNode not supported

✓ WKInterfaceMovie

Visual effects

SKEffectNode using CoreImage Filter

SKEffectNode using SKShader

SpriteKit on Apple Watch

NEW

Compatibility

Audio playback

✗ SKAudioNode not supported

✓ SKAction playSoundFileNamed

Video playback

✗ SKVideoNode not supported

✓ WKInterfaceMovie

Visual effects

✗ SKEffectNode using CoreImage Filter

SKEffectNode using SKShader

SpriteKit on Apple Watch

Compatibility

NEW

Audio playback

- ✗ SKAudioNode not supported
- ✓ SKAction playSoundFileNamed

Video playback

- ✗ SKVideoNode not supported
- ✓ WKInterfaceMovie

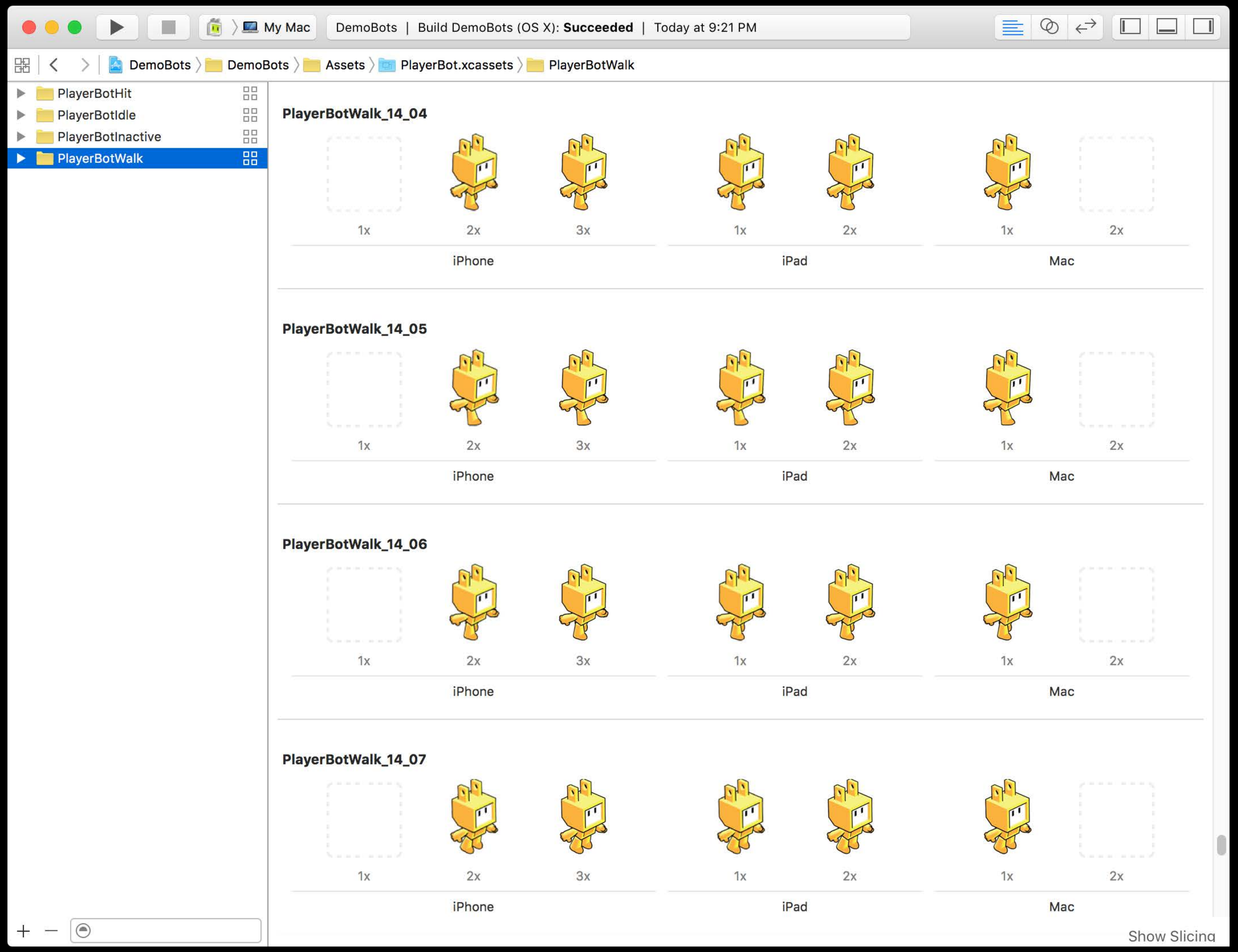
Visual effects

