

クソゲーを作って学ぶ

Swift勉強会

<https://github.com/MasayaHayashi724/kuso-game-hands-on/blob/master/kuso-game-hands-on.pdf>

今日の目的

- ✗ Swiftができるようになる
- ✗ ゲームが作れるようになる
- ✗ クソゲーが作れるようになる
- ○ クソみたいなゲームが作れるようになる
- ○ みんな仲良くなる
- ○ Swiftが好きになる
- ○ 何か他にもアプリを作ってみたくなる

お品書き

1. Swiftの簡単な文法解説 (15分)
2. ハンズオンでゲーム作ってみる (2~3時間)
3. オリジナルのゲーム作ってみる (4~5時間?)
(ハンズオンで作ったゲームを発展させてもOK)
(4. リリースしてみる (30分~1時間))

Swiftの簡単な文法解説

変数定義

```
let height: Double = 25.5 // letは定数、再代入できない
var score: Int = 0 // varは変数、再代入できる
let nameArray: [String] = ["masaya", "hayashi"] // 配列
var array = [1, 2, 3] // 型が明確なら省略できる
var didWin: Bool = false // Bool型(true, false)
let dict: [String: Int] = ["masaya": 6, "hayashi": 7]
// 辞書型というのもある
```

オプショナル型

```
let gender: Int? = nil
```

Swiftの簡単な文法解説

if文

```
let score = 12
if score > 10 {
    print("You did it!")
} else {
    print("Try again!")
}
```

```
let didWin = true
if didWin {
    print("You win!")
} else {
    print("You lose...")
}
```

Swiftの簡単な文法解説

for文

```
for i in 0..<10 {  
    print(i)  
}
```

```
for i in 0...10 {  
    print(i)  
}
```

```
let numbers = [1, 2, 3, 4, 5]  
for number in numbers { // for i in 0..<5 {  
    print(number)        //     print(numbers[i])  
}                          // }
```

Swiftの簡単な文法解説

enumとswitch文

```
enum Difficulty {  
    case easy  
    case normal  
    case hard  
}
```

```
var difficulty: Difficulty = .normal  
  
switch difficulty {  
    case .easy: print("easy")  
    case .normal: print("normal")  
    case .hard: print("hard")  
}
```

Swiftの簡単な文法解説

おわり

2Dゲームを作ってみよう

Swift vs Unity

Swift (Spritekit, Scenekit, Metal)

- 無料で簡単
- Appleが作っている

Unity

- Androidも作れる
- Asset Store

素材の探し方

良いフリー素材を探すのは結構時間がかかります。
あらかじめ探しておくと当日の開発がスムーズに。

おすすめサイト

- ぴぽや <http://blog.pipoya.net/>
- illust AC <https://www.ac-illust.com/>
- Rド <http://www.geocities.co.jp/Milano-Cat/3319/muz/002.html>
- Folce-zero <http://folce.zatunen.com>

では、作ってみましょう！

↓完成品↓

<https://github.com/MasayaHayashi724/save-the-earth>

<https://itunes.apple.com/jp/app/id1265444161>

シューティングゲームを作る

1. プロジェクトの作成
2. メニュー画面とゲーム画面の作成
3. 宇宙船を追加 (傾けると移動、タップでミサイル)
4. 小惑星を追加 (ランダム位置に出現して迫ってくる)
5. 衝突処理 (ミサイル-小惑星、小惑星-宇宙船or地球)
6. スコアとライフを導入
7. ゲーム終了処理
8. ベストスコアの保存

実機でビルドする方法

- <http://qiita.com/DKN915/items/7a2ce97f3758e2daf486>
- 上の記事を参考にしてね。

好きなゲームを作つてみよう！

or

さっきのゲームを進化させよう！

作ったゲームをリリースする

- 時間があればやりましょう
- 後日やってもOK

衝突判定

- physicsBody : 衝突を判定する範囲
- isDynamic : trueにしておく
- categoryBitMask : このノードのID
- contactTestBitMask : 衝突対象のcategoryBitMaskとのANDが1以上になれば、衝突する
- collisionBitMask : とりあえず0にしておこう