

1204315 - Wireless Mobile Application Programming

Manasawee Kaenampornpan

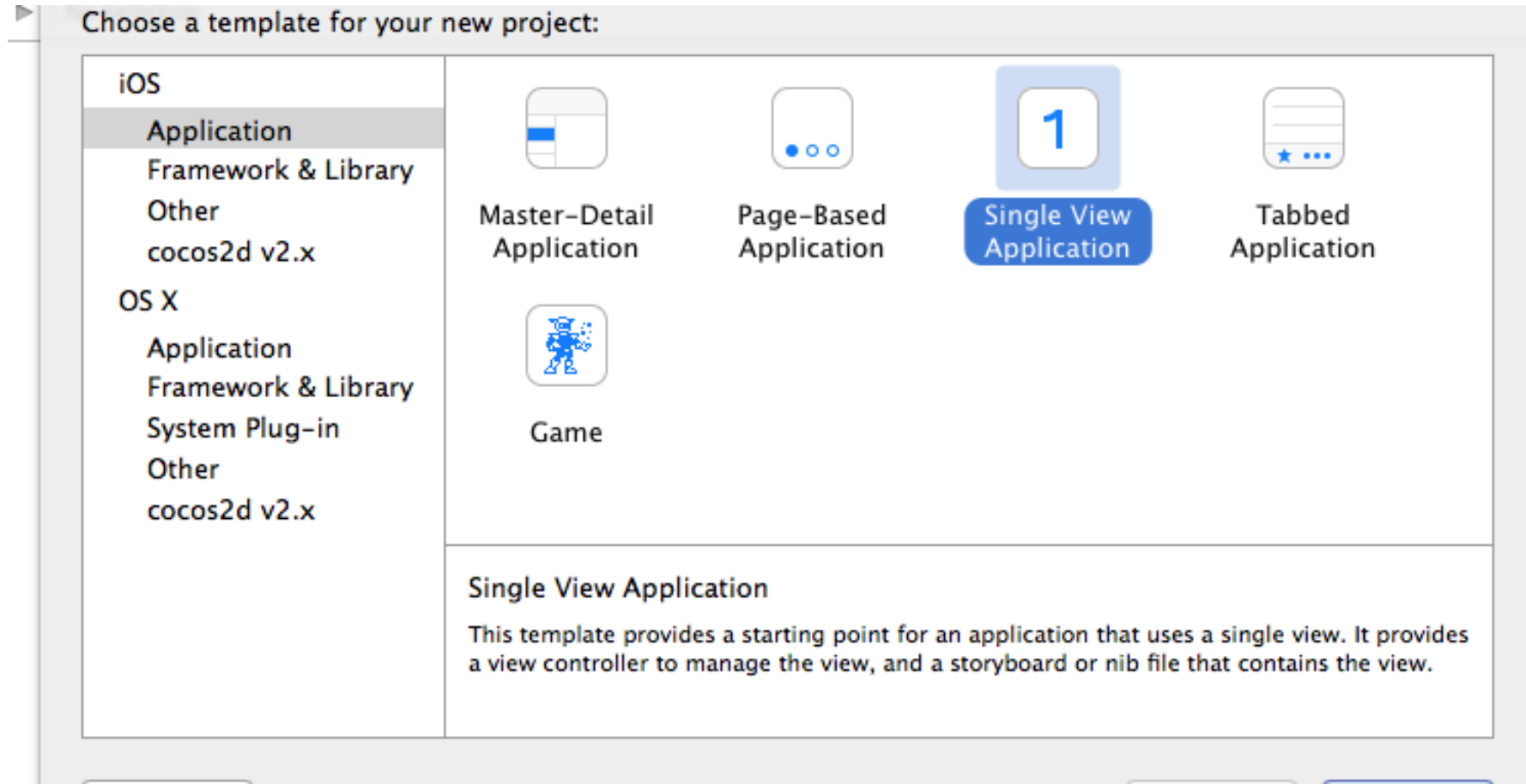
มนัสวี แก่นอำพรพันธ์

manasaweek@gmail.com

Maharakham University

UIKit Animation

Create single view project



เพิ่ม สีเหลี่ยม สร้างจาก **UIView** ก่อน **ViewDidLoad**

```
var square: UIView!
```

เป็น optional (!) เพราะเรายังไม่ได้สร้างค่าในฟังก์ชัน `init`



ใช้ สีเหลี่ยม ใน viewDidLoad()

```
override func viewDidLoad() {  
    super.viewDidLoad()  
  
    square = UIView(frame: CGRect(x: 100, y: 100, width: 100, height: 100))  
    square.backgroundColor = UIColor.grayColor()  
    view.addSubview(square)
```

Build and Run

เพิ่ม ตัวแปร แรงดึงดูด ก่อน **ViewDidLoad**

```
var animator: UIDynamicAnimator!  
var gravity: UIGravityBehavior!
```

เป็น optional (!) เพราะเรายังไม่ได้สร้างค่าในฟังก์ชัน `init`



เรียกใช้ ตัวแปร แรงดึงดูด ใน **ViewDidLoad**

```
animator = UIDynamicAnimator(referenceView: view)
gravity = UIGravityBehavior(items: [square])
animator.addBehavior(gravity)
```