- ✗ SKEffectNode using CoreImage Filter
- ✓ SKEffectNode using SKShader

SpriteKit Best Practices

SpriteKit Tips & Tricks

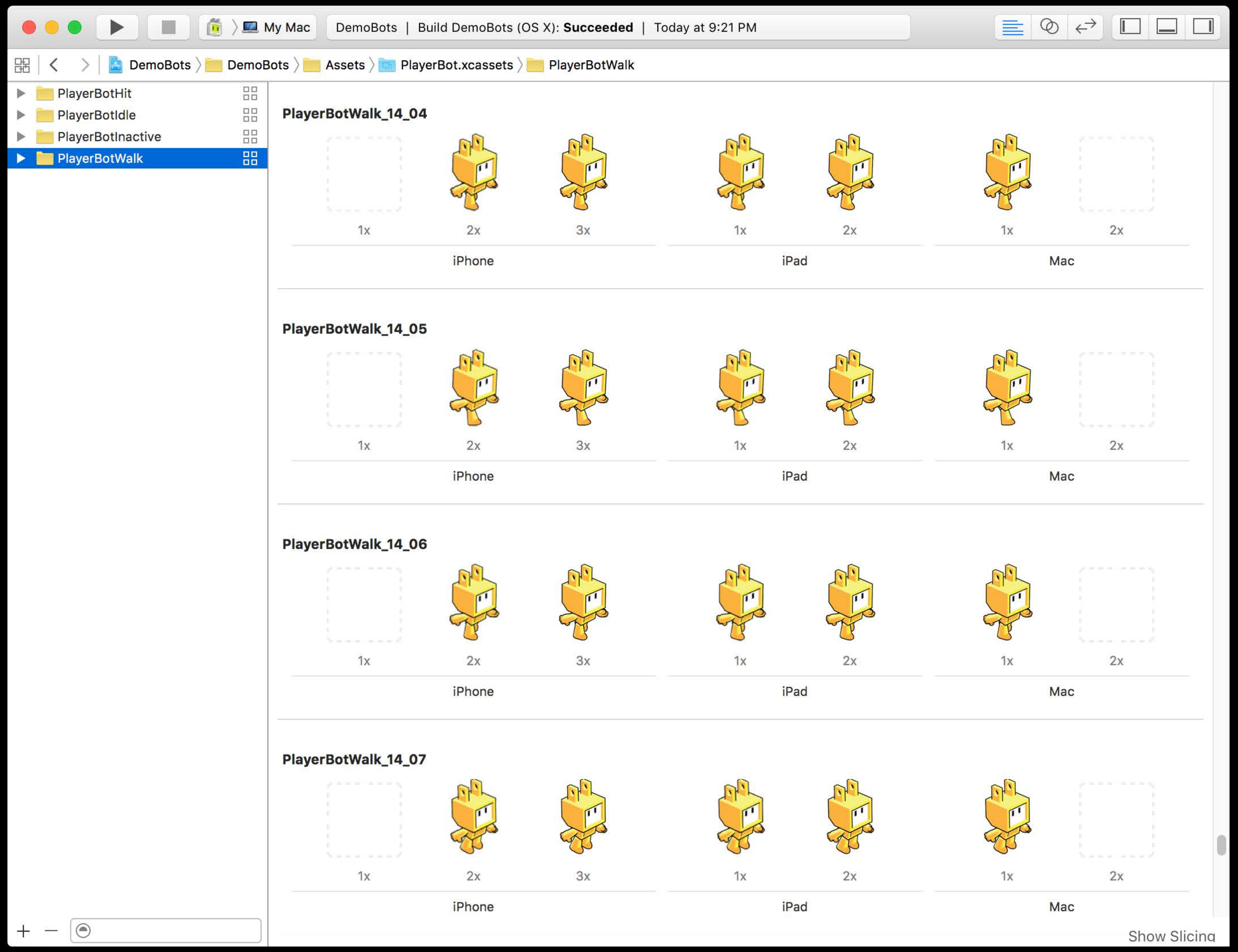
Asset Catalog



SpriteKit Tips & Tricks

Asset Catalog

SpriteKit is fully integrated with Asset Catalog

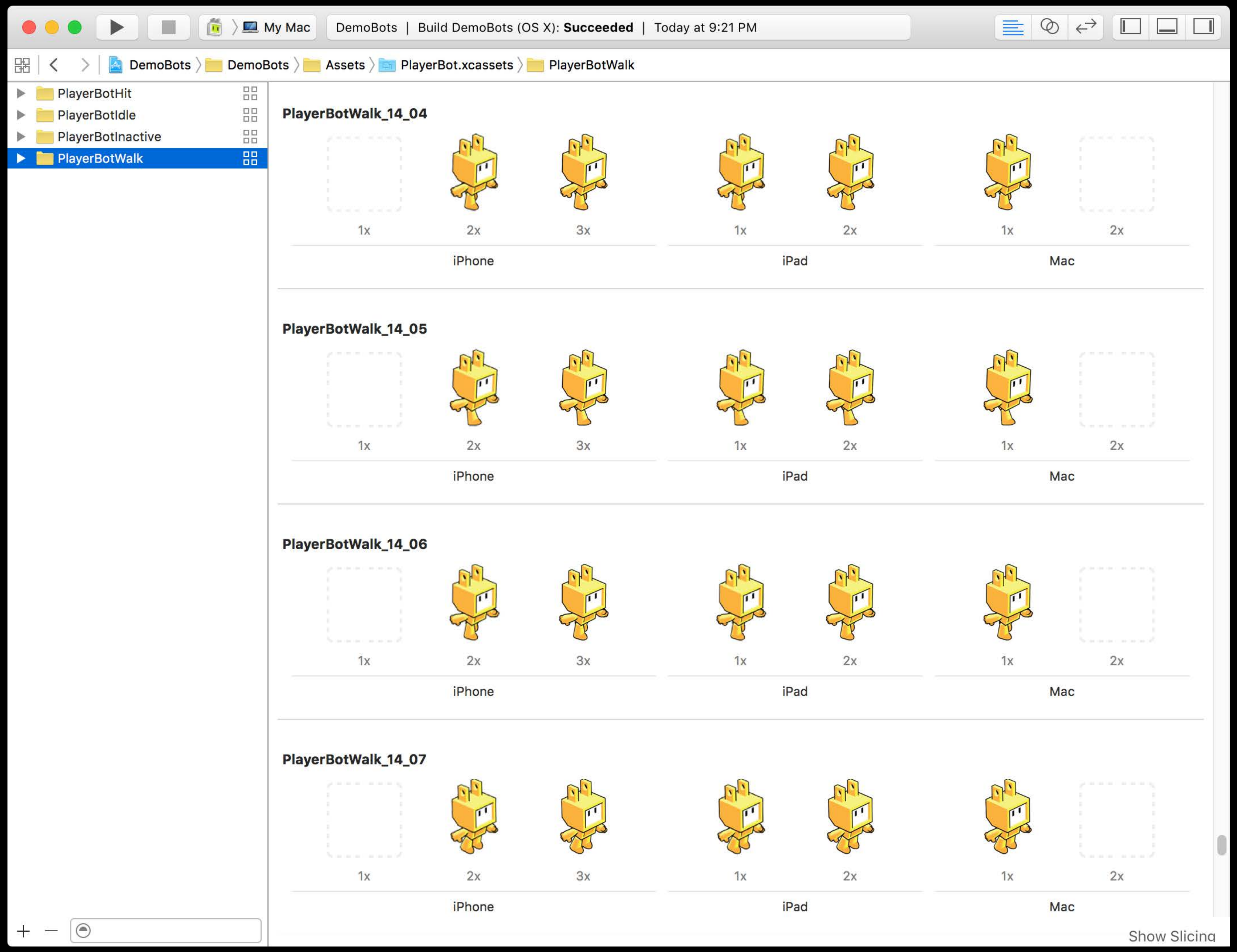


SpriteKit Tips & Tricks

