Swift Playgrounds 3

Jonathan Penn, Playgrounds Engineer
Grace Kendall, Playgrounds Engineer
Joy Forbes, Developer Education Engineer
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
Sprite Kit  \((F = ma)\)
Swift Playgrounds
Raw Accelerometer Data
Sprite Kit  \( (F = ma) \)
Swift Playgrounds
Raw Accelerometer Data
SpriteKit  \((F = ma)\)
Hole Placement
Swift Playgrounds
Raw Accelerometer Data
Sprite Kit  ($F = ma$)
Hole Placement
New on iPad

Grace Kendall, Playgrounds Engineer
let circle = Circle()
circle.draggable = true
let circle = Circle()
circle.draggable = true
let circle = Circle()
circle draggable = true
Shapes

Canvas
Create
Touch
Animate

 Modules

MyFiles
Math
Graphics

Edit

Modules

Math

Edit

Modules

Euclidean.swift
Calculus.swift
DiffEQ.swift

Edit

Edit
let circle = Circle()
circle.draggable = true

let colorChanger = ColorChanges()
let randomAlpha =
colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha: randomAlpha)
let circle = Circle()
circle.draggable = true

let colorChanger = ColorChanges()
let randomAlpha = colorChanger.getRandomTransparency()
circle.color = circle.color.withAlpha(alpha: randomAlpha)
let circle = Circle()
circle.draggable = true

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

let colorChanger = ColorChanger()
let randomAlpha = colorChanger.get()
circle.color = circle.color.withAlpha(randomAlpha)
let x: Int = 5

let y: Double = x

Cannot convert value of type 'Int' to specified type 'Double'

Insert "Double(x)".
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)
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: 3, z: 2)`

3. Call `plane.place(character:st:)` using the character and position you created, like this: `plane.place(character: hopper, at: newPosition)`

```swift
func detectedPlane(plane: Plane) {
    playSound(.boing)
    // Create your character and Point objects here
    let hopper = Character(name: CharacterName.hopper)
    let newPosition = Point(x: 3, z: 2)
    plane.place(character: hopper, at: newPosition)
}
```
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: 3, z: 2)
3. Call plane.place(character: hopper) using the character and position you created, like this: plane.place(character: hopper, at: newPosition)

```swift
func detectedPlane(plane: Plane) {  

    playSound(.beep)  
    // Create your character and Point objects here  
    let hopper = Character(name: CharacterName.hopper)  
    let newPosition = Point(x: 3, z: 2)  

    plane.place(character: hopper, at: newPosition)  
}
```
// 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()
    }
New for Authors

Joy Forbes, Developer Education Engineer
Module Mode
Module Mode
None, Limited, and Full
None
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
Limited
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.
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(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(absoluteValue: CGFloat) -> CGVector {
        var next = self
        return next
    }
}
Blu’s Adventure
Assemble Your Camera
Flashy Photos
Rock, Paper, Scissors
Sonic Workshop
Brick Breaker
Full
```swift
// Size changed event handler. Updates the layout of the components for the new size.
func onChangeSize(size: Size) {
    updateLayoutFor(cameraSize: size)
}
```

```
func onPhotoTaken(image: Image) {
    photoTaken.notifyInputs(image)
}
```
import UIKit

// Set up the scene.
let scene = Scene()

scene.backgroundColor = .white
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: UIFont.boldSystemFont(ofSize: 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 0...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: 460))
}

// Configure the graphic menu in the upper-right corner of the scene. Tap to choose a graphic from the menu and
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserModuleMode</td>
<td>String</td>
<td>Full</td>
</tr>
</tbody>
</table>
Auto-Opening and Activating Files
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)
}
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
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserModuleSourceFilesToOpen</td>
<td></td>
<td>Item 0: UserModule/MyFiles.playgroundmodule/Sources/Crystal.swift</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Item 1: UserModule/MyFiles.playgroundmodule/Sources/CaveGlitter.swift</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Item 2: UserModule/MyFiles.playgroundmodule/Sources/GraphicCluster.swift</td>
</tr>
<tr>
<td>Key</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>UserModuleSourceFilesToOpen</td>
<td>Array</td>
<td>(3 items)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>UserModule/MyFiles.playgroundmodule/Sources/Crystal.swift</td>
</tr>
<tr>
<td>Item 1</td>
<td>String</td>
<td>UserModule/MyFiles.playgroundmodule/Sources/CaveGlitter.swift</td>
</tr>
<tr>
<td>Item 2</td>
<td>String</td>
<td>UserModule/MyFiles.playgroundmodule/Sources/GraphicCluster.swift</td>
</tr>
</tbody>
</table>
scene.backgroundImage = 🌞
playMusic(Music.turtle)

