Swift Cheat Sheet

Variables

```
var myInt = 1
var myExplicitInt: Int = 1 // explicit type
var x = 1, y = 2, z = 3 // declare multiple integers
myExplicitInt = 2 // set to another integer value
```

Random

```
import Foundation

        var randomNumber : Int = Int(arc4random())%50
        var randomNumberNoLimit : Int = Int(arc4random())
        var ran = arc4random_uniform(150)
```

Constants

```
let myInt = 1
myInt = 2 // compile-time error!
```

Strings

```
var myString = "a"
let myImmutableString = "c"
myString += "b" // ab
myString = myString + myImmutableString // abc
myImmutableString += "d" // compile-time error!

let count = 7
let message = "There are \(count) days in a week"
```

Func toInt()

```
1  let string = "42"
2  if let number = string.toInt() {
3      println("Got the number: \(number)")
4  } else {
5      println("Couldn't convert to a number")
6  }
7  // prints "Got the number: 42"
```

Logical Operators

```
var happy = true
var sad = !happy // logical NOT, sad = false
var everyoneHappy = happy && sad // logical AND, everyoneHappy = false
var someoneHappy = happy || sad // logical OR, someoneHappy = true
```

Printing

```
let name = "swift"
println("Hello")
println("My name is \(name)")
print("See you ")
print("later")
/*  Hello
    My name is swift
    See you later */
```

Tuple

```
let tipAndTotalNamed = (tipAmt:4.00, total:25.19)
tipAndTotalNamed.tipAmt
tipAndTotalNamed.total
```

Array

```
var colors = ["red", "blue"]
var moreColors: String[] = ["orange", "purple"] // explicit type
colors.append("green") // [red, blue, green]
colors += "yellow" // [red, blue, green, yellow]
colors += moreColors // [red, blue, green, yellow, orange, purple]

var days = ["mon", "thu"]
var firstDay = days[0] // mon
days.insert("tue", atIndex: 1) // [mon, tue, thu]
days[2] = "wed" // [mon, tue, wed]
days.removeAtIndex(0) // [tue, wed]
```

```
// Array
var shoppingList = ["catfish", "water", "lemons"]
shoppingList[1] = "bottle of water"             // update
shoppingList.count                              // size of array (3)
shoppingList.append("eggs")
shoppingList += "Milk"

// Array slicing
var fibList = [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 5]
fibList[4..6] // [3, 5]. Note: the end range value is exclusive
fibList[0..fibList.endIndex] // all except last item
// Subscripting returns the Slice type, instead of the Array type.
// You may need to cast it to Array in order to satisfy the type checker
Array(fibList[0..4])

// Variants of creating an array. All three are equivalent.
var emptyArray1 = String[]()
var emptyArray2: String[] = []
var emptyArray3: String[] = String[]()
```

Dictionaries

```
var days = ["mon": "monday", "tue": "tuseday"]
days["tue"] = "tuesday" // change the value for key "tue"
days["wed"] = "wednesday" // add a new key/value pair

var moreDays: Dictionary = ["thu": "thursday", "fri": "friday"]
moreDays["thu"] = nil // remove thu from the dictionary
moreDays.removeValueForKey("fri") // remove fri from the dictionary
```

```
// Dictionary
var occupations = [
   "Malcolm": "Captain",
   "kaylee": "Mechanic"
]
occupations["Jayne"] = "Public Relations"
var emptyDictionary = Dictionary<String, Float>()
```

Conditionals

```
//IF STATEMENT
let happy = true
if happy {
    println("We're Happy!")
} else {
    println("We're Sad :('")
}
// We're Happy!

let speed = 28
if speed <= 0 {
    println("Stationary")
} else if speed <= 30 {
    println("Safe speed")
} else {
    println("Too fast!")
}
// Safe speed


//SWITCH STATEMENT
let n = 2
switch n {
case 1:
    println("It's 1!")
case 2...4:
    println("It's between 2 and 4!")
case 5, 6:
    println("It's 5 or 6")
default:
    println("Its another number!")
}
// It's between 2 and 4!
```

```
if (onSaleInferred) {
  println("\(nameInferred) on sale for \(priceInferred)!")
} else {
  println("\(nameInferred) at regular price: \(priceInferred)!")
}
```

```
// Switch
let vegetable = "red pepper"
switch vegetable {
case "celery":
   let vegetableComment = "Add some raisins and make ants on a log."
case "cucumber", "watercress":
   let vegetableComment = "That would make a good tea sandwich."
case let x where x.hasSuffix("pepper"):
   let vegetableComment = "Is it a spicy \(x)?"
default: // required (in order to cover all possible input)
   let vegetableComment = "Everything tastes good in soup."
}

// Switch to validate plist content
let city:Dictionary<String, AnyObject> = [
   "name" : "Qingdao",
   "population" : 2_721_000,
   "abbr" : "QD"
]
switch (city["name"], city["population"], city["abbr"]) {
   case (.Some(let cityName as NSString),
      .Some(let pop as NSNumber),
      .Some(let abbr as NSString))
   where abbr.length == 2:
      println("City Name: \(cityName) | Abbr.:\(abbr) Population: \(pop)")
   default:
      println("Not a valid city")
}
```