Asset Catalog

SpriteKit is fully integrated with Asset Catalog

- Use sprite atlas for minimal draw calls

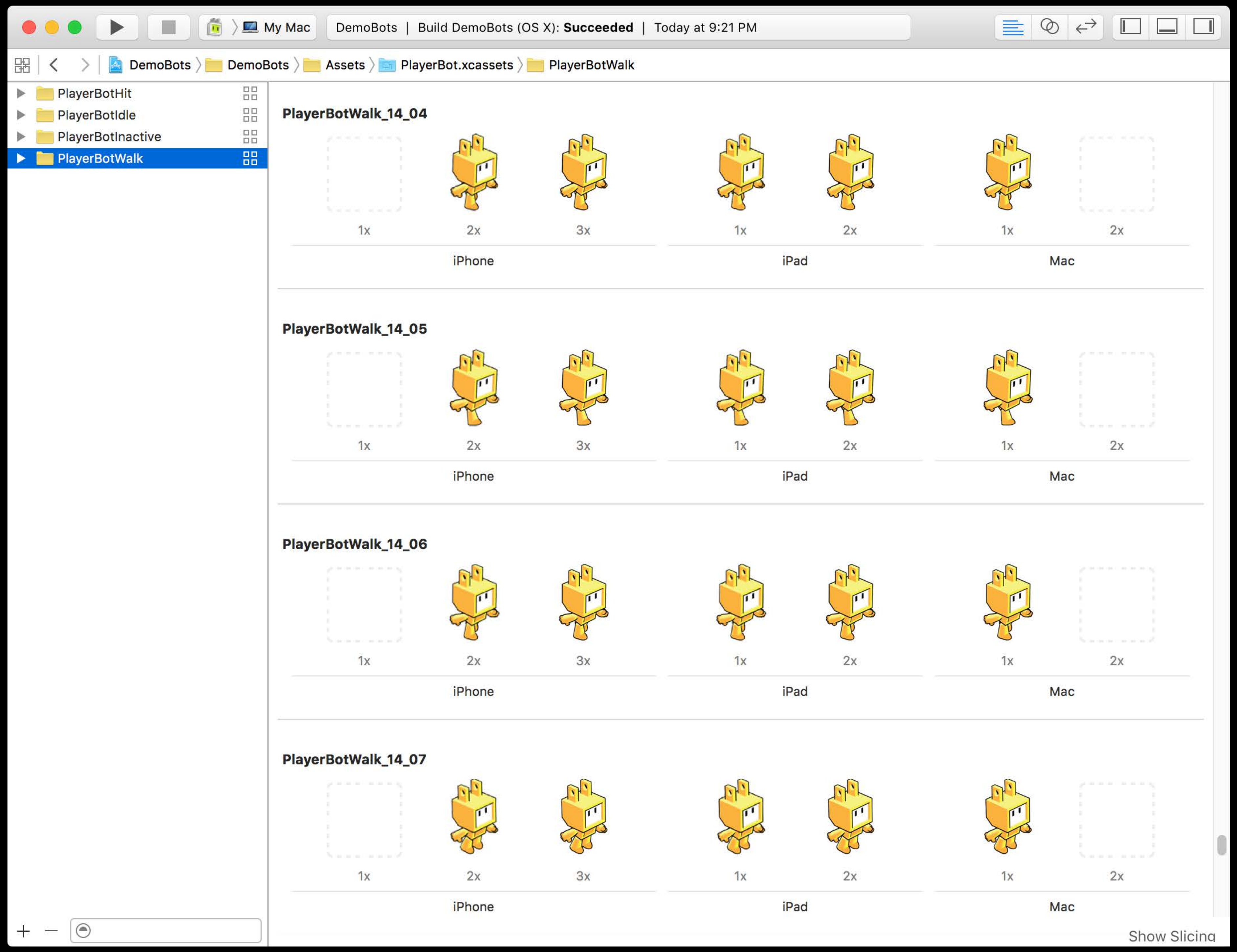


SpriteKit Tips & Tricks

Asset Catalog

SpriteKit is fully integrated with Asset Catalog

- Use sprite atlas for minimal draw calls
- Support assets of multiple size (1x, 2x, 3x)

