

步驟

STEP 1：依照 5.1 節建立一個 Swift GameKit 專案

先設計遊戲的畫面。請打開 Xcode 開發工具，並且依照 5.1 節建立一個 Swift GameKit 專案，這裡取名為“MyWord”，並且透過繪圖軟體來設計遊戲的畫面。

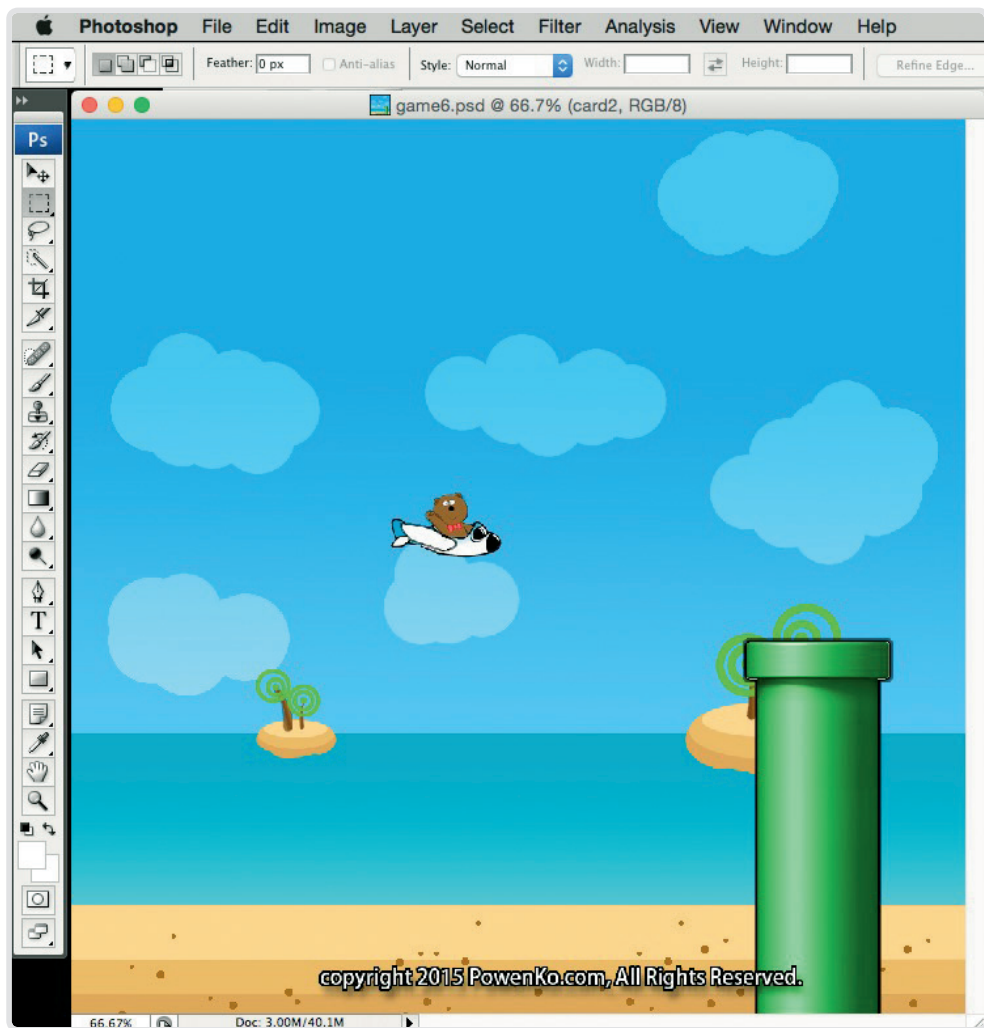


圖 14-2 飛飛熊畫面設計

STEP 2：添加遊戲的圖片

可以使用光碟中準備好的圖片，或是自行準備的遊戲圖片也可以。

透過滑鼠拖拉檔案的方法，把圖檔拖曳到 Xcode 的專案中，也可以透過滑鼠右鍵選取“new group”建立新資料夾“pic”，這樣可以讓專案看起來更整齊。

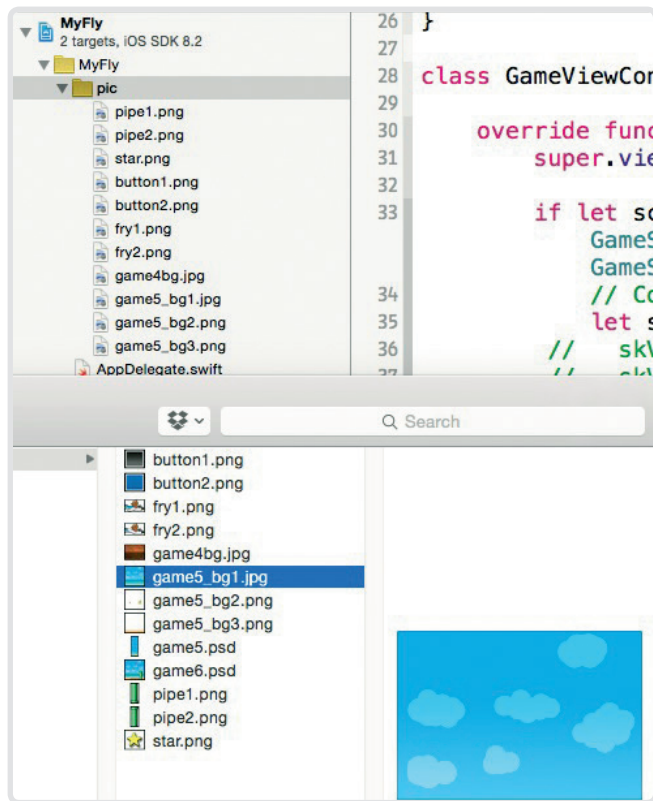


圖 14-3 透過滑鼠拖拉檔案的方法，把圖檔拖曳到 Xcode 的專案中

STEP 3：修改程式

接下來我們調整程式，把遊戲的畫面排列出來。

範例程式

範例：ch14\14-1\1\MyFly\GameScene.swift

```
01 import SpriteKit
02
03 class GameScene: SKScene {
04     override func didMoveToView(view: SKView) {
05         let myScore = SKLabelNode(fontNamed:"AmericanTypewriter-Bold")
06
07         override func didMoveToView(view: SKView) {
08             //////////////// / 文字
09             let myLabel = SKLabelNode(fontNamed:"Chalkduster")
10             myLabel.text = "copyright 2015 PowenKo.com, All Rights Reserved.";
11             myLabel.fontSize = 32;
12             myLabel.position = CGPoint(x:CGRectGetMidX(self.frame),y: 100);
13             self.addChild(myLabel)
14             //////////////// 分數
15             myScore.horizontalAlignmentMode =
16                 SKLabelHorizontalAlignmentMode.Left // 文字靠左
17             myScore.text = "30";
18             myScore.fontSize = 50;
19             myScore.fontColor = UIColor(red: 1/255, green: 1/255,
20                 blue: 1/255, alpha: 150/255) // 顏色
21             myScore.position = CGPoint(x:30,y: Int(self.size.height)-160);
22             self.addChild(myScore)
23
24             //////////////// 背景
25             var bgImage = SKSpriteNode(imageNamed: "game5_bg1.jpg")
26             bgImage.zPosition = -1
27             bgImage.position = CGPoint(x:CGRectGetMidX(self.frame),
28                 y:CGRectGetMidY(self.frame));
29             self.addChild(bgImage)
30
31             //////////////// 主角
```

```
29     var fly = SKSpriteNode(imageNamed: "fly1.png")
30     fly.zPosition = 2
31     fly.position = CGPoint(x:CGRectGetMidX(self.frame),
32                            y:CGRectGetMidY(self.frame));
33     self.addChild(fly)
34 }
```

