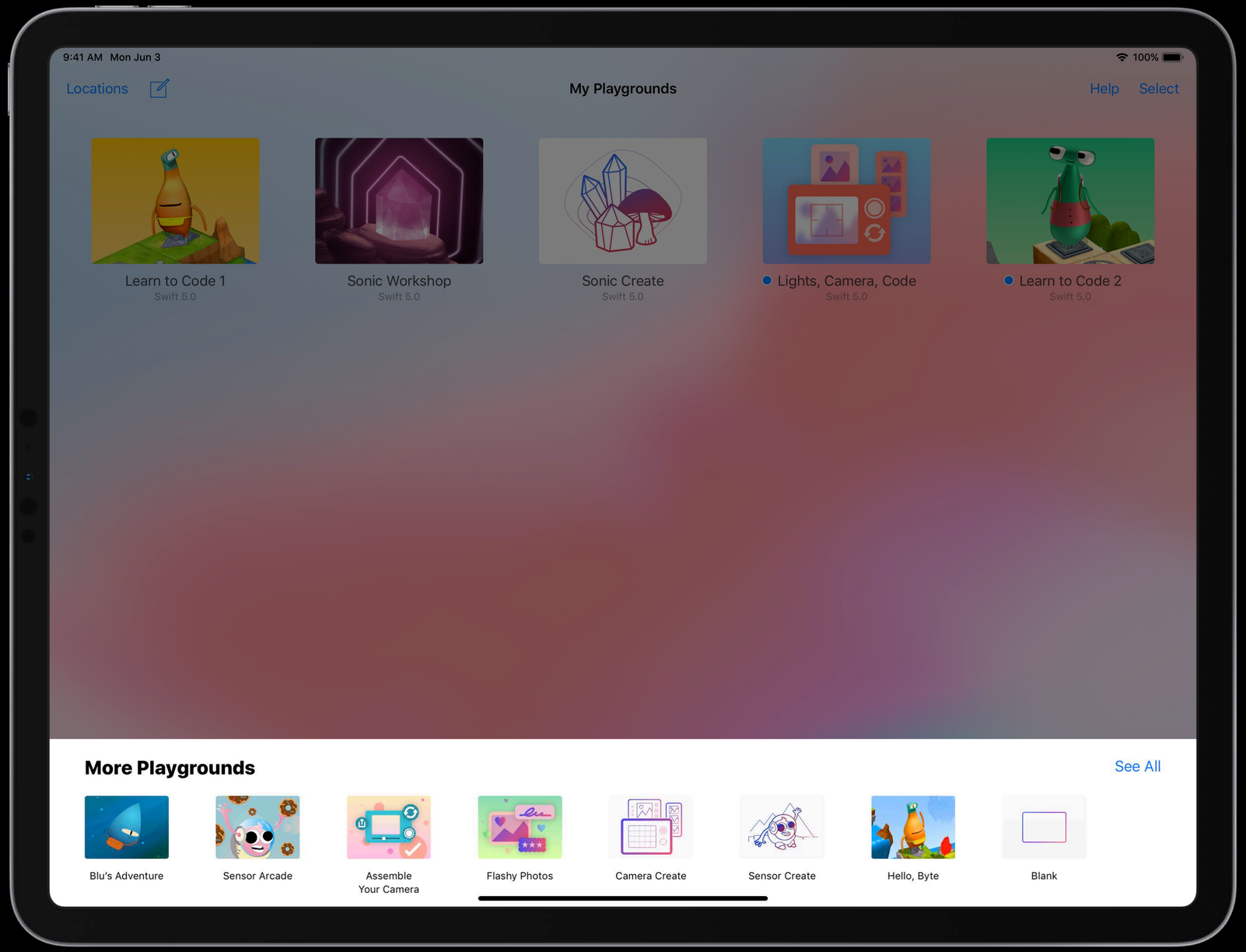
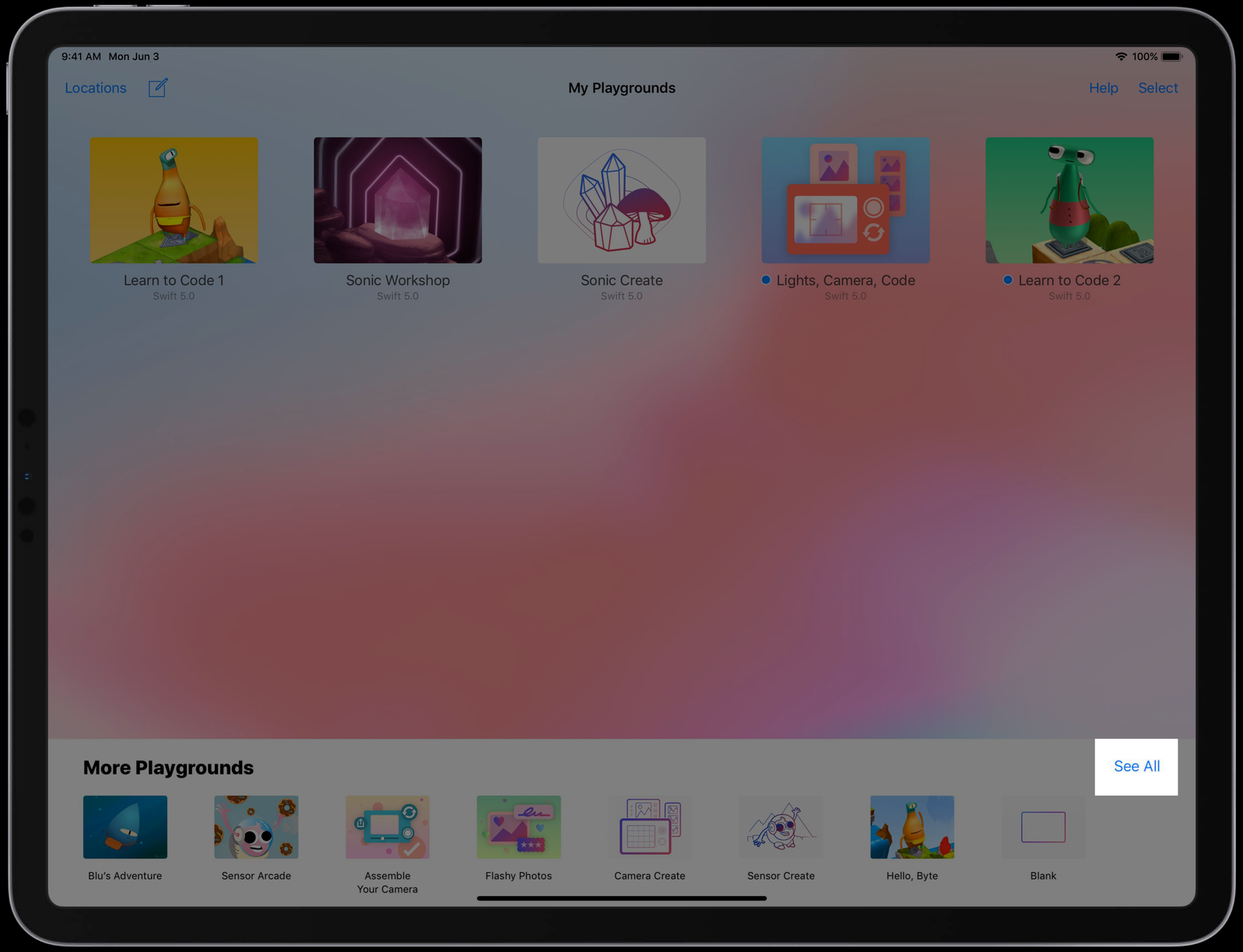
#WWDC19

Swift Playgrounds 3

Jonathan Penn, Playgrounds Engineer
Grace Kendall, Playgrounds Engineer
Joy Forbes, Developer Education Engineer


9:41 AM Mon Jun 3
100%
Locations
My Playgrounds
Help
Select
Learn to Code 1
Swift 5.0
Sonic Workshop
Swift 5.0
Sonic Create
Swift 5.0
Lights, Camera, Code
Swift 5.0
Learn to Code 2
Swift 5.0
More Playgrounds
See All
Blu's Adventure
Sensor Arcade
Assemble Your Camera
Flashy Photos
Camera Create
Sensor Create
Hello, Byte
Blank


9:41 AM Mon Jun 3
100%
Locations
My Playgrounds
Help
Select
Learn to Code 1
Swift 5.0
Sonic Workshop
Swift 5.0
Sonic Create
Swift 5.0
Lights, Camera, Code
Swift 5.0
Learn to Code 2
Swift 5.0
More Playgrounds
See All
Blu's Adventure
Sensor Arcade
Assemble Your Camera
Flashy Photos
Camera Create
Sensor Create
Hello, Byte
Blank


9:41 AM Mon Jun 3
100%
Locations
My Playgrounds
Help
Select
Learn to Code 1
Swift 5.0
Sonic Workshop
Swift 5.0
Sonic Create
Swift 5.0
Lights, Camera, Code
Swift 5.0
Learn to Code 2
Swift 5.0
More Playgrounds
See All
Blu's Adventure
Sensor Arcade
Assemble Your Camera
Flashy Photos
Camera Create
Sensor Create
Hello, Byte
Blank

9:41 AM Mon Jun 3

100%

More Playgrounds

Edit Done

Learn to Code

Learn to Code 1

Fundamentals of Swift

Swift 5.0 Edition

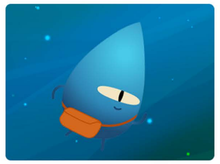
GET

Learn to Code 2

Beyond the Basics

Swift 5.0 Edition

GET

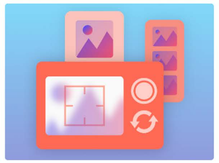


Blu's Adventure

Exploring graphics, coordinates, and touch events

Swift 5.0 Edition

GET

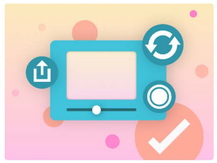


Lights, Camera, Code

Combine the fun of taking photos and writing code

Swift 5.0 Edition

GET

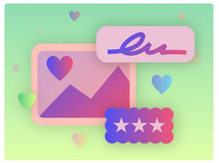


Assemble Your Camera

Design the camera of your dreams and then build it from scratch

Swift 5.0 Edition

GET



Flashy Photos

Design and build your own photo editor

Swift 5.0 Edition

GET

Challenges

See All

Sonic Workshop

Intermediate

Swift 5.0 Edition

GET

Code Machine

Beginner

Swift 5.0 Edition

GET

Augmented Reality

Intermediate

Swift 5.0 Edition

GET

Battle

Intermedi

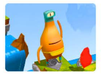
Swift 5.0

Sensor Arcade

Intermediate

Swift 5.0 Edition

GET



Hello, Byte

Beginner

Swift 5.0 Edition

GET

Cipher

Intermediate

Swift 5.0 Edition

GET

Brick

Intermedi

Swift 5.0

Starting Points

See All

Blank

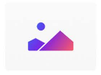
Swift 5.0 Edition

GET

Sonic Create

Swift 5.0 Edition

GET



Shapes

Swift 5.0 Edition

GET

Graph

Swift 5.0

Camera Create

Swift 5.0 Edition

GET

Sensor Create

Swift 5.0 Edition

GET

Answers

Swift 5.0 Edition

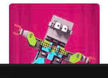
GET

Puzzle

Swift 5.0

MeeBot Playgrounds THIRD PARTY

See All



MeeBot Dances

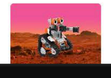
Code your own dance moves for MeeBo...

GET

MeeBot Template

Code your own dance moves for MeeBo...

GET



AstroBot Mars Mission

Control AstroBot to repair the Mars base

GET

9:41 AM Mon Jun 3 100%

Starting Points

See All

Blank
Swift 5.0 Edition
GET

Sonic Create
Swift 5.0 Edition
GET

Shapes
Swift 5.0 Edition
GET

Graph
Swift 5.0

Camera Create
Swift 5.0 Edition
GET

Sensor Create
Swift 5.0 Edition
GET

Answers
Swift 5.0 Edition
GET

Puzzle
Swift 5.0

MeeBot Playgrounds THIRD PARTY

See All

MeeBot Dances
Code your own dance moves for MeeBo...
GET

MeeBot Template
Code your own dance moves for MeeBo...
GET

AstroBot Mars Mission
Control AstroBot to repair the Mars base
GET

Sphero THIRD PARTY

See All

Sphero Arcade 1
Code classic games with Sphero.
GET

Sphero Arcade 2
Code classic games with Sphero.
GET

Sphero Arcade 3
Code classic games with Sphero.
GET

Spher
Build your

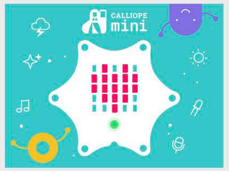
From Other Publishers

Playgrounds from leading developers, educators, and robotics engineers

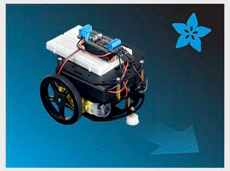
LEGO MINDSTORMS EV3
Build, code, and learn with LEGO® Education and Swift Pla...

Tello by Ryze
Program Tello drones with Swift.

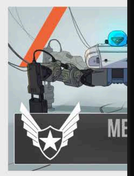
Makeblock
Program with the Makeblock Explorer Kit, mBot, or Codey Ro...



Calliope mini
Control the Calliope mini lights, pins, buttons, and sensors.



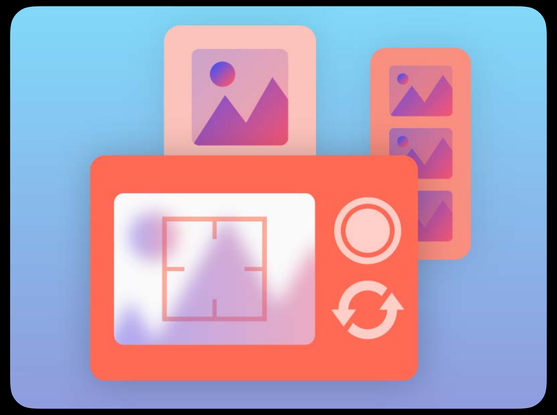
Adafruit
Control your own Adafruit robot using Bluetooth.



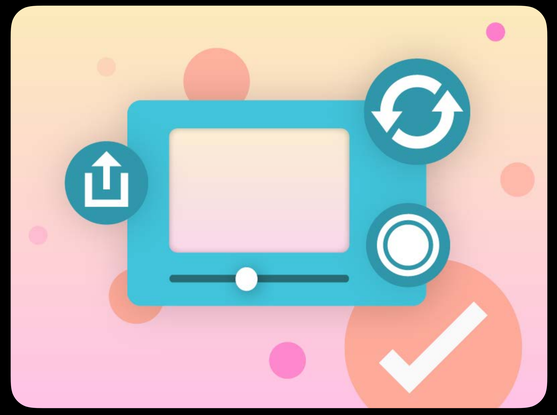
Mekamon
The MekAcademy
code with Mekamo

All content from these publishers is provided for free, and without warranty from Apple.