// Call your function.
// 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.
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserModuleSourceFileToActivate</td>
<td>String</td>
<td>UserModule/MyFiles.playgroundmodule/Sources/Crystals.swift</td>
</tr>
</tbody>
</table>
Code Completion Directives
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.
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.
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.
main.swift

// #-code-completion(everything, hide)
// #-code-completion(everything, hide)
// #-code-completion(currentmodule, show)
// #-code-completion(everything, hide)
// #-code-completion(currentmodule, show)
// #-code-completion(module, show, MyFiles)
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserModuleCodeCompletionDirectives</td>
<td>Array</td>
<td>(4 times)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>everything, hide</td>
</tr>
<tr>
<td>Item 1</td>
<td>String</td>
<td>currentmodule, show</td>
</tr>
<tr>
<td>Item 2</td>
<td>String</td>
<td>module, show, MyFiles, UIKit</td>
</tr>
<tr>
<td>Item 3</td>
<td>String</td>
<td>identifier, show, public, private</td>
</tr>
</tbody>
</table>

**NEW**
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserModuleCodeCompletionDirectives</td>
<td>Array</td>
<td>(4 items)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>everything, hide</td>
</tr>
<tr>
<td>Item 1</td>
<td>String</td>
<td>currentmodule, show</td>
</tr>
<tr>
<td>Item 2</td>
<td>String</td>
<td>module, show, MyFiles, UIKit</td>
</tr>
<tr>
<td>Key</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>UserModuleCodeCompletionDirectives</td>
<td>Array</td>
<td>(4 items)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>everything, hide</td>
</tr>
<tr>
<td>Item 1</td>
<td>String</td>
<td>currentmodule, show</td>
</tr>
<tr>
<td>Item 2</td>
<td>String</td>
<td>module, show, MyFiles, UIKit</td>
</tr>
<tr>
<td>Item 3</td>
<td>String</td>
<td>identifier, show, public, private</td>
</tr>
</tbody>
</table>
Swift Cutscenades
Frameworks — SpriteKit, UIKit, and CoreAnimation
Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode
Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode

Interface Builder
Frameworks — SpriteKit, UIKit, and CoreAnimation

Xcode

Interface Builder

Localization
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pages</td>
<td>Array</td>
<td>(3 items)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>Intro.cutscenepage</td>
</tr>
<tr>
<td>Item 1</td>
<td>String</td>
<td>00.playgroundpage</td>
</tr>
<tr>
<td>Item 2</td>
<td>String</td>
<td>Outro.cutscenepage</td>
</tr>
</tbody>
</table>
Localized Code Comments

//Código
//Code
//コード
import UIKit

// `addGraphicCluster`
函数将图形显示在视图中，并在按住手指过每个图形时播放其声音。当你在点击

public func addGraphicCluster(image: UIImage, sounds: [Sound], at point: CGPoint) {
    var graphics = [Graphic]{}

    //
    // 针对调用函数时指定的音效数组中的每个音效，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 {
                //
                // 轻点两次以播放所有音效，并允许所有图形同时发射。
                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音效，并使其单发发光。
                playSound(sounds[count], volume: 100)
                graphic.glow()
let lightSensor = LightSensor(configuration: .front)
```swift
let message = "/*/localizable-zone(welcome2)*/Hello!/**-end-localizable-zone*/"
```
Module Mode

Code Completion

Swift Cutscenes

Localized Code Comments
My Great App

Copyright © 2019 Great Developers
Demo
More Information

developer.apple.com/wwdc19/405

Swift Playgrounds Lab  Wednesday, 9:00AM