For Loops

```
for var index = 1; index < 3; ++index {
        // loops with index taking values 1,2
}
for index in 1..3 {
        // loops with index taking values 1,2
}
for index in 1...3 {
        // loops with index taking values 1,2,3
}

let colors = ["red", "blue", "yellow"]
for color in colors {
    println("Color: \(color)")
}
// Color: red
// Color: blue
// Color: yellow

let days = ["mon": "monday", "tue": "tuesday"]
for (shortDay, longDay) in days {
    println("\(shortDay) is short for \(longDay)")
}
// mon is short for monday
// tue is short for tuesday
```

While Loops

```
var count = 1
while count < 3 {
    println("count is \(count)")
    ++count
}
// count is 1
// count is 2

count = 1
while count < 1 {
    println("count is \(count)")
    ++count
}
//



count = 1
do {
    println("count is \(count)")
    ++count
} while count < 3
// count is 1
// count is 2

count = 1
do {
    println("count is \(count)")
    ++count
} while count < 1
// count is 1
```

Functions

```
func iAdd(a: Int, b: Int) -> Int {
  return a + b
}
iAdd(2, 3) // returns 5

func eitherSide(n: Int) -> (nMinusOne: Int, nPlusOne: Int) {
  return (n-1, n+1)
}
eitherSide(5) // returns the tuple (4,6)
```

Classes

```
// A parent class of Square
class Shape {
    init() {
    }

    func getArea() -> Int {
        return 0;
    }
}

// A simple class `Square` extends `Shape`
class Square: Shape {
    var sideLength: Int

    // Custom getter and setter property
    var perimeter: Int {
        get {
            return 4 * sideLength
        }
        set {
            sideLength = newValue / 4
        }
    }

    init(sideLength: Int) {
        self.sideLength = sideLength
        super.init()
    }

    func shrink() {
        if sideLength > 0 {
            --sideLength
        }
    }

    override func getArea() -> Int {
        return sideLength * sideLength
    }
}
var mySquare = Square(sideLength: 5)
print(mySquare.getArea()) // 25
mySquare.shrink()
print(mySquare.sideLength) // 4

// Access the Square class object,
// equivalent to [Square class] in Objective-C.
Square.self

//example for 'willSet' and 'didSet'
class StepCounter {
    var totalSteps: Int = 0 {
        willSet(newTotalSteps) {
            println("About to set totalSteps to \(newTotalSteps)")
        }
        didSet {
            if totalSteps > oldValue {
```

```
import Foundation

class TipCalculatorModel {

  var total: Double
  var taxPct: Double
  var subtotal: Double {
    get {
      return total / (taxPct + 1)
    }
  }

  init(total:Double, taxPct:Double) {
    self.total = total
    self.taxPct = taxPct
  }

  func calcTipWithTipPct(tipPct:Double) -> Double {
    return subtotal * tipPct
  }

  func returnPossibleTips() -> [Int: Double] {

    let possibleTipsInferred = [0.15, 0.18, 0.20]
    let possibleTipsExplicit:[Double] = [0.15, 0.18, 0.20]

    var retval = [Int: Double]()
    for possibleTip in possibleTipsInferred {
      let intPct = Int(possibleTip*100)
      retval[intPct] = calcTipWithTipPct(possibleTip)
    }
    return retval

  }
}
```