Enter a Subscription URL

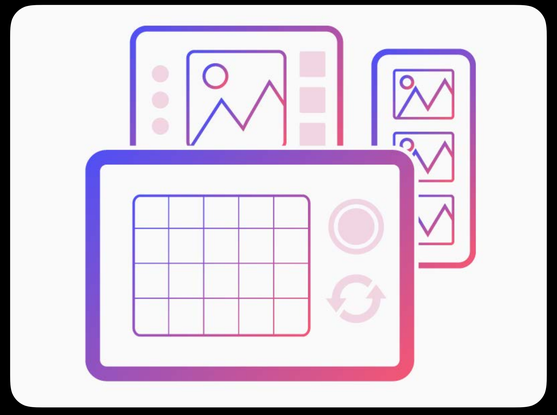


Lights, Camera, Code

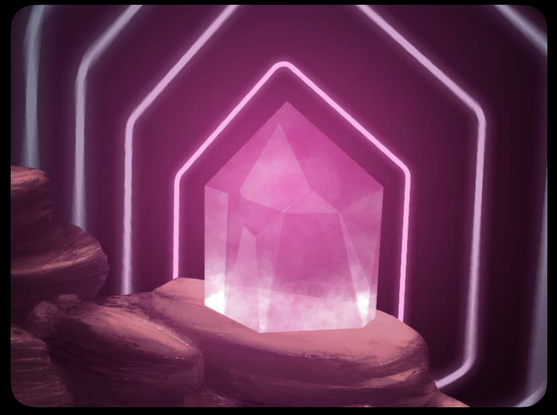


Assemble Your Camera

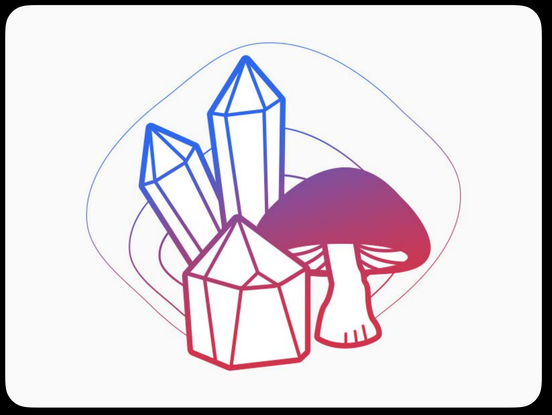
Flashy Photos



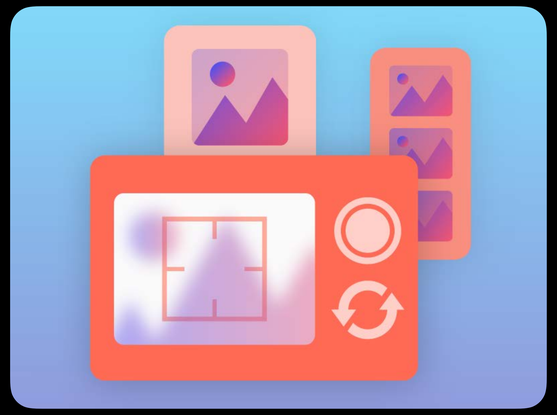
Camera Create



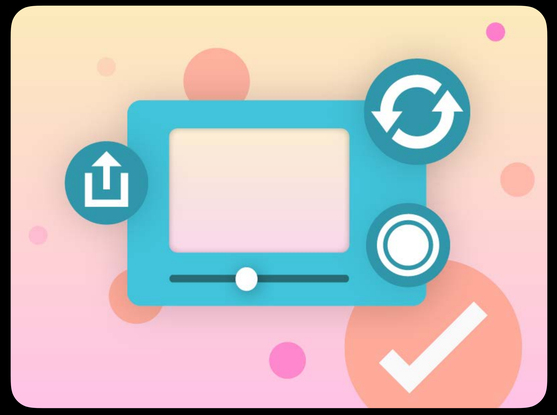
Sonic Workshop



Sonic Create

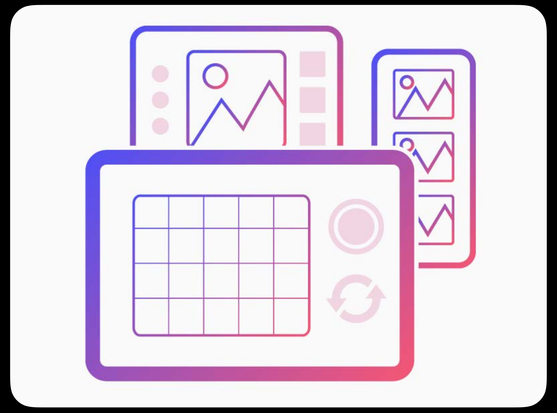


Lights, Camera, Code

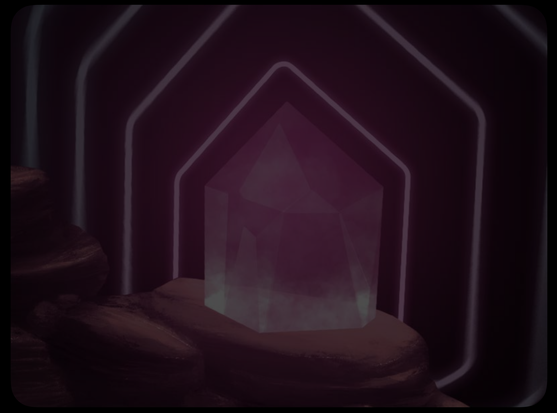


Assemble Your Camera

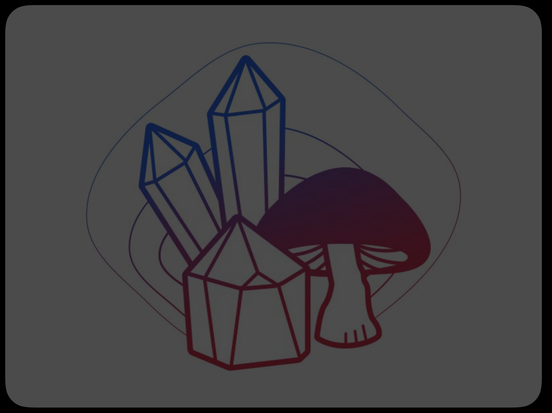
Flashy Photos



Camera Create



Sonic Workshop



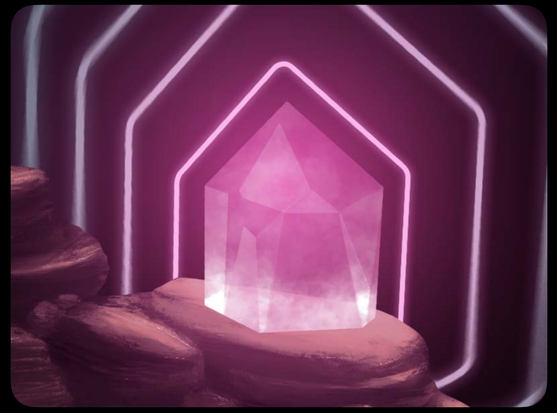
Sonic Create

Lights, Camera, Code

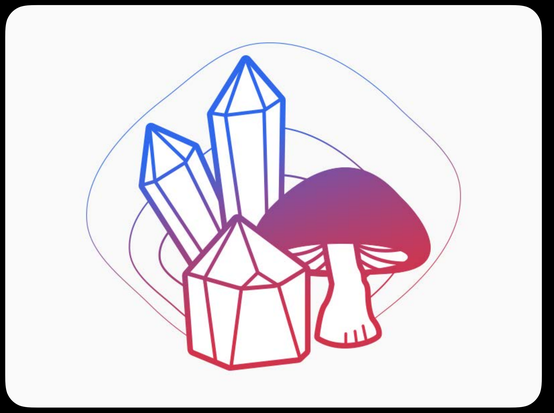
Assemble Your Camera

Flashy Photos

Camera Create



Sonic Workshop



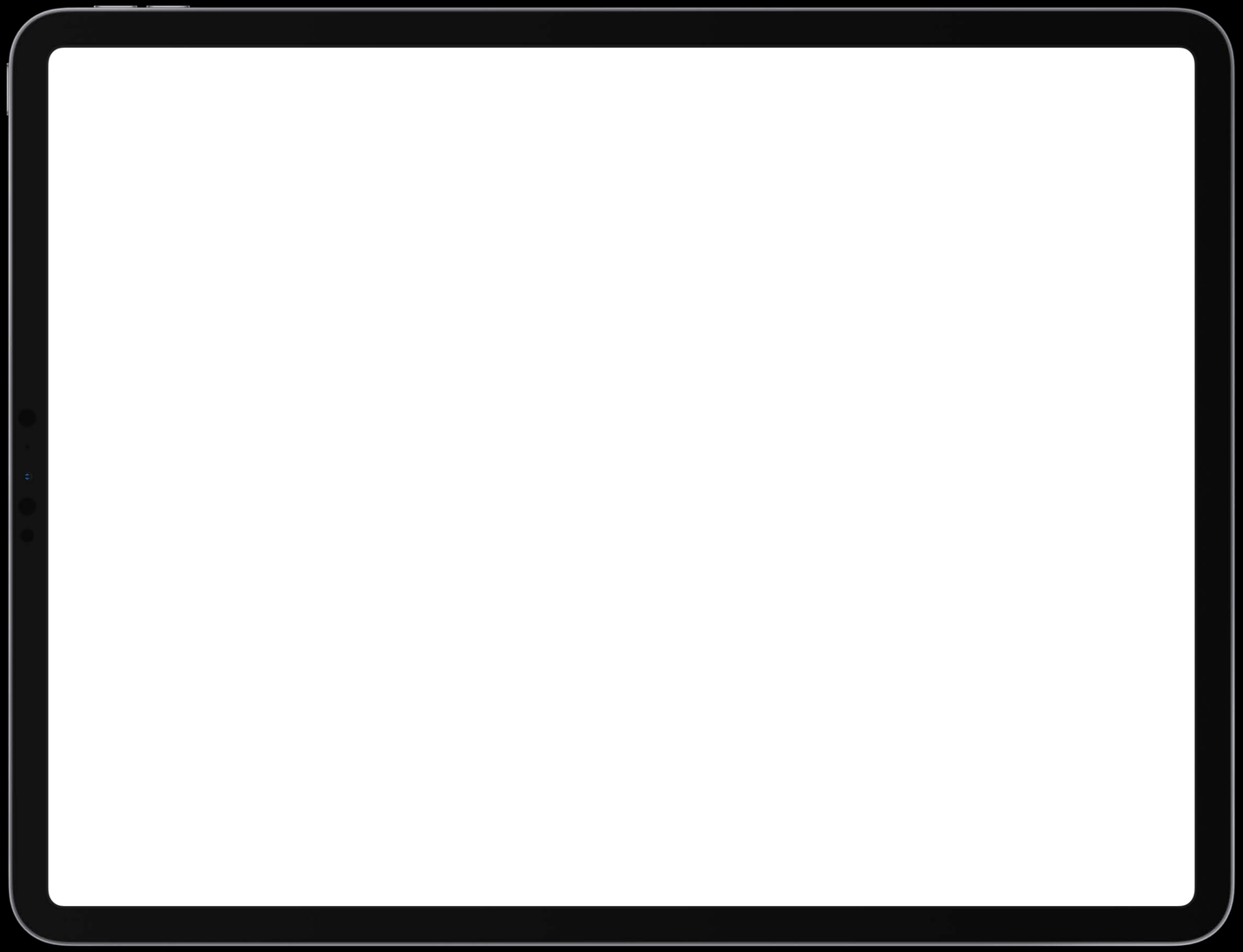
Sonic Create

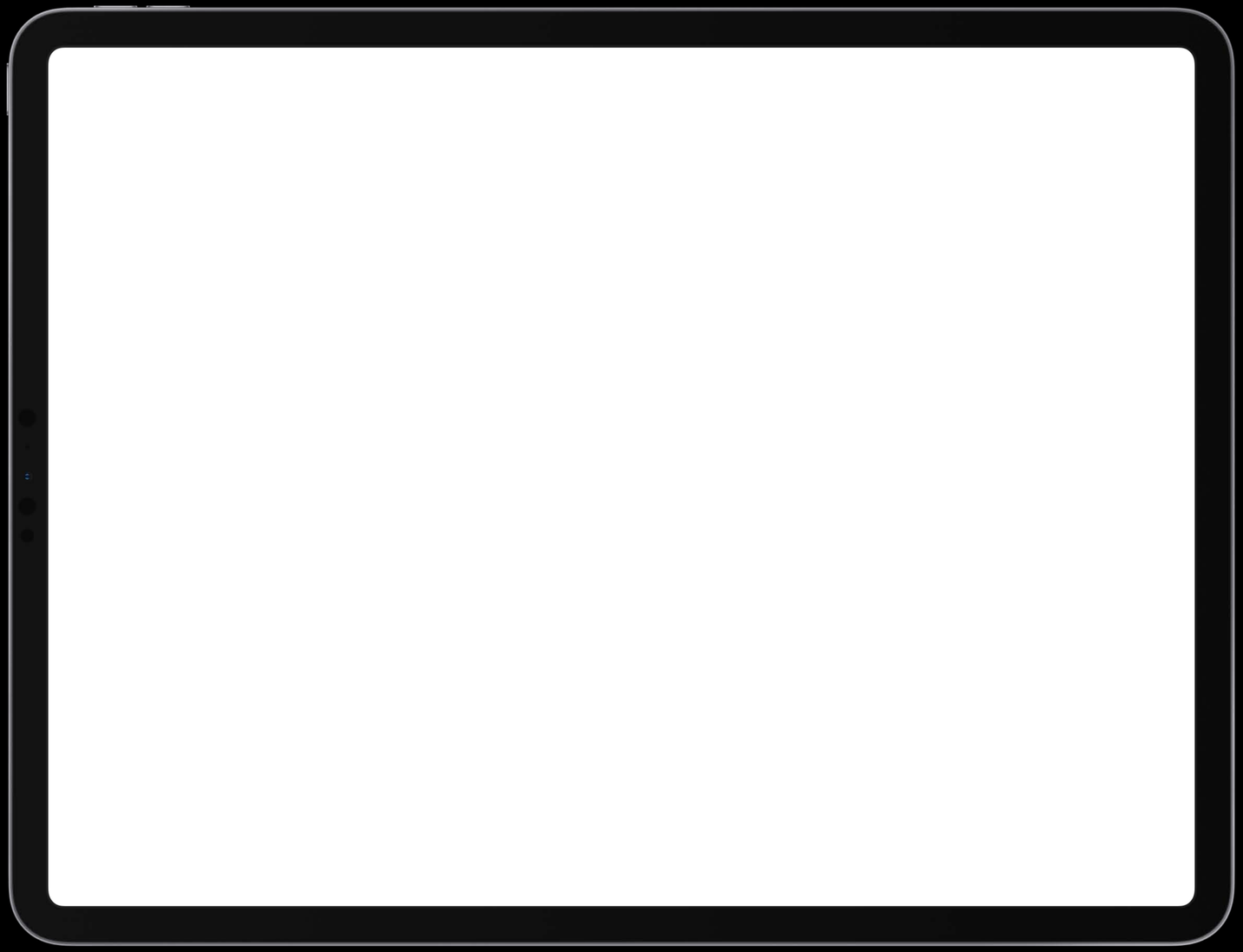
Demo — User Modules

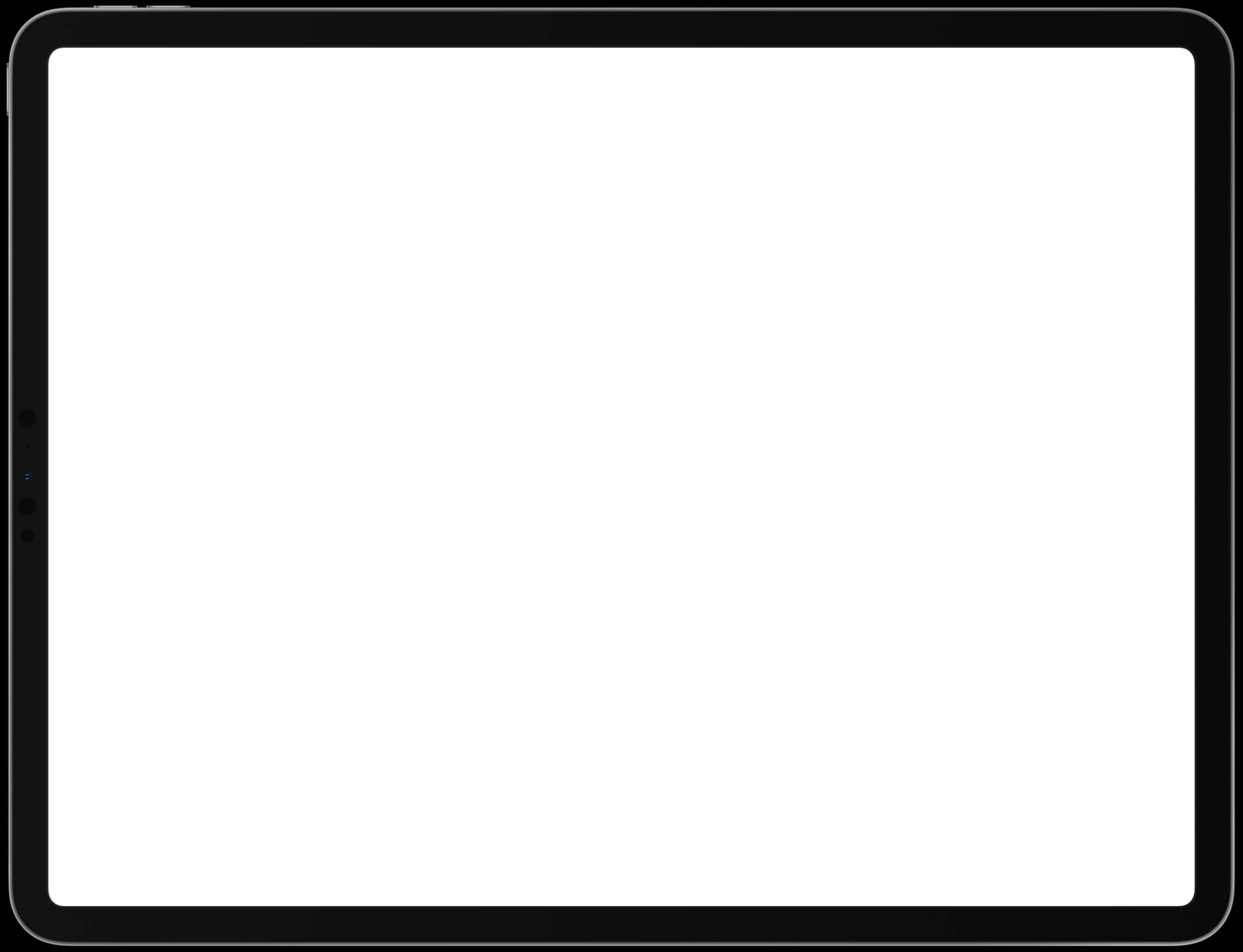
iPad Capabilities

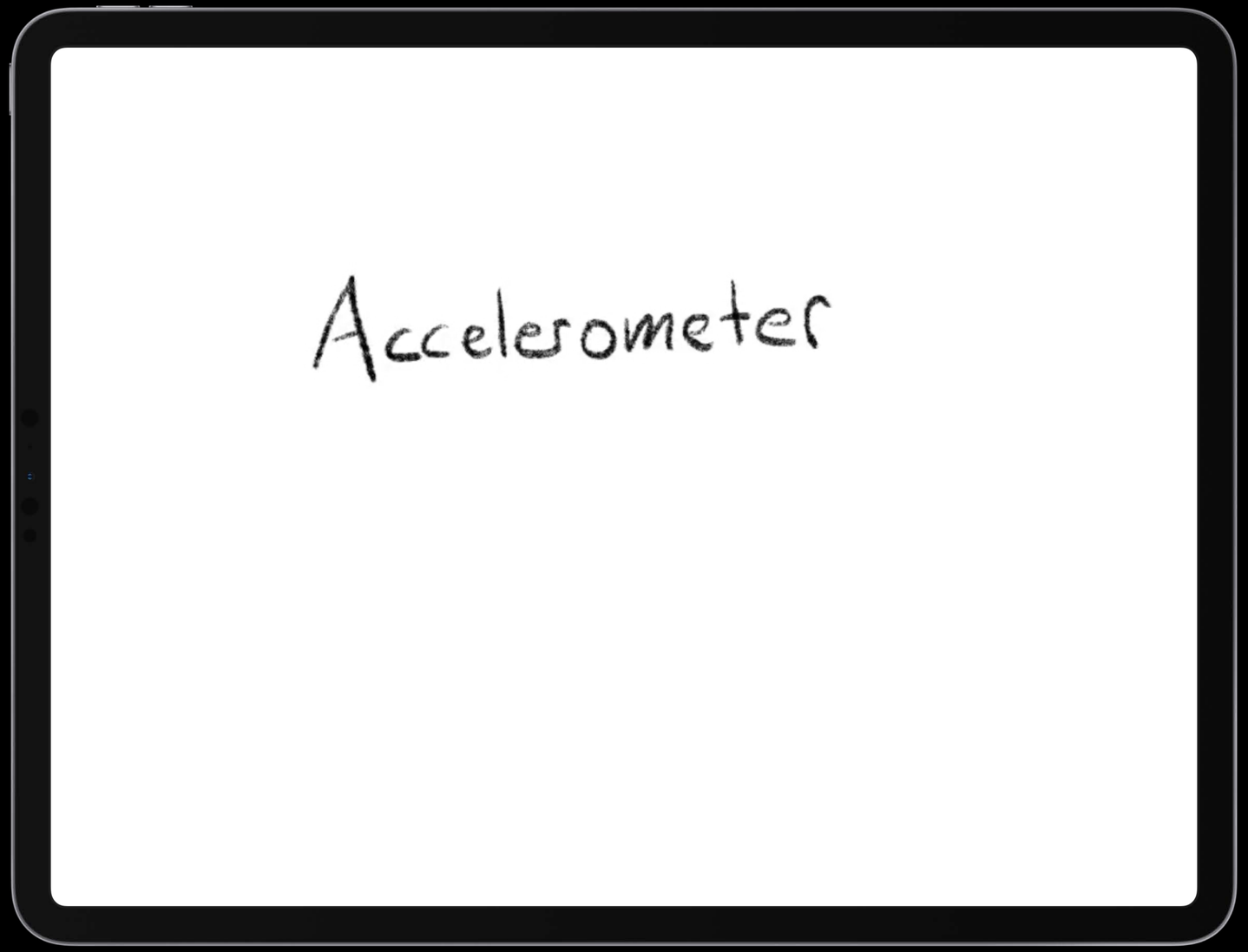
Authoring on the Mac

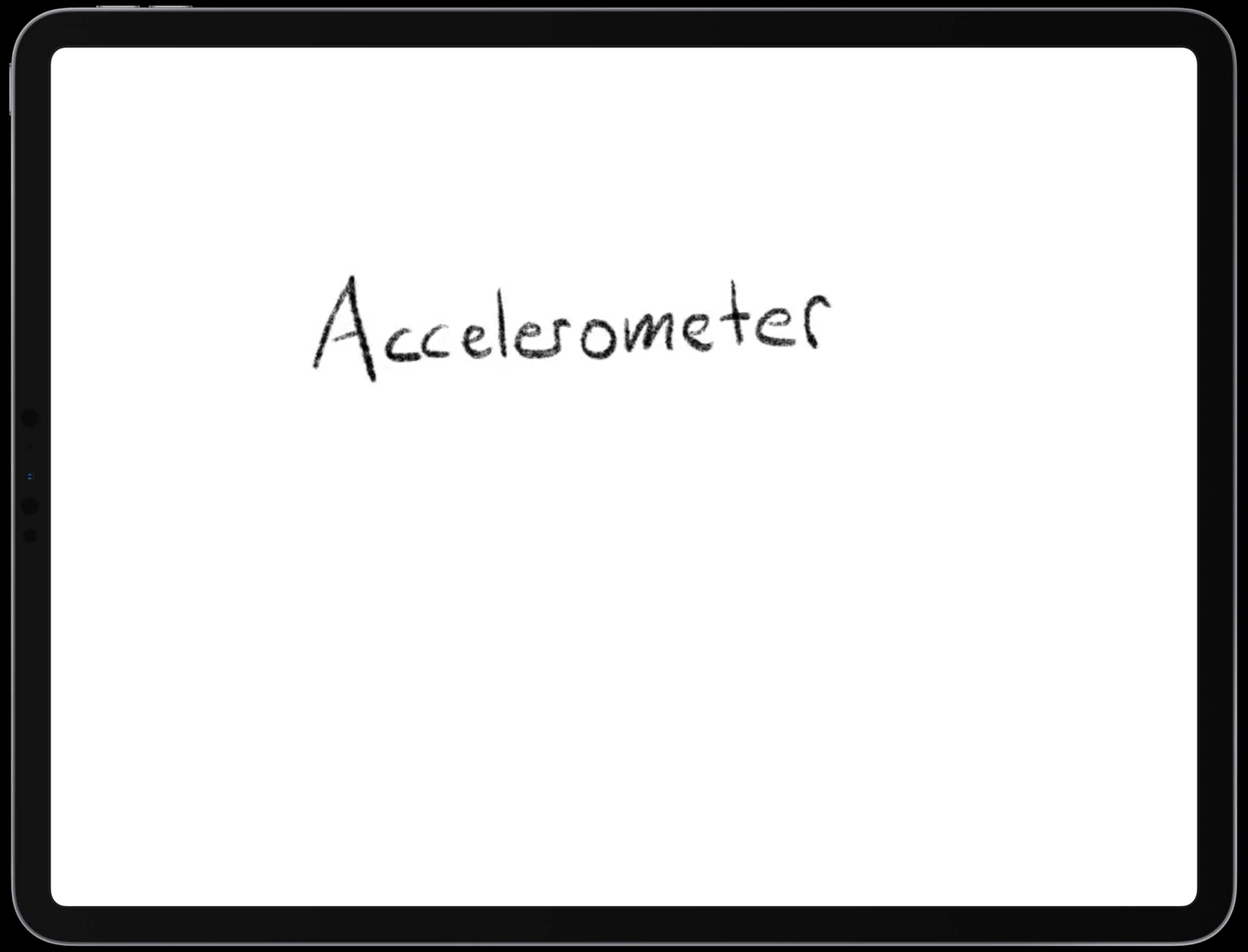
Demo — Vision and Core ML








Accelerometer


Accelerometer


Accelerometer
Games


Accelerometer
Games

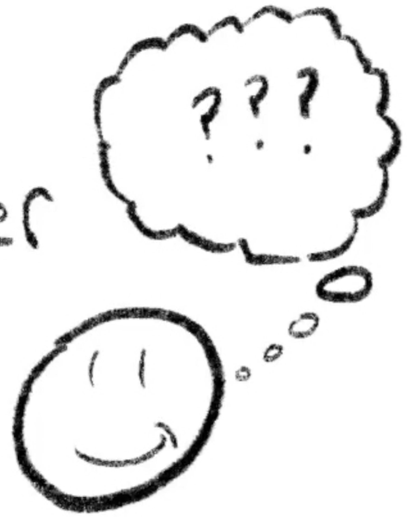

Accelerometer
Games
Physicality


Accelerometer
Games
Physicality

Accelerometer

Games

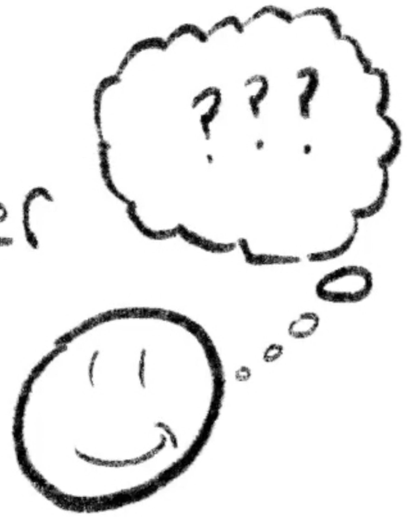
Physicality

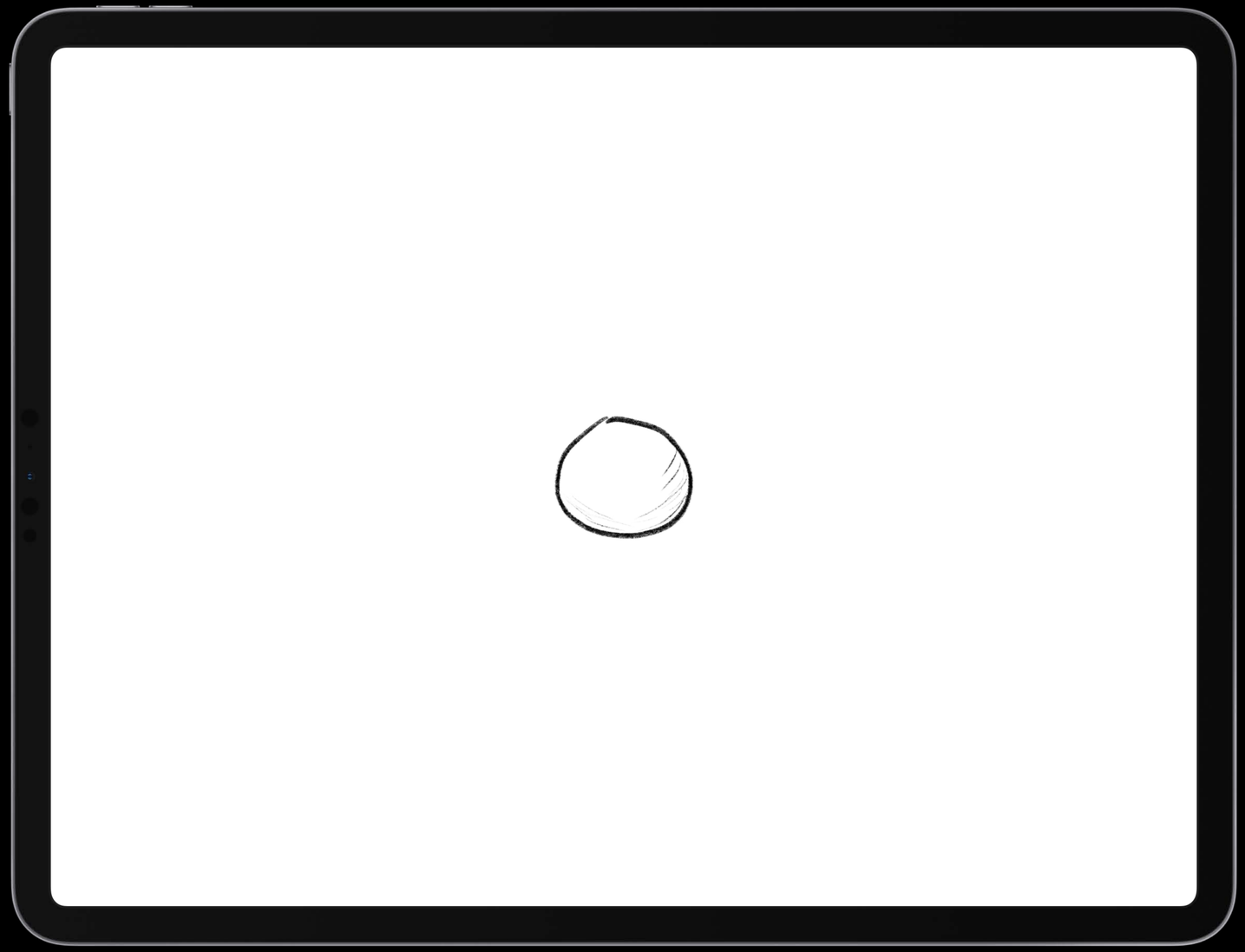


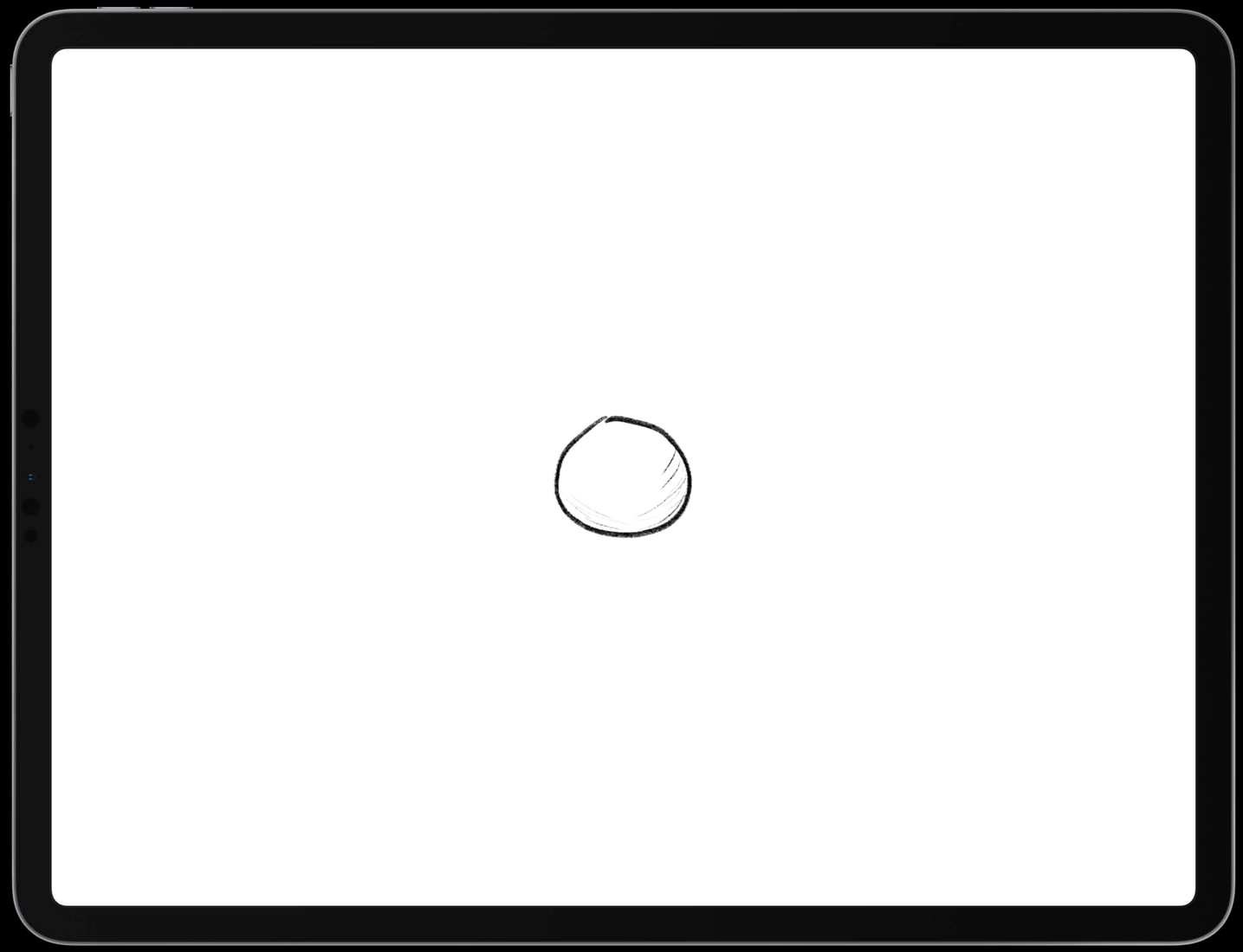
Accelerometer

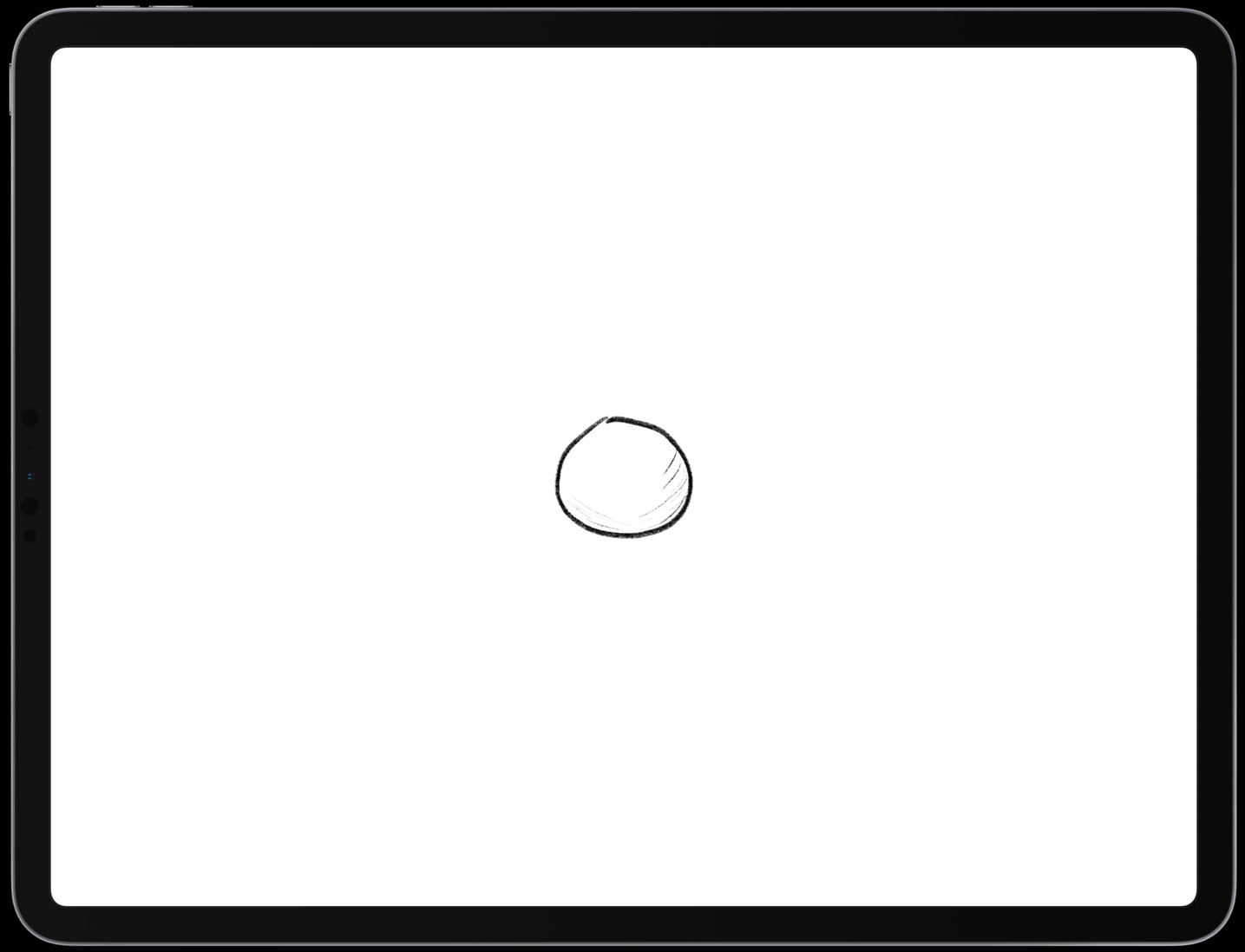
