iOS 從零開始

Swift語法補充

蔡智強 Denny Tsai

denny@hpd.io

https://iosdev.hpd.io

空值 (Nil)

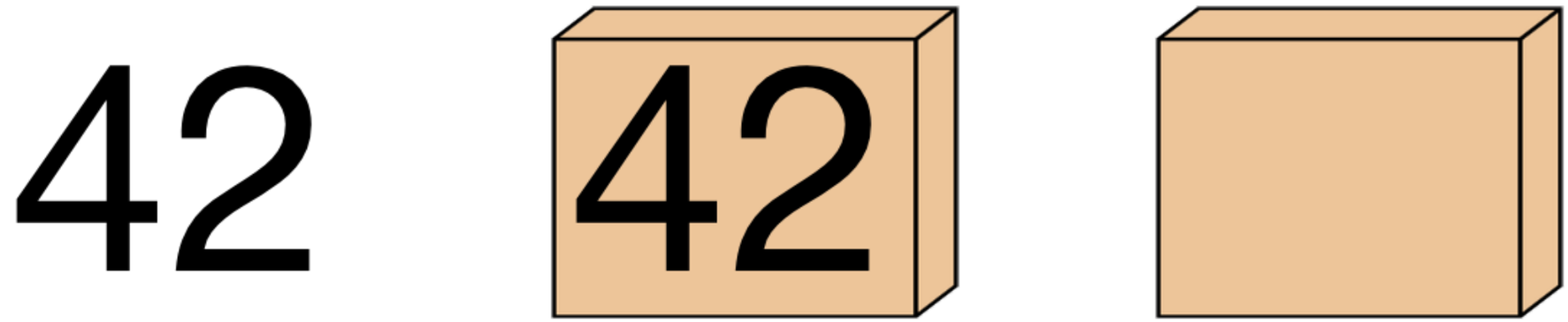
- 表示沒有值、不存在
- 不等於 0 或者 空字串 ("") 或者空陣列或者空字典

Optional

- 想像成是一種包裝，在資料型別後面加上?符號
- Optional可以是某資料型別的值或者是nil
- 減少對於傳送變數時候的猜測

- 任何資料型別都可以使用Optional
- 資料型別加上！代表預先拆開的Optional

Optional




```
Int
Int?
Int?
```

http://lithium3141.com/blog/2014/06/19/learning-swift-optional-types/

使用變數流程

- 判斷變數是否為Optional
  - 是: 解開Optional包裝
  - 否: 一定有值、不需要做沒有值的處理

拆開包裝

- Optional binding. if-let
- 用驚嘆號硬拆！

Examples

```
var aInt: Int?

aInt
aInt = 15

var bInt: Int
bInt = aInt
```

```
if let aInt = aInt {
    bInt = aInt
}

bInt = aInt!
```

註解 (Comment)

- 單行註解由 // 開頭
- 多行註解由 /* 開頭 */ 結尾

```
var number = 1 // 這是單行註解

/*
這是多行註解
註解第二行
*/
var hello = "World!"
```

作用範圍 (Scope)

- 最簡單的辨認方法：最近的大括號
- Code block: { ... }

```
var a = "Hello"
var b = [1, 2, 3, 4, 5]

func testFunc() {
    var x = false

    for i in b {
        let c = 10
    }
}
```

型別轉換 (Casting)

- 通常是把比較上層的型別轉成比較下層的型別
  例如：如果拿到UILabel，可以向下轉換成比較精確的MyLabel型別
- 因為轉換有可能會失敗，所以轉換結果會使用Optional包住
- `if let downcastVar = anyVar as? MyVar { ... }`
- `downcastVar = anyVar as! MyVar`

型別轉換 (Casting)

```
class MyLabel: UILabel {

    var myLabelProperty = 10

}

var anyLabel: UILabel
let myLabel = MyLabel()

anyLabel = myLabel
anyLabel.myLabelProperty

if let downcastLabel = anyLabel as? MyLabel {
    downcastLabel.myLabelProperty
}
```