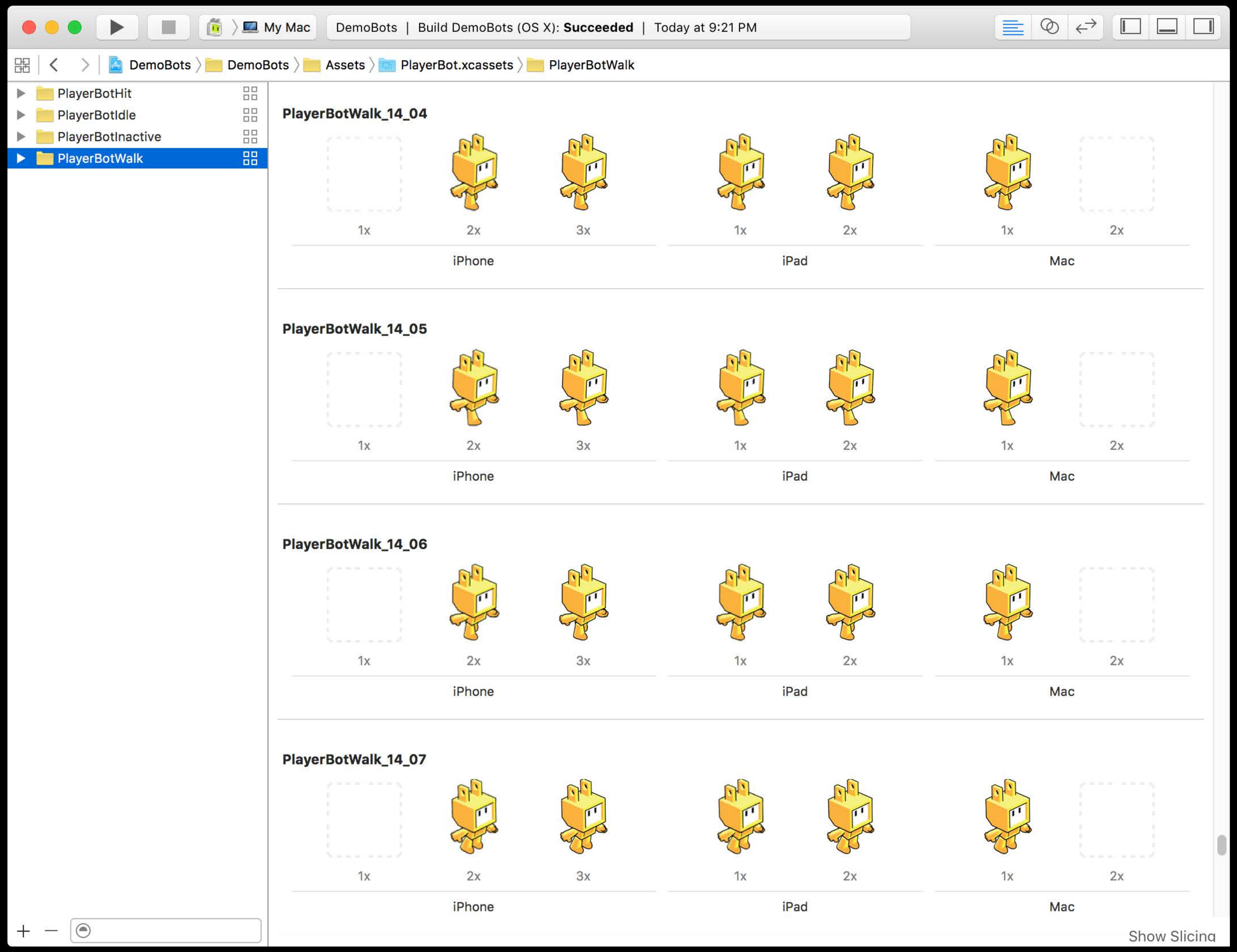


SpriteKit Tips & Tricks

Asset Catalog

SpriteKit is fully integrated with Asset Catalog

- Use sprite atlas for minimal draw calls
- Support assets of multiple size (1x, 2x, 3x)
- Support for On-Demand Resources (iOS, tvOS)