ถ้ามีการอัพเดทค่าเมื่อค่าตัวแปรมีการเปลี่ยนแปลงจะใช้ get

Enum

- ใช้กับจำนวนเต็ม

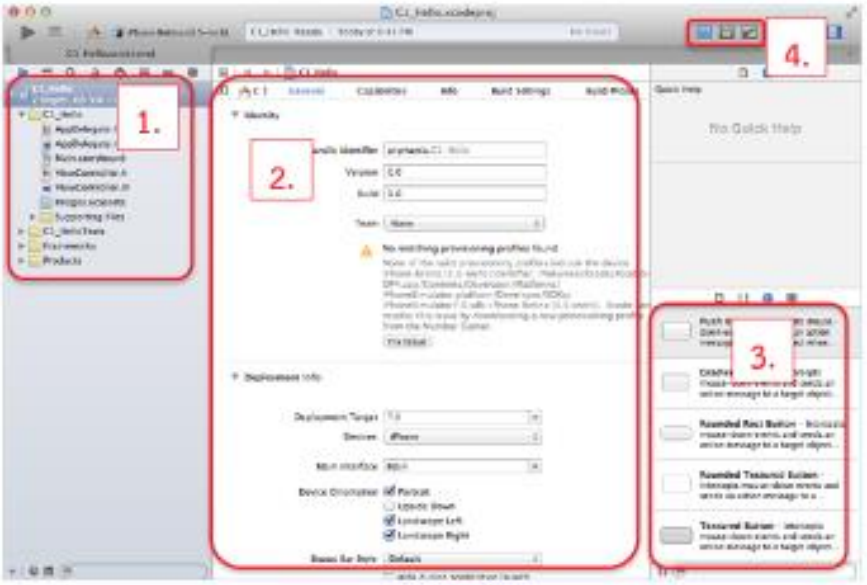
```
enum FirstYearOfDecade: Int
{
    case Seventies = 1970
    case Eighties  = 1980
    case Nineties  = 1990
}
```

```
switch otherDecade
{
case .Nineties:
    println("I love the 90s")
default:
    println("What decade are you talking about?")
}
```

- ใช้ระบุค่าใน **LevelProgress**

```
enum LevelProgress
{
    case Completed(stars: Int)
    case AttemptedButIncomplete(percentComplete: Double, lastAttemptedDate: NSDate)
    case Unattempted
}
```

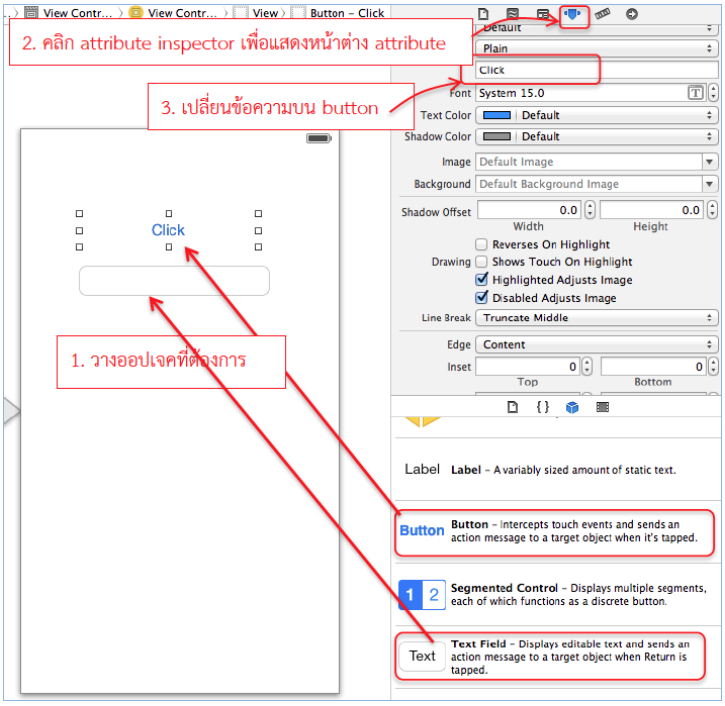
```
func printLevelDetails(level: LevelProgress)
{
    switch level
    {
    case .Unattempted:
        println("Didn't attempt")
    case .AttemptedButIncomplete(let percentComplete, let lastAttemptedDate):
        println("About \(percentComplete) complete, last tried \(lastAttemptedDate).")
    case .Completed(let stars):
        println("Complete with \(stars) stars.")
    }
}
```