Games

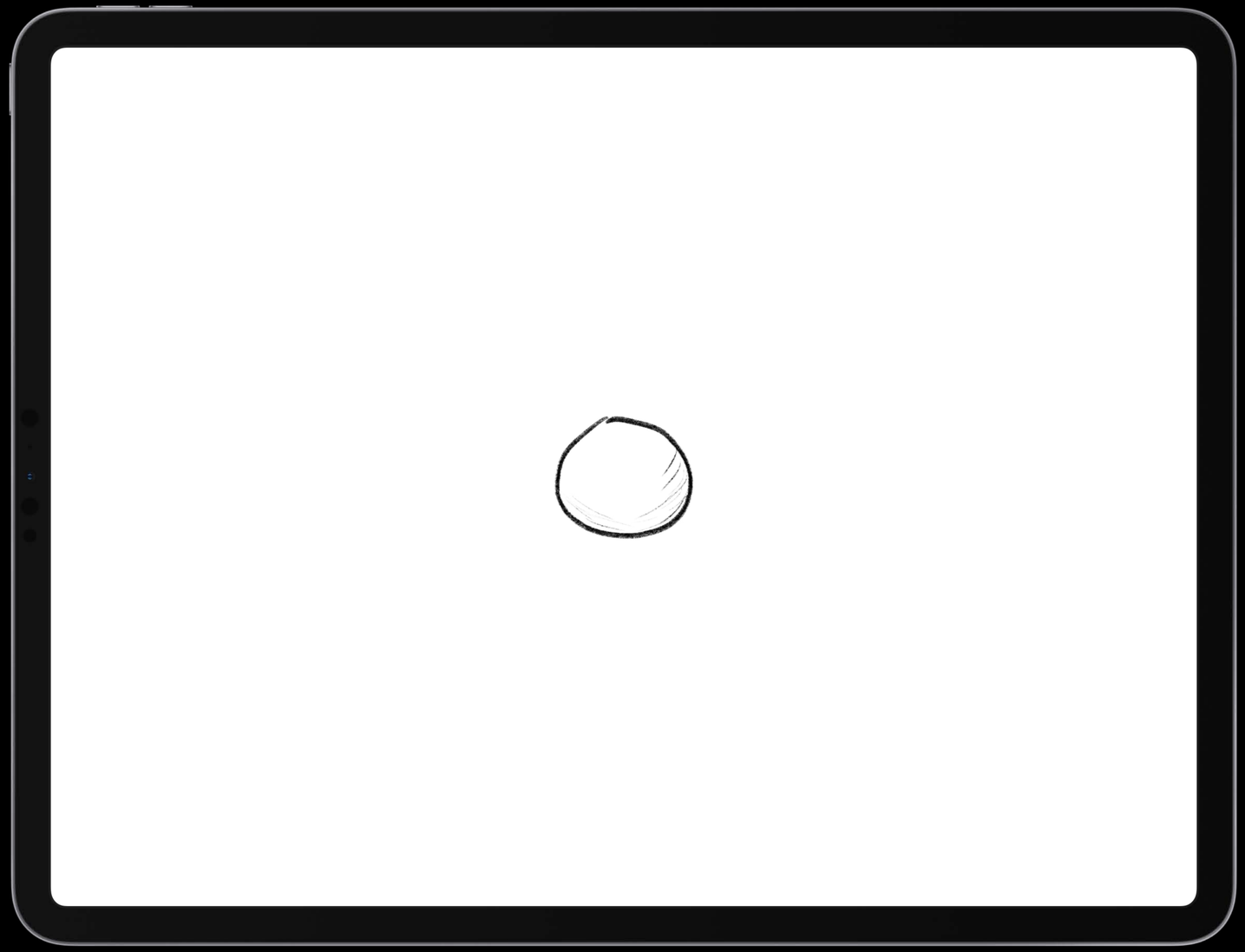
Physicality

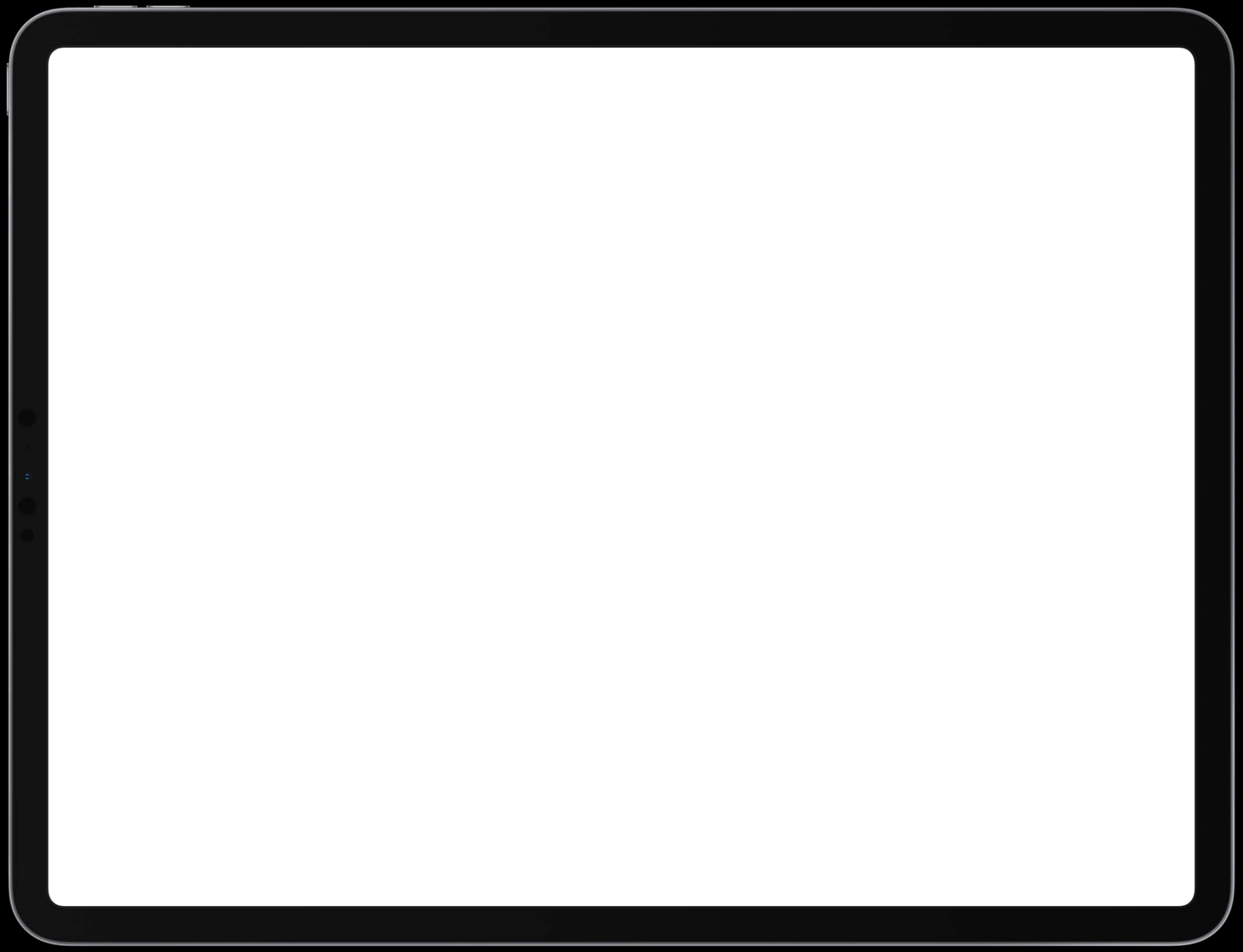


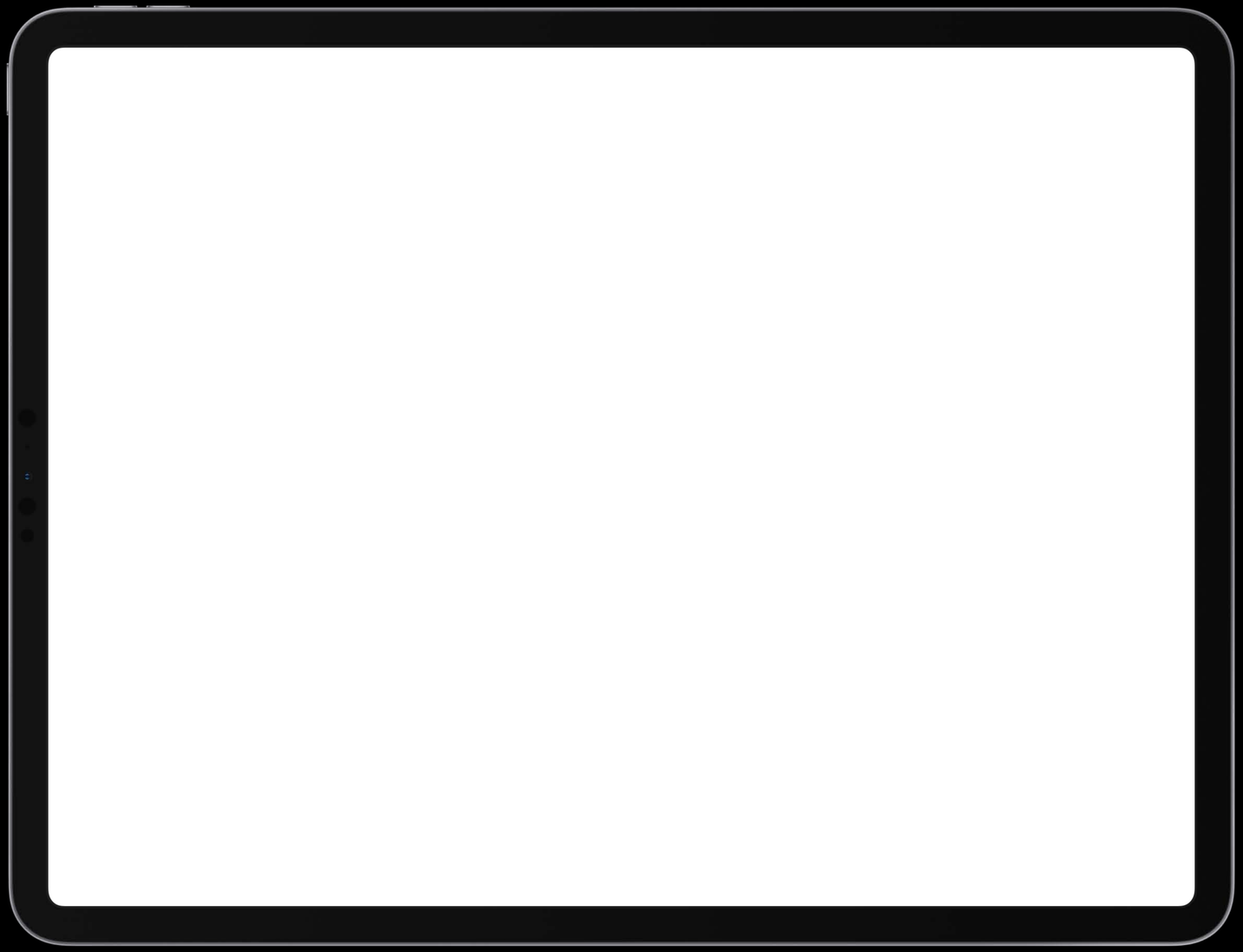


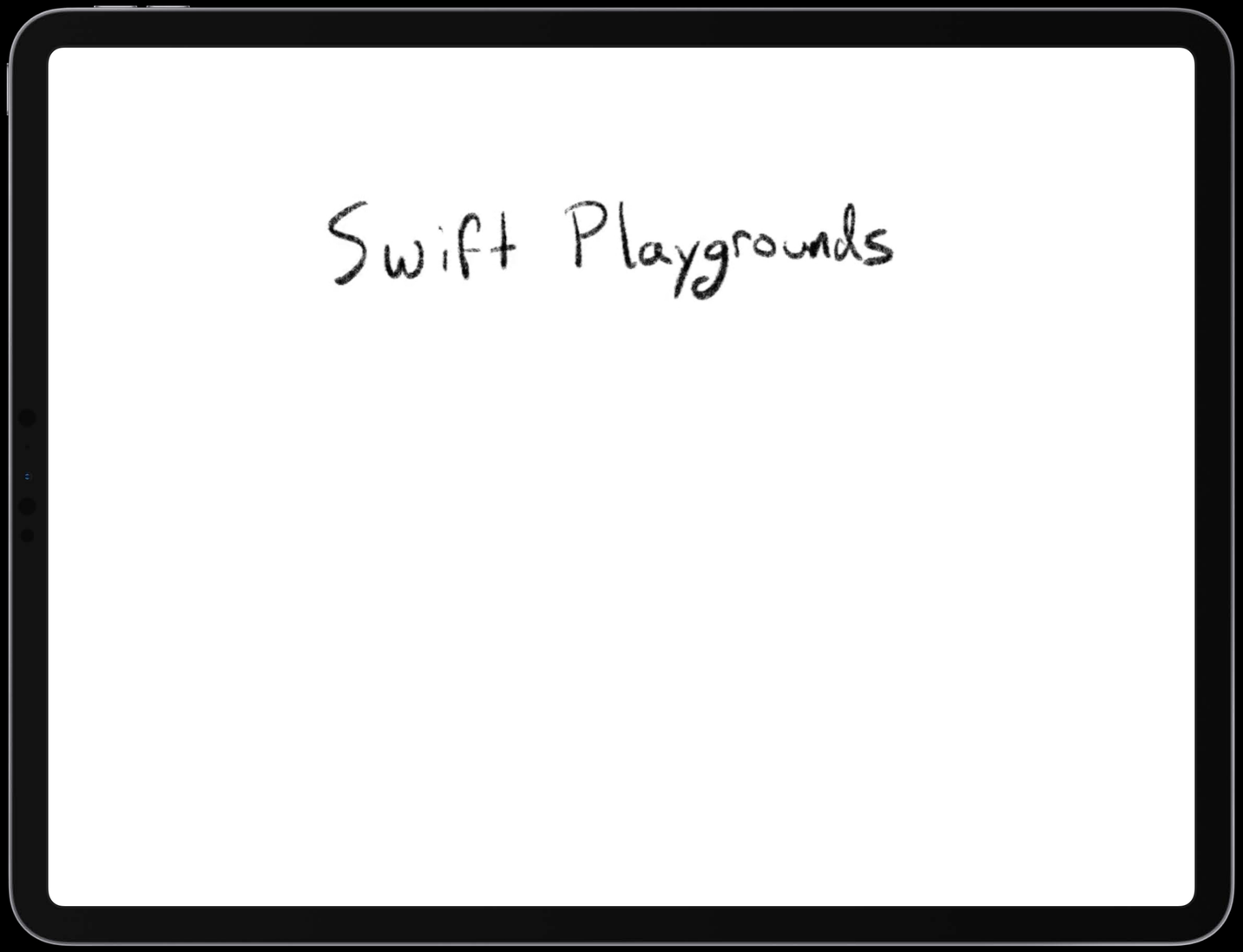


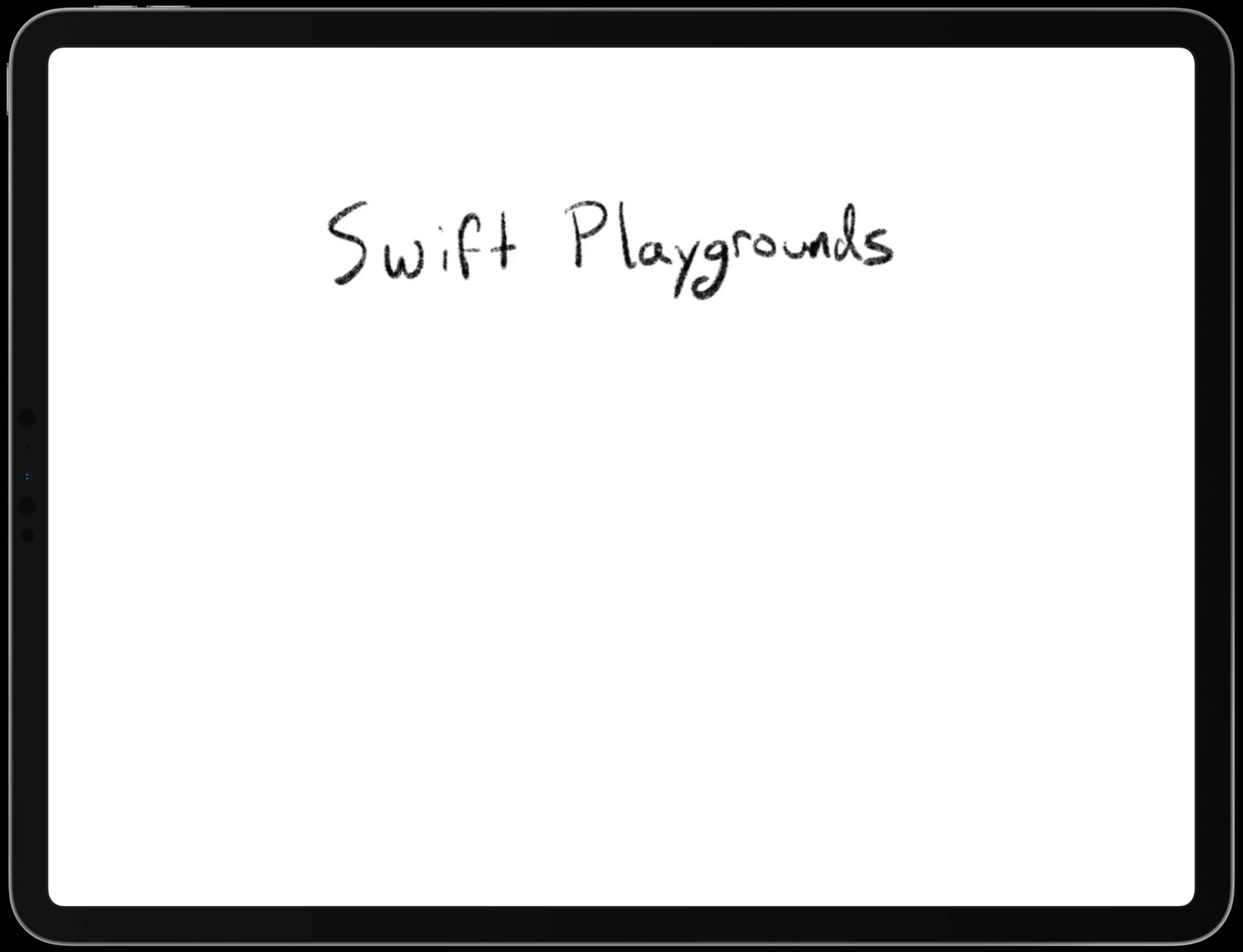






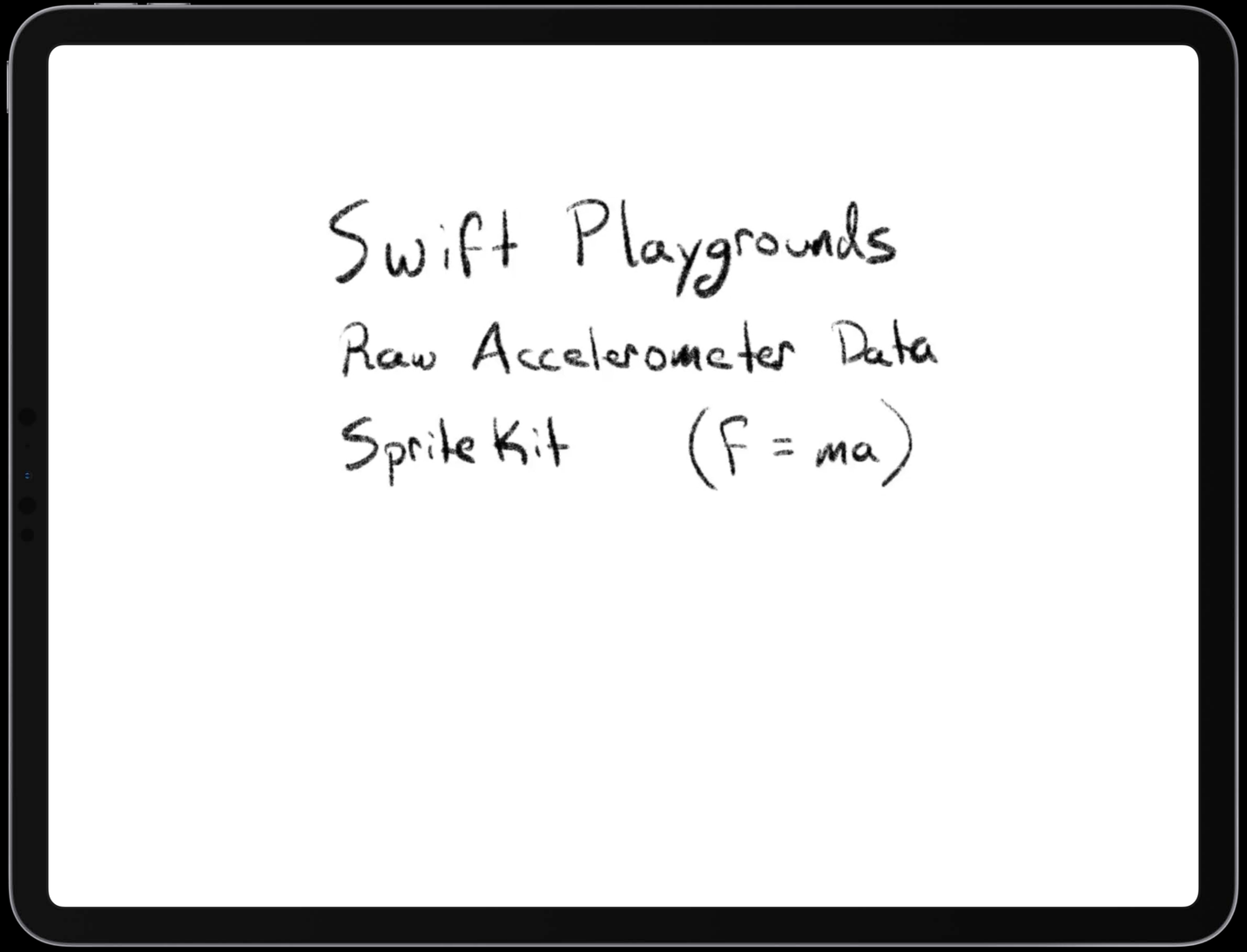


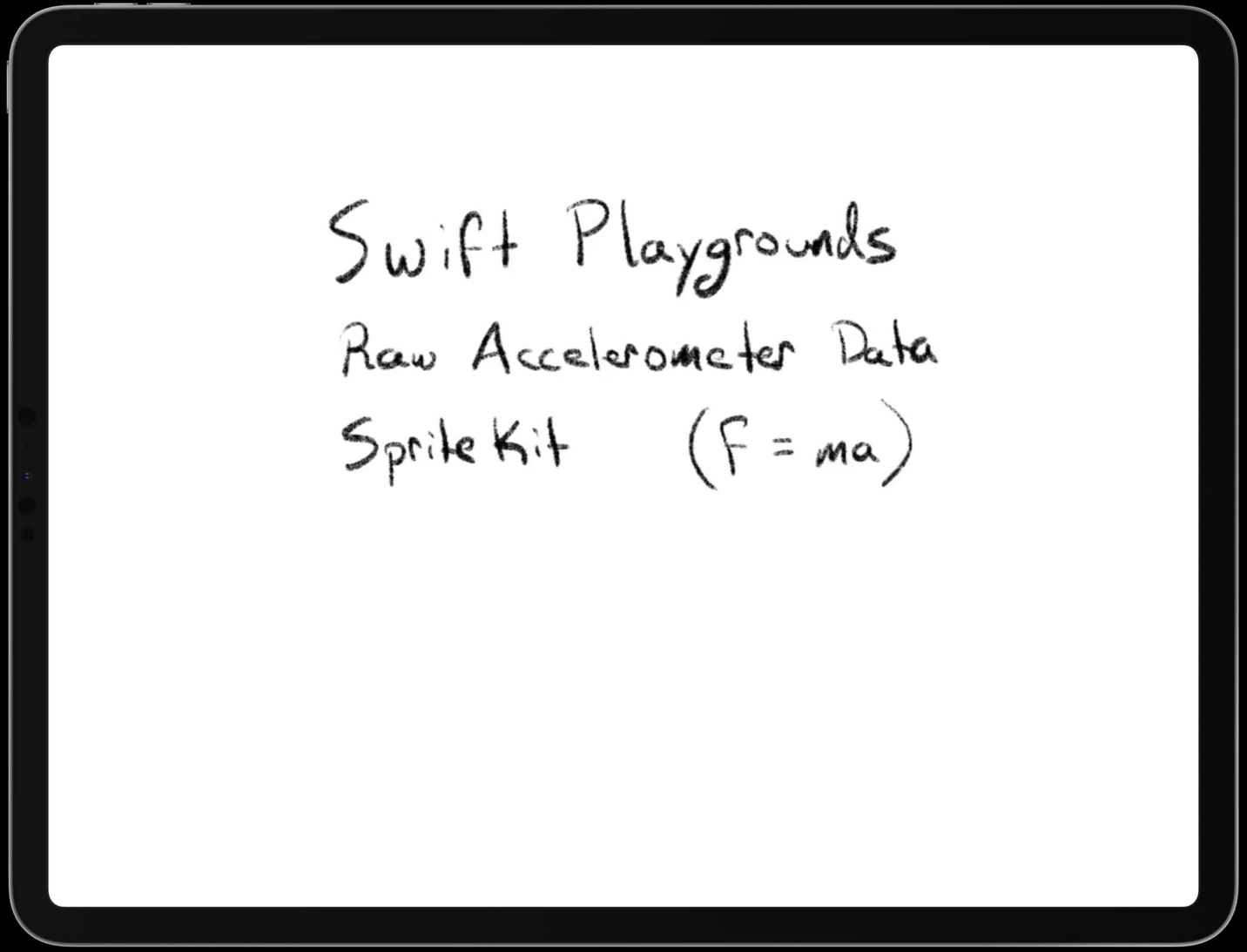

Swift Playgrounds

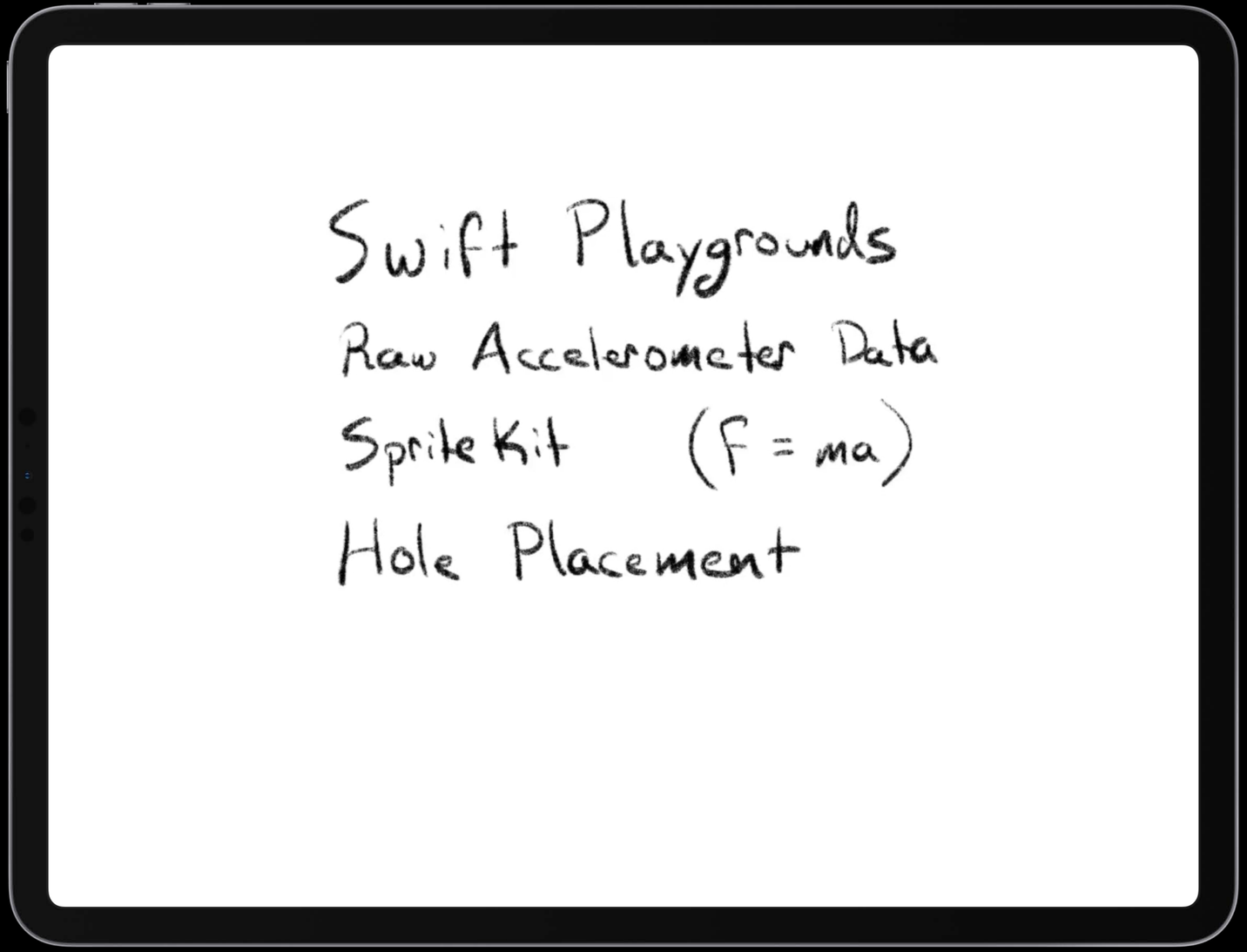

Swift Playgrounds

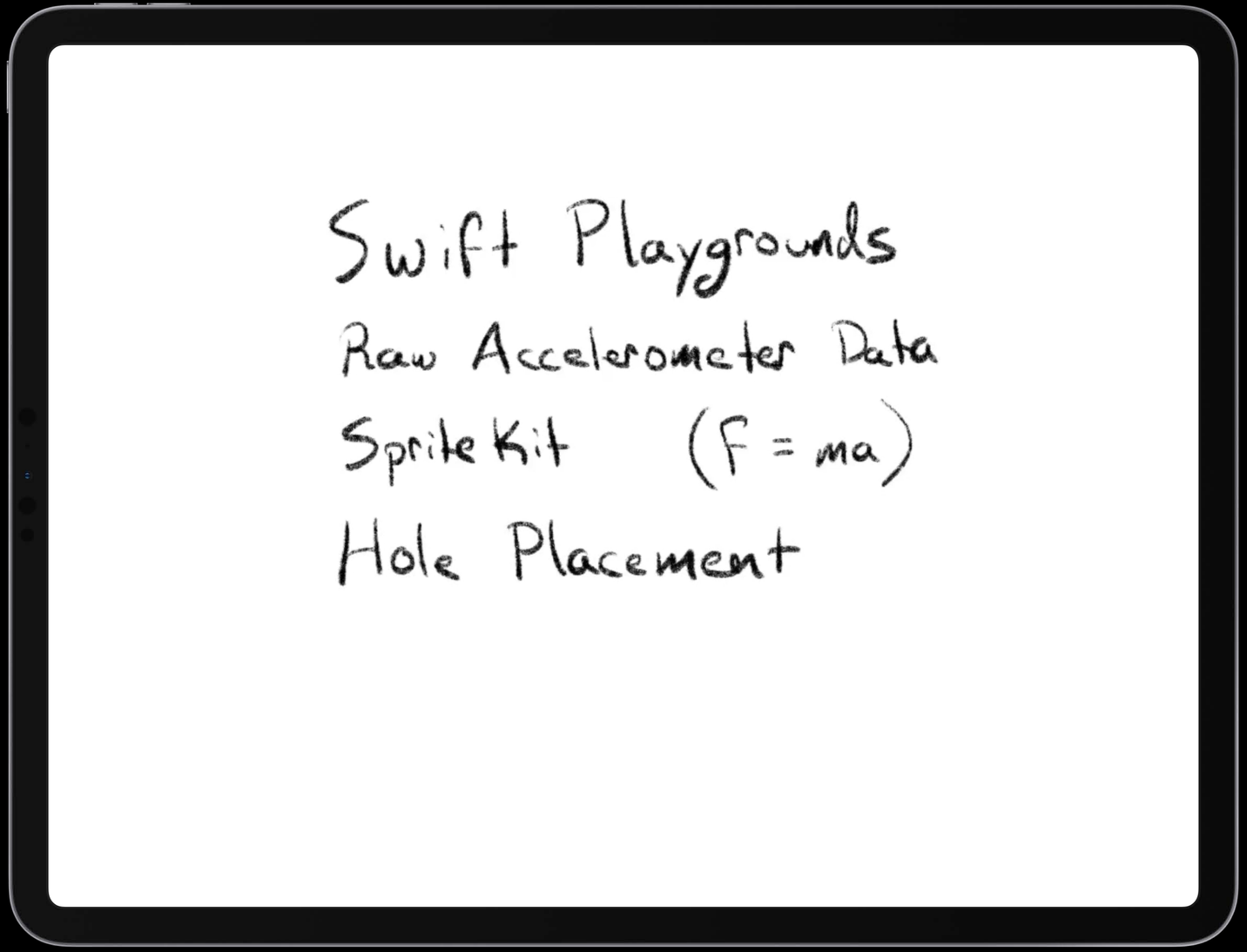

Swift Playgrounds
Raw Accelerometer Data


Swift Playgrounds
Raw Accelerometer Data


Swift Playgrounds
Raw Accelerometer Data
SpriteKit (F = ma)


Swift Playgrounds
Raw Accelerometer Data
SpriteKit (F = ma)


Swift Playgrounds
Raw Accelerometer Data
SpriteKit (F = ma)
Hole Placement

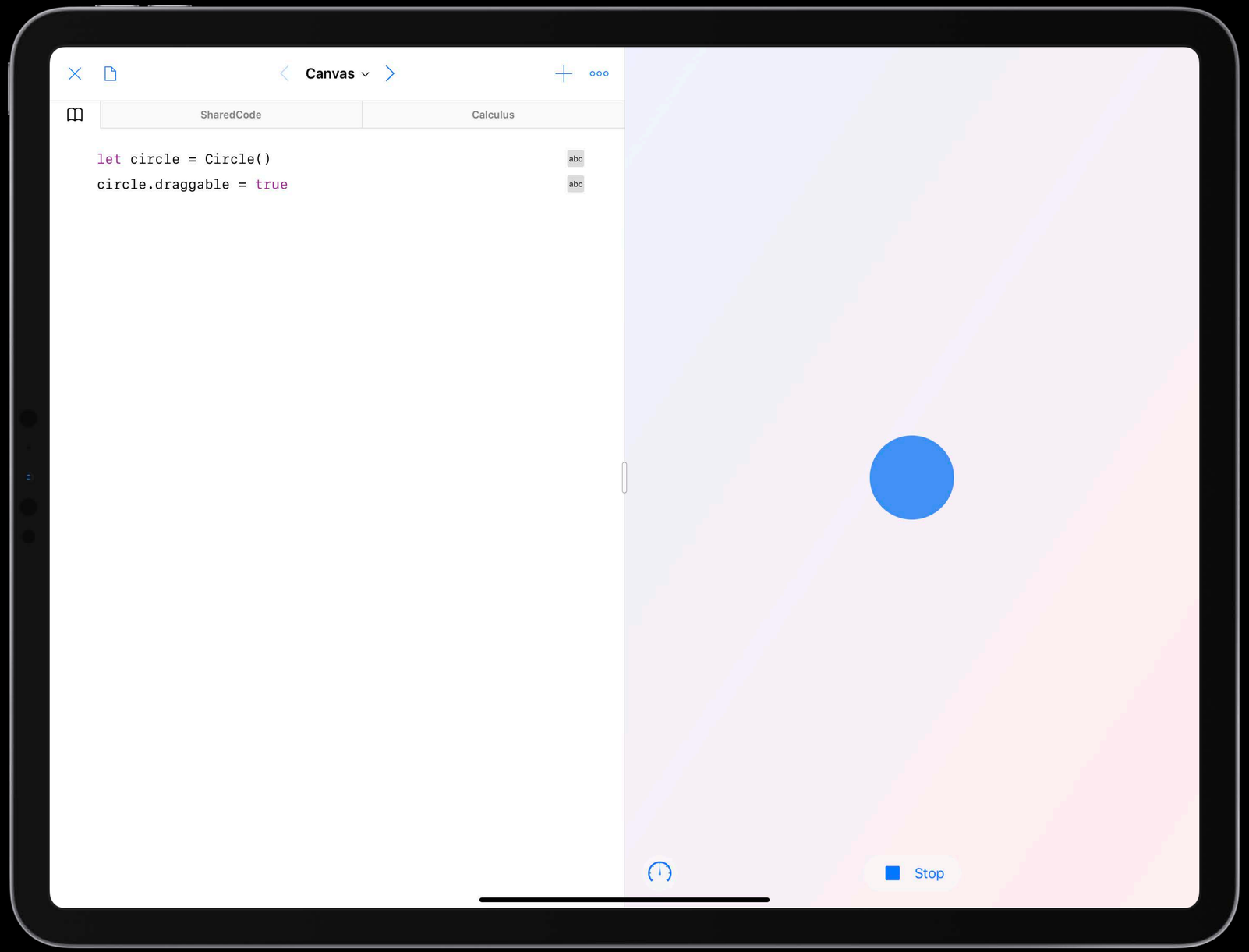

Swift Playgrounds
Raw Accelerometer Data
SpriteKit (F = ma)
Hole Placement

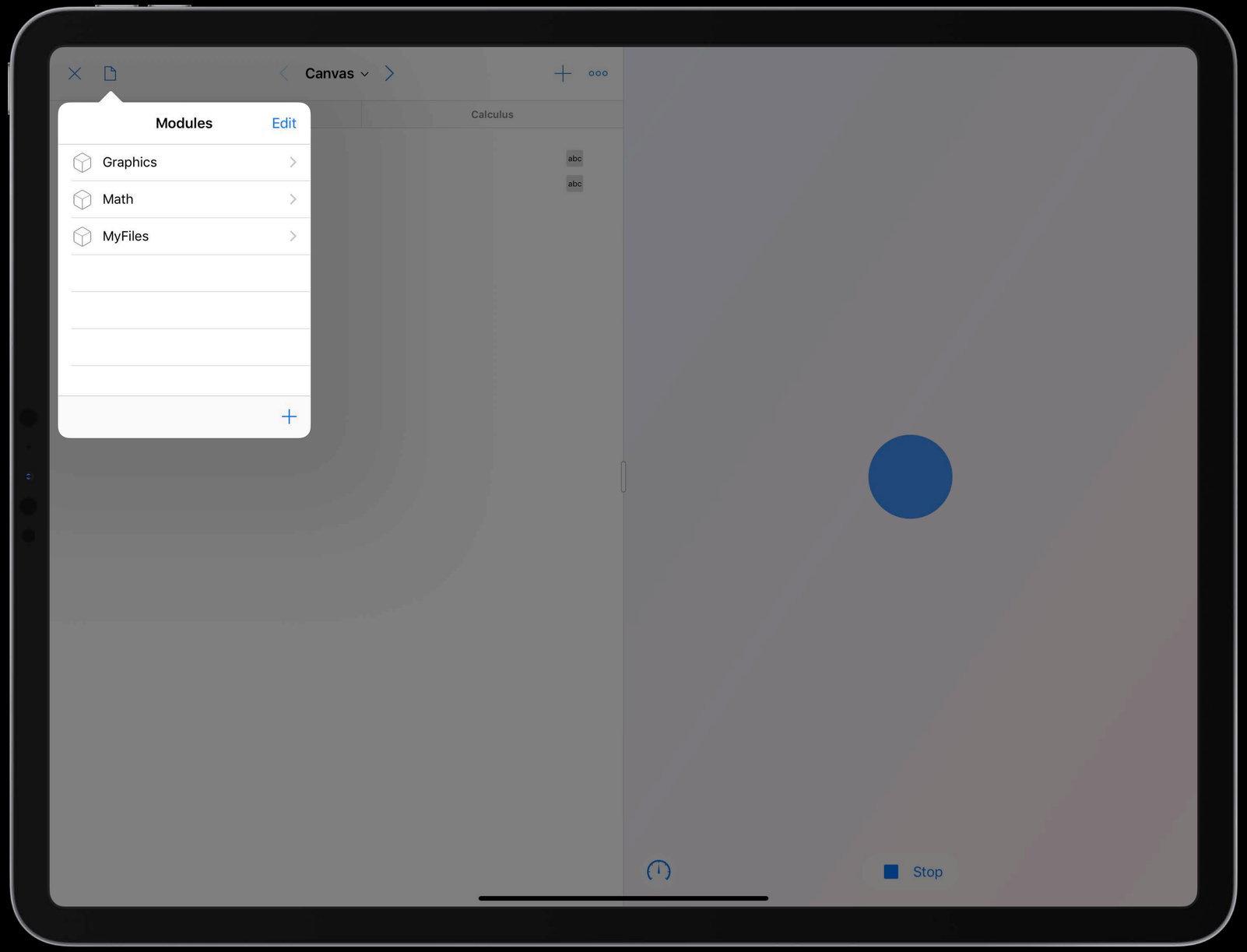
Demo

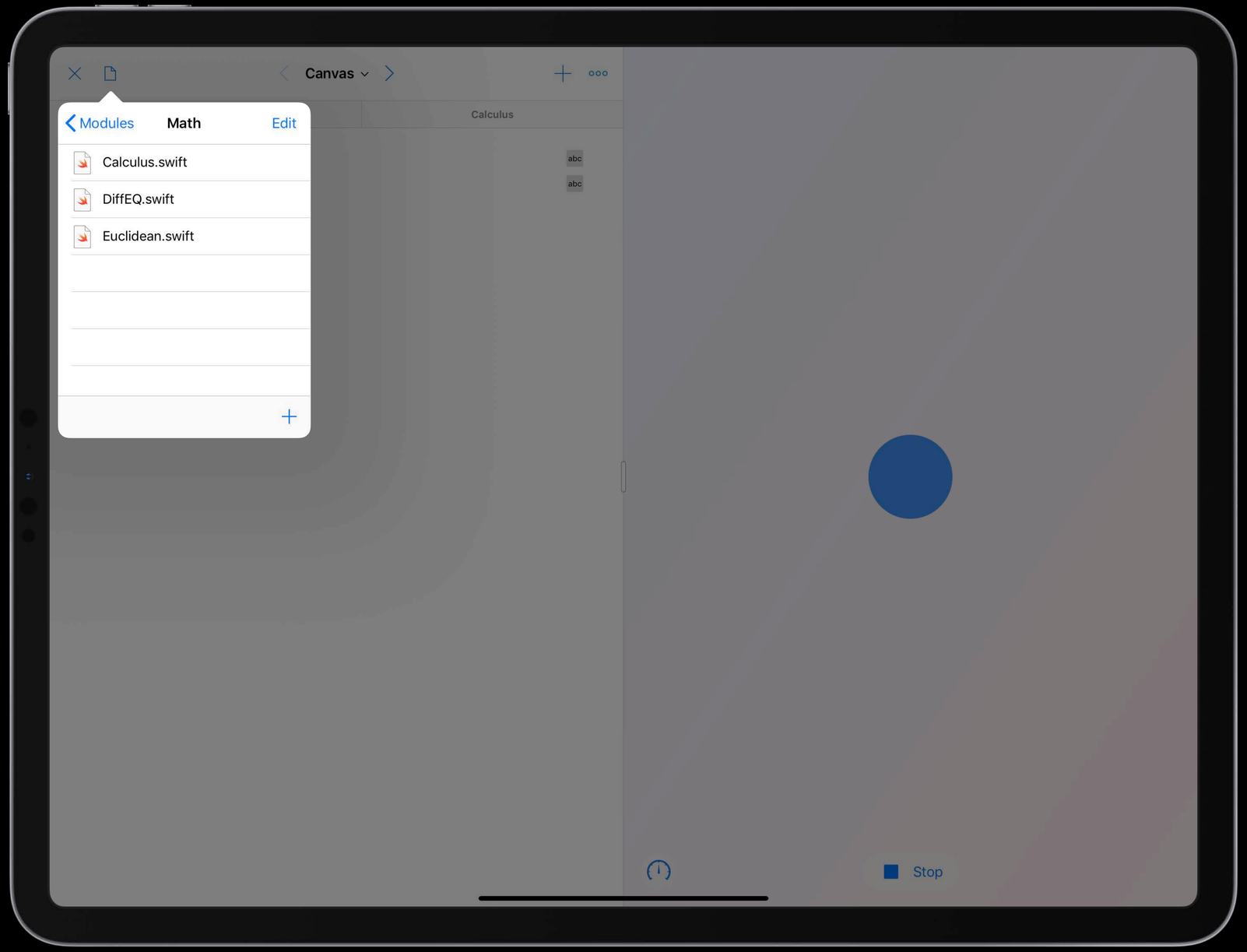
Jonathan Penn, Playgrounds Engineer

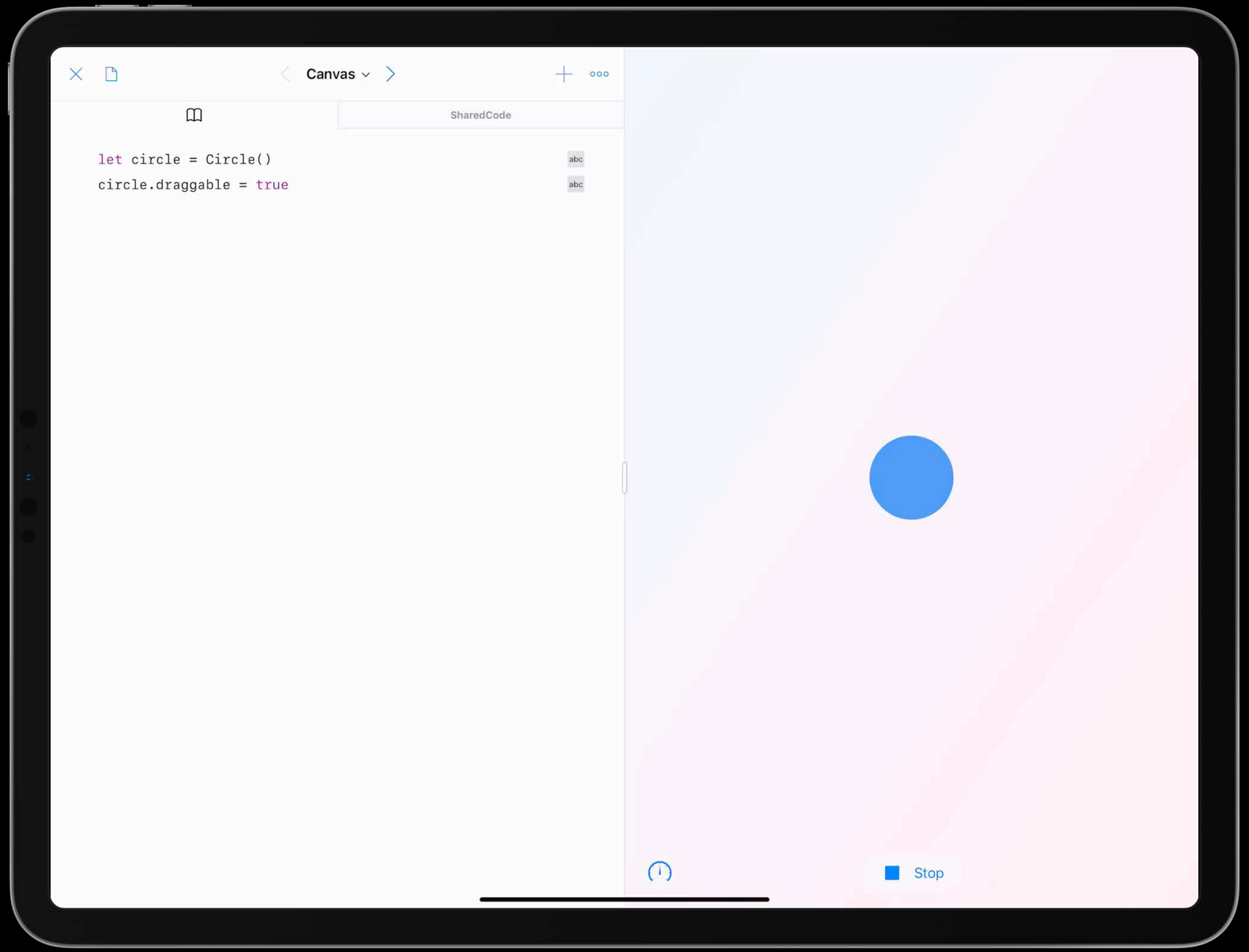
New on iPad

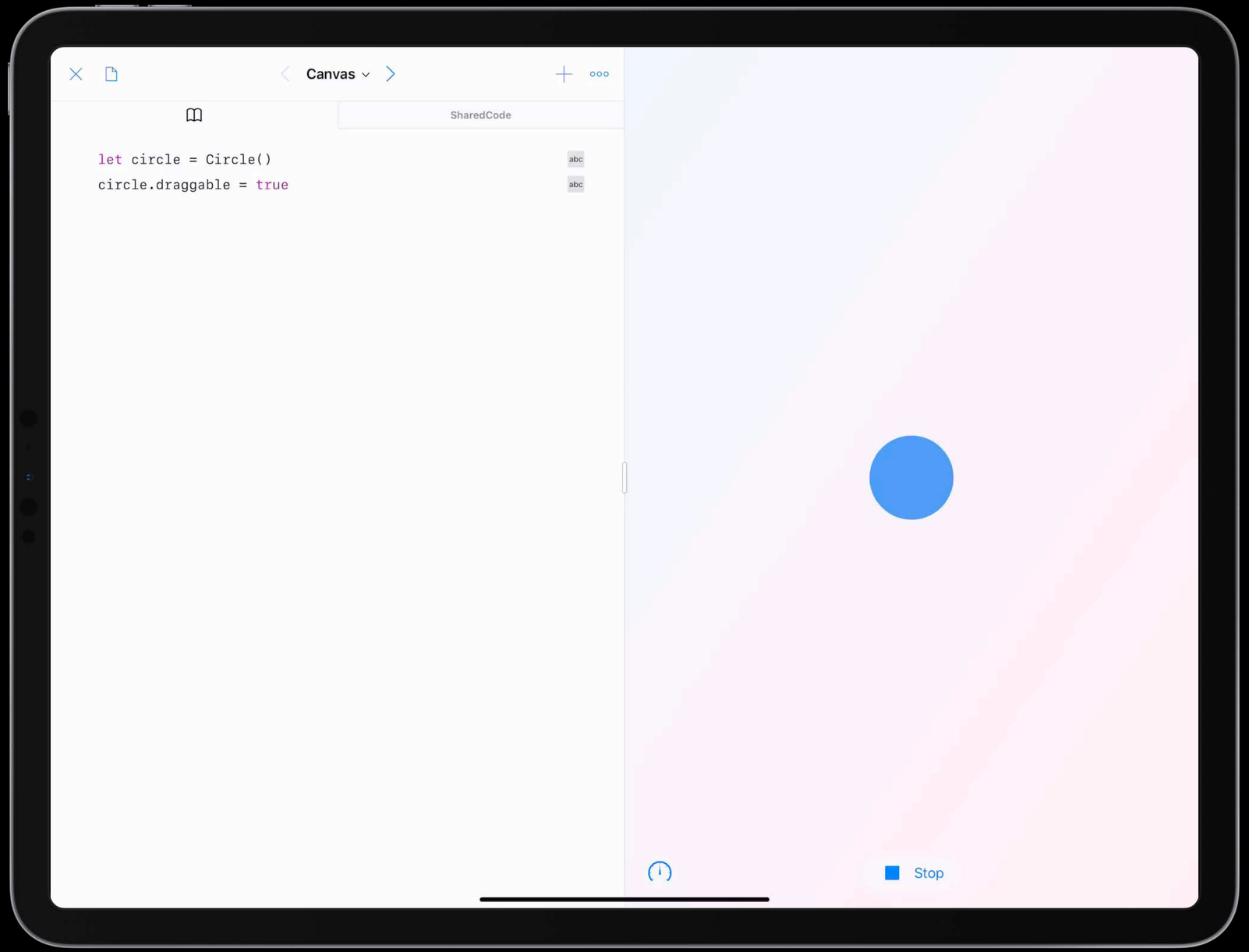
Grace Kendall, Playgrounds Engineer


Canvas
SharedCode
Calculus
let circle = Circle()
circle.draggable = true
abc
abc
Stop

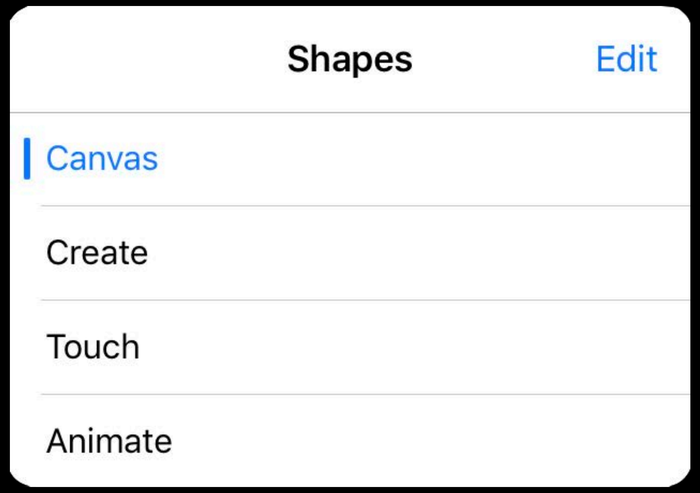

Canvas
Modules
Edit
Graphics
Math
MyFiles
Calculus
abc
abc
Stop

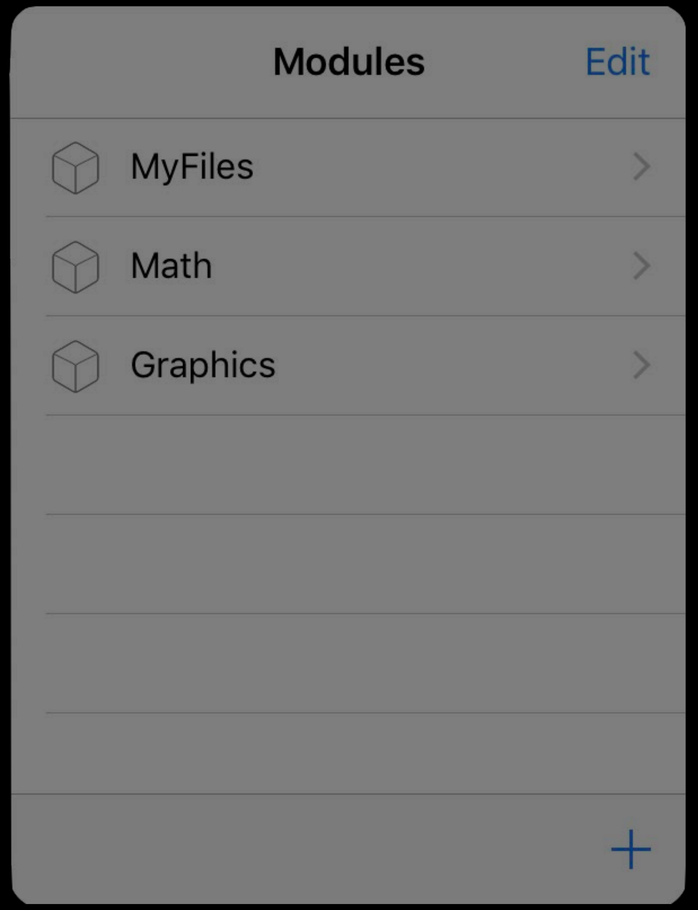

Canvas
Modules
Math
Edit
Calculus.swift
DiffEQ.swift
Euclidean.swift
Calculus
abc
abc
Stop