程式解說

- ✓ 第 22 ~ 27 行：顯示背景。
- ✓ 第 28 ~ 32 行：顯示主角。

看到的遊戲畫面如下圖所示。

執行結果

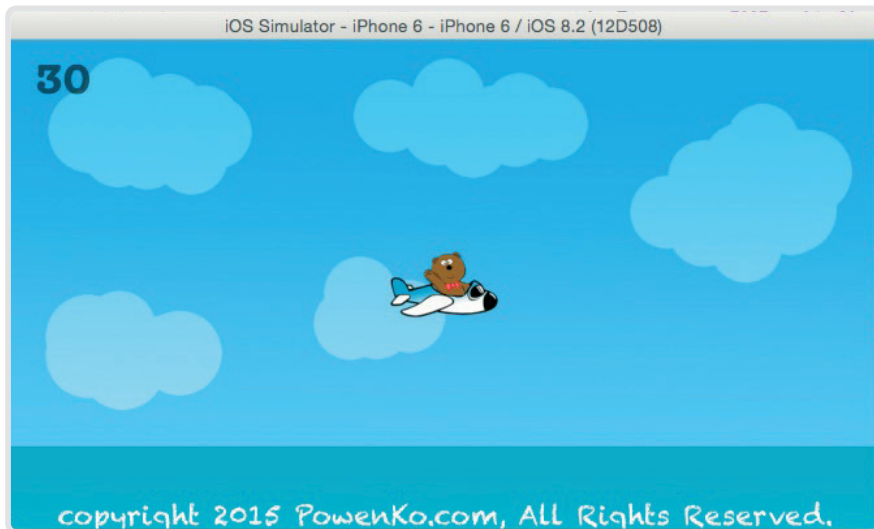


圖 14-4 執行結果

STEP 4：修改程式

接下來將主角透過兩張圖片的切換做出動畫效果，讓主角看起來就像是在天空飛翔的樣子。

範例程式

範例：ch14\14-1\1\MyFly\GameScene.swift

```
01 import SpriteKit
02
03 class GameScene: SKScene {
04     .....
05     ////////////////////////////////////////////////// 主角
06     var fly1 = SKTexture(imageNamed: "fly1.png")           // 主角圖片 1
07     var fly2 = SKTexture(imageNamed: "fly2.png")           // 主角圖片 2
08     let ani1 = SKAction.animateWithTextures(
09         [fly1,fly2],timePerFrame:0.25)                     // 0.25 秒切換
10     let ani2 = SKAction.repeatActionForever(ani1)           // 重複動畫
11
12     // var fly = SKSpriteNode(imageNamed: "fly1.png")
13     var fly = SKSpriteNode(texture: fly1)
14     fly.runAction(ani2)
15     fly.zPosition = 2
16     fly.position = CGPoint(x:CGRectGetMidX(self.frame),
17                             y:CGRectGetMidY(self.frame));
18
19     self.addChild(fly)
20 }
21 .....
```

程式解說

- ✓ 第 6 ~ 7 行：指定動畫圖片。
- ✓ 第 8 ~ 10 行：設定每 0.25 秒不斷切換圖片動畫效果。

執行結果

執行後就會看到主角展翅高飛的樣子。

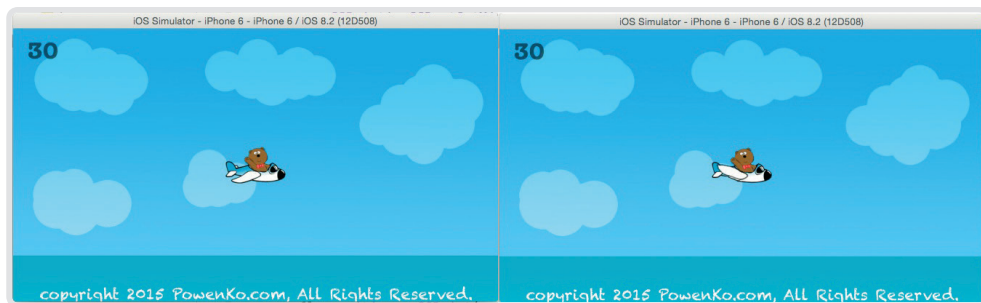


圖 14-5 執行結果

教學影片

完整教學影片請參考光碟中的 `swift_14_1_fly`。

14.2 物理動作

目的

Flappy Bird 有趣之處在於，主角在天空飛翔，但因為地心引力的關係，需要玩家點選畫面讓主角繼續在天空中飛翔。本章節將會介紹 如何加上物理動作反應，添加主角因為地心引力的關係往下降，且當用戶點選畫面時，就會往上飛的物理效果。

步驟

STEP 1：延續上一個章節

延續上一個章節的範例繼續執行。

STEP 2：修改程式，加上地心引力

在這之前要先介紹物理效果，在 SKScene 場景中，可以透過 `self.physicsWorld.gravity` 參數指定物理效果。如果要物體往下降的話，可以透過：

```
CGVectorMake(0.0, -5.0)
```

如果要往上飄的話，可以用：

```
CGVectorMake(0.0, 5.0)
```

範例程式

範例：ch14\14-2\1\MyFly\GameScene.swift

```
01 import SpriteKit
02 class GameScene: SKScene, SKPhysicsContactDelegate {
03     let myScore = SKLabelNode(fontNamed:"AmericanTypewriter-Bold")
04
05     override func didMoveToView(view: SKView) {
06         .....
07         //////////// 主角
08         var fly1 = SKTexture(imageNamed: "fly1.png")
09         var fly2 = SKTexture(imageNamed: "fly2.png")
10         let ani1 = SKAction.animateWithTextures(
11             [fly1,fly2],timePerFrame:0.25)
12         let ani2 = SKAction.repeatActionForever(ani1)
13         var fly = SKSpriteNode(texture: fly1)
14         fly.runAction(ani2)
15         fly.zPosition = 2
16         fly.position = CGPoint(x:CGRectGetMidX(self.frame),
17             y:CGRectGetMidY(self.frame));
17         self.addChild(fly)
18         fly.physicsBody = SKPhysicsBody(rectangleOfSize: fly.size) // 加上物理
19         fly.physicsBody?.dynamic = true
20         fly.physicsBody?.allowsRotation = false
21
22         //////////// 設計世界的物理
23         self.physicsWorld.gravity = CGVectorMake(0.0,-5.0) // 地心引力往下
24         self.physicsWorld.contactDelegate = self
25     }
26
27     ...
28 }
```