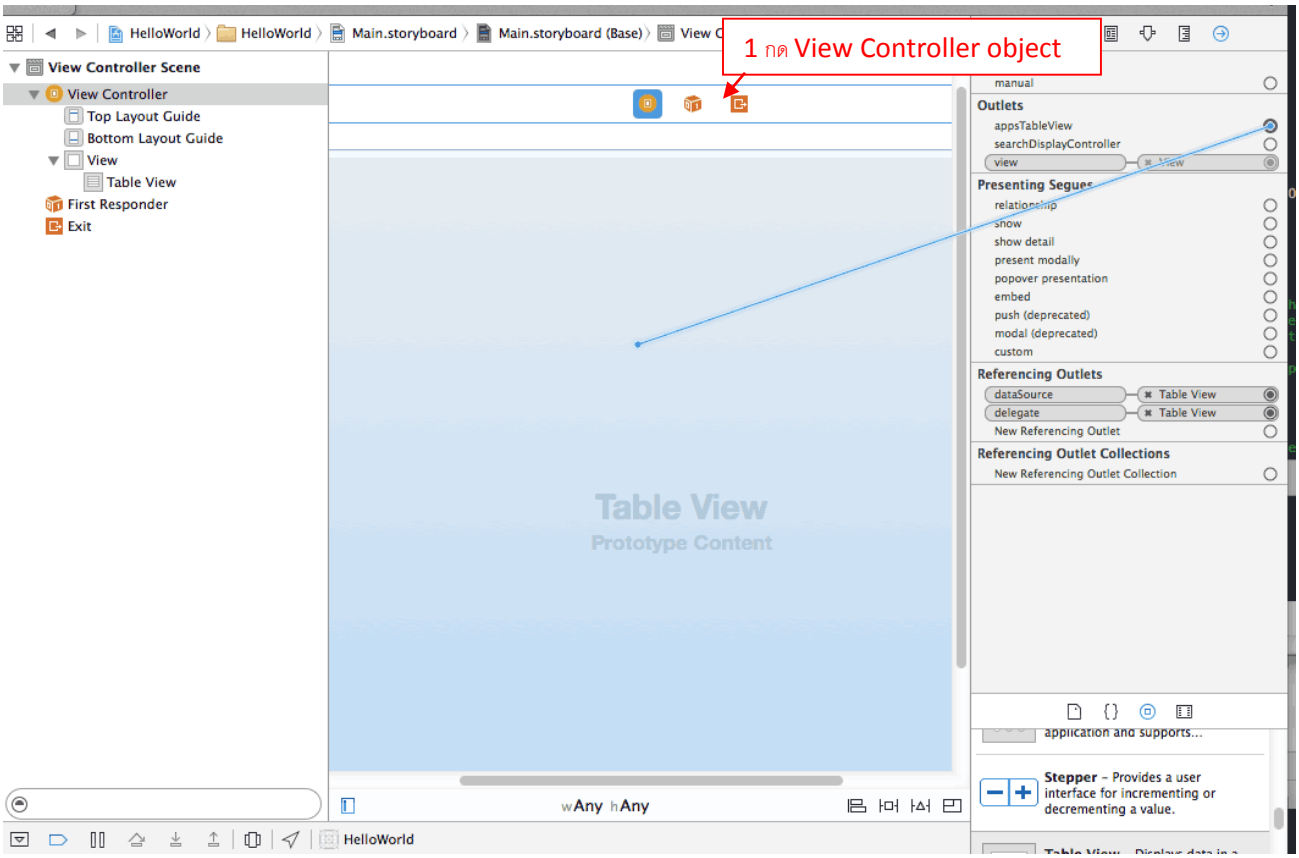
1. project navigator เพื่อเลือกไฟล์ที่เราต้องการทางาน หลักๆตอนนี้มี 3 ไฟล์คือ Main.storyboard ViewController.h และ ViewController.m
2. file editor แสดงรายละเอียดไฟล์ที่เราเลือก
3. object library เก็บออปเจคต่างๆที่เราจะนามาใช้ในโปรแกรม เช่น button, textfield, label
4. เมนู show editor menu เพื่อเลือกรูปแบบการแสดงไฟล์ มีที่เราสลับใช้ไปมา 2 แบบคือ standard editor และ Assistant

สร้าง **GUI Output**

1. ลาก **object** มาใส่ใน **storyboard** เช่น **Tableview, imageview, textview**



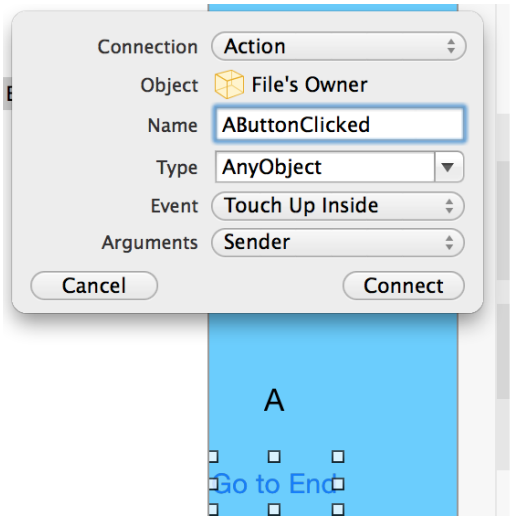
2. ประกาศตัวแปรบนโค้ด **@IBOutlet var appsTableView : UITableView!** (หรือจะกด **control +** ลากจาก **storyboard** มาที่โค้ด สร้างเป็น **outlet**)
3. เชื่อมต่อ **UI** กับ **Outlet** โดยกด **outlets +** ลากมาที่ **storyboard** แต่ต้องกด**view controller object** ที่ 1 ก่อน



4.

สร้าง **GUIAction**

1. ลาก **object** มาใส่ใน **storyboard** เช่น **button**
2. กด **control +** ลากจาก **storyboard** มาที่โค้ด สร้างเป็น **Action**




```
//  Copyright (c) 2014 Brendan Lee. All rights reserved.
//

import UIKit

class AViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view,
            typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }


}
```

3. จะได้ **@IBAction func AButtonClicked(sender: AnyObject) {** เพิ่มการทำงาน**}**