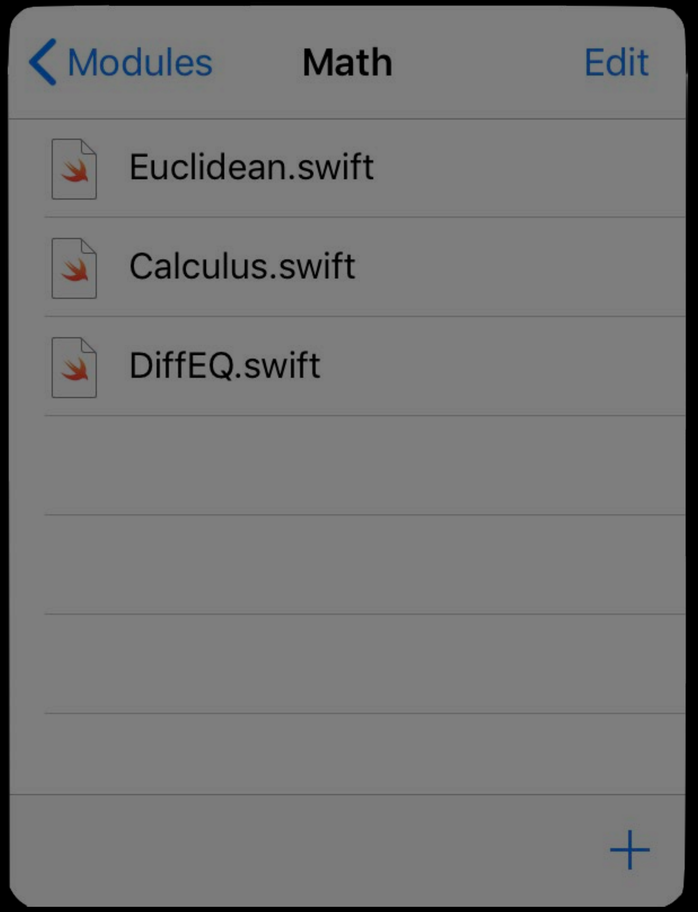

Canvas
SharedCode
let circle = Circle()
circle.draggable = true
abc
abc
Stop


Canvas
SharedCode
let circle = Circle()
circle.draggable = true
abc
abc
Stop

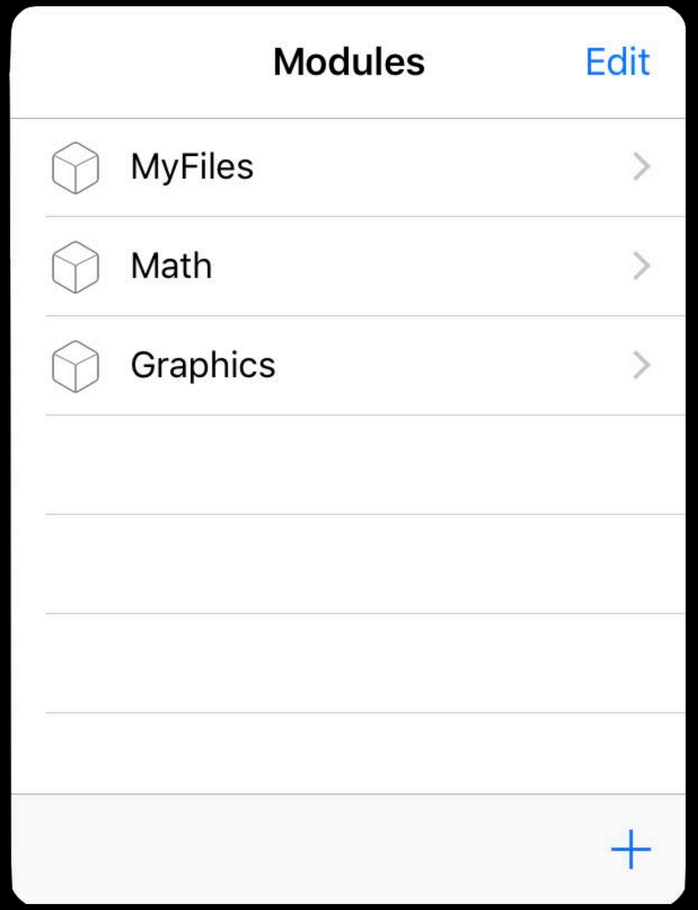
Shapes

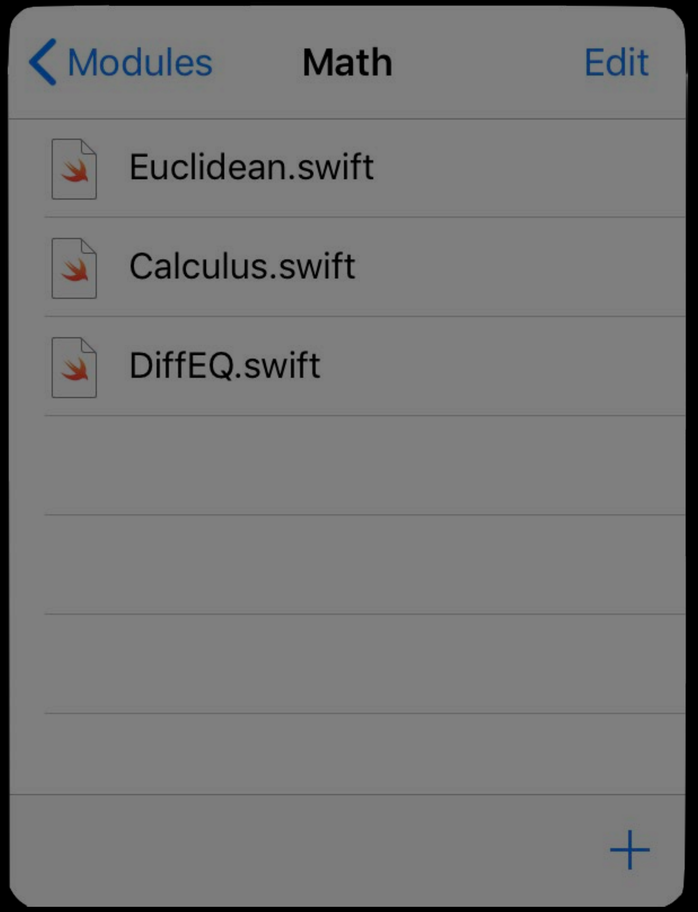

Shapes
Edit
Canvas
Create
Touch
Animate


Modules
Edit
MyFiles
Math
Graphics

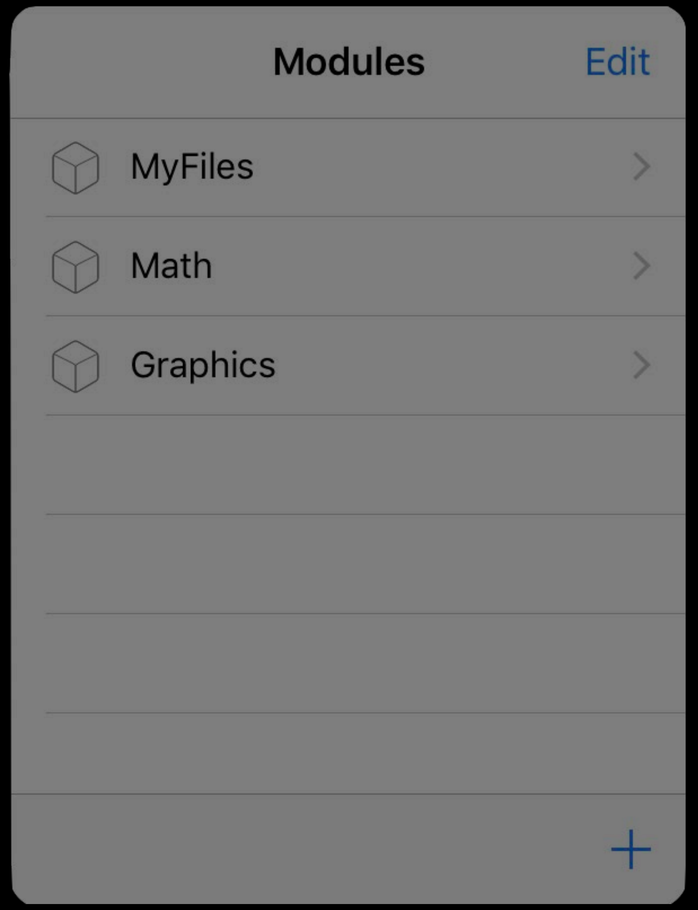

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

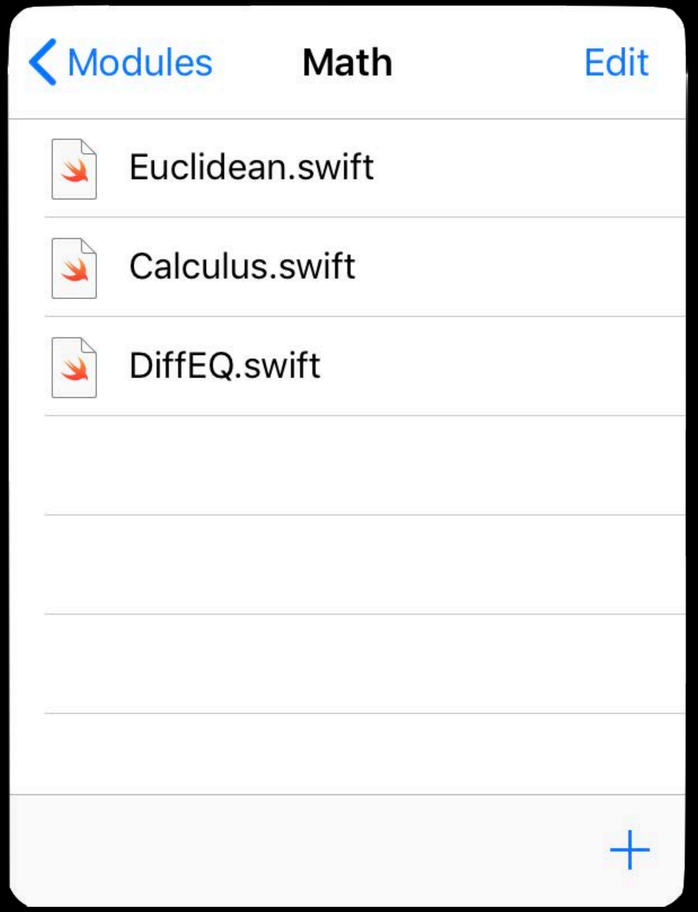

Shapes
Edit
Canvas
Create
Touch
Animate

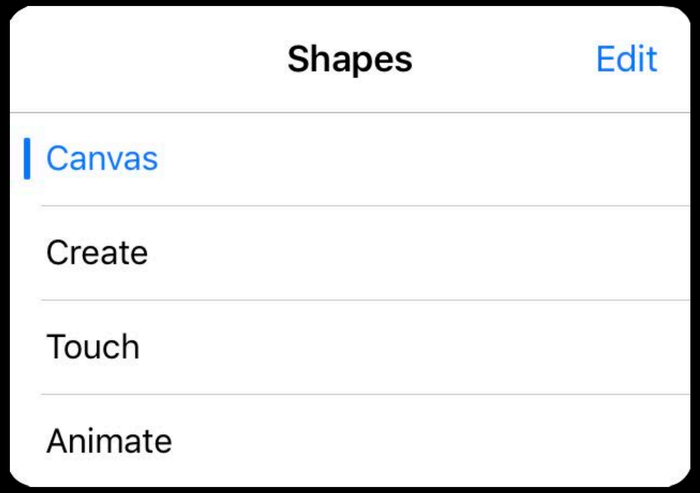

Modules
Edit
MyFiles
Math
Graphics

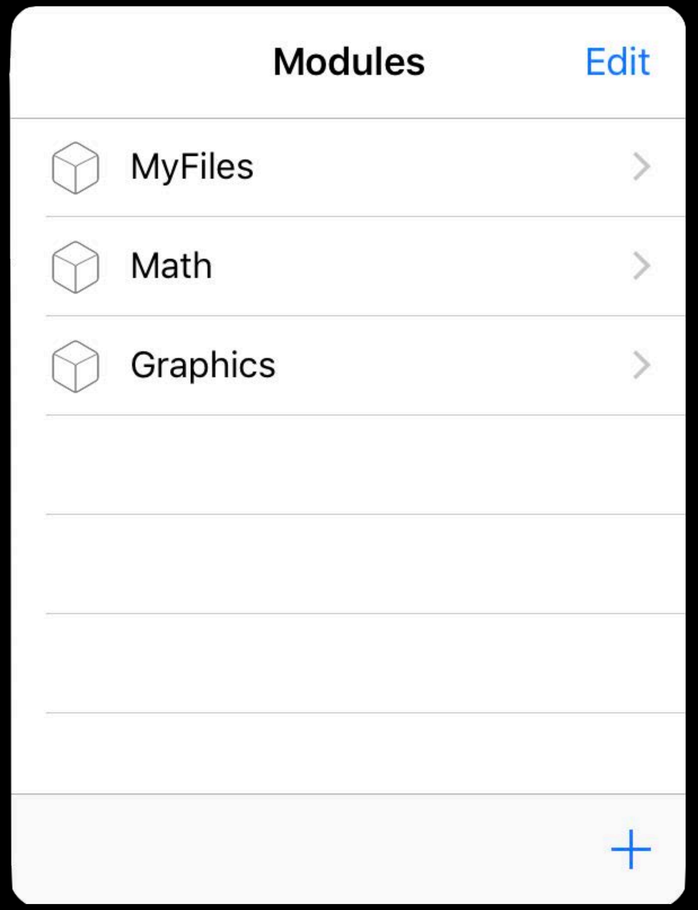

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

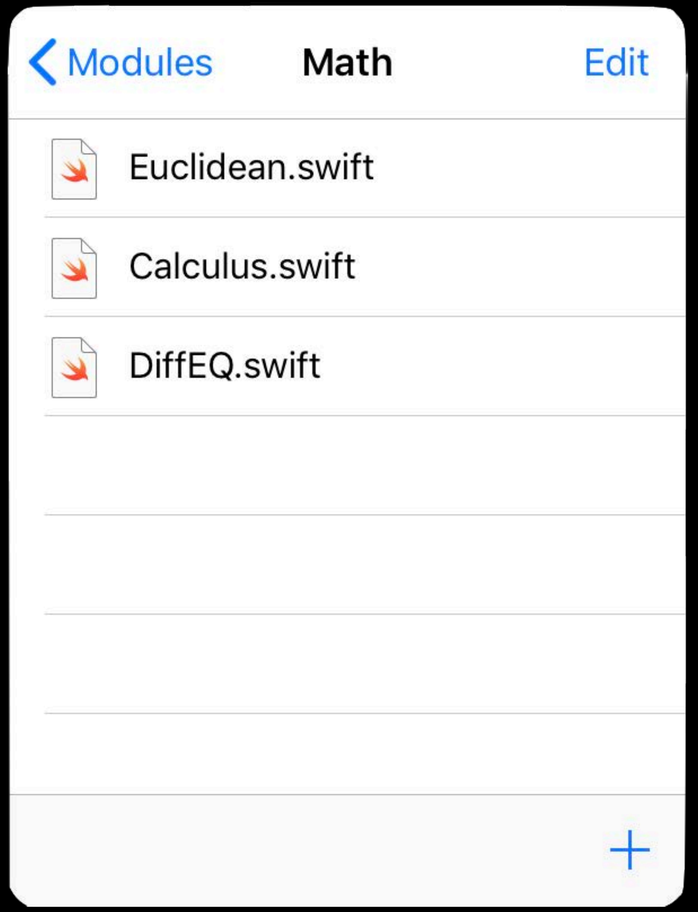

Shapes
Edit
Canvas
Create
Touch
Animate


Modules
Edit
MyFiles
Math
Graphics

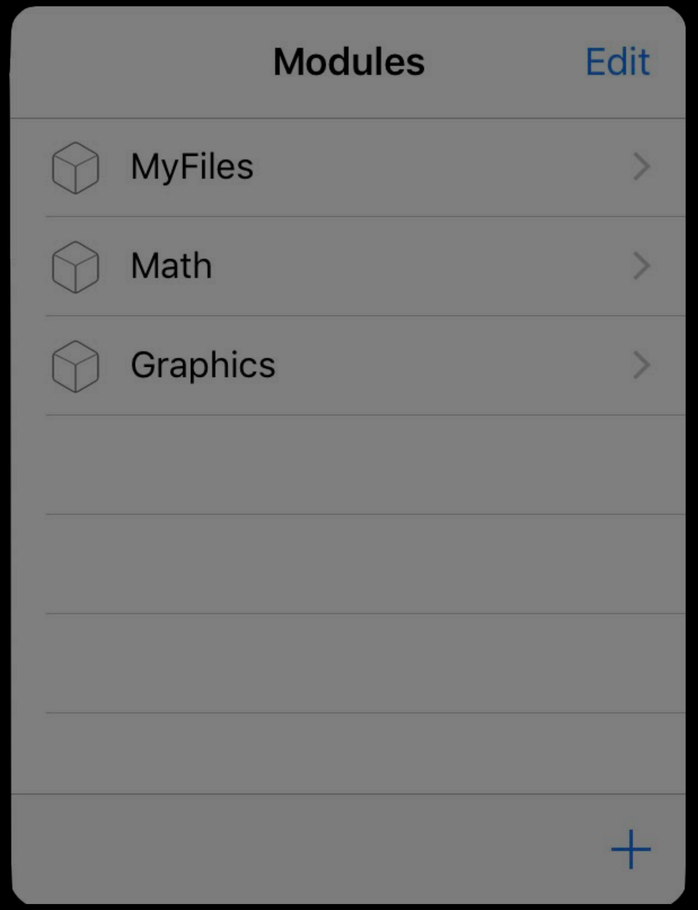

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

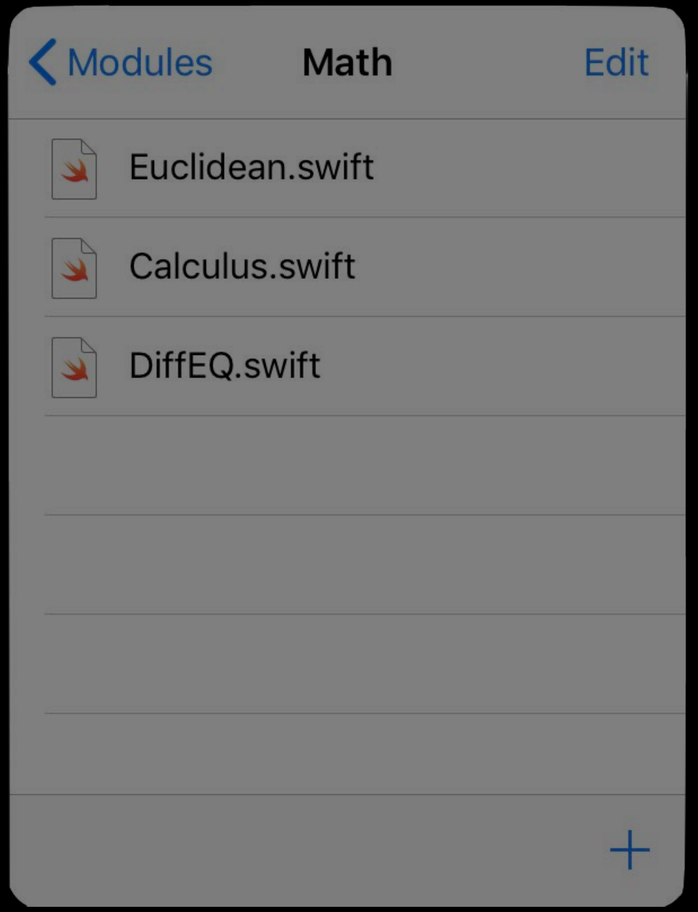

Shapes
Edit
Canvas
Create
Touch
Animate

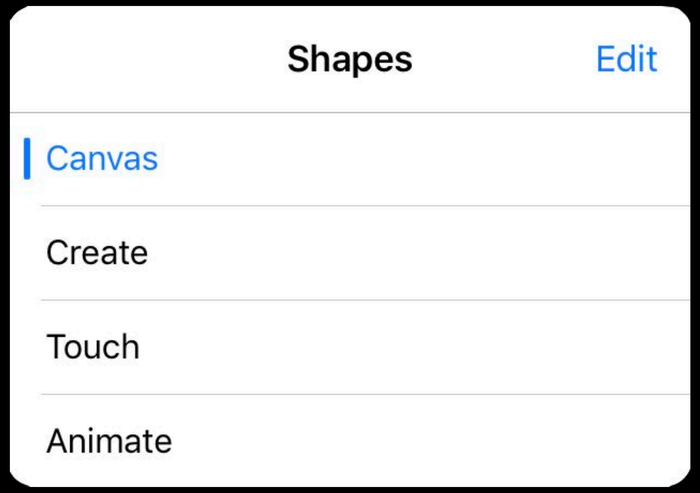

Modules
Edit
MyFiles
Math
Graphics

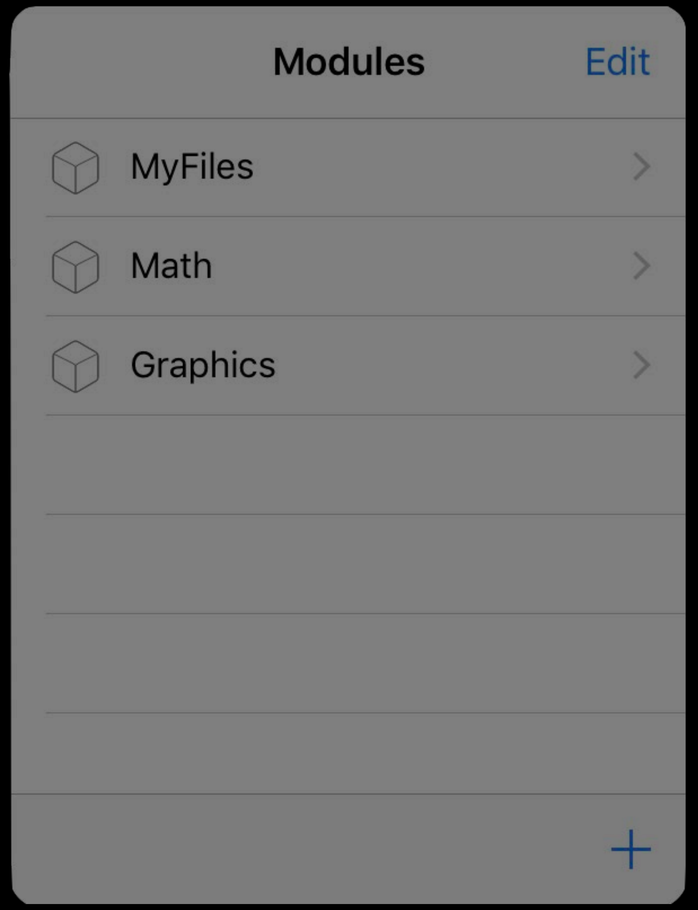

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

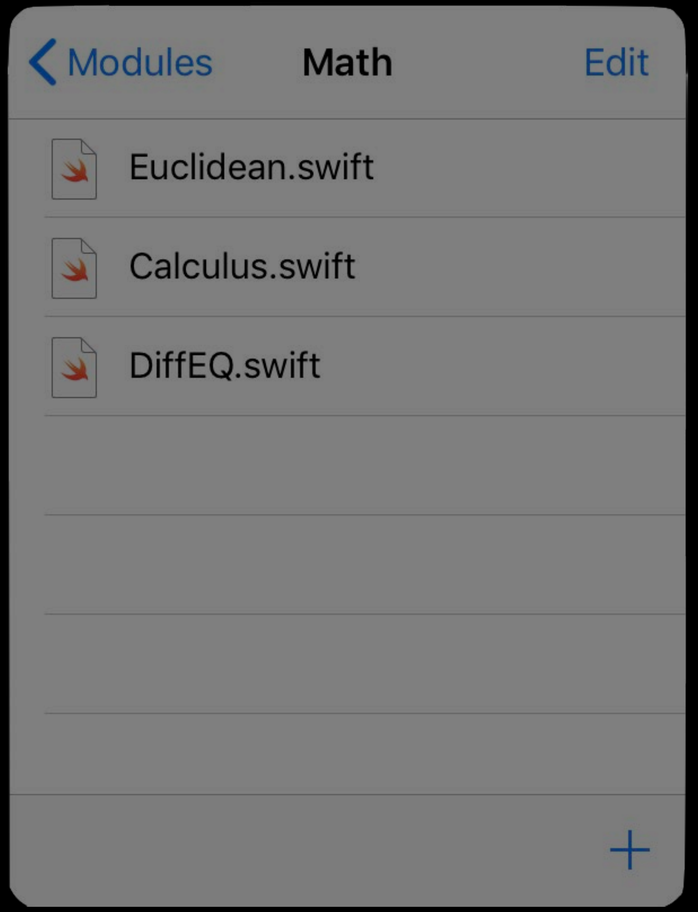

Shapes
Edit
Canvas
Create
Touch
Animate

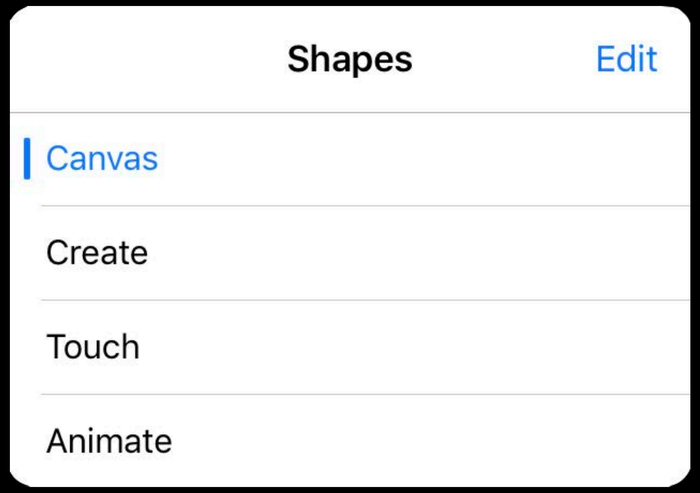

Modules
Edit
MyFiles
Math
Graphics

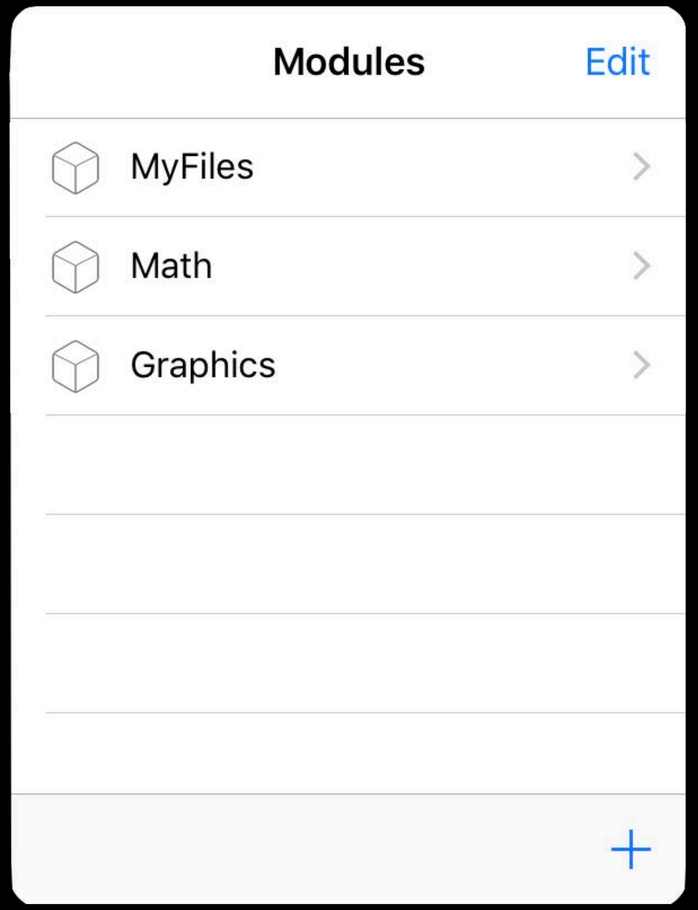

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

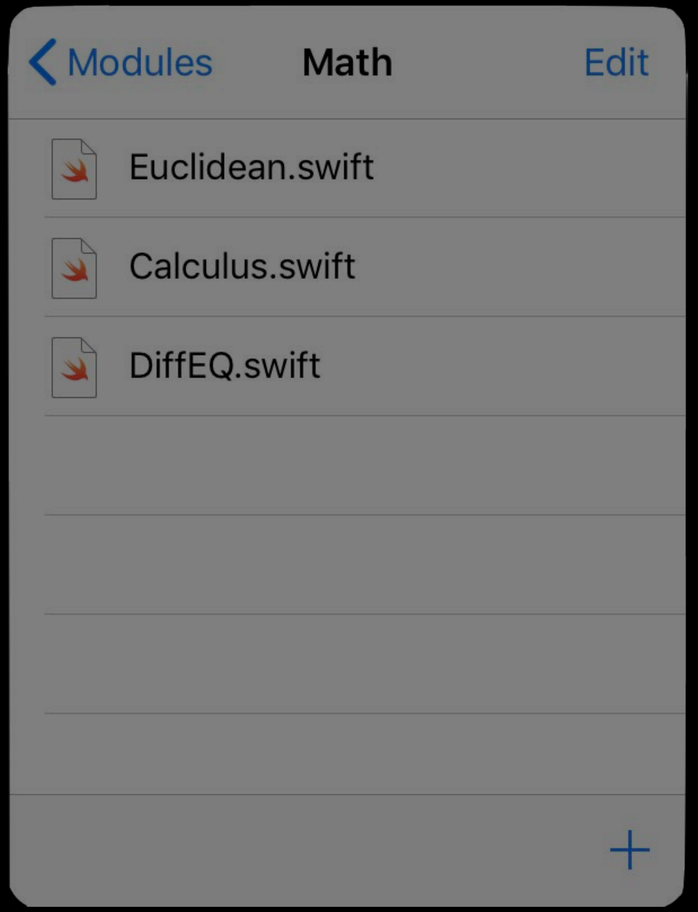

Shapes
Edit
Canvas
Create
Touch
Animate

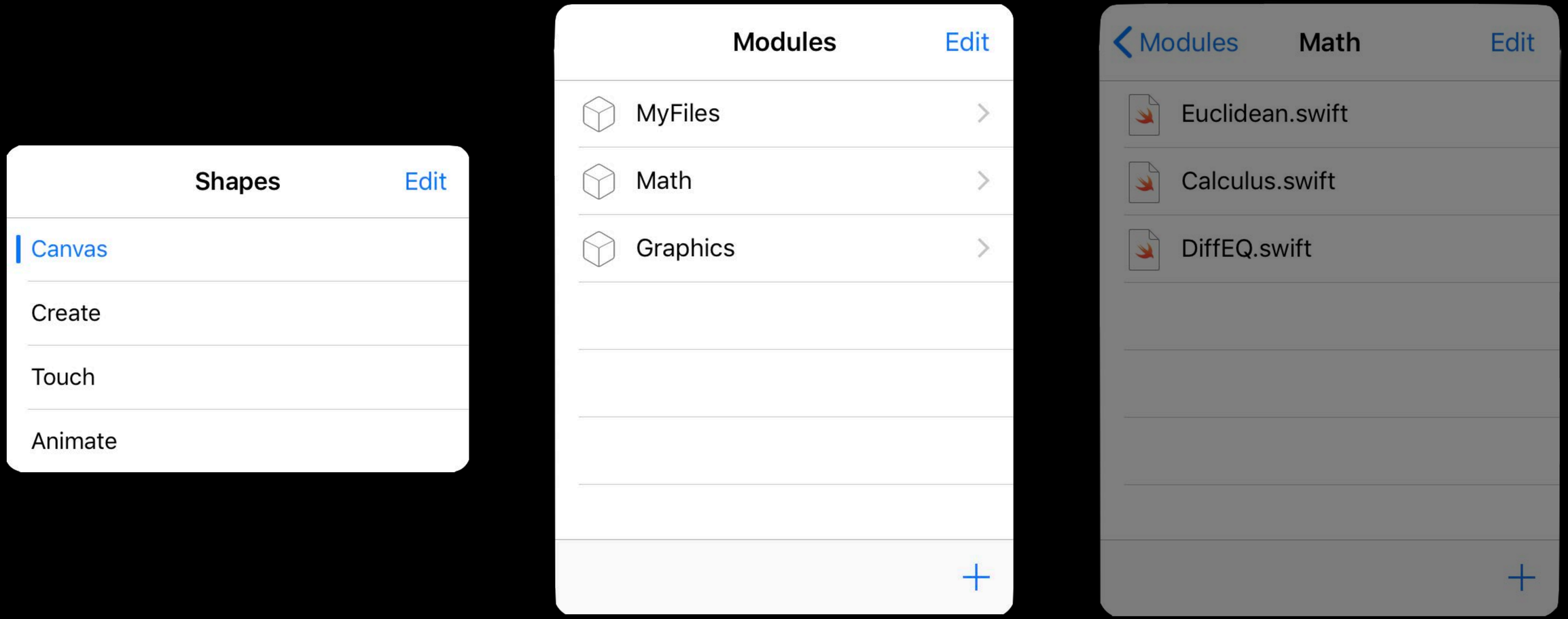

Modules
Edit
MyFiles
Math
Graphics


Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

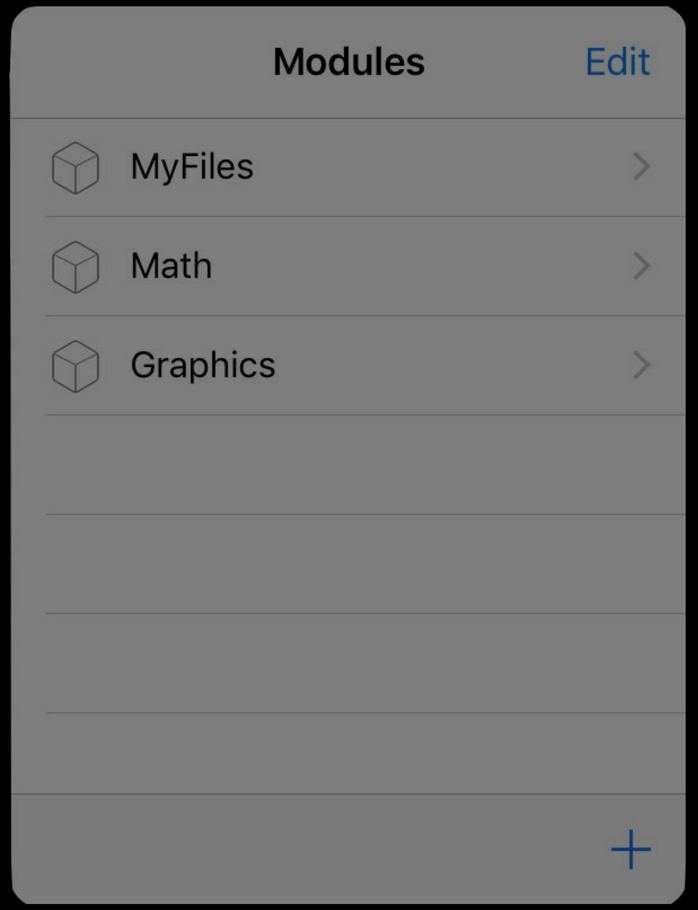

Shapes
Edit
Canvas
Create
Touch
Animate

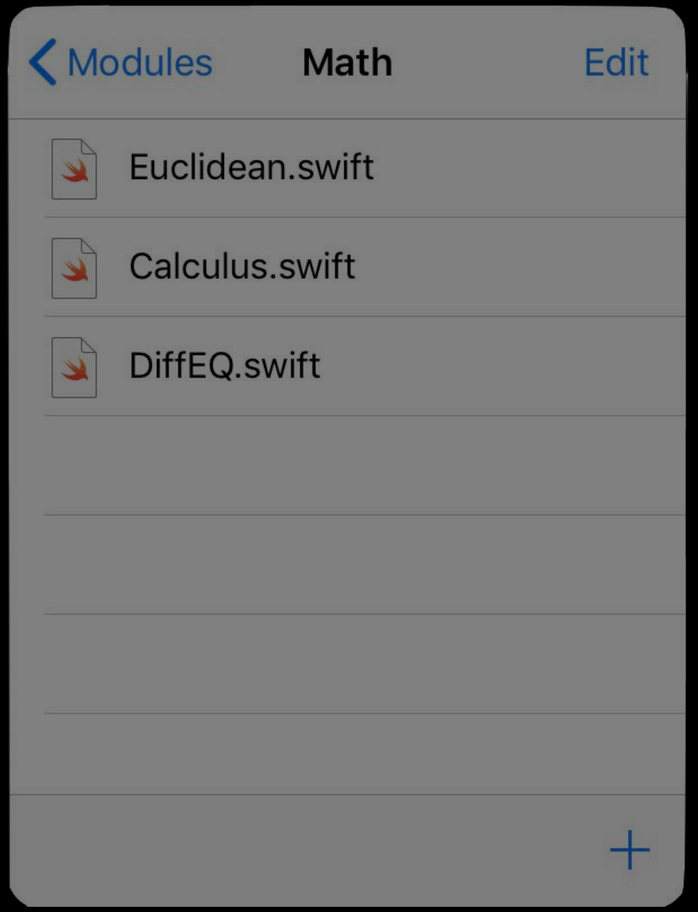

Modules
Edit
MyFiles
Math
Graphics


Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

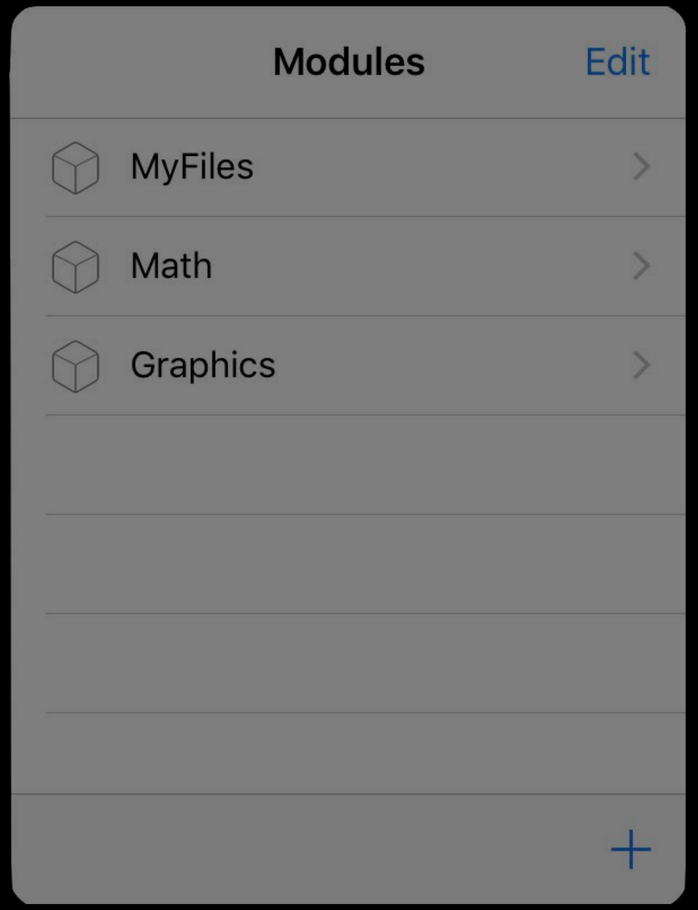

Shapes
Edit
Canvas
Create
Touch
Animate
Modules
Edit
MyFiles
Math
Graphics
Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