程式解說

- ✓ 第 18 行：設定物理大小為實際圖片的大小。
- ✓ 第 19 行：設定動態物理計算，也就是此物件會受到物理反應自動改變位置，在此是讓主角往下掉。
- ✓ 第 20 行：該物理反應時，是否會允許旋轉。
- ✓ 第 23 行：設計世界的物理反應是地心引力往下 -5。
- ✓ 第 24 行：物理反應的觸發反應事件，是哪個類別反應，這裡指定的是自己。

在 SKNode 中，每一個物件都有 SKPhysicsBody 物理屬性，但是需要特別設定，所以會看到主角透過指定的物理反應，產生與世界的物理效果。

```
fly.physicsBody = SKPhysicsBody(rectangleOfSize: fly.size) // 加上物理
```

```
fly.physicsBody?.dynamic = true
```

```
fly.physicsBody?.allowsRotation = false
```

執行結果

看到的遊戲畫面如下圖所示，因為物理反應的設定，主角在一啟動時，就會自動往下掉。

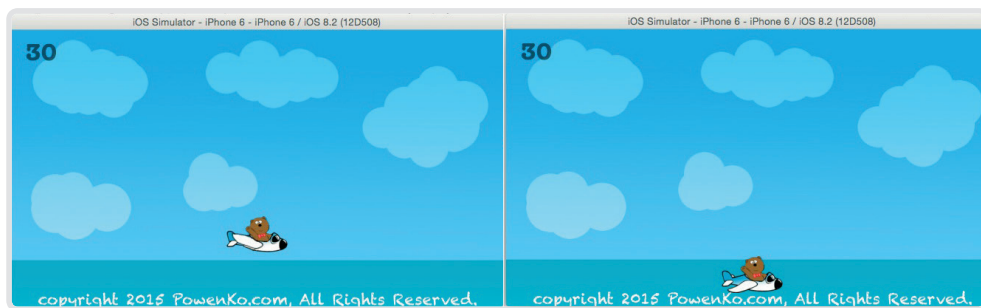


圖 14-6 執行結果

STEP 3：修改程式—跳躍動作

本步驟將調整程式，增加用戶點選螢幕後，做出跳躍的動作，就是把主角施加外力向上推。

範例程式

範例：ch14\14-2\MyFly\GameScene.swift

```
01 import SpriteKit
02 class GameScene: SKScene, SKPhysicsContactDelegate {
03
04     let myScore = SKLabelNode(fontNamed:"AmericanTypewriter-Bold")
05     var fly:SKSpriteNode = SKSpriteNode() // 主角物件變數
06     ...
07     override func touchesBegan(touches: Set<UITouch>, withEvent event: UIEvent?) {
08         self.fly.physicsBody?.velocity = CGVectorMake(0,0)
09         self.fly.physicsBody?.applyImpulse(CGVectorMake(0,200)) // 向上施力
10
11     }
12 }
```

程式解說

- ✓ 第 8 行：設定物件的重力加速度為 0,0。
- ✓ 第 9 行：向上推 200 的力量，也可以依照遊戲的難易度，調整這個數字大小。

執行結果

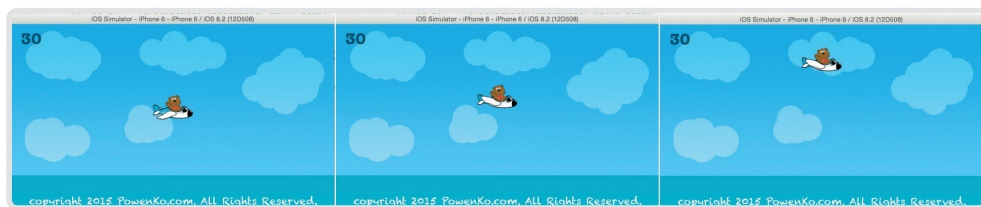


圖 14-7 執行結果

教學影片

完整教學影片請參考光碟中的 swift_14_2_physics。