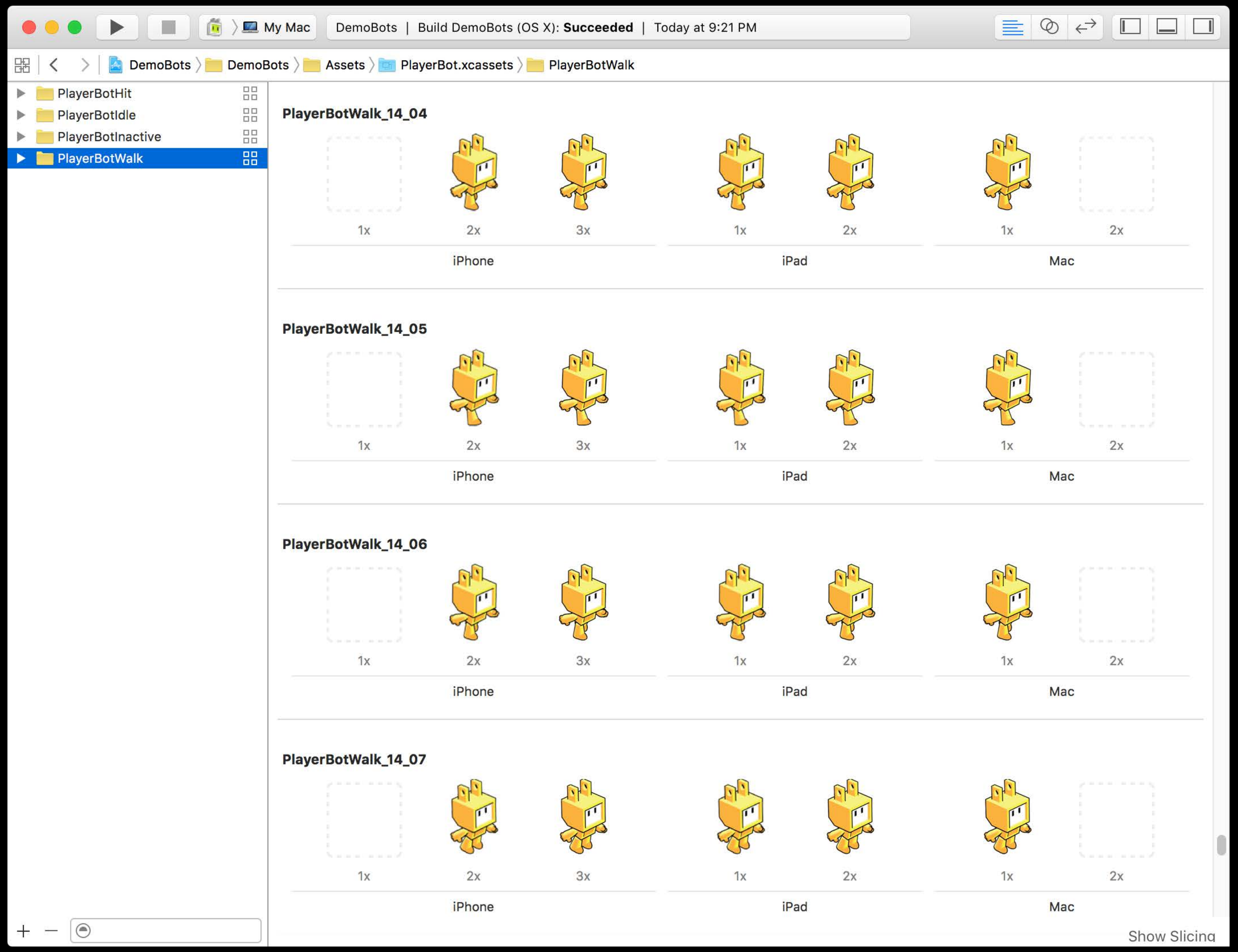


SpriteKit Tips & Tricks

Asset Catalog

SpriteKit is fully integrated with Asset Catalog

- Use sprite atlas for minimal draw calls
- Support assets of multiple size (1x, 2x, 3x)
- Support for On-Demand Resources (iOS, tvOS)
- Compiles necessary assets into runtime binary



SpriteKit Tips & Tricks

Performance

NEW

SpriteKit Tips & Tricks

Performance

NEW

Performance tuning and battery life improvements

SpriteKit Tips & Tricks

Performance

NEW

Performance tuning and battery life improvements

- SpriteKit now only renders when necessary

SpriteKit Tips & Tricks

Performance

NEW

Performance tuning and battery life improvements

- SpriteKit now only renders when necessary
- Additional ways to control the frame rate

SpriteKit Tips & Tricks

Performance

NEW

Performance tuning and battery life improvements

- SpriteKit now only renders when necessary
- Additional ways to control the frame rate

```
// Specify the desired FPS.
skView.preferredFramesPerSecond = 30

@objc public protocol SKViewDelegate : NSObjectProtocol {
    // Dynamically control the render rate.
    // - return YES to initiate an update and render for the target time.
    // - return NO to skip update and render for this target time.
    @objc public func view(_ view: SKView, shouldRenderAtTime time: TimeInterval) -> Bool
}
```

SpriteKit Tips & Tricks

Performance

NEW

Performance tuning and battery life improvements

- SpriteKit now only renders when necessary
- Additional ways to control the frame rate

```
// Specify the desired FPS.
skView.preferredFramesPerSecond = 30
```

```
@objc public protocol SKViewDelegate : NSObjectProtocol {
    // Dynamically control the render rate.