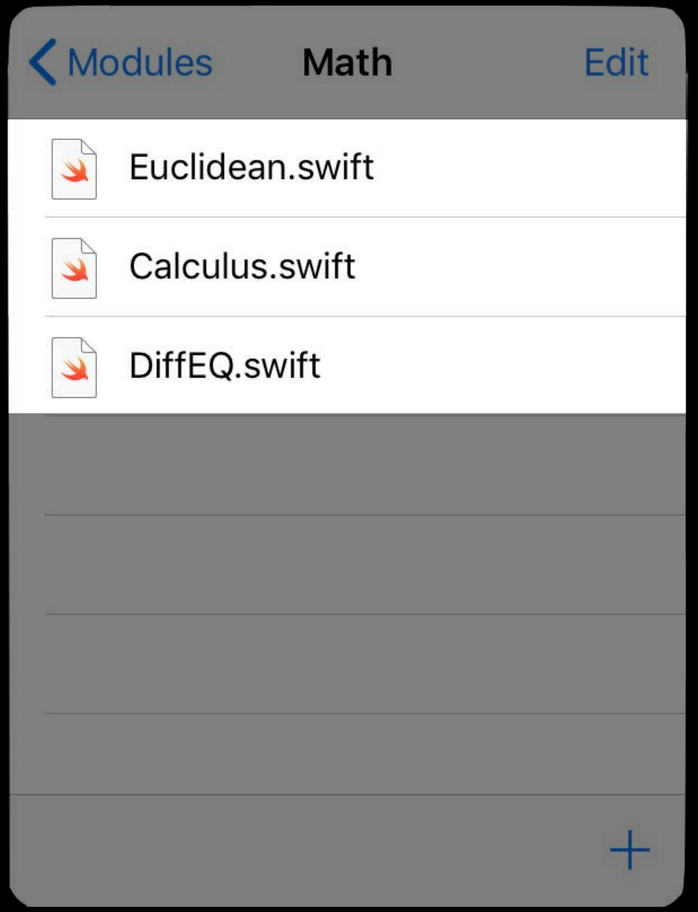

Shapes
Edit
Canvas
Create
Touch
Animate

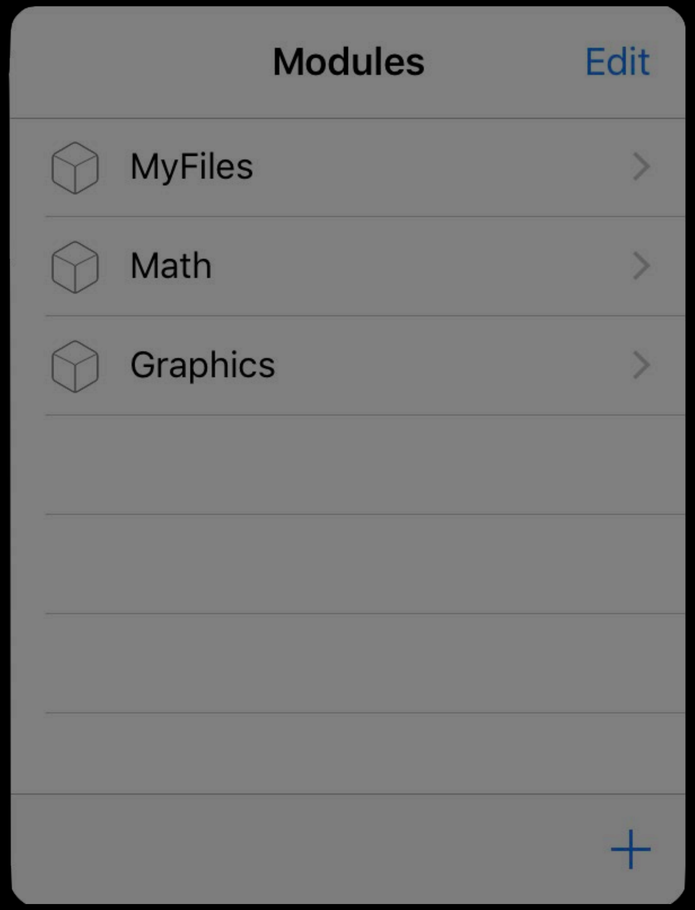

Modules
Edit
MyFiles
Math
Graphics

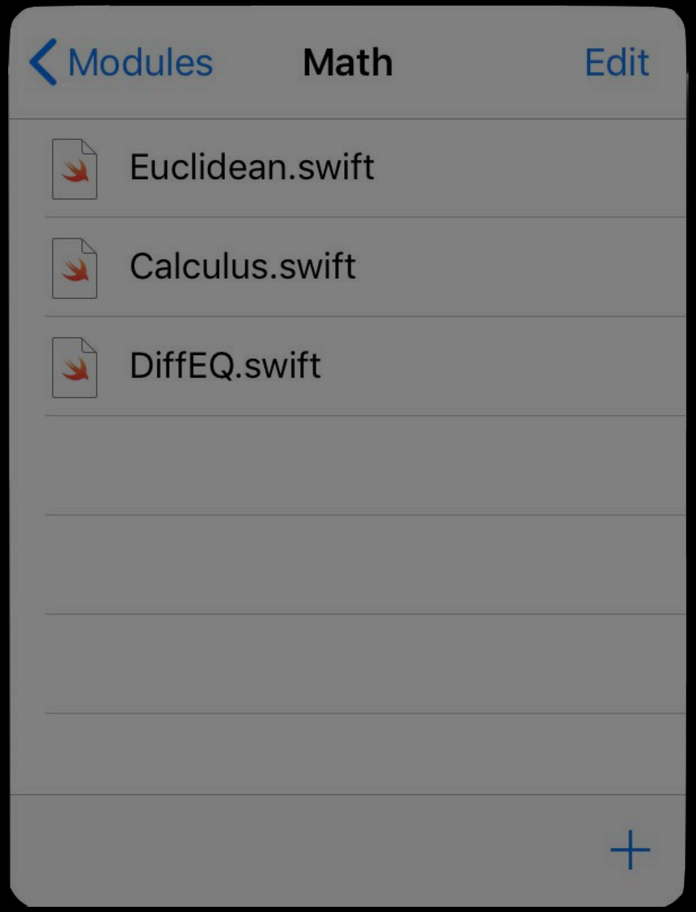

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

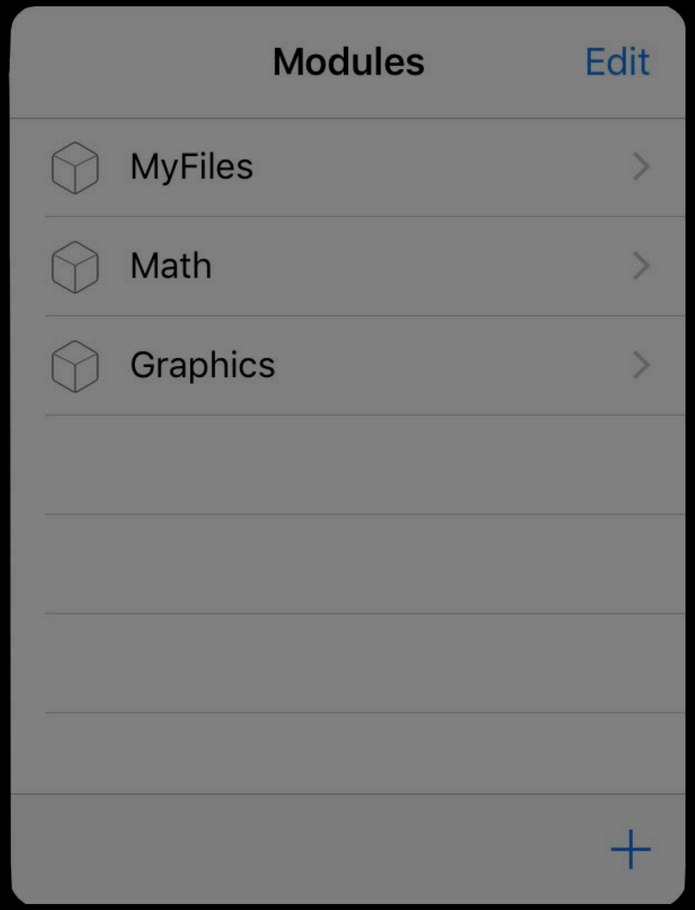

Shapes
Edit
Canvas
Create
Touch
Animate

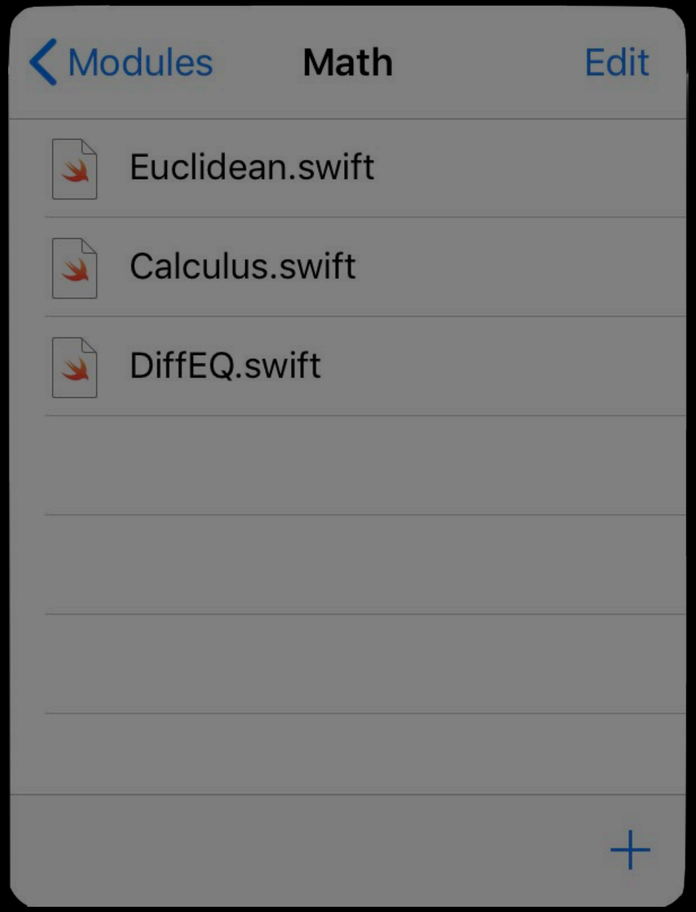

Modules
Edit
MyFiles
Math
Graphics

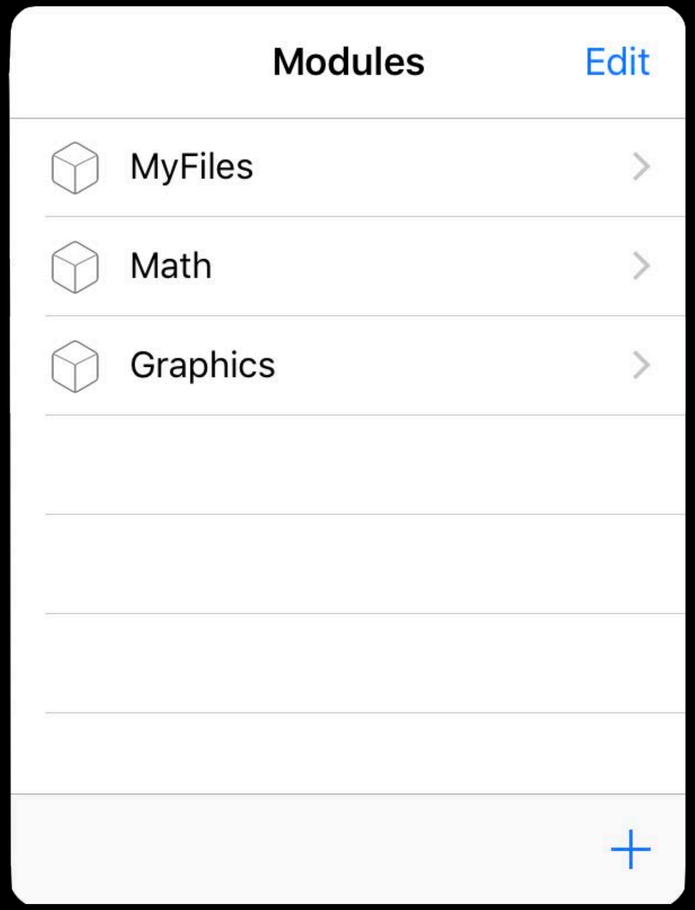

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

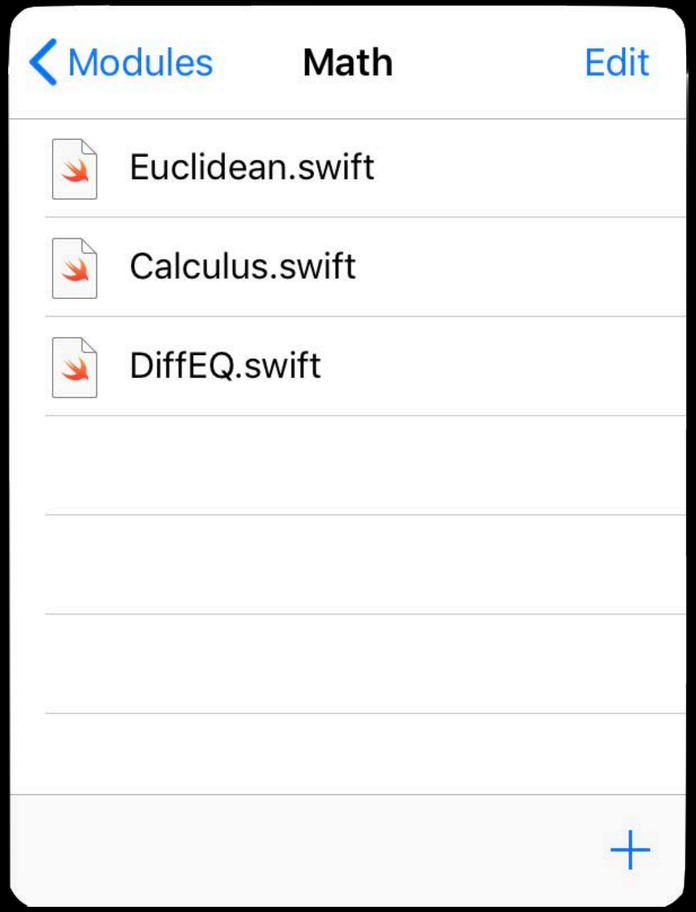

Modules
Edit
MyFiles
Math
Graphics

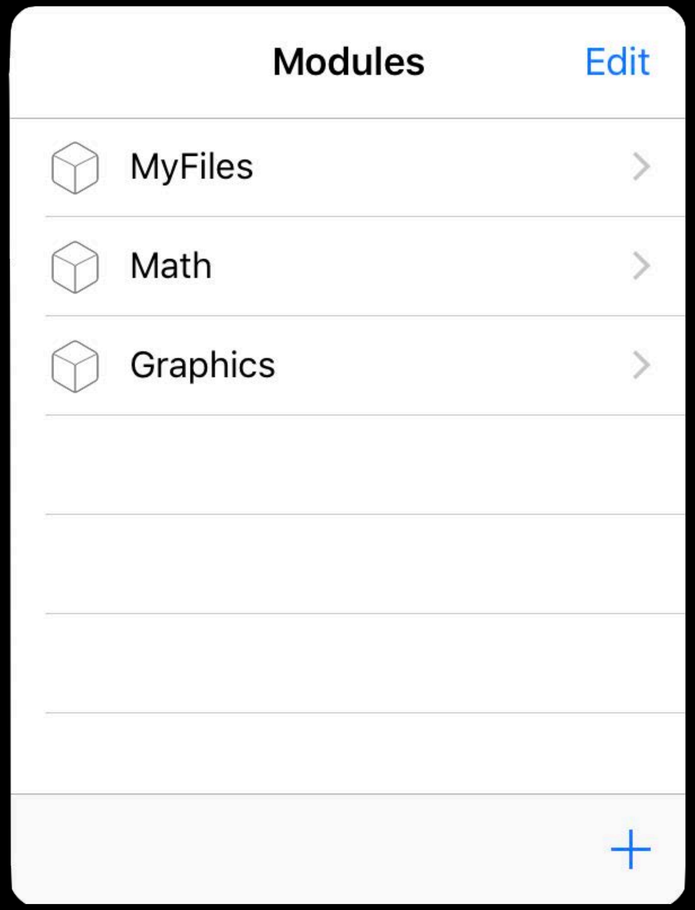

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

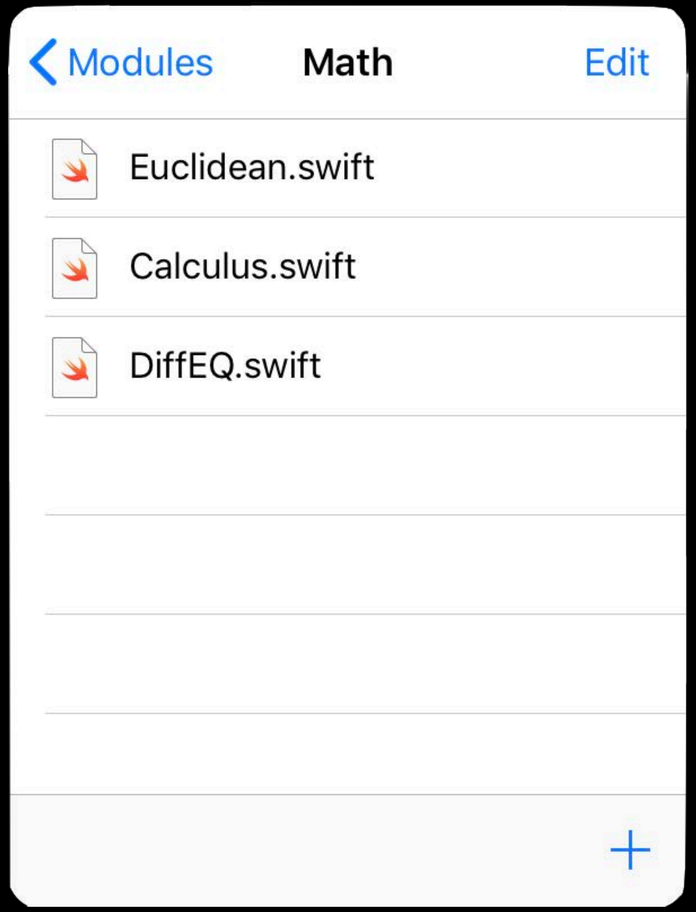

Modules
Edit
MyFiles
Math
Graphics

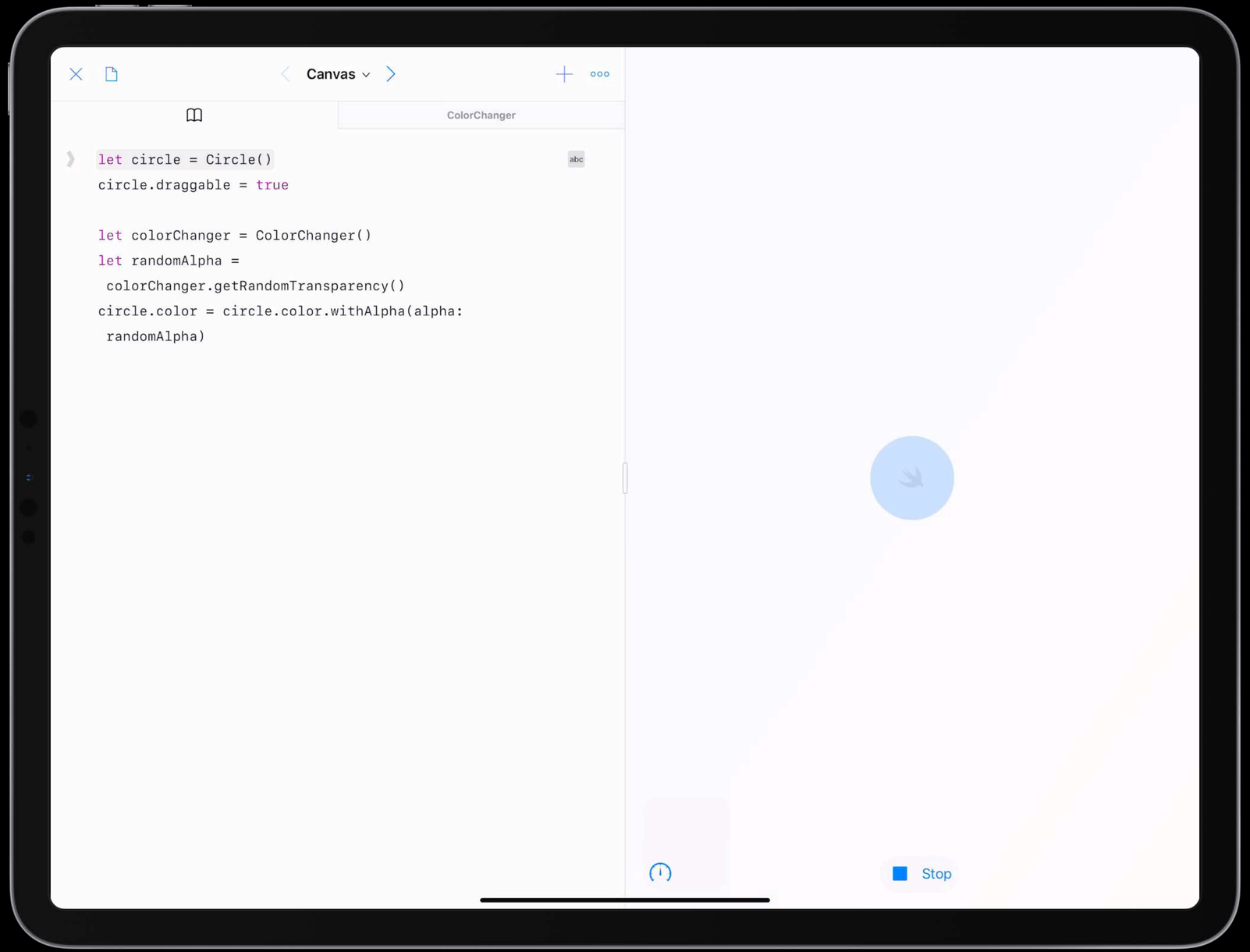

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

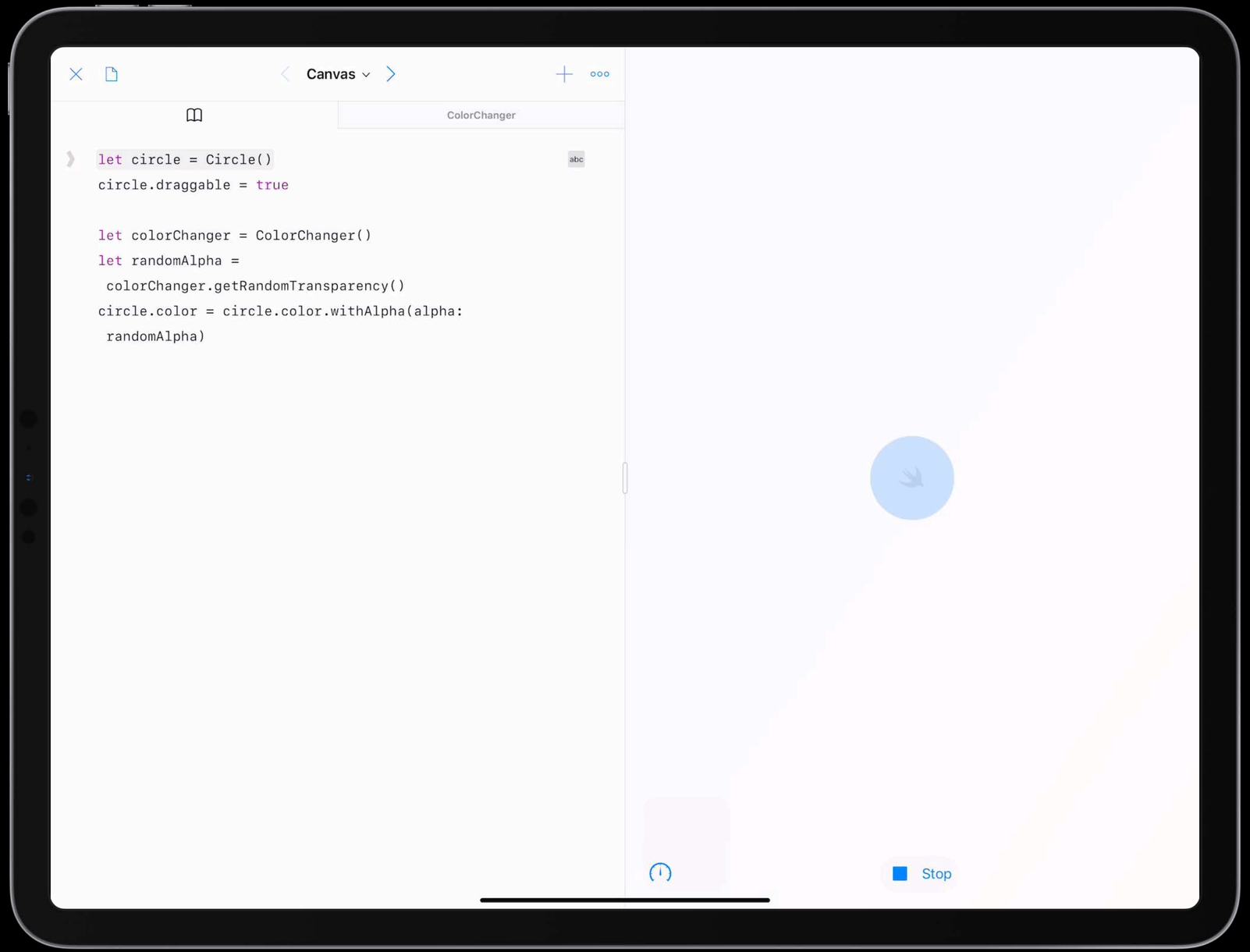

Modules
Edit
MyFiles
Math
Graphics

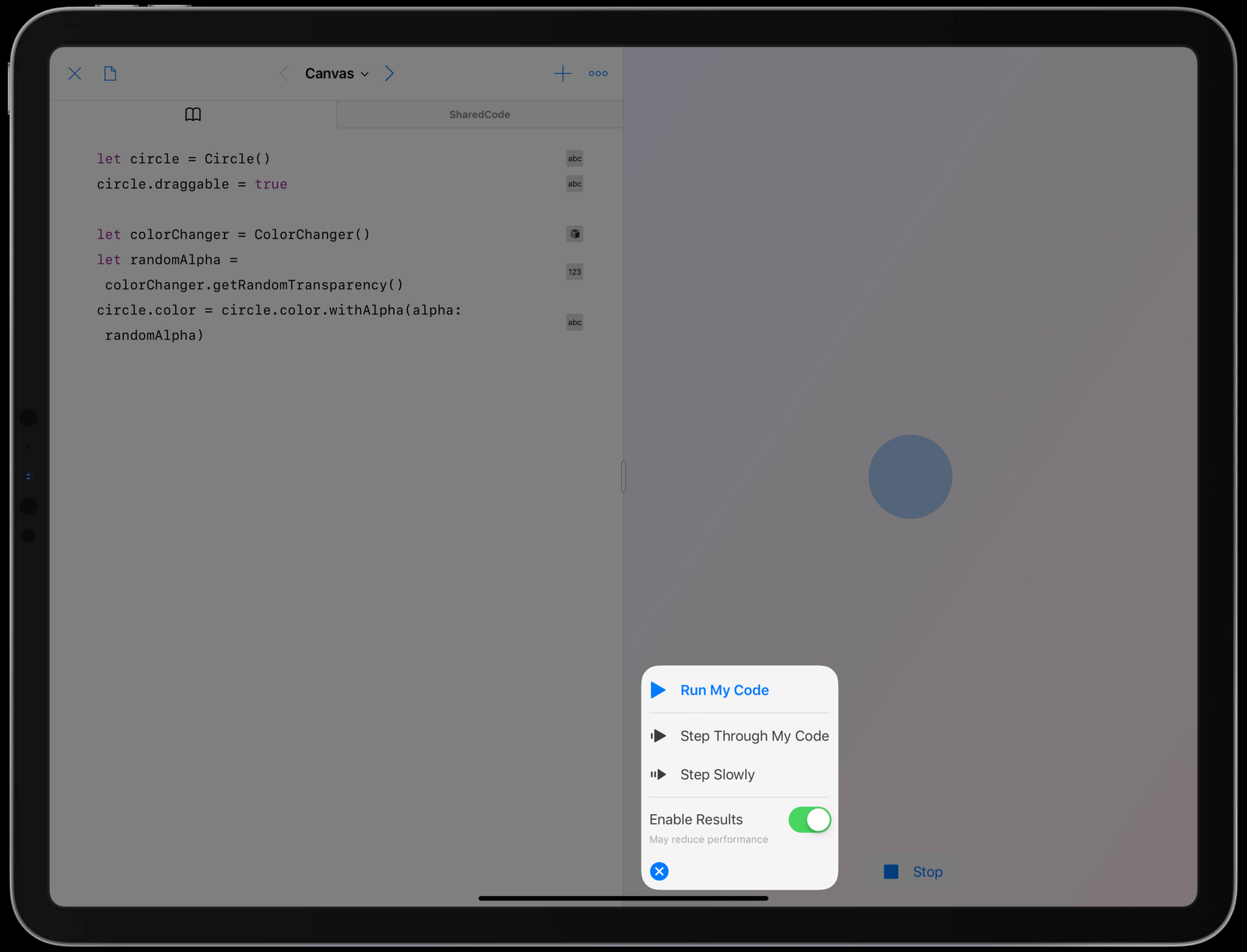

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

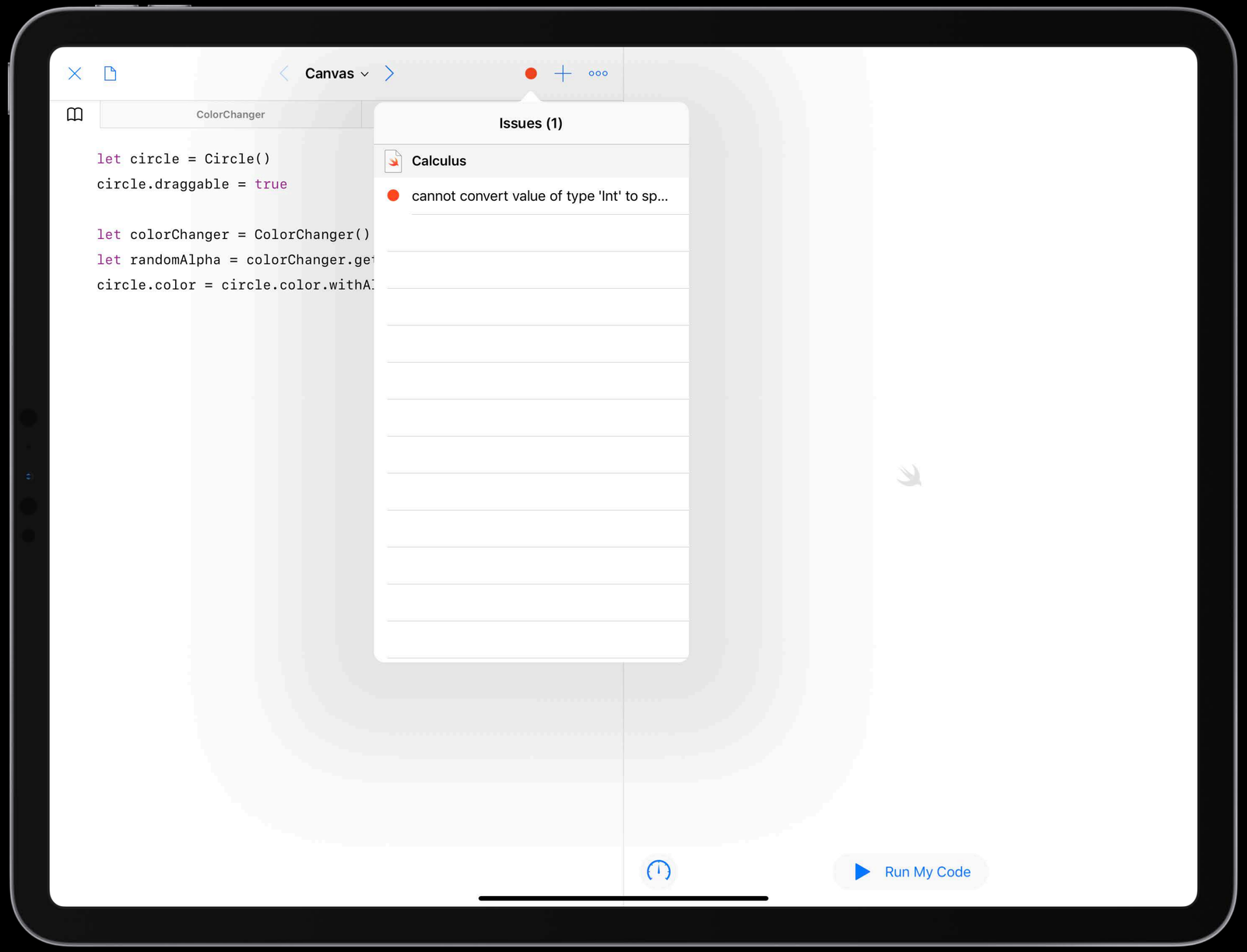

Modules
Edit
MyFiles
Math
Graphics

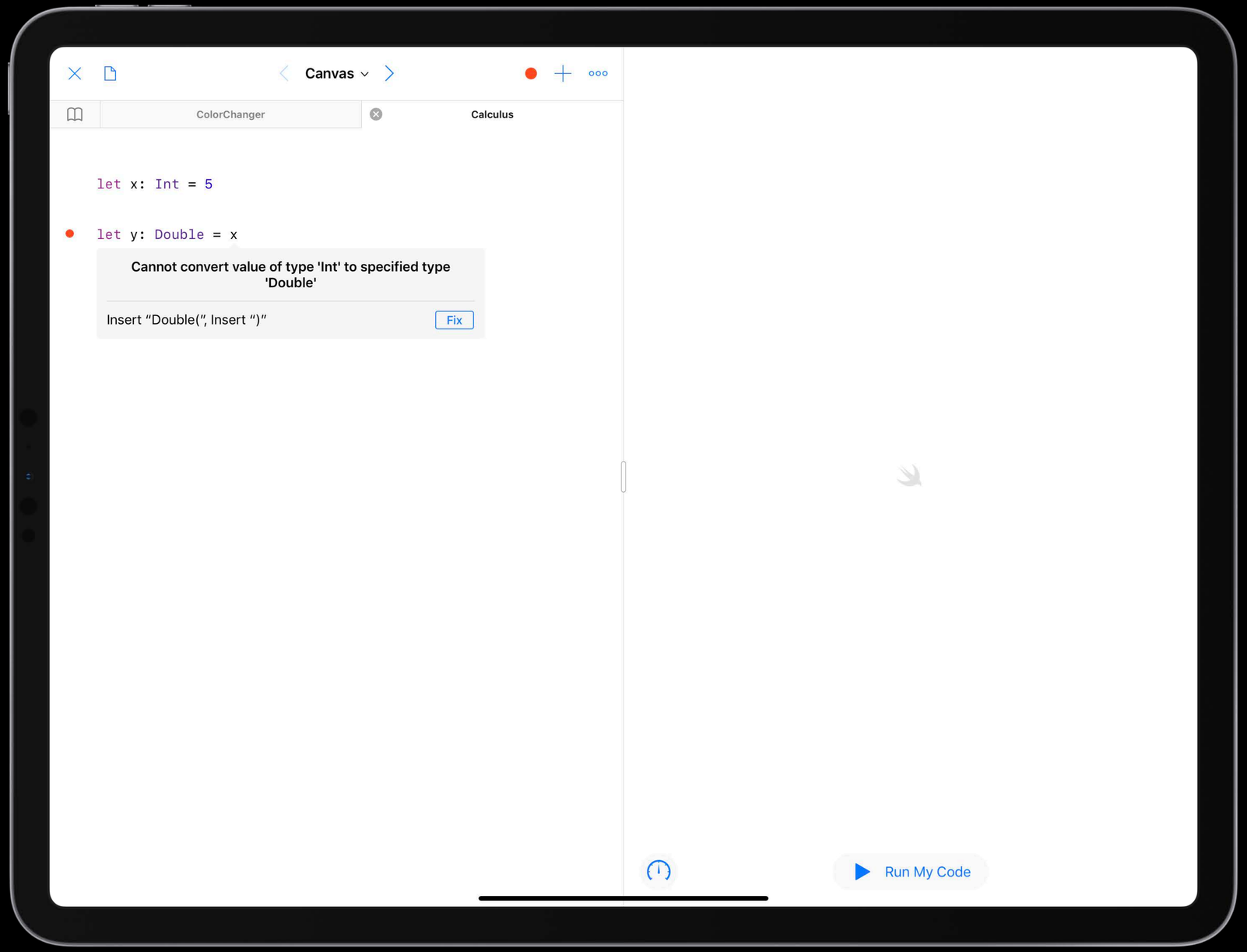

Modules
Math
Edit
Euclidean.swift
Calculus.swift
DiffEQ.swift

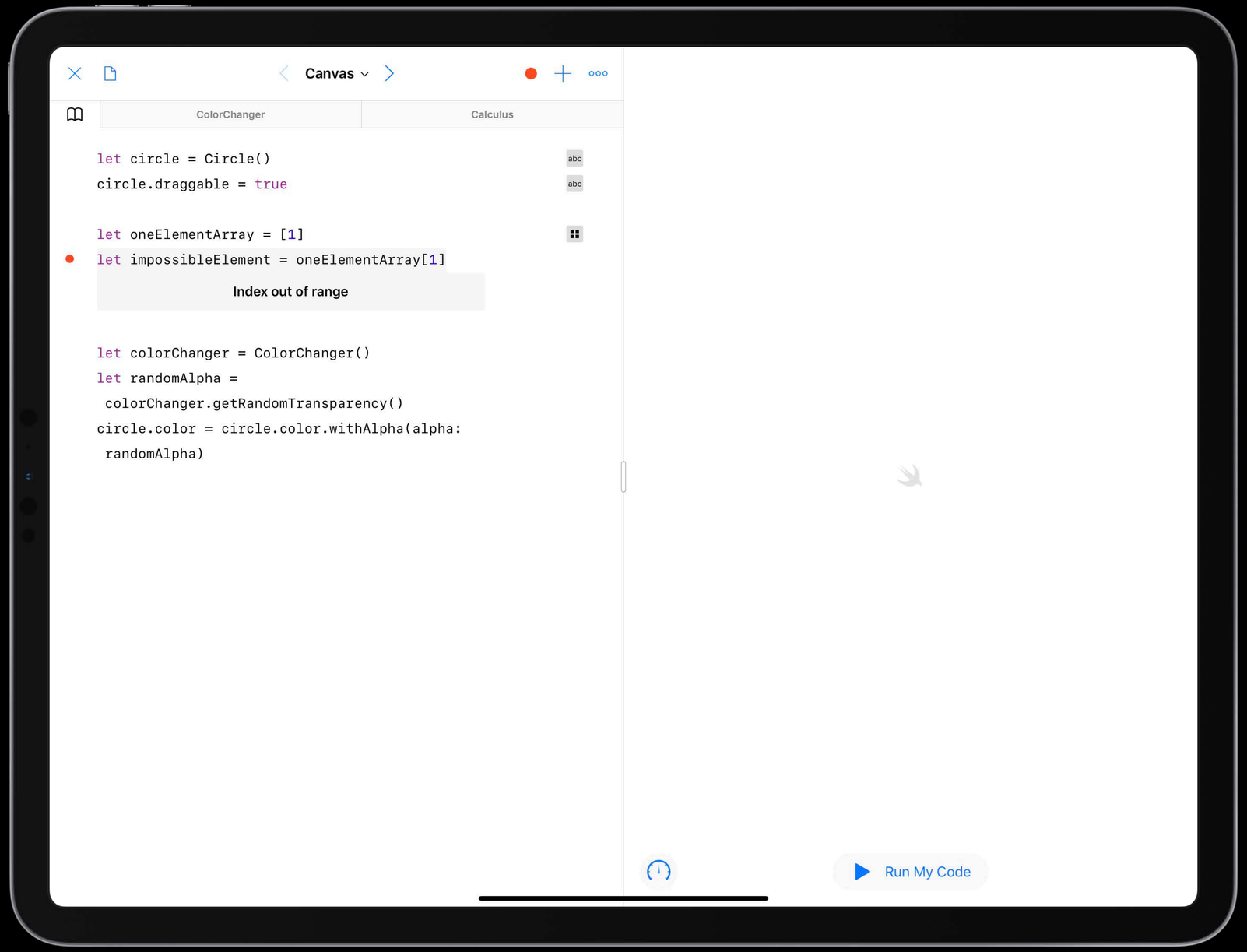

Canvas
ColorChanger
let circle = Circle()
circle.draggable = true
let colorChanger = ColorChanger()
let randomAlpha =
colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha:
randomAlpha)
abc
Stop


Canvas
ColorChanger
let circle = Circle()
circle.draggable = true
let colorChanger = ColorChanger()
let randomAlpha =
colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha:
randomAlpha)
abc
Stop


Canvas
SharedCode
let circle = Circle()
circle.draggable = true
let colorChanger = ColorChanger()
let randomAlpha =
colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha:
randomAlpha)
Run My Code
Step Through My Code
Step Slowly
Enable Results
May reduce performance
Stop


Canvas
ColorChanger
let circle = Circle()
circle.draggable = true
let colorChanger = ColorChanger()
let randomAlpha = colorChanger.ge
circle.color = circle.color.withA
Issues (1)
Calculus
cannot convert value of type 'Int' to sp...
Run My Code


Canvas
ColorChanger
Calculus
let x: Int = 5
let y: Double = x
Cannot convert value of type 'Int' to specified type 'Double'
Insert “Double(”, Insert “)”
Fix
Run My Code


Canvas
ColorChanger
Calculus
let circle = Circle()
circle.draggable = true
let oneElementArray = [1]
let impossibleElement = oneElementArray[1]
Index out of range
let colorChanger = ColorChanger()
let randomAlpha =
colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha:
randomAlpha)
Run My Code

A Character Is Born!

Goal: Set your character on a plane.

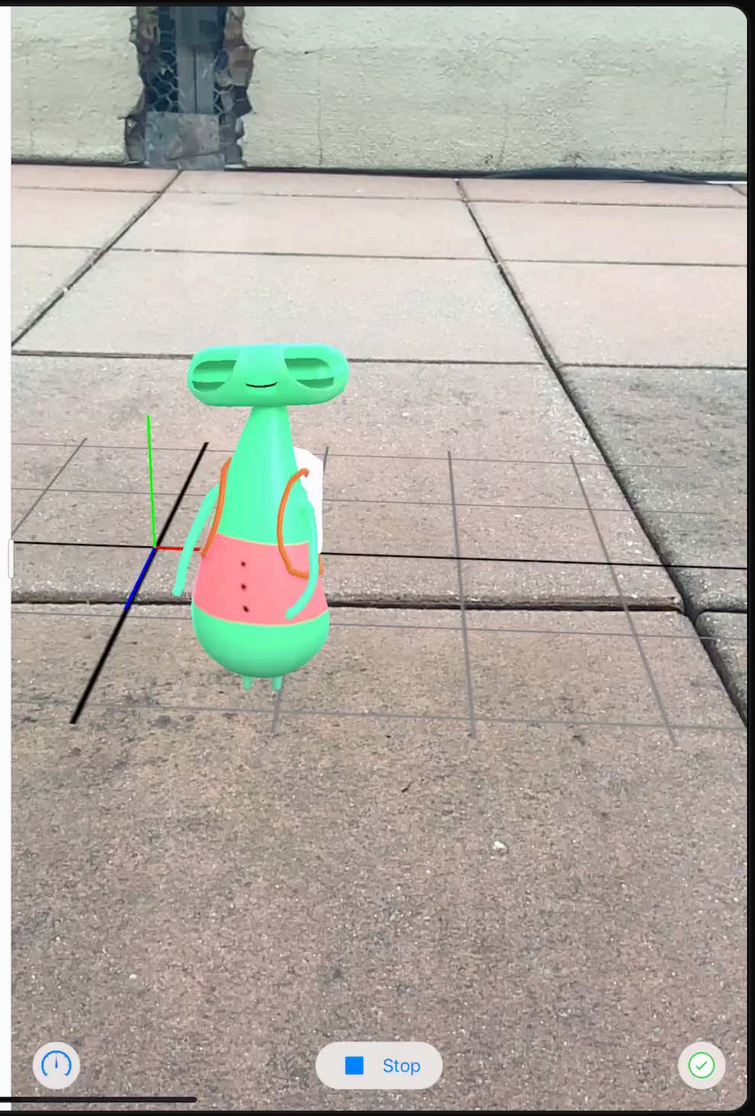
Before you bring Byte & Friends into the world, you need to decide where you want to place them, just like in Learn to Code 2.

Using code, you'll create a new character object, decide on a `Point` you want to place the character, and then place the character on each plane you discover. Use the *X* and *Z* values to place a character. As you change these values, the character will move backward, forward, right, and left on top of the plane.

Try this:

1. Create a new character object, like this: `let hopper = Character(name: CharacterName.hopper)`
2. Create a new position, using a `Point`, like this: `let newPosition = Point(x: 1, z: 2)`
3. Call `plane.place(character:at:)` using the character and position you created, like this: `plane.place(character: hopper, at: newPosition)`

```
func detectedPlane(plane: Plane) {
    playSound(.boing)
    // Create your character and Point objects here
    let hopper = Character(name: CharacterName.hopper)
    let newPosition = Point(x: 1, z: 2)

    plane.place(character: hopper, at: newPosition)
}
```



A Character Is Born!

Goal: Set your character on a plane.

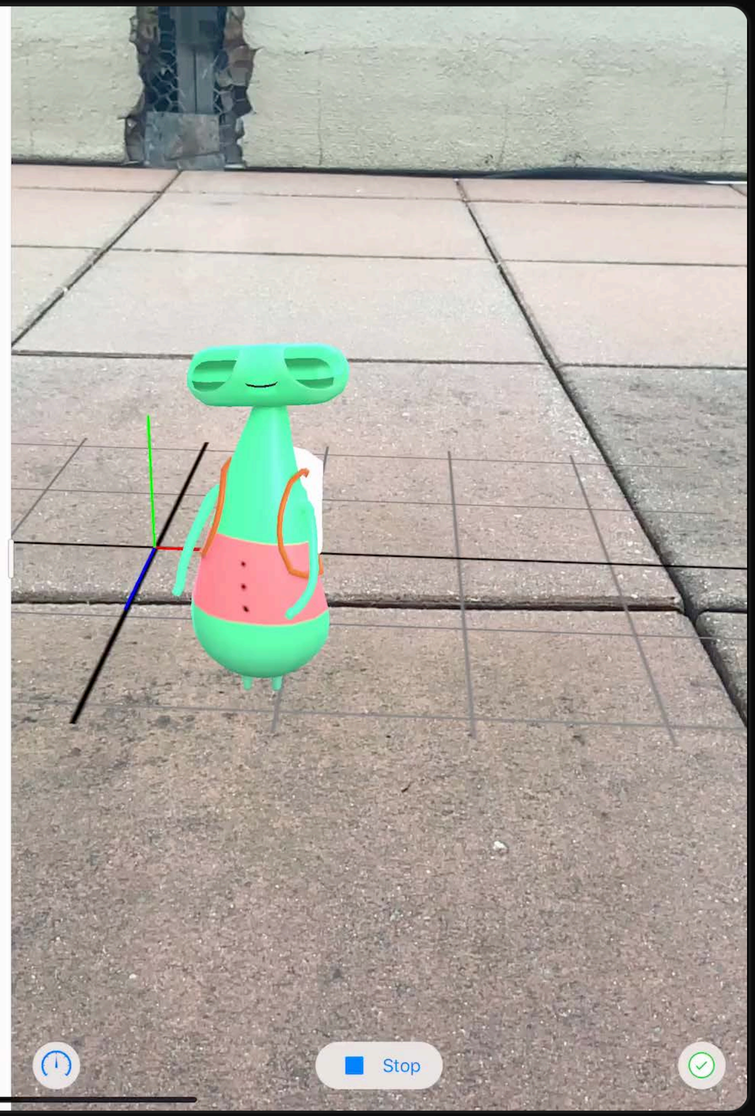
Before you bring Byte & Friends into the world, you need to decide where you want to place them, just like in Learn to Code 2.

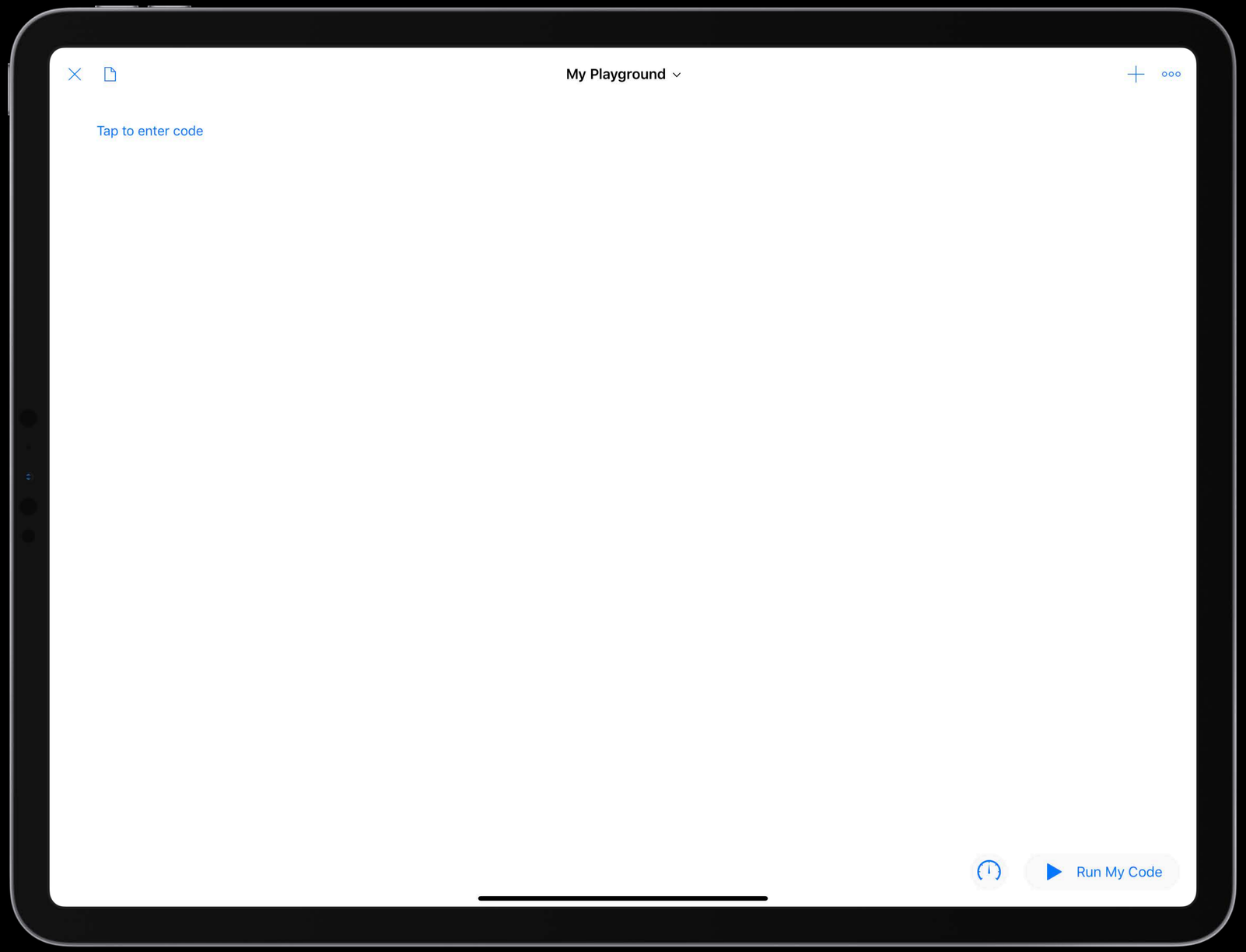
Using code, you'll create a new character object, decide on a `Point` you want to place the character, and then place the character on each plane you discover. Use the *X* and *Z* values to place a character. As you change these values, the character will move backward, forward, right, and left on top of the plane.