Build and Run

เพิ่ม ตัวแปร ขอบเขตของ Animation ก่อน ViewDidLoad

```
var collision: UICollisionBehavior!
```

เป็น optional (!) เพราะเรายังไม่ได้สร้างค่าในฟังก์ชัน init



เรียกใช้ ตัวแปร ขอบเขต ใน **ViewDidLoad**

```
collision = UICollisionBehavior(items: [square])  
collision.translatesReferenceBoundsIntoBoundary = true  
animator.addBehavior(collision)
```

Build and Run

เพิ่ม ตัวแปร สิ่งกีดขวางใน **ViewDidLoad**

```
let barrier = UIView(frame: CGRect(x: 0, y: 300, width: 130, height: 20))
```

เรียกใช้ ตัวแปร สิ่งกีดขวางใน **ViewDidLoad**

```
barrier.backgroundColor = UIColor.redColor()  
view.addSubview(barrier)
```

Build and Run

แก้ ตัวแปร ขอบเขต ใน ViewDidLoad

```
collision = UICollisionBehavior(items: [square])  
collision.translatesReferenceBoundsIntoBoundary = true  
animator.addBehavior(collision)
```

แก้เป็น

```
collision = UICollisionBehavior(items: [square, barrier])
```


เพิ่ม คุณลักษณะ ให้ ตัวแปร ขอบเขต ใน ViewDidLoad

```
// add a boundary that has the same frame as the barrier  
collision.addBoundaryWithIdentifier("barrier", forPath: UIBezierPath(rect: barrier.frame))
```

The above code adds an invisible boundary that has the same frame as the barrier view. The red barrier remains visible to the user but not to the dynamics engine, while the boundary is visible to the dynamics engine but not the user. As the square falls, it appears to interact with the barrier, but it actually hits the immovable boundary instead.

Build and Run

เพิ่ม โค้ดใน viewDidLoad

```
collision.action = {  
    println("\(NSStringFromCGAffineTransform(square.transform)) \(NSStringFromCGPoint(square.center))")  
}
```

Build and Run

- ดูค่าจุดกลางขอสีเหลี่ยมที่เปลี่ยนไป

เพิ่ม protocol `UICollisionBehaviorDelegate` ใน
`ViewController.swift` และ ประการตั้งค่าเริ่มต้นใน
`ViewDidLoad`

```
class ViewController: UIViewController, UICollisionBehaviorDelegate {
```

```
collision.collisionDelegate = self
```

เพิ่ม ฟังก์ชัน collisionBehavior ใน viewController.swift

```
func collisionBehavior(behavior: UICollisionBehavior!, beganContactForItem item: UIDynamicItem!,  
withBoundaryIdentifier identifier: NSString!, atPoint p: CGPoint) {  
    println("Boundary contact occurred - \(identifier)")  
}
```

Build and Run

เพิ่ม โค้ดใน collisionBehavior func ใน
viewController.swift

```
let collidingView = item as UIView
collidingView.backgroundColor = UIColor.yellowColor()
UIView.animateWithDuration(0.3) {
    collidingView.backgroundColor = UIColor.grayColor()
}
```


Build and Run

เพิ่มคุณลักษณะการชน โดยเพิ่มโค้ด ในท้ายของ **ViewDidLoad**

```
let itemBehaviour = UIDynamicItemBehavior(items: [square])  
itemBehaviour.elasticity = 0.6  
animator.addBehavior(itemBehaviour)
```

Build and Run

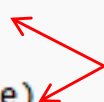
- สร้างเหมือนเส้นเชื่อม คล้ายๆสปริง

User Interaction

เพิ่ม โค้ดใน collisionBehavior func ใน viewController.swift

collisionBehavior(beganContactForItem:withBoundaryIdentifier:atPoint:)

```
if (!firstContact) {  
    firstContact = true  
  
    let square = UIView(frame: CGRect(x: 30, y: 0, width: 100, height: 100))  
    square.backgroundColor = UIColor.grayColor()  
    view.addSubview(square)  
    collision.addItem(square)  
    gravity.addItem(square)  
  
    let attach = UIAttachmentBehavior(item: collidingView, attachedToItem:square)  
    animator.addBehavior(attach)  
}
```



เพิ่มอีกสี่เหลี่ยมและใส่คุณสมบัติ

Build and Run

เพิ่ม ตัวแปร viewDidLoad

```
var square: UIView!  
var snap: UISnapBehavior!
```

เพิ่มโค้ด func ในท้ายของ viewController

```
override func touchesEnded(touches: NSSet, withEvent event: UIEvent) {  
    if (snap != nil) {  
        animator.removeBehavior(snap)  
    }  
  
    let touch = touches.anyObject() as UITouch  
    snap = UISnapBehavior(item: square, snapToPoint: touch.locationInView(view))  
    animator.addBehavior(snap)  
}
```


Build and Run

- กดหน้าจอดูว่าเกิดไร

References

- <http://www.raywenderlich.com/76147/uikit-dynamics-tutorial-swift>
- <http://mathewsanders.com/prototyping-iOS-iPhone-iPad-animations-in-swift/>