    // - return YES to initiate an update and render for the target time.
    // - return NO to skip update and render for this target time.
    @objc public func view(_ view: SKView, shouldRenderAtTime time: TimeInterval) -> Bool
}
```

Summary

Summary

What's New in SpriteKit

Summary

What's New in SpriteKit

- Scene Outline View

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge
- Tile Maps

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge
- Tile Maps
- Warp Transformation

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge
- Tile Maps
- Warp Transformation
- Per-Node Attributes for Custom Shaders

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge
- Tile Maps
- Warp Transformation
- Per-Node Attributes for Custom Shaders
- Focus Interaction on Apple TV

Summary

What's New in SpriteKit

- Scene Outline View
- GameplayKit Integration
- FPS Performance Gauge
- Tile Maps
- Warp Transformation
- Per-Node Attributes for Custom Shaders
- Focus Interaction on Apple TV
- SpriteKit on Apple Watch

More Information

https://developer.apple.com/wwdc16/610

Related Sessions

Go Live with ReplayKit	Mission	Tuesday 10:00AM
Focus Interaction on tvOS	Mission	Wednesday 4:00PM
Visual Debugging with Xcode	Presidio	Wednesday 4:00PM
Controlling Game Input for Apple TV	Mission	Wednesday 5:00PM
What’s New in GameplayKit	Pacific Heights	Thursday 9:00AM
Advances in SceneKit Rendering	Mission	Thursday 11:00AM

Related Sessions

What's New in Game Center	Mission	Friday 10:00AM
Game Technologies for Apple Watch	Mission	Friday 3:00PM

Labs

Game Center Lab	Graphics, Games, and Media Lab A	Friday 12:00PM
SpriteKit Lab	Graphics, Games, and Media Lab B	Friday 12:00PM
watchOS Graphics and Games Lab	Graphics, Games, and Media Lab B	Friday 4:00PM

WWDC16