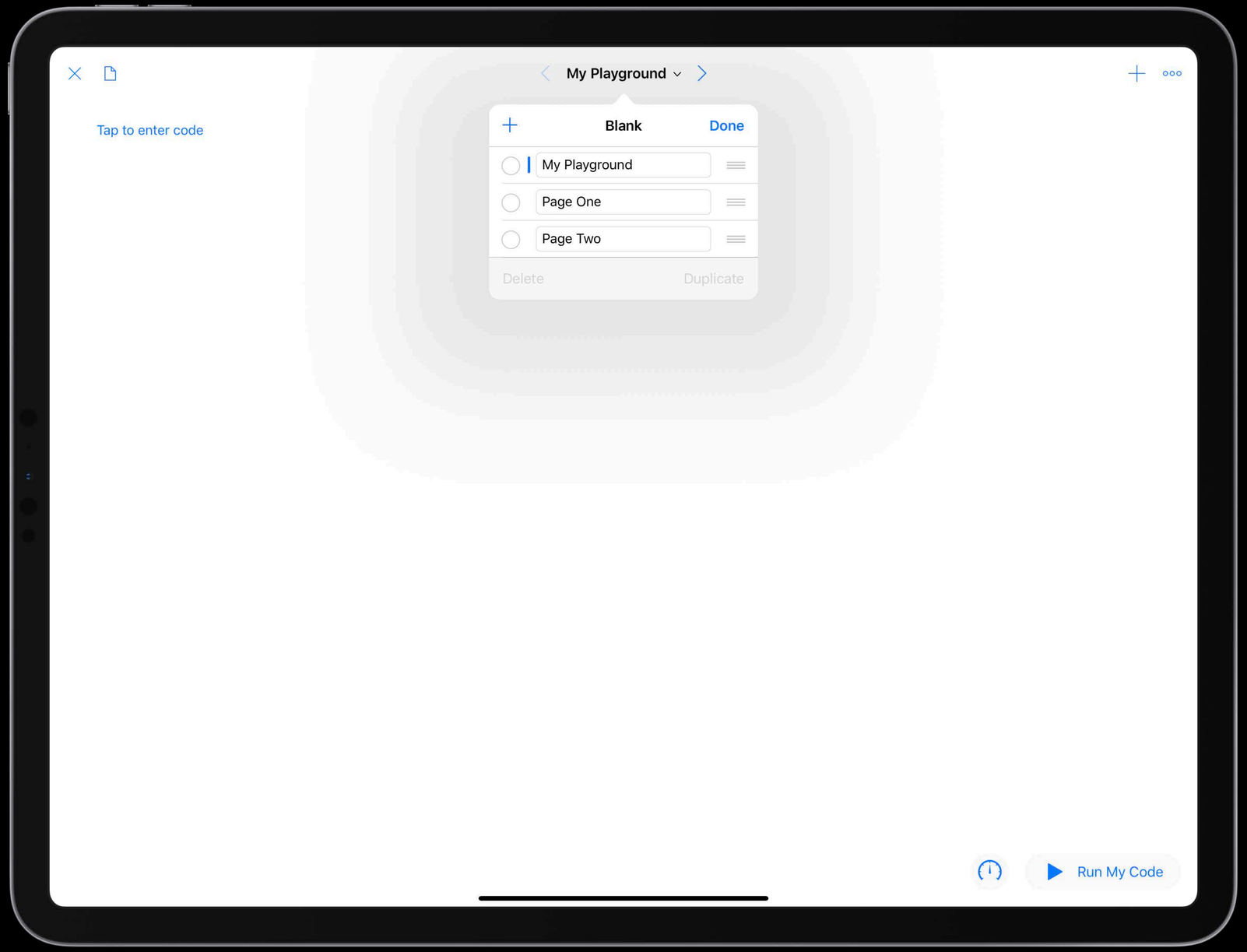
Try this:

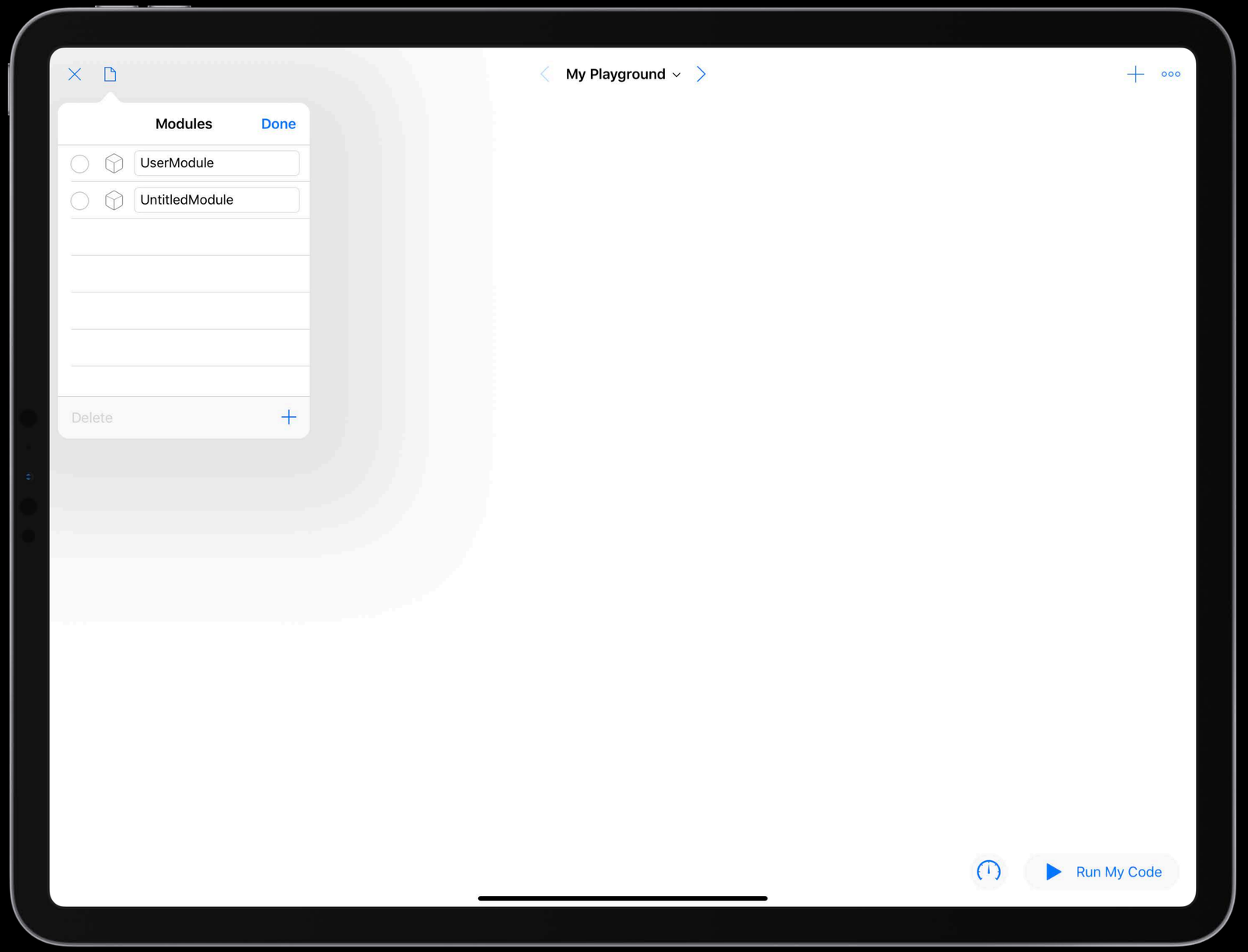
1. Create a new character object, like this: `let hopper = Character(name: CharacterName.hopper)`
2. Create a new position, using a `Point`, like this: `let newPosition = Point(x: 1, z: 2)`
3. Call `plane.place(character:at:)` using the character and position you created, like this: `plane.place(character: hopper, at: newPosition)`

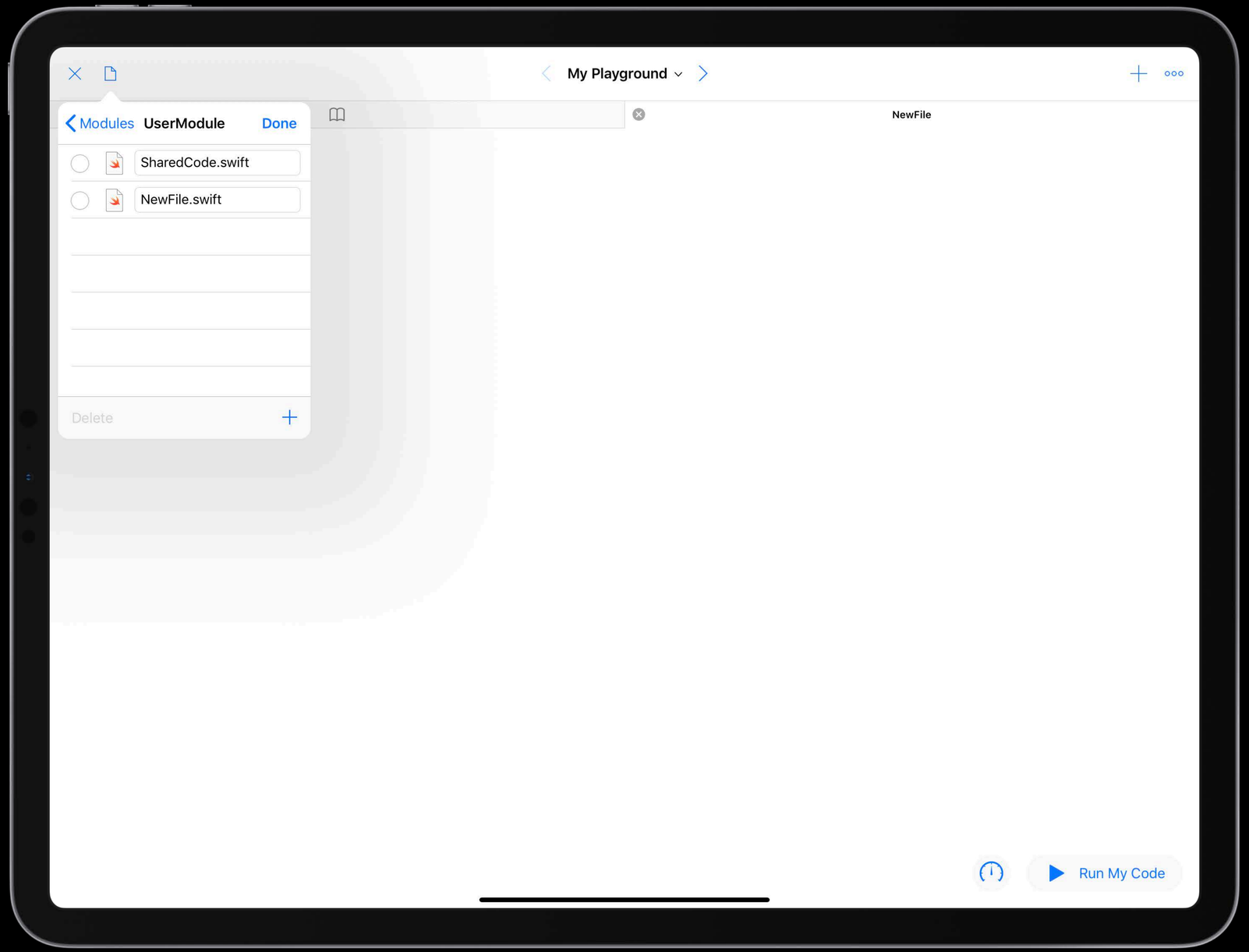
```
func detectedPlane(plane: Plane) {
    playSound(.boing)
    // Create your character and Point objects here
    let hopper = Character(name: CharacterName.hopper)
    let newPosition = Point(x: 1, z: 2)

    plane.place(character: hopper, at: newPosition)
}
```

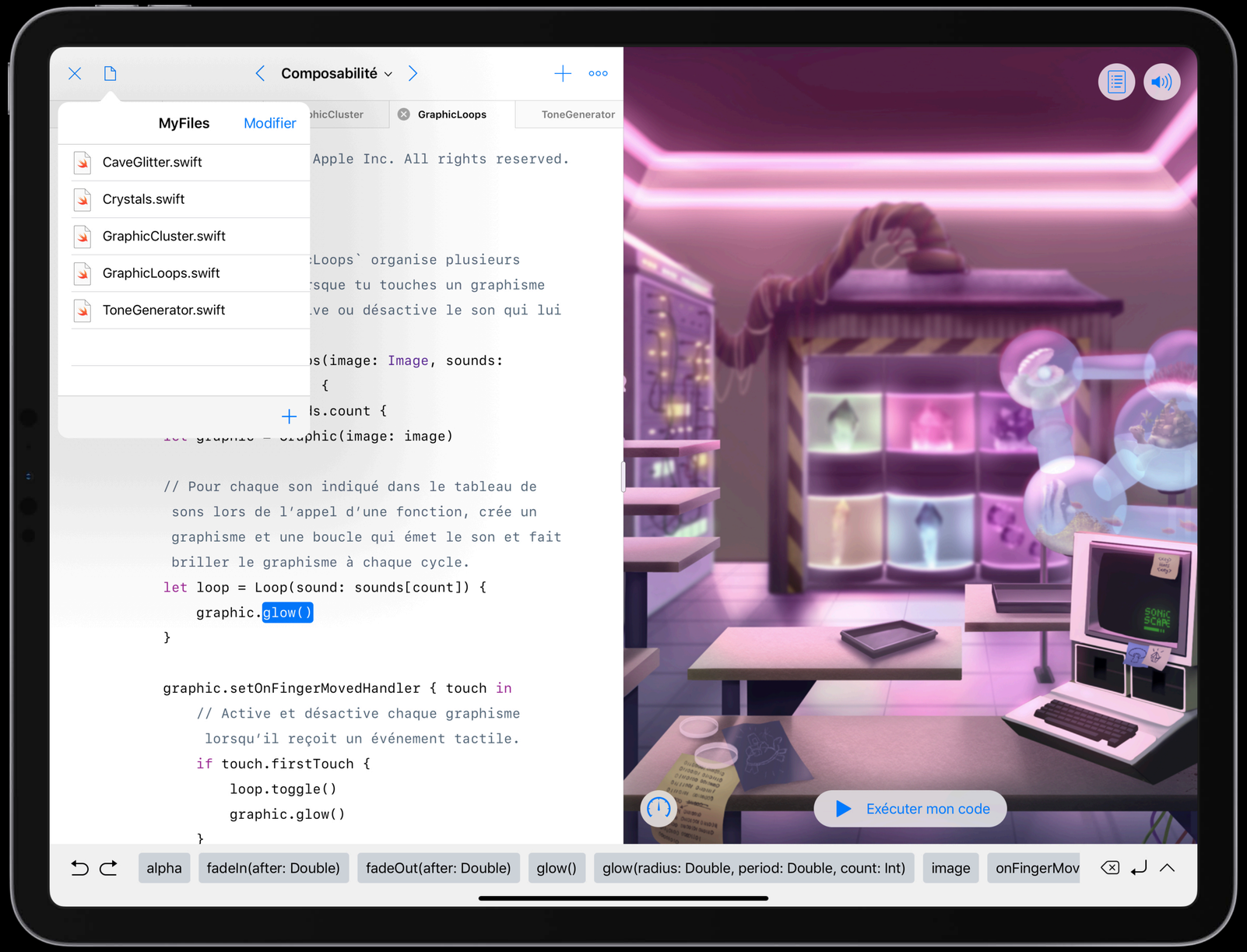



My Playground
Tap to enter code
Run My Code


My Playground
Tap to enter code
Blank
Done
My Playground
Page One
Page Two
Delete
Duplicate
Run My Code

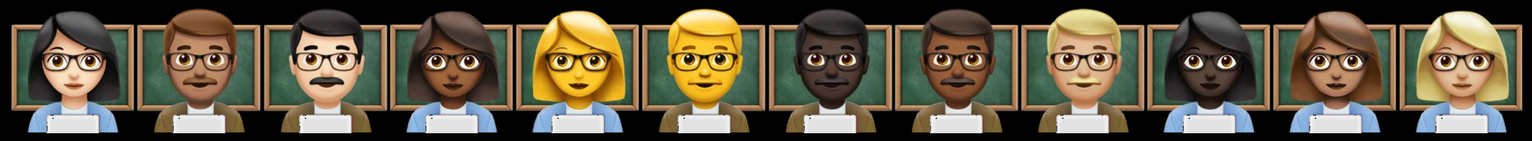

Modules
Done
UserModule
UntitledModule
Delete
My Playground
Run My Code

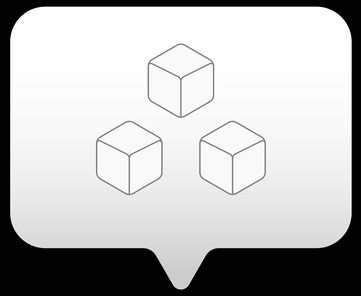

My Playground
Modules
UserModule
Done
SharedCode.swift
NewFile.swift
Delete
NewFile
Run My Code


Composabilité
MyFiles
Modifier
CaveGlitter.swift
Crystals.swift
GraphicCluster.swift
GraphicLoops.swift
ToneGenerator.swift
GraphicCluster
GraphicLoops
ToneGenerator
Apple Inc. All rights reserved.
Loops` organise plusieurs
rsque tu touches un graphisme
ve ou désactive le son qui lui
s(image: Image, sounds:
{
ls.count {
phic(image: image)
// Pour chaque son indiqué dans le tableau de
sons lors de l'appel d'une fonction, crée un
graphisme et une boucle qui émet le son et fait
briller le graphisme à chaque cycle.
let loop = Loop(sound: sounds[count]) {
graphic.glow()
}
graphic.setOnFingerMovedHandler { touch in
// Active et désactive chaque graphisme
lorsqu'il reçoit un événement tactile.
if touch.firstTouch {
loop.toggle()
graphic.glow()
}
Exécuter mon code
alpha
fadeIn(after: Double)
fadeOut(after: Double)
glow()
glow(radius: Double, period: Double, count: Int)
image
onFingerMov

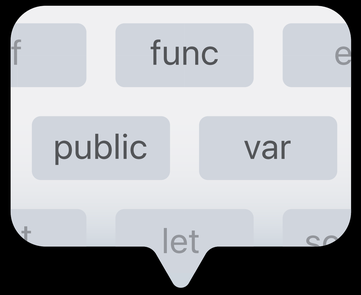
New for Authors

Joy Forbes, Developer Education Engineer





Module Mode

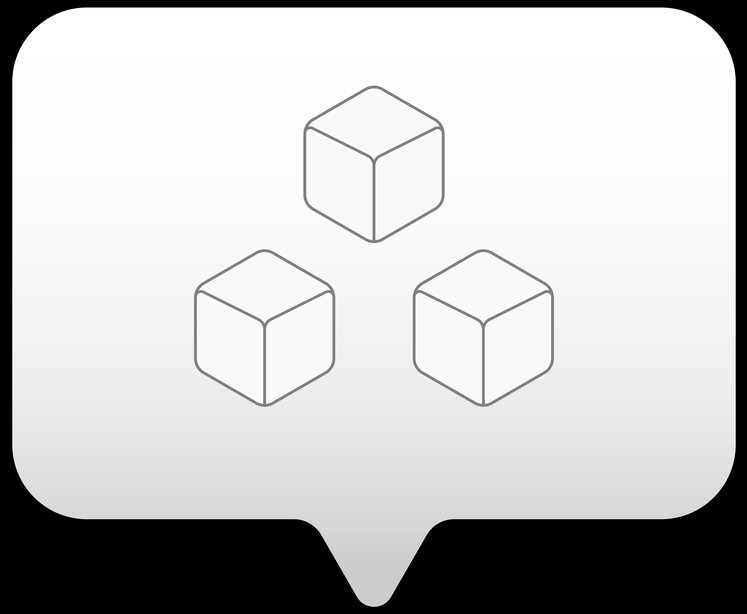


Code Completion

Swift Cutscenes

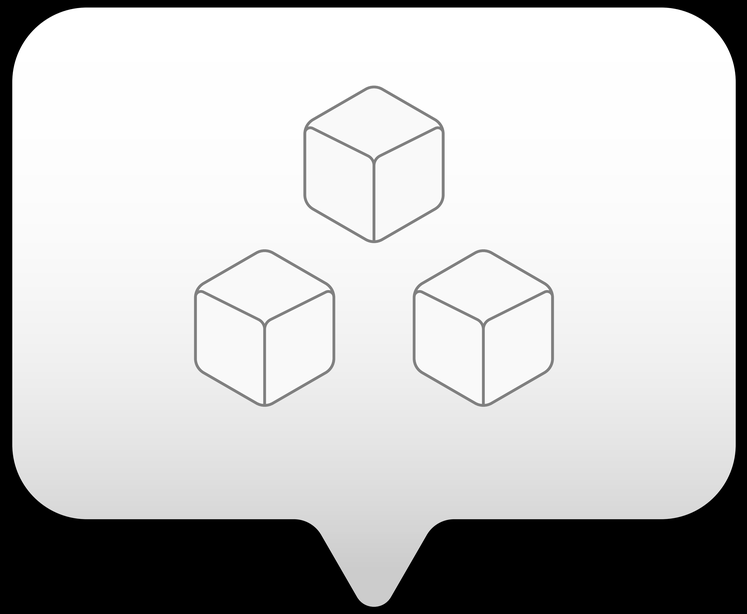
Localized Code Comments

NEW



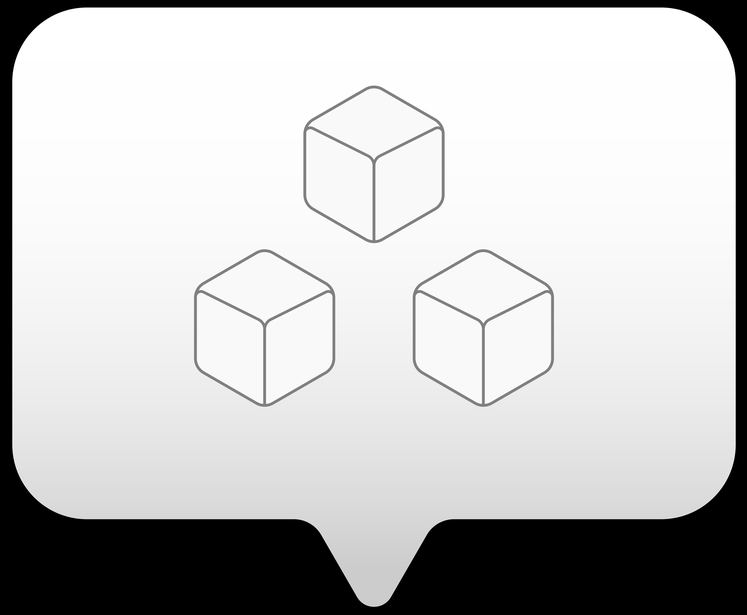
Module Mode

NEW

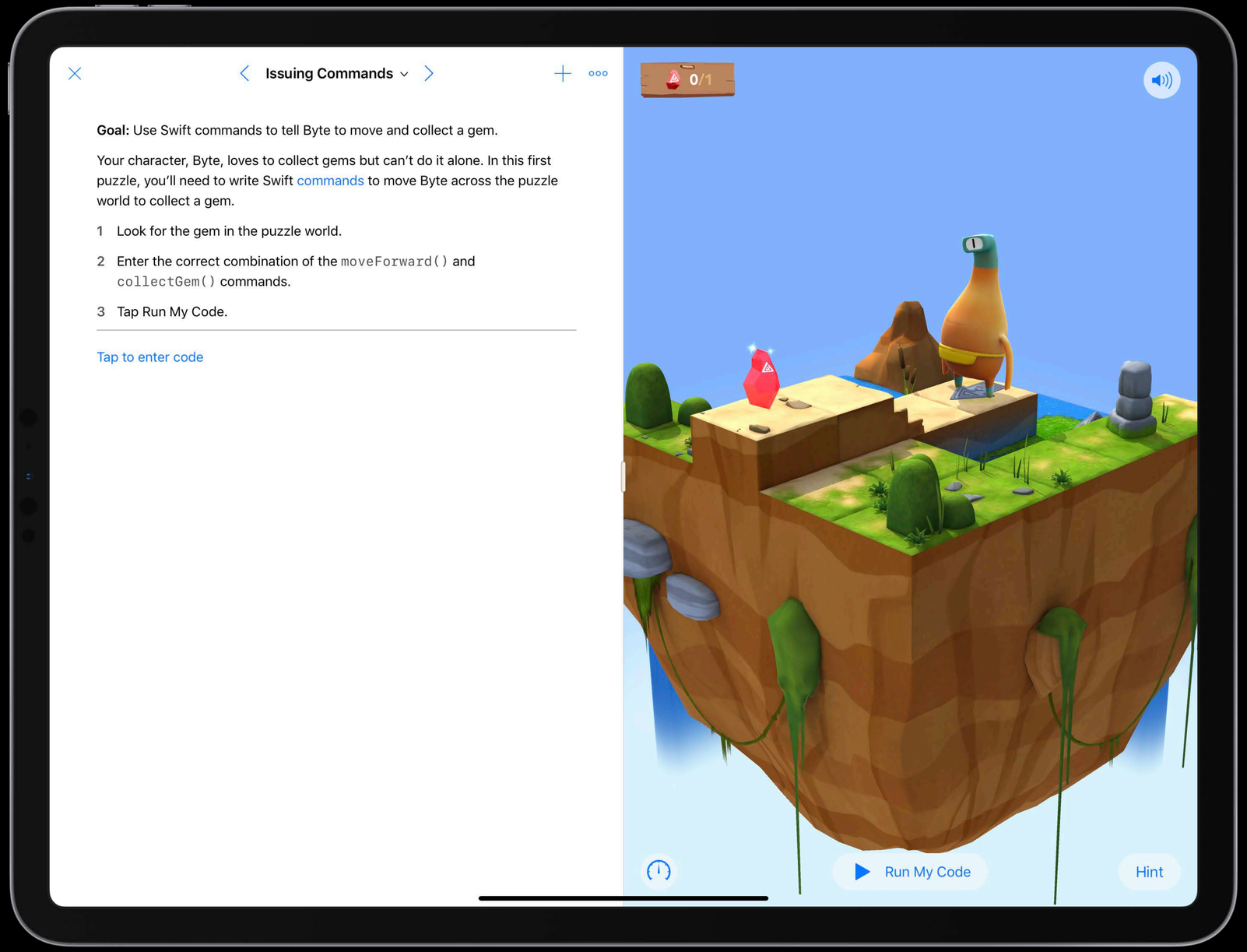


Module Mode

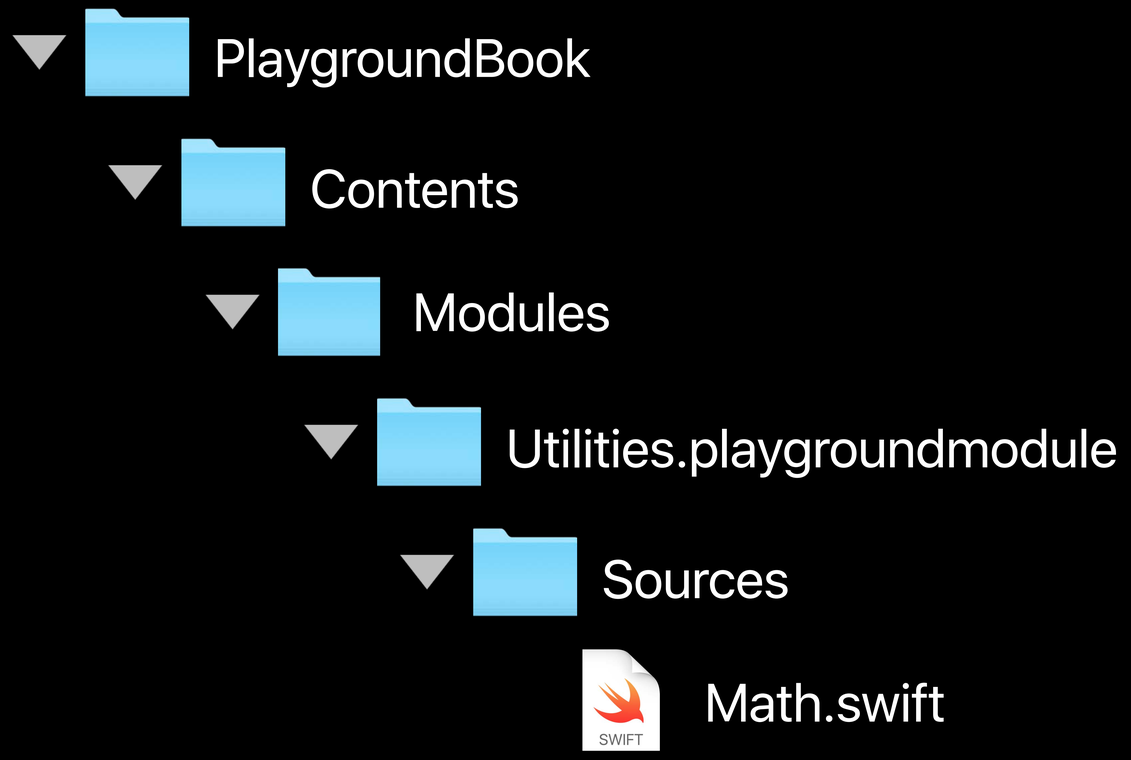
None, Limited, and Full



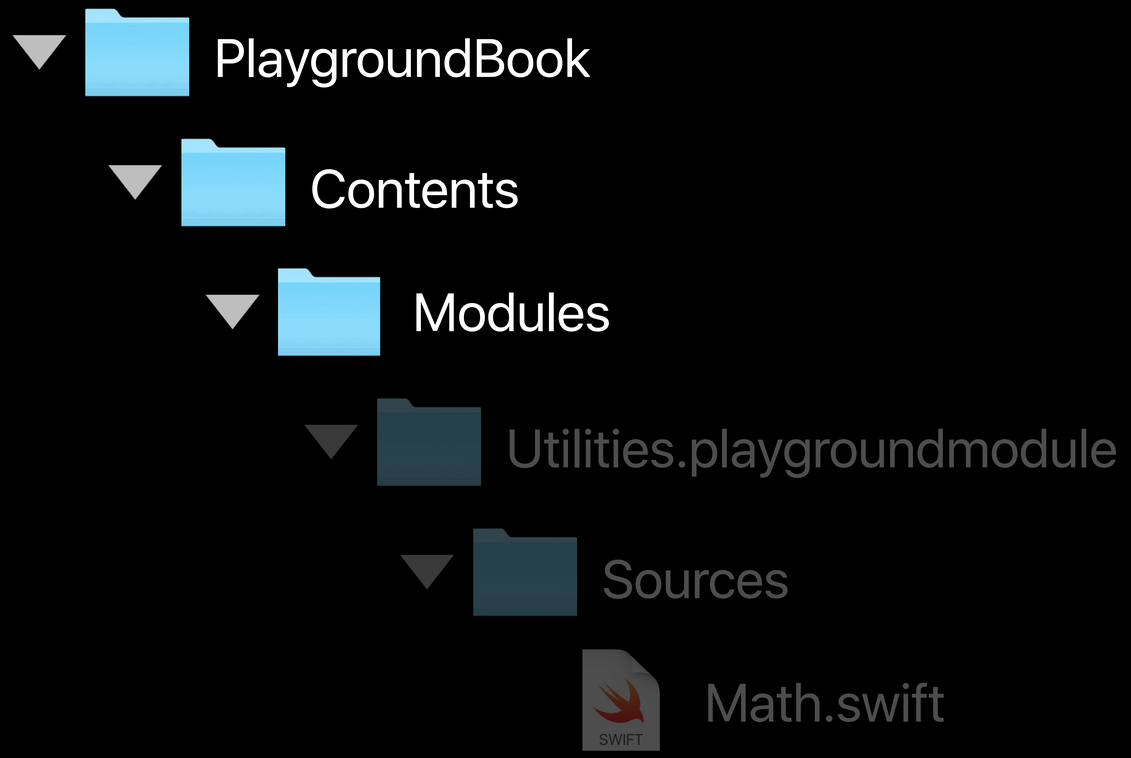
None

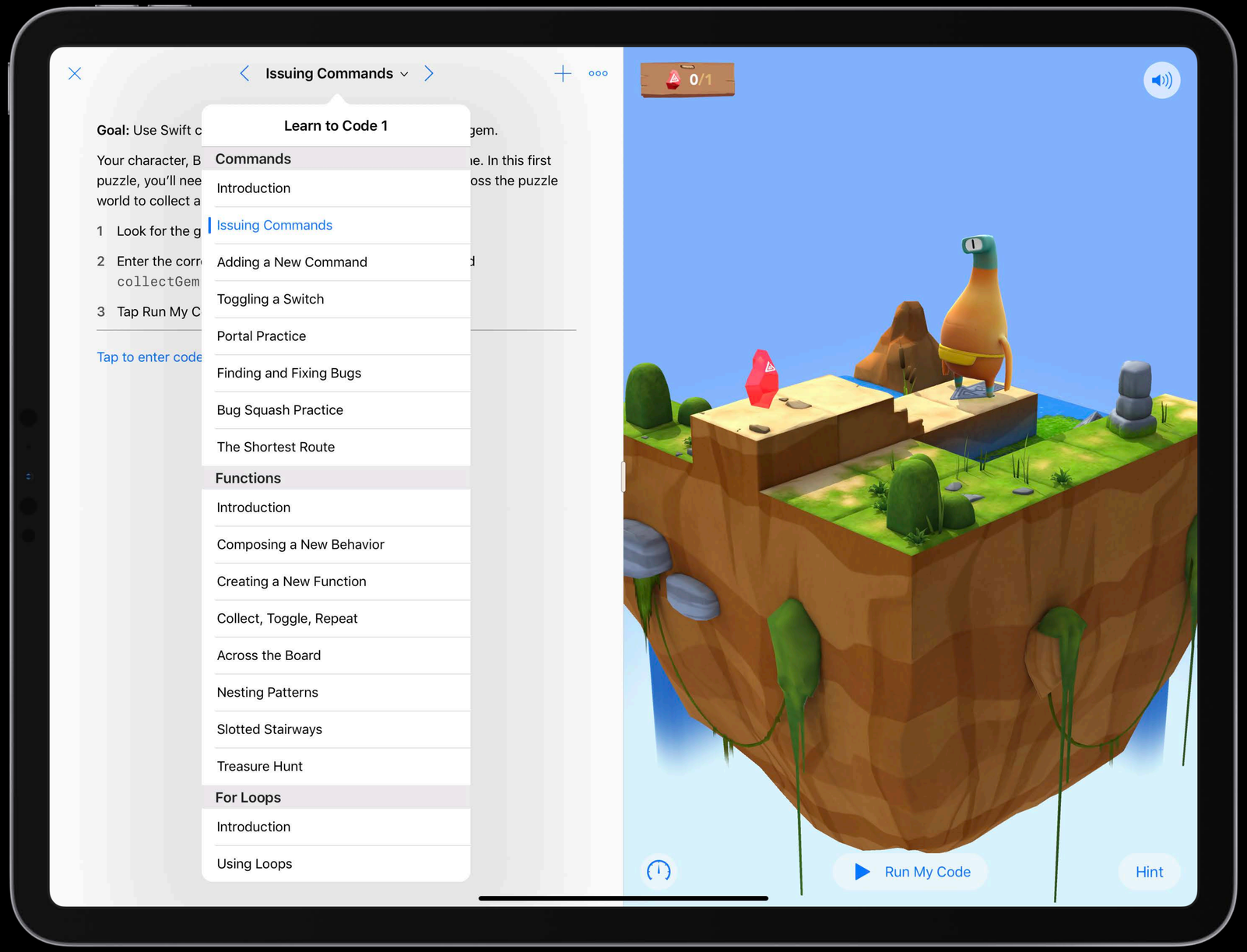

Issuing Commands
Goal: Use Swift commands to tell Byte to move and collect a gem.
Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.
1 Look for the gem in the puzzle world.
2 Enter the correct combination of the moveForward() and collectGem() commands.
3 Tap Run My Code.
Tap to enter code
0/1
Run My Code
Hint

NEW


PlaygroundBook
Contents
Modules
Utilities.playgroundmodule
Sources
SWIFT
Math.swift

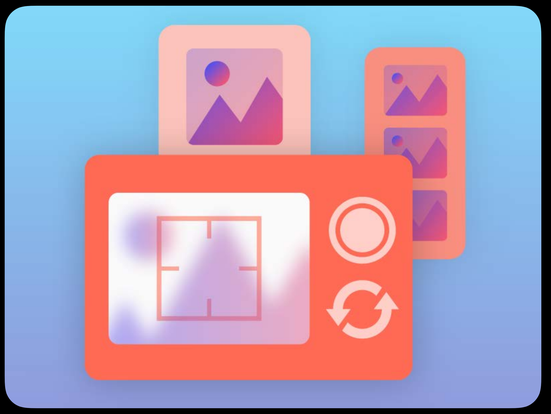
NEW


PlaygroundBook
Contents
Modules
Utilities.playgroundmodule
Sources
SWIFT
Math.swift


Issuing Commands
Learn to Code 1
Commands
Introduction
Issuing Commands
Adding a New Command
Toggling a Switch
Portal Practice
Finding and Fixing Bugs
Bug Squash Practice
The Shortest Route
Functions
Introduction
Composing a New Behavior
Creating a New Function
Collect, Toggle, Repeat
Across the Board
Nesting Patterns
Slotted Stairways
Treasure Hunt
For Loops
Introduction
Using Loops
Goal: Use Swift c
gem.
Your character, B
e. In this first
puzzle, you'll nee
oss the puzzle
world to collect a
1 Look for the g
2 Enter the corr
d
collectGem
3 Tap Run My C
Tap to enter code
0/1
Run My Code
Hint

Learn to Code 1

Learn to Code 2



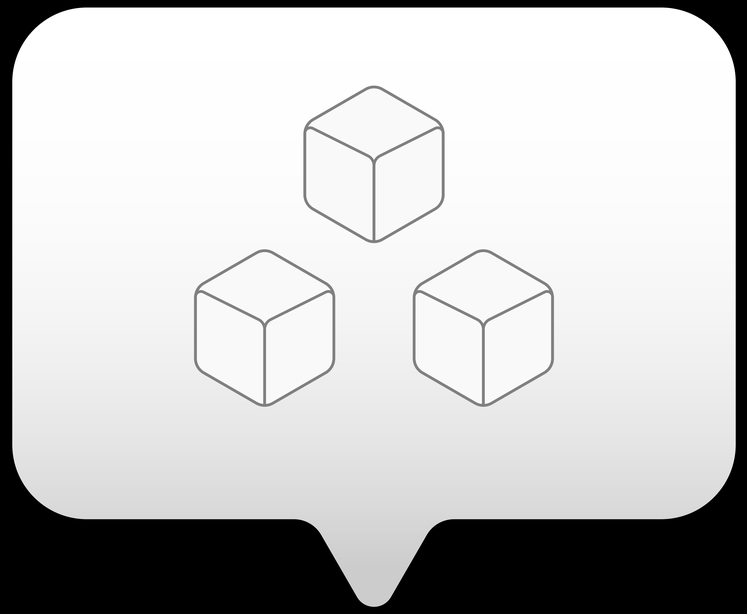
Lights, Camera, Code

Hello Byte

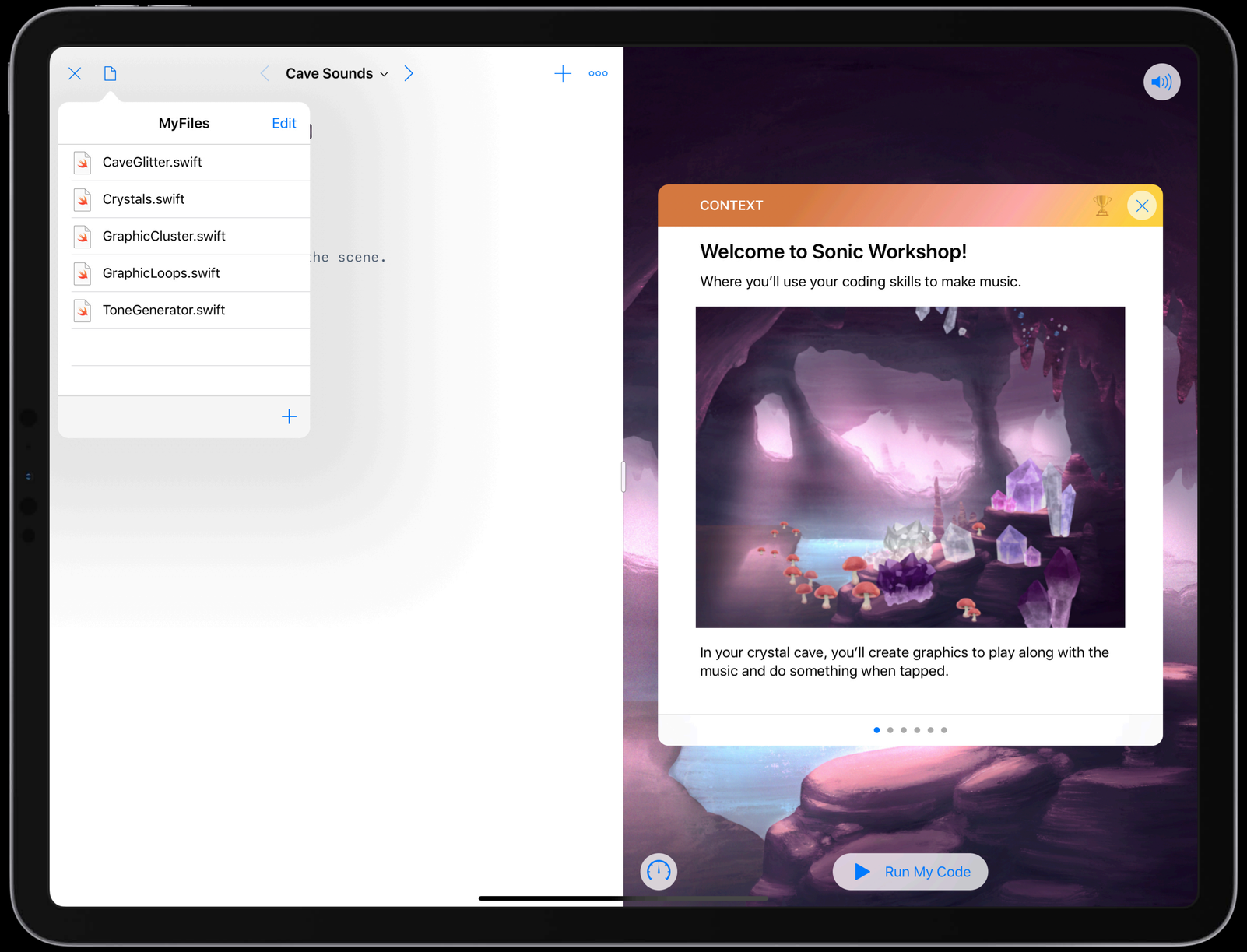
Code Machine

Sensor Arcade

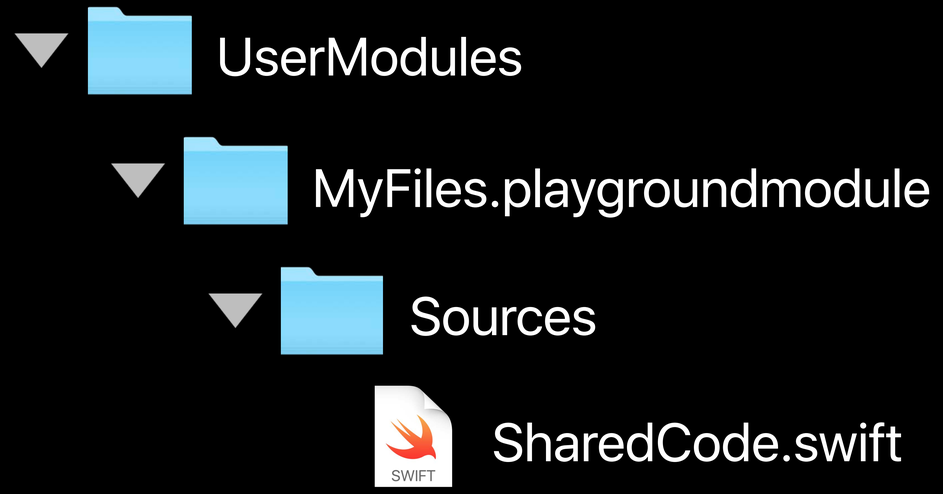
NEW



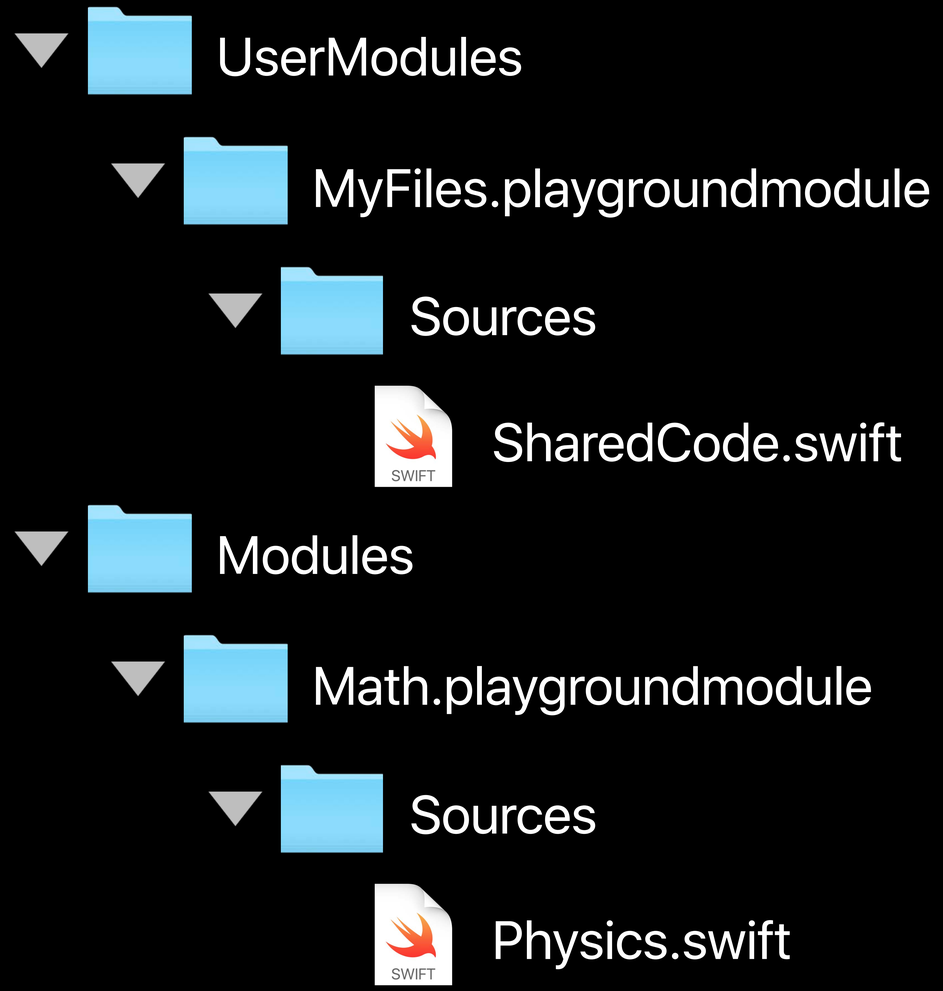
Limited

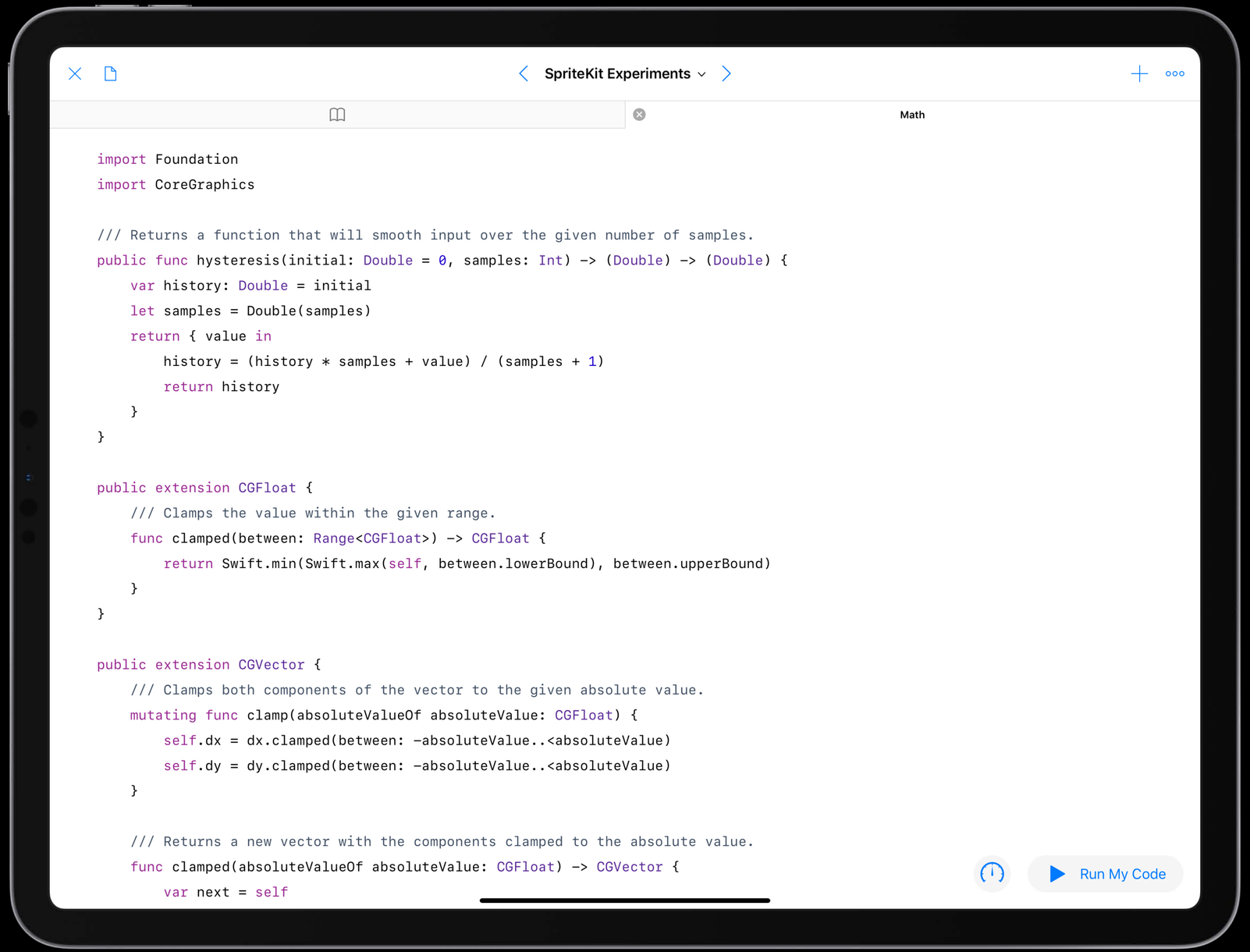

Cave Sounds
MyFiles
Edit
CaveGlitter.swift
Crystals.swift
GraphicCluster.swift
GraphicLoops.swift
ToneGenerator.swift
CONTEXT
Welcome to Sonic Workshop!
Where you'll use your coding skills to make music.
In your crystal cave, you'll create graphics to play along with the music and do something when tapped.
Run My Code

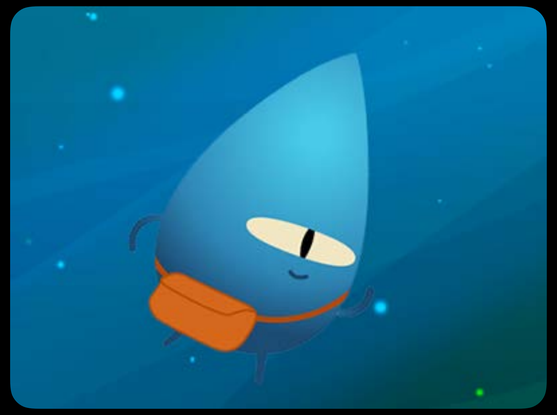
NEW


UserModules
MyFiles.playgroundmodule
Sources
SWIFT
SharedCode.swift

NEW


UserModules
MyFiles.playgroundmodule
Sources
SwiftCode.swift
Modules
Math.playgroundmodule
Sources
Physics.swift


SpriteKit Experiments
Math
import Foundation
import CoreGraphics
/// Returns a function that will smooth input over the given number of samples.
public func hysteresis(initial: Double = 0, samples: Int) -> (Double) -> (Double) {
var history: Double = initial
let samples = Double(samples)
return { value in
history = (history * samples + value) / (samples + 1)
return history
}
}
public extension CGFloat {
/// Clamps the value within the given range.
func clamped(between: Range<CGFloat>) -> CGFloat {
return Swift.min(Swift.max(self, between.lowerBound), between.upperBound)
}
}
public extension CGVector {
/// Clamps both components of the vector to the given absolute value.
mutating func clamp(absoluteValueOf absoluteValue: CGFloat) {
self.dx = dx.clamped(between: -absoluteValue..<absoluteValue)
self.dy = dy.clamped(between: -absoluteValue..<absoluteValue)
}
/// Returns a new vector with the components clamped to the absolute value.
func clamped(absoluteValueOf absoluteValue: CGFloat) -> CGVector {
var next = self
Run My Code

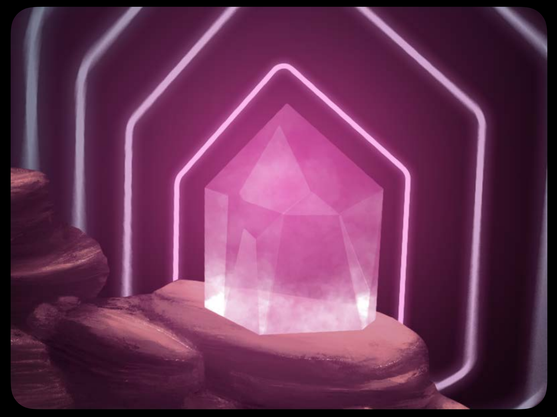


Blu’s Adventure

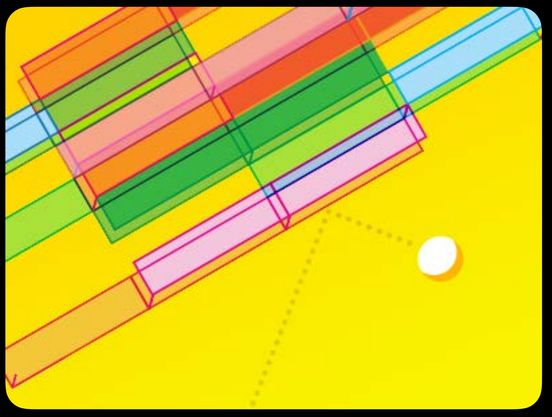
Assemble Your Camera

Flashy Photos

Rock, Paper, Scissors

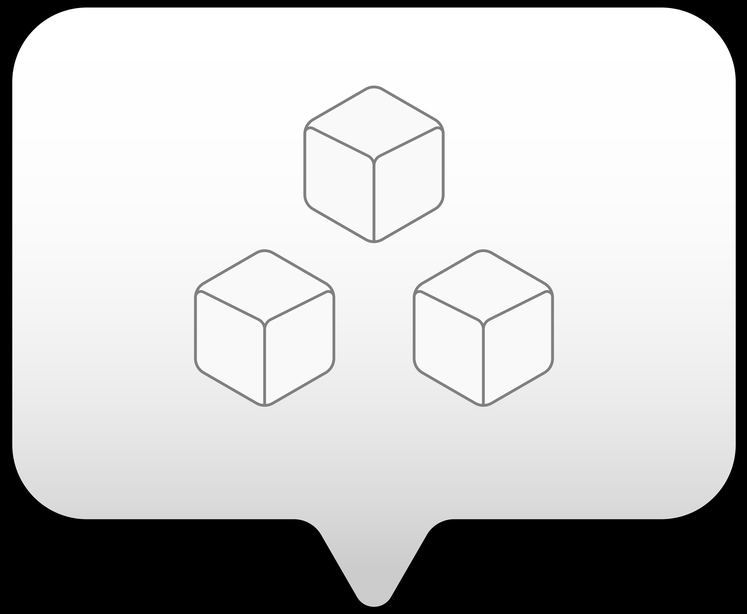


Sonic Workshop

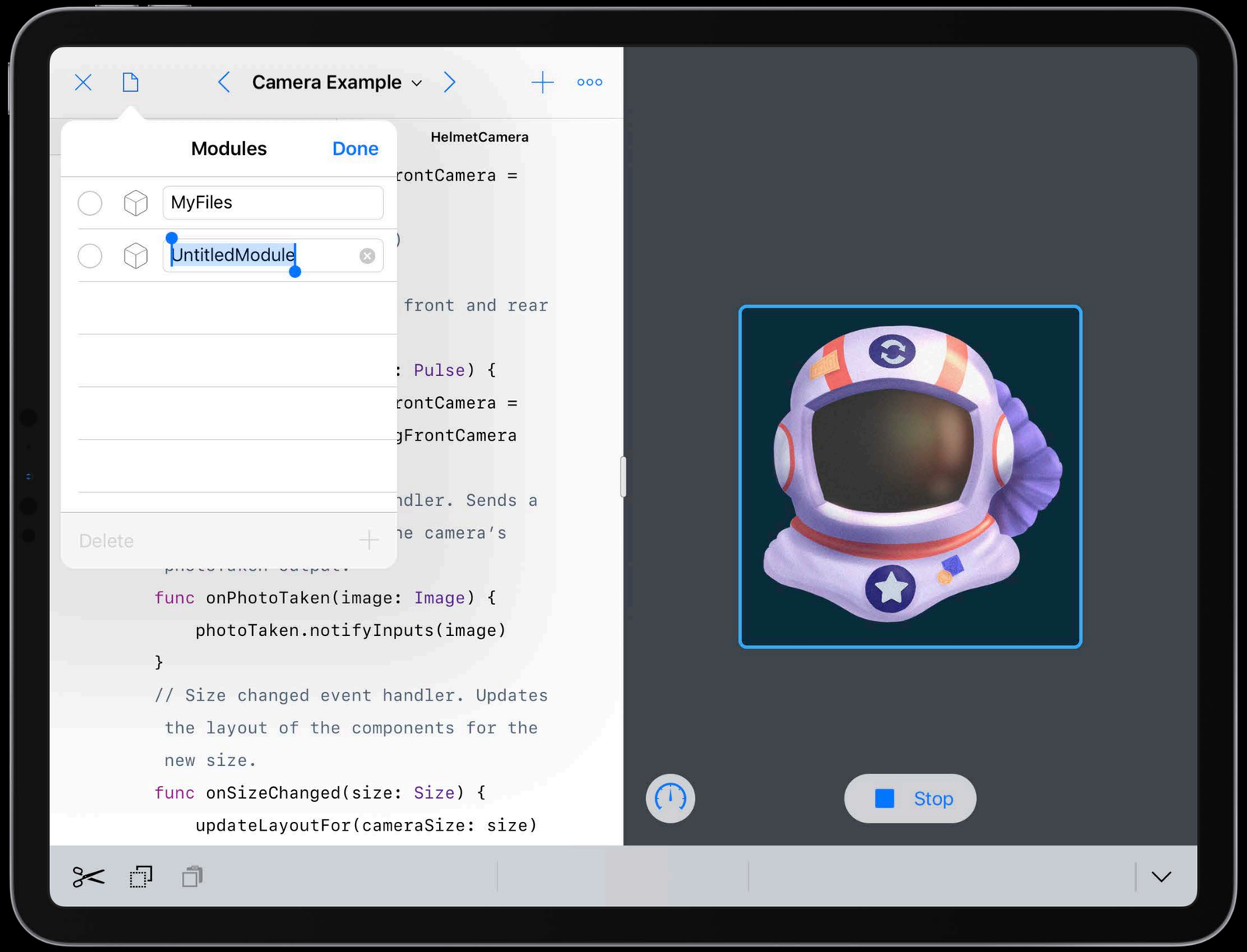


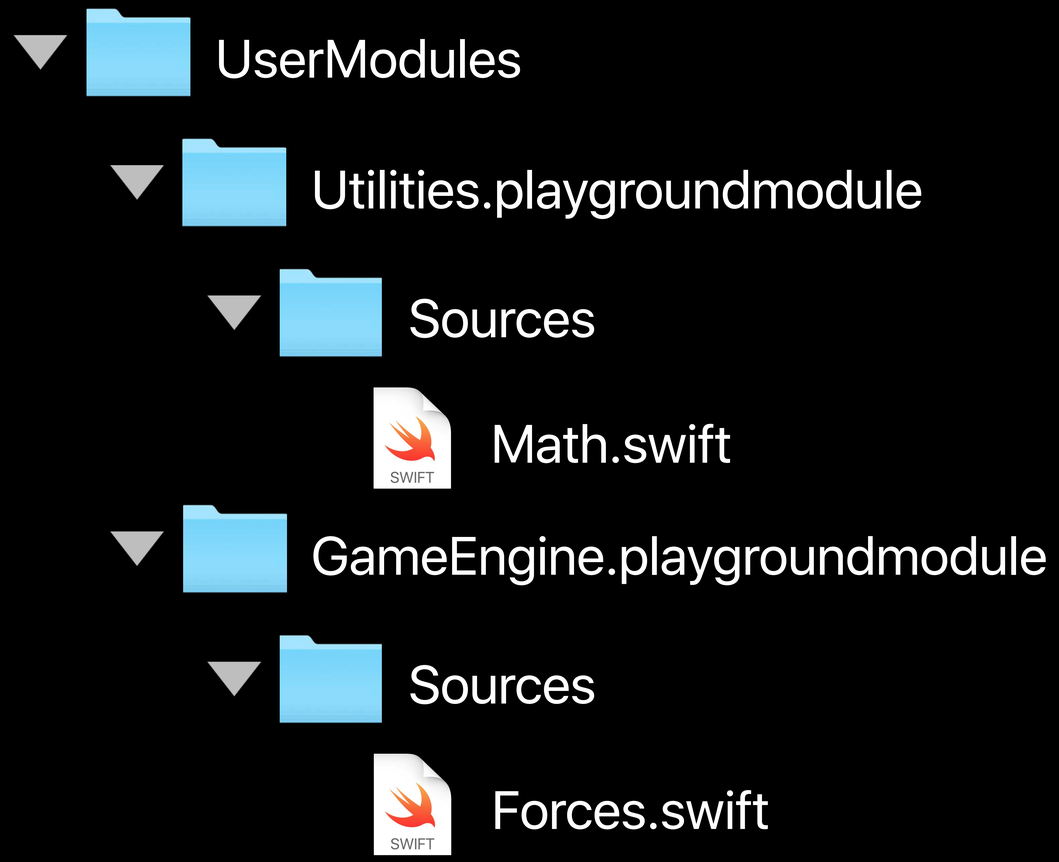
Brick Breaker

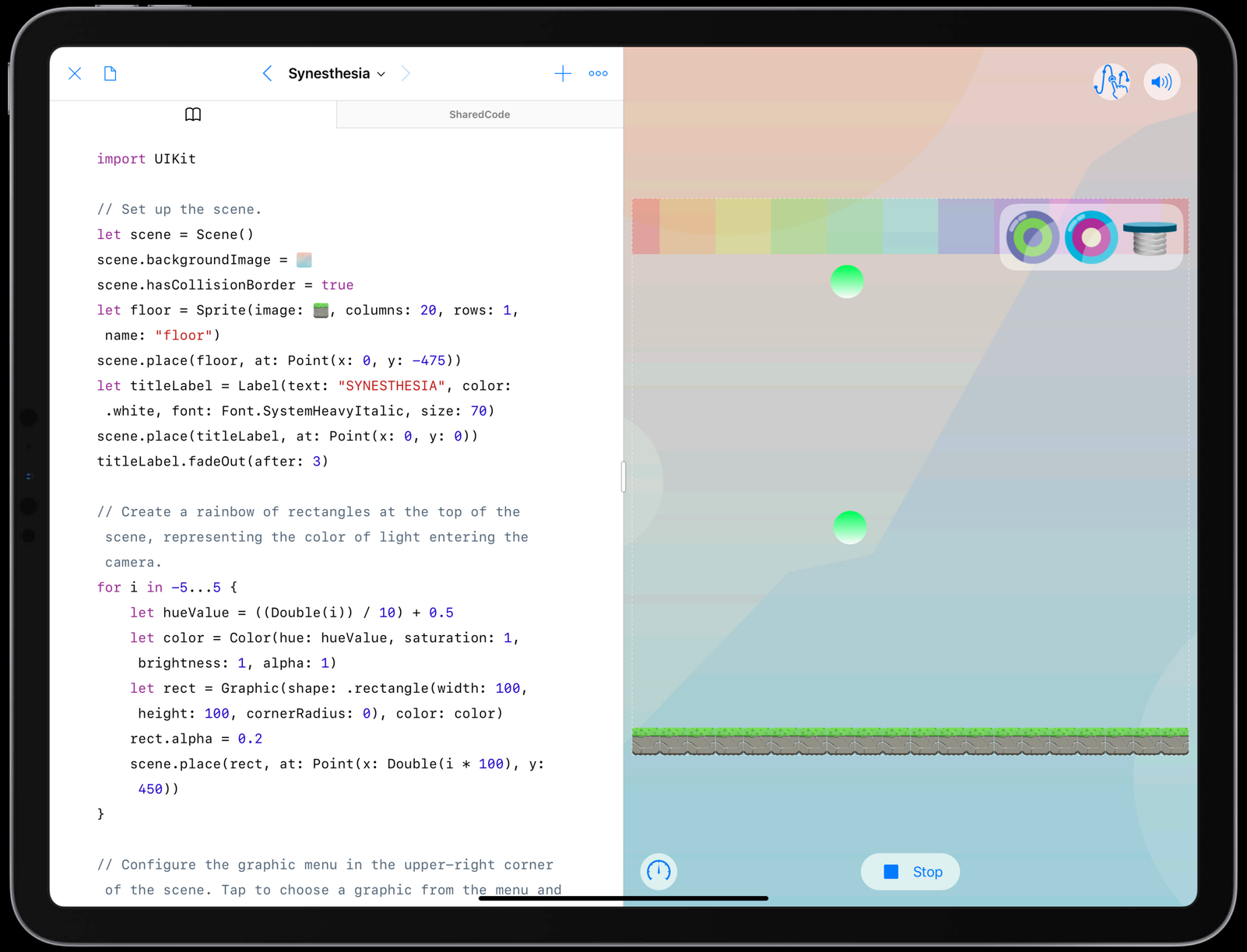
NEW



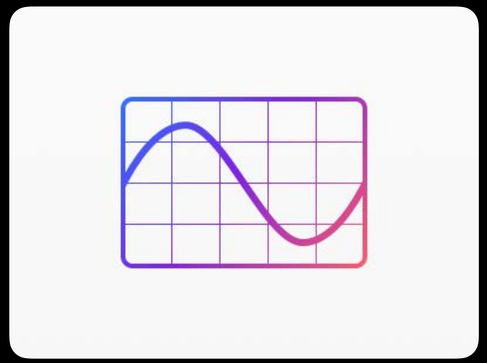
Full


Camera Example
Modules
Done
MyFiles
UntitledModule
Delete
HelmetCamera
rontCamera =
front and rear
: Pulse) {
rontCamera =
gFrontCamera
ndler. Sends a
he camera's
func onPhotoTaken(image: Image) {
photoTaken.notifyInputs(image)
}
// Size changed event handler. Updates
the layout of the components for the
new size.
func onSizeChanged(size: Size) {
updateLayoutFor(cameraSize: size)
Stop


UserModules
Utilities.playgroundmodule
Sources
Math.swift
SWIFT
GameEngine.playgroundmodule
Sources
Forces.swift
SWIFT

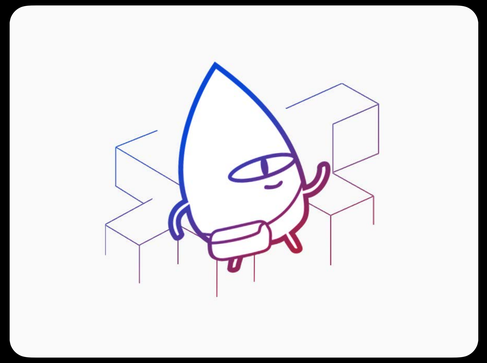

Synesthesia
SharedCode
import UIKit
// Set up the scene.
let scene = Scene()
scene.backgroundImage =
scene.hasCollisionBorder = true
let floor = Sprite(image: , columns: 20, rows: 1,
name: "floor")
scene.place(floor, at: Point(x: 0, y: -475))
let titleLabel = Label(text: "SYNESTHESIA", color:
.white, font: Font.SystemHeavyItalic, size: 70)
scene.place(titleLabel, at: Point(x: 0, y: 0))
titleLabel.fadeOut(after: 3)
// Create a rainbow of rectangles at the top of the
scene, representing the color of light entering the
camera.
for i in -5...5 {
let hueValue = ((Double(i)) / 10) + 0.5
let color = Color(hue: hueValue, saturation: 1,
brightness: 1, alpha: 1)
let rect = Graphic(shape: .rectangle(width: 100,
height: 100, cornerRadius: 0), color: color)
rect.alpha = 0.2
scene.place(rect, at: Point(x: Double(i * 100), y:
450))
}
// Configure the graphic menu in the upper-right corner
of the scene. Tap to choose a graphic from the menu and
Stop

Shapes

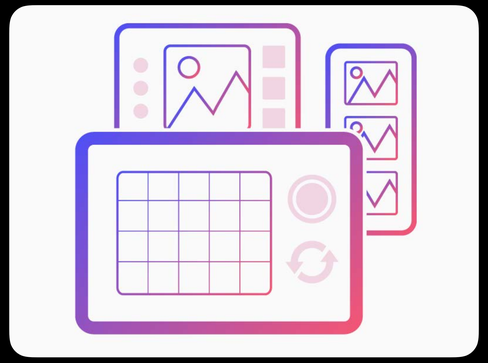


Graphing

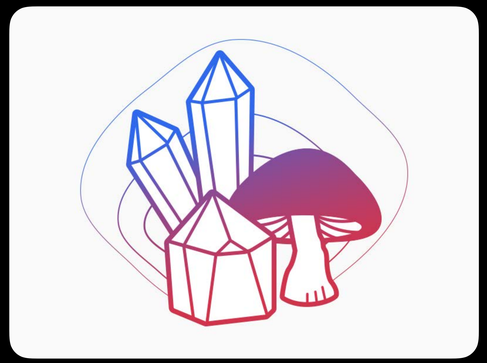
Answers



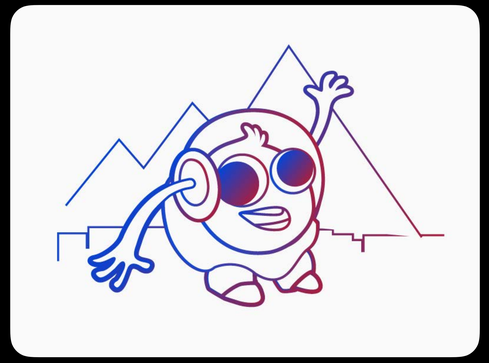
Puzzle World



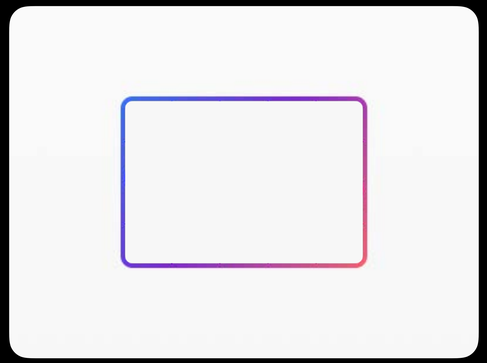
Camera Create



Sonic Create



Sensor Create



Blank

PlaygroundBook

Manifest.plist

Key	Type	Value
UserModuleMode	String	Full

NEW

Auto-Opening and Activating Files

CaveScape

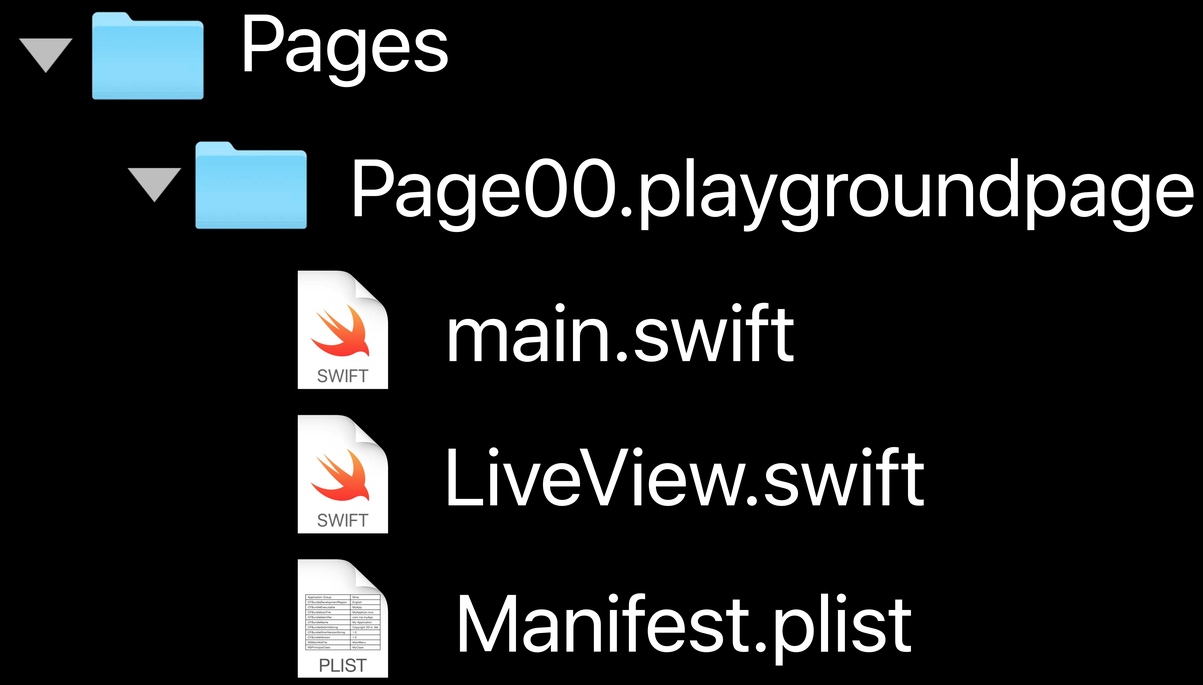
Crystals | Band | CaveGlitter | GraphicCluster | GraphicLoops | ToneGenerator

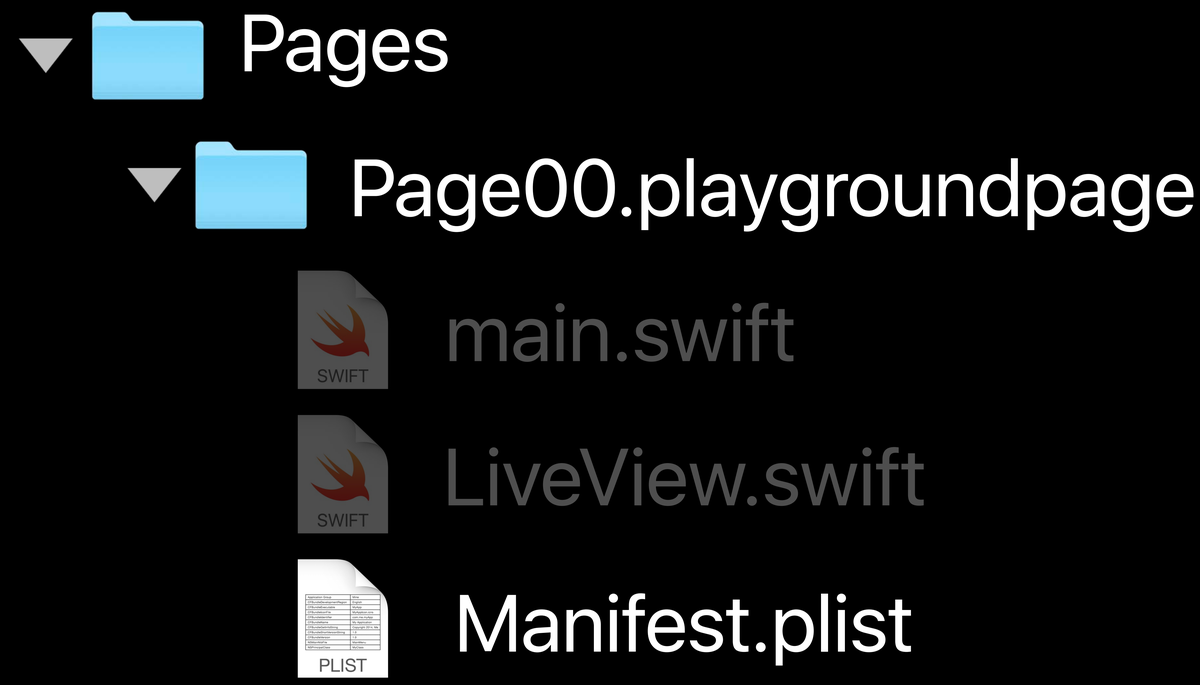
```
scene.backgroundImage = 

scene.isGridVisible = false

// Decorative moss
let mossValues = [
    BandProperties(scale: 0.5, point: Point(x: -20, y: -300)),
    BandProperties(scale: 0.3, point: Point(x: -70, y: -330)),
    BandProperties(scale: 0.5, point: Point(x: -400, y: -320))
]
for moss in mossValues {
    let mossGraphic = Graphic(image: )
    mossGraphic.scale = moss.scale
    scene.place(mossGraphic, at: moss.point)
}

// Graphic loops
```

CaveScape

Crystals | Band | CaveGlitter | GraphicCluster | GraphicLoops | ToneGenerator

```
scene.backgroundImage = 

scene.isGridVisible = false

// Decorative moss
let mossValues = [
    BandProperties(scale: 0.5, point: Point(x: -20, y: -300)),
    BandProperties(scale: 0.3, point: Point(x: -70, y: -330)),
    BandProperties(scale: 0.5, point: Point(x: -400, y: -320))
]
for moss in mossValues {
    let mossGraphic = Graphic(image: )
    mossGraphic.scale = moss.scale
    scene.place(mossGraphic, at: moss.point)
}

// Graphic loops
```


Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist


Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist

Manifest.plist

NEW

Key	Type	Value
UserModuleSourceFilesToOpen		

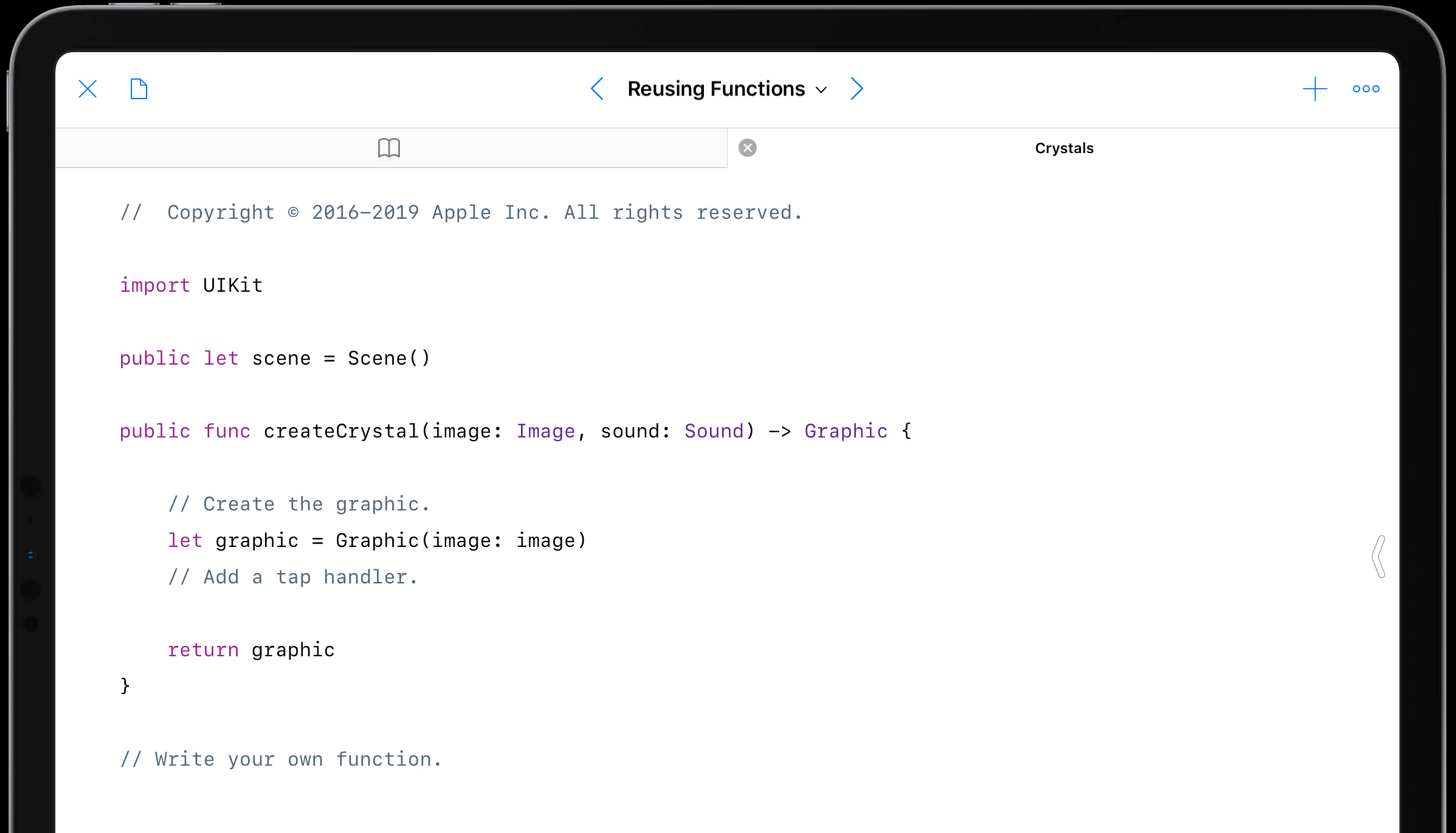
Manifest.plist

NEW

Key	Type	Value
UserModuleSourceFilesToOpen	Array	(3 items)
Item 0	String	UserModule/MyFiles.playgroundmodule/Sources/Crystal.swift
Item 1	String	UserModule/MyFiles.playgroundmodule/Sources/CaveGlitter.swift
Item 2	String	UserModule/MyFiles.playgroundmodule/Sources/GraphicCluster.swift

Crystals

```
scene.backgroundImage = 🏔
playMusic(Music.turtle)

// Call your function.
```


Reusing Functions
Crystals
// Copyright © 2016–2019 Apple Inc. All rights reserved.
import UIKit
public let scene = Scene()
public func createCrystal(image: Image, sound: Sound) -> Graphic {
// Create the graphic.
let graphic = Graphic(image: image)
// Add a tap handler.
return graphic
}
// Write your own function.

Manifest.plist

NEW

Key	Type	Value
UserModuleSourceFileToActivate	String	UserModule/MyFiles.playgroundmodule/Sources/Crystals.swift

NEW

Code Completion Directives

Issuing Commands

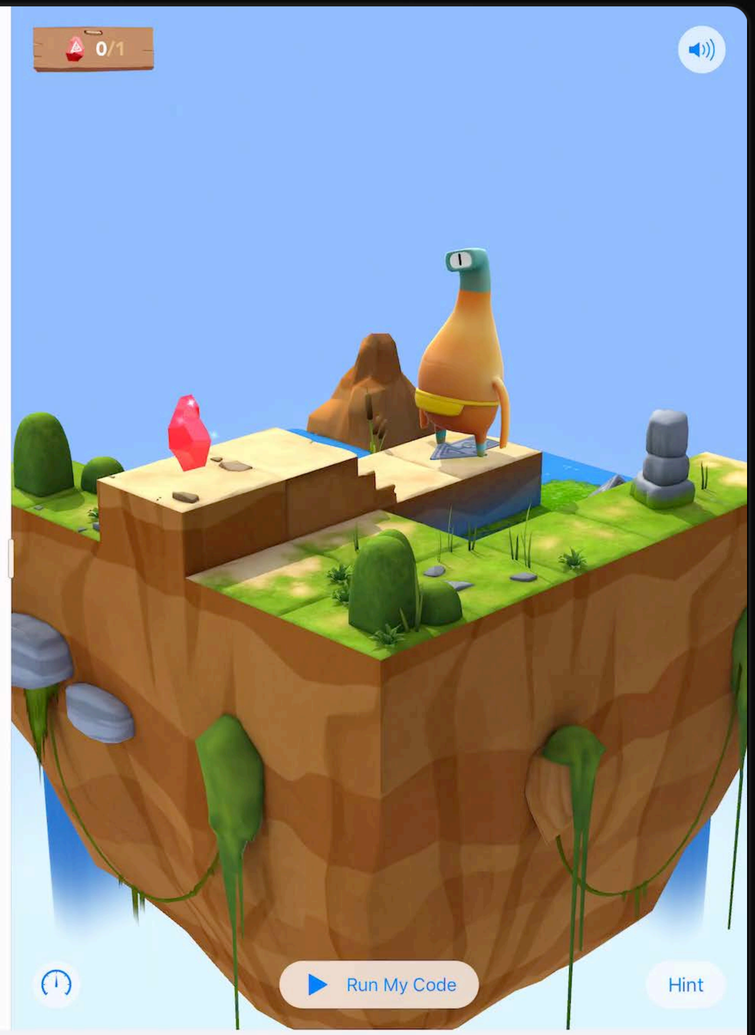
Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.



collectGem() moveForward()

Issuing Commands

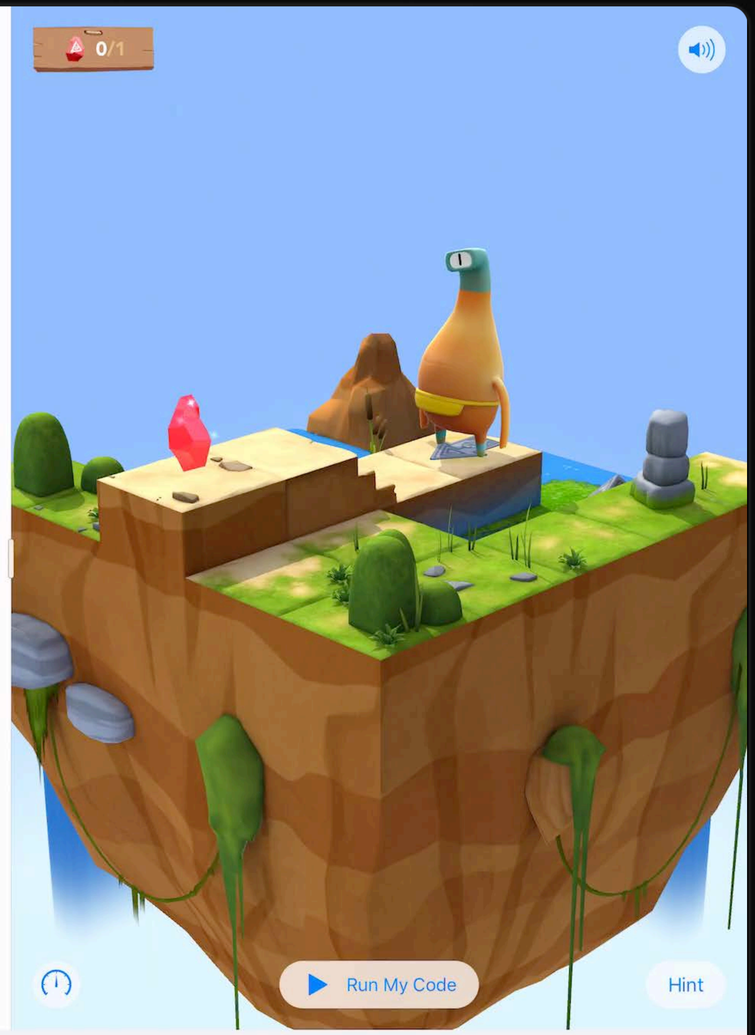
Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.



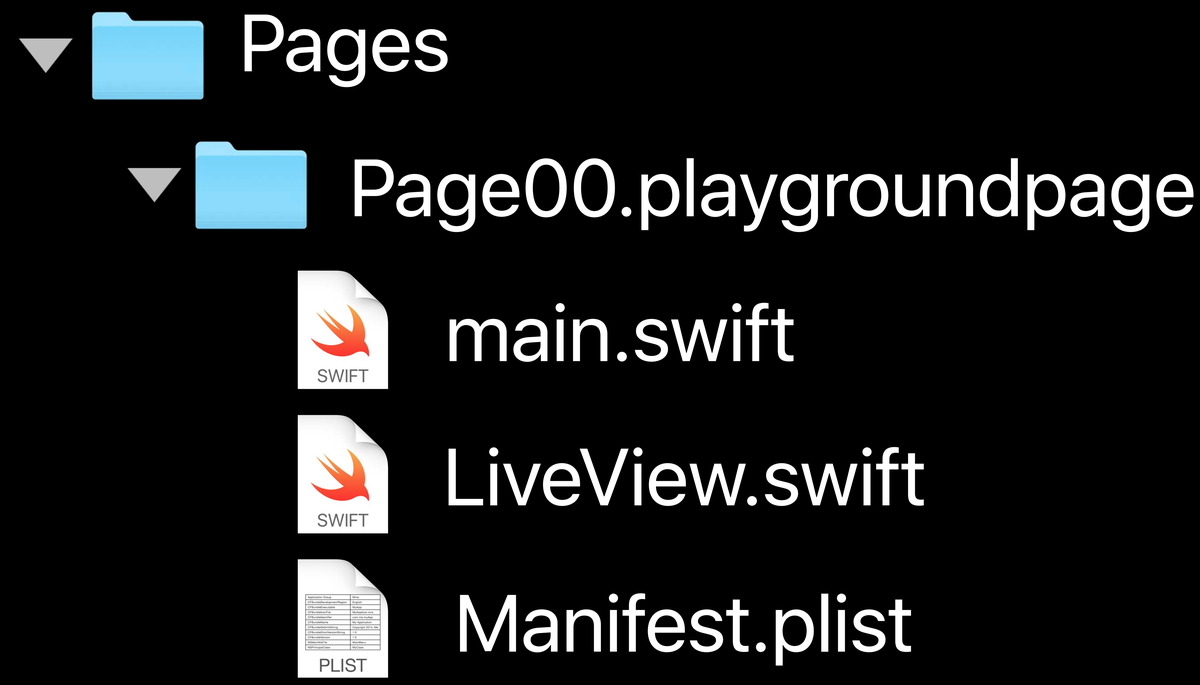
collectGem() moveForward()

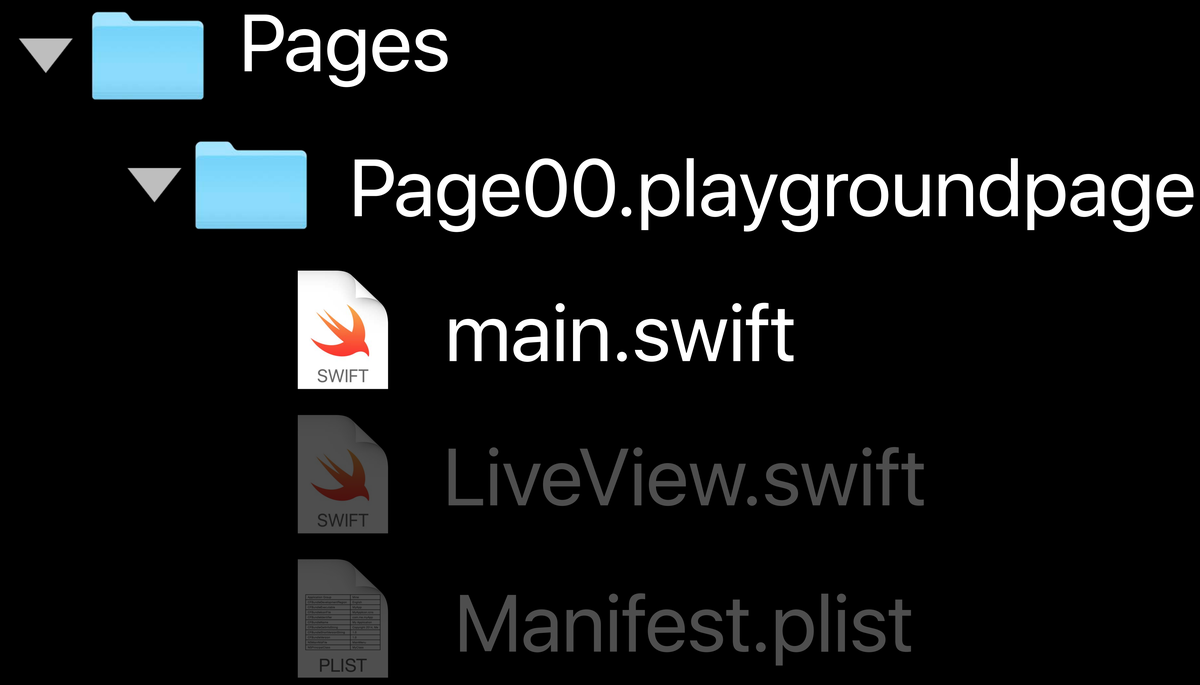
1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

collectGem() moveForward()


Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist

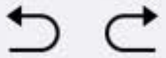

Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist

main.swift

let var if for while func App Button CameraView Color Label

main.swift

```
//#-code-completion(everything, hide)
```



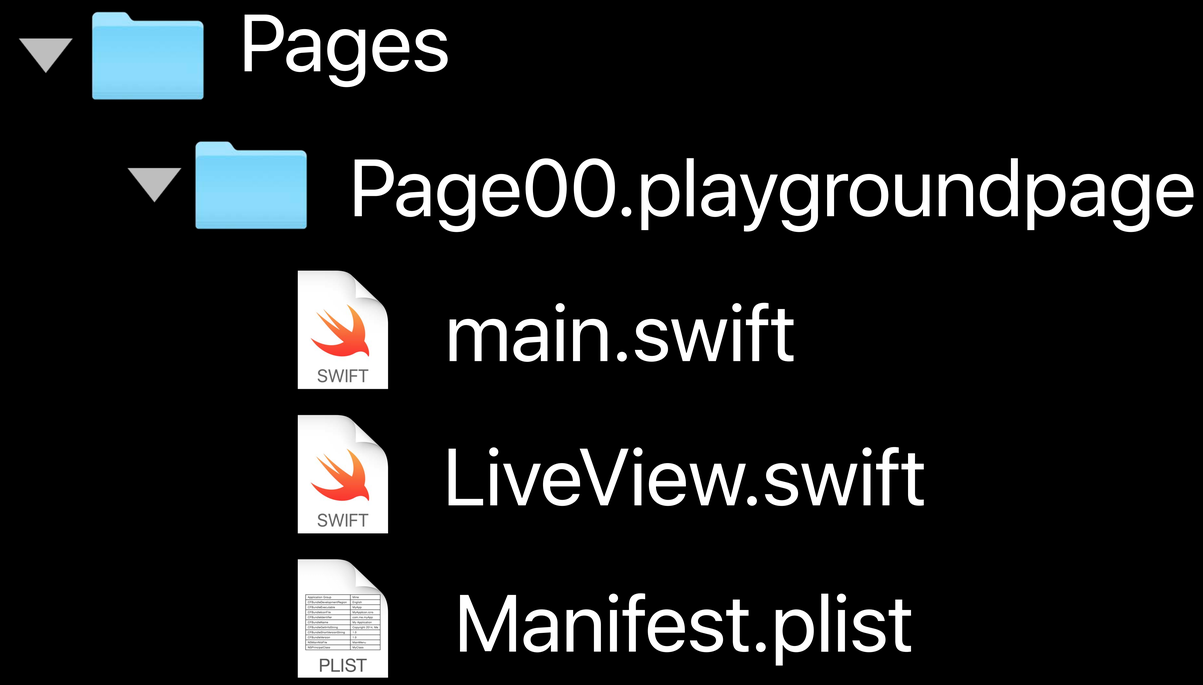

main.swift

```
//#-code-completion(everything, hide)
//#-code-completion(currentmodule, show)
```

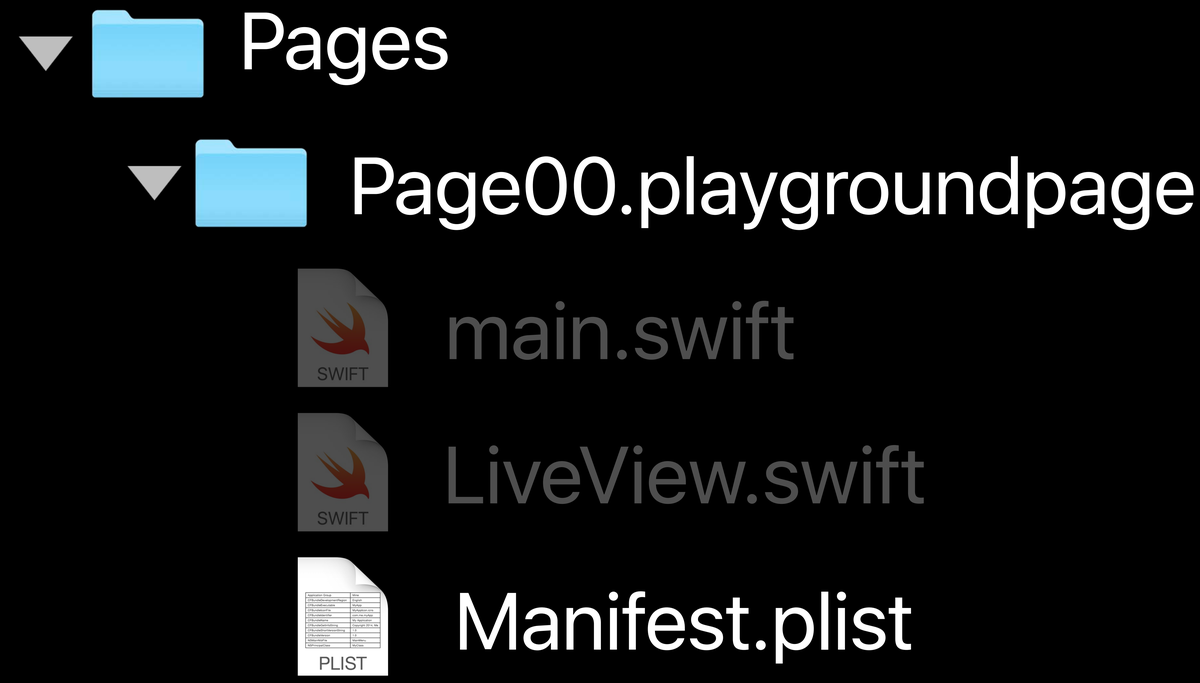
main.swift

```
//#-code-completion(everything, hide)
//#-code-completion(currentmodule, show)
//#-code-completion(module, show, MyFiles)
```

NEW


Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist

NEW


Pages
Page00.playgroundpage
SWIFT
main.swift
SWIFT
LiveView.swift
PLIST
Manifest.plist

Manifest.plist

NEW

Key	Type	Value
UserModuleCodeCompletionDirectives		

Manifest.plist

NEW

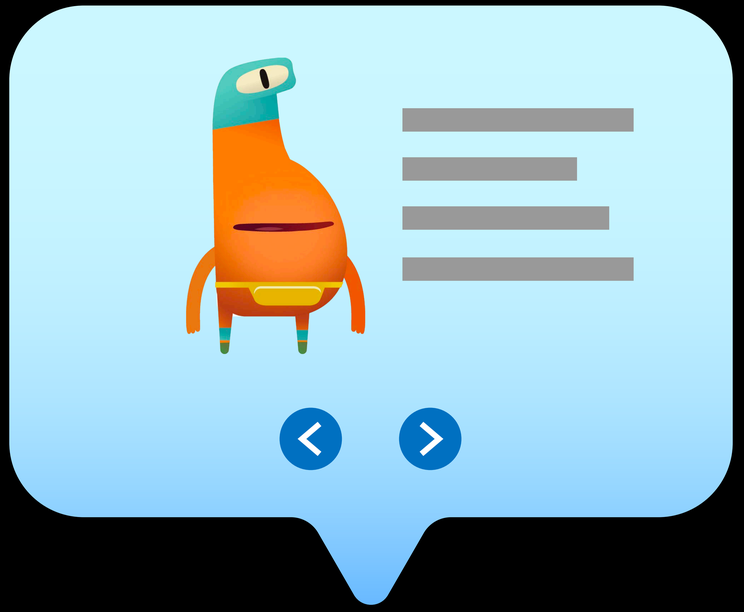
Key	Type	Value
UserModuleCodeCompletionDirectives	Array	(4 items)
Item 0	String	everything, hide
Item 1	String	currentmodule, show
Item 2	String	module, show, MyFiles, UIKit

Manifest.plist

NEW

Key	Type	Value
UserModuleCodeCompletionDirectives	Array	(4 items)
Item 0	String	everything, hide
Item 1	String	currentmodule, show
Item 2	String	module, show, MyFiles, UIKit
Item 3	String	identifier, show, public, private

NEW



Swift Cutscenes


Introduction


Introduction


HTML


HTML
5


HTML
5

NEW

NEW

Frameworks — SpriteKit, UIKit, and CoreAnimation

NEW

Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode

NEW

Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode

Interface Builder

NEW

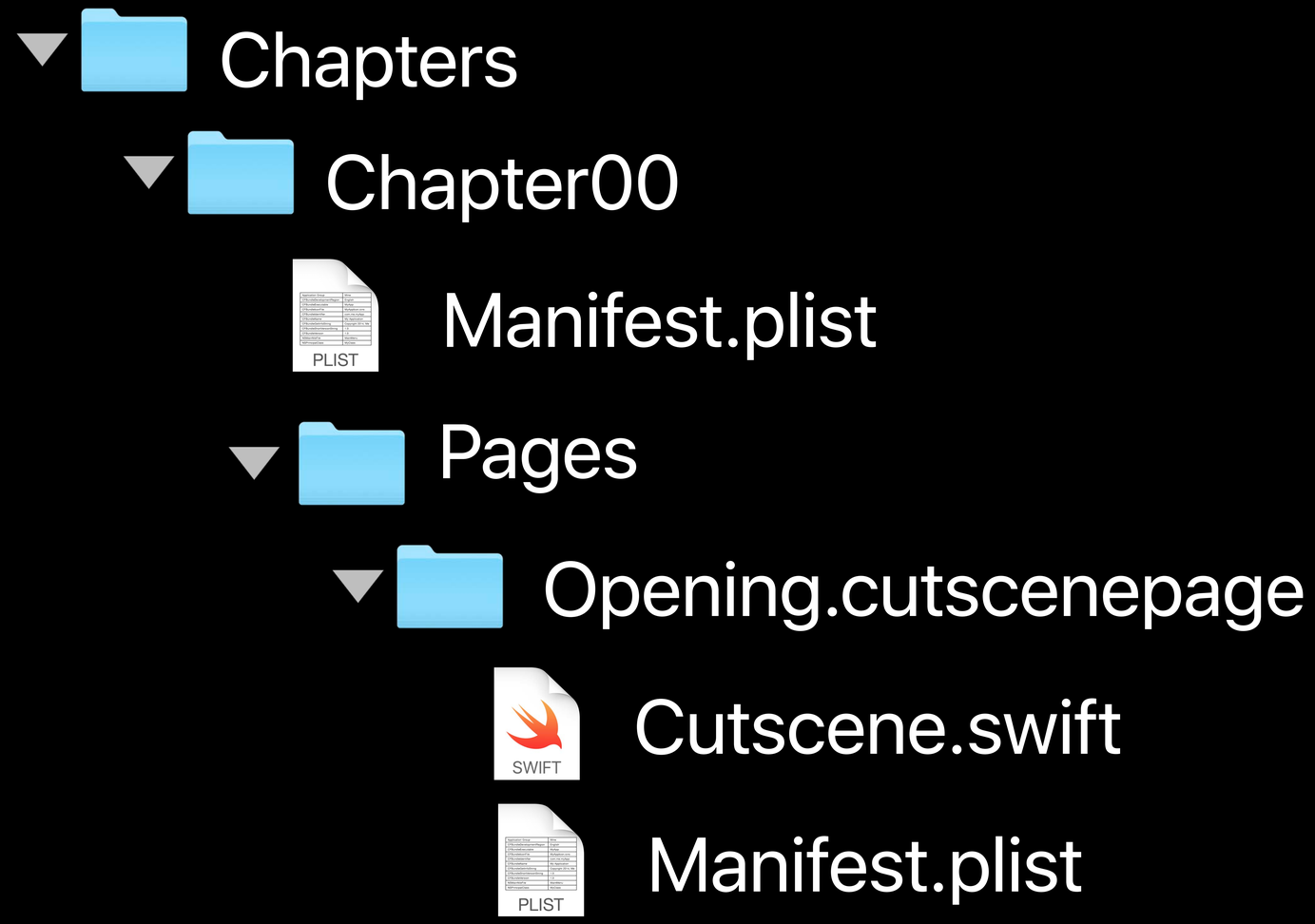
Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode

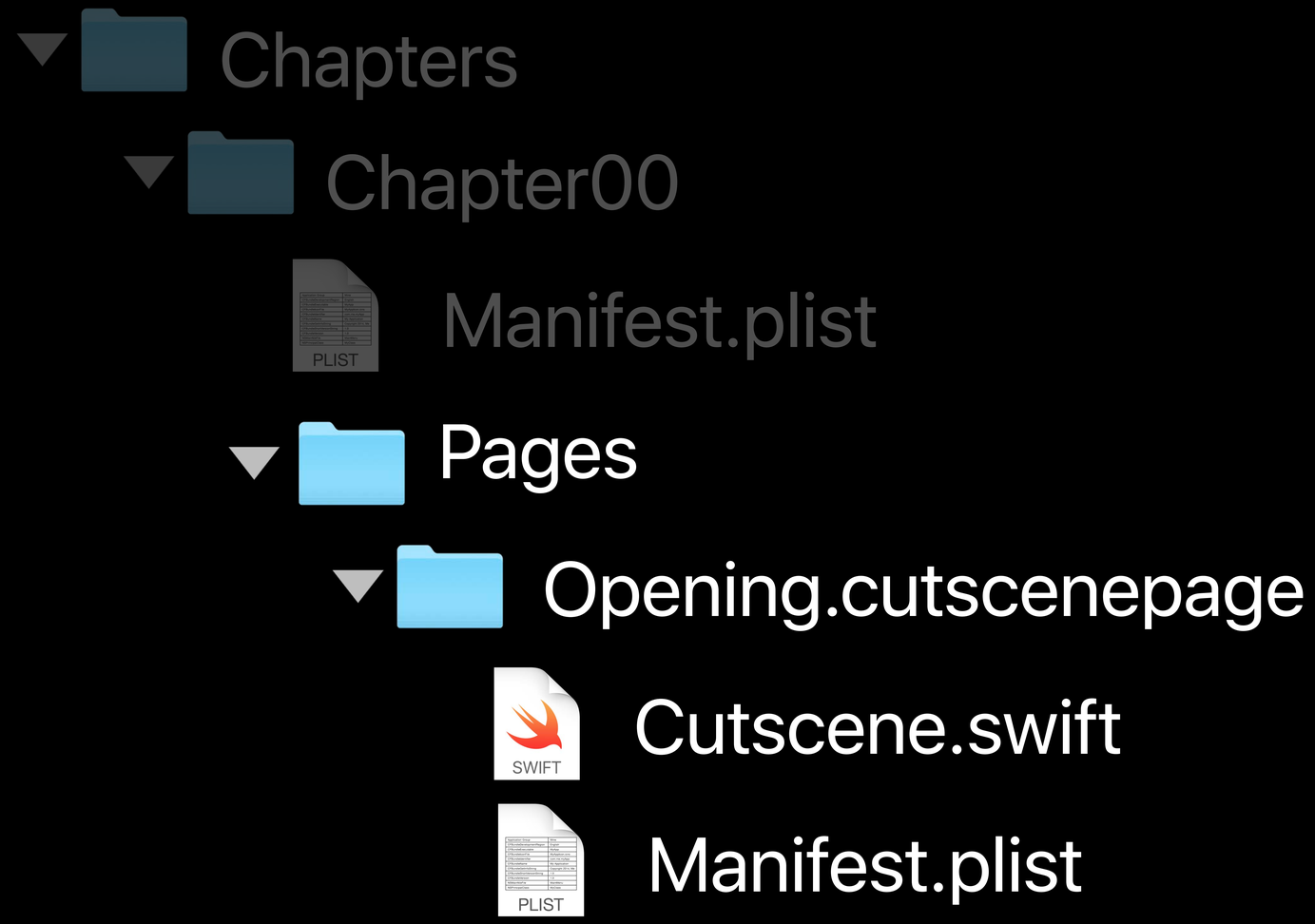
Interface Builder

Localization

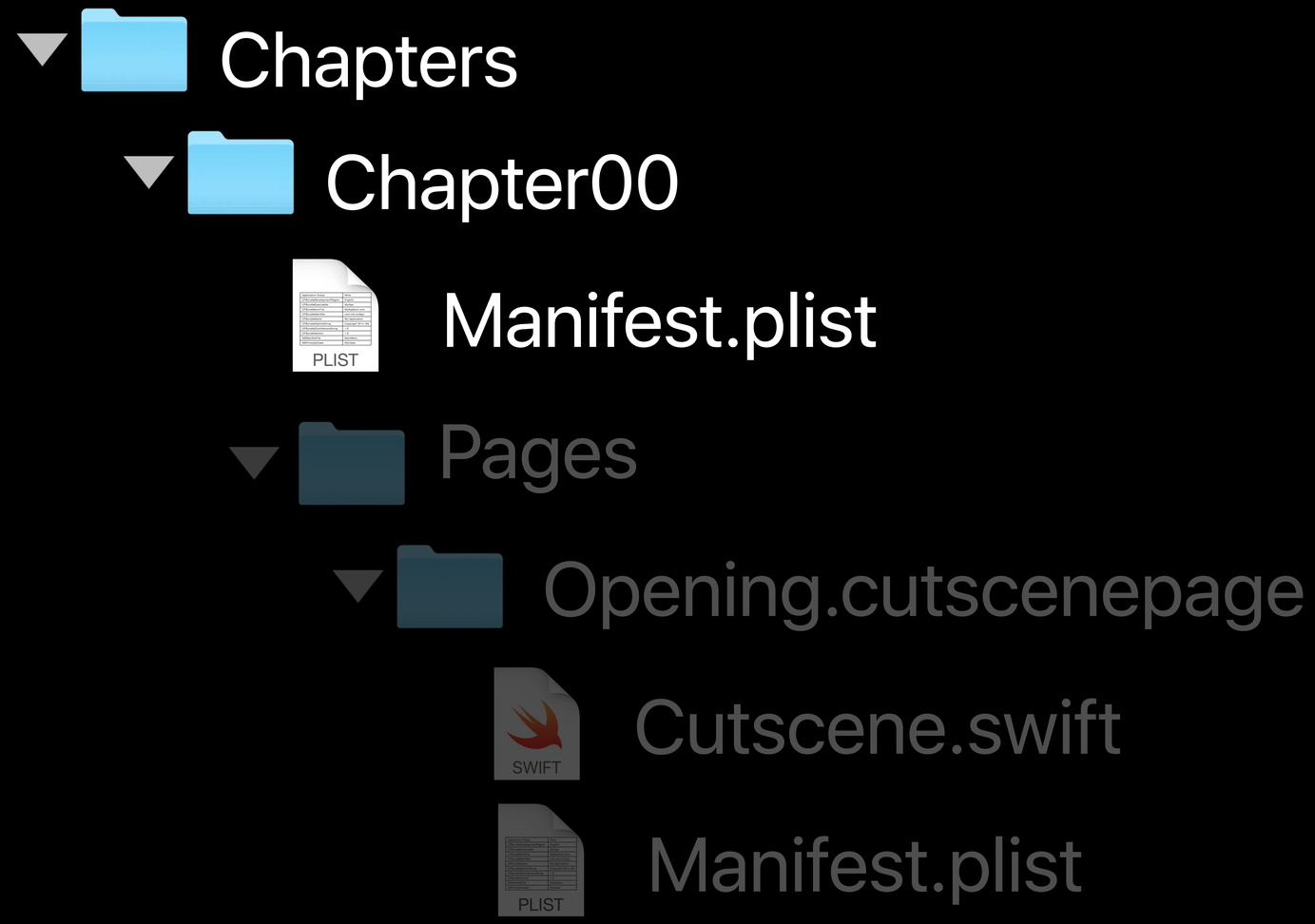
NEW


Chapters
Chapter00
PLIST
Manifest.plist
Pages
Opening.cutscenepage
SWIFT
Cutscene.swift
PLIST
Manifest.plist

NEW


Chapters
Chapter00
PLIST
Manifest.plist
Pages
Opening.cutscenepage
SWIFT
Cutscene.swift
PLIST
Manifest.plist

NEW


Chapters
Chapter00
PLIST
Manifest.plist
Pages
Opening.cutscenepage
SWIFT
Cutscene.swift
PLIST
Manifest.plist

Manifest.plist

NEW

Key	Type	Value
Pages	Array	(3 items)
Item 0	String	Intro.cutscenepage
Item 1	String	00.playgroundpage
Item 2	String	Outro.cutscenepage

NEW

Localized Code Comments

可组合性

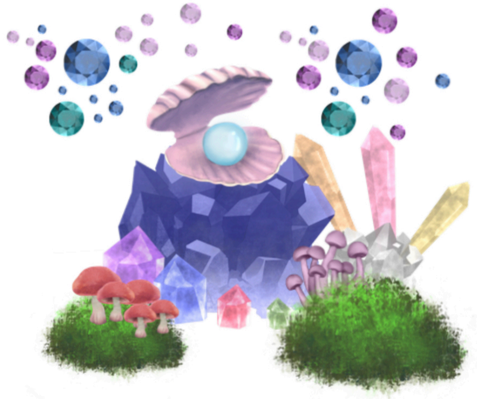
Crystals | CaveGlitter | GraphicCluster | GraphicLoop | ToneG

```
import UIKit

// `addGraphicCluster`
 函数将图形显示在扇形中，并在你手指拖过每个图形时播放其声音。当你轻点两
 下集群时，函数会同时播放所有声音。
public func addGraphicCluster(image: Image, sounds:
 [Sound], at point: Point) {
    var graphics = [Graphic]()

    //
     针对调用函数时指定的声音数组中的每个声音，用提供的图像创建图形并
     放置。
    for count in 0..<sounds.count {
        let graphic = Graphic(image: image)

        graphic.setOnFingerMovedHandler { touch in
            if touch.doubleTap {
                //
                 轻点两下集群以播放所有声音，并使所有图形同时发光。
                for sound in sounds {
                    playSound(sound, volume: 100)
                }

                for graphic in graphics {
                    graphic.glow()
                }
            } else if touch.firstTouchInGraphic {
                // 依次轻点每个图形以播放其声音，并使它单独发光。
                playSound(sounds[count], volume: 100)
                graphic.glow()
```

情境

调用函数

现在你已经在共享文件中编写了几个你自己的函数，了解如何从其他共享文件中调用预构建函数来创建完整的音乐场景吧。



运行我的代码

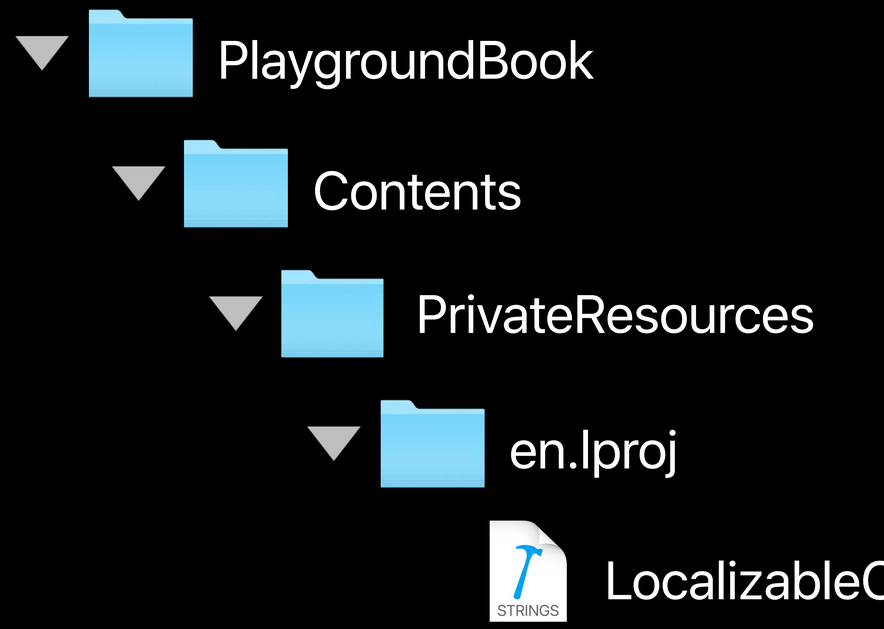
NEW

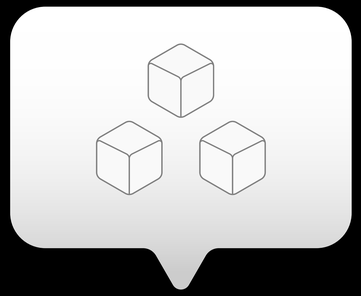
```
//#-localizable-zone(ex02k1)
// Create an instance of LightSensor to gather light.
//#-end-localizable-zone
let lightSensor = LightSensor(configuration: .front)
```

NEW

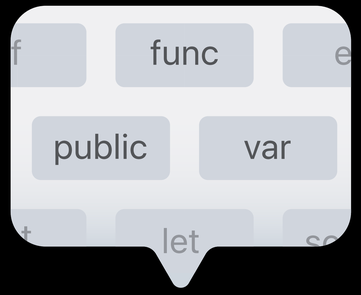
```
let message = "/*#-localizable-zone(welcome2)*/Hello!/*#-end-
    localizable-zone*/"
```

NEW


PlaygroundBook
Contents
PrivateResources
en.lproj
STRINGS
LocalizableCode.strings



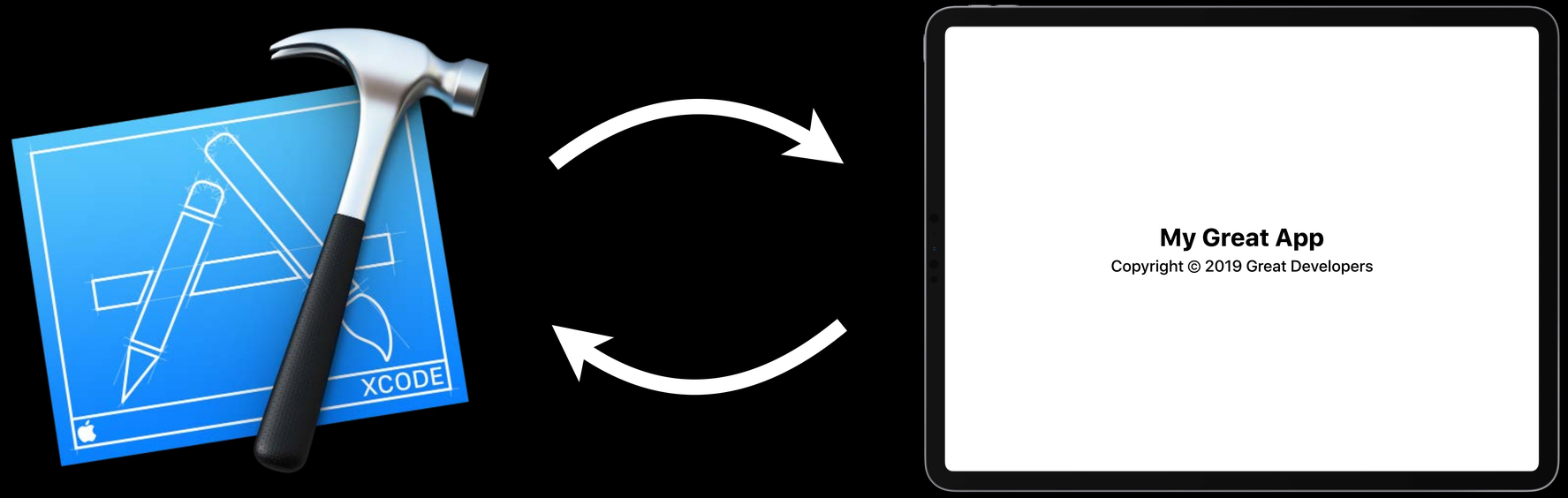
Module Mode



Code Completion

Swift Cutscenes

Localized Code Comments


XCODE
My Great App
Copyright © 2019 Great Developers

Demo

Grace Kendall, Playgrounds Engineer

WWDCSwiftPlaygrounds2019.github.io

More Information

developer.apple.com/wwdc19/405

Swift Playgrounds Lab

Wednesday, 9:00AM


WWDC19