M.C. María Antonia Ruíz Díaz - Facultad de
Ciencias de la Computación - BUAP

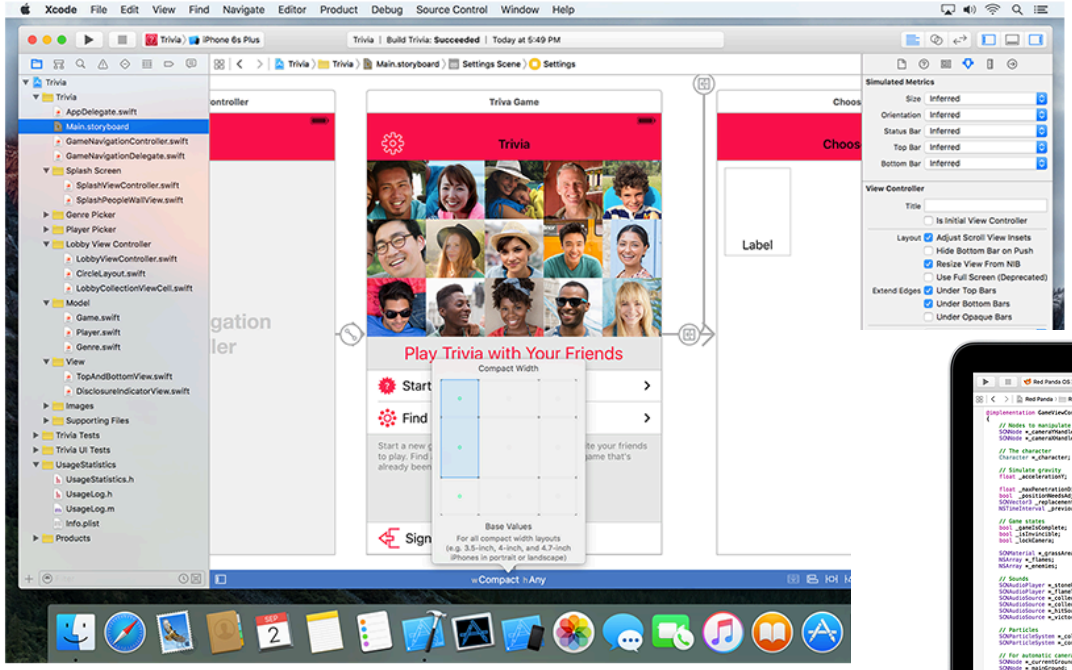
Introducción

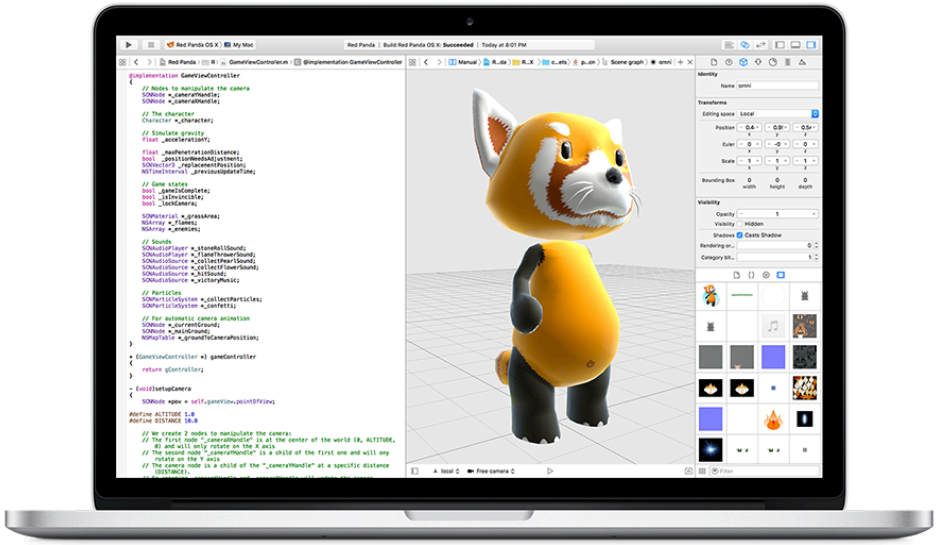
Desarrollo de Aplicaciones Móviles para iOS

El IDE Xcode 7.2

- Entorno de desarrollo integrado de Apple Inc., se suministra gratuitamente con Mac OS.
- Incluye la colección de compiladores del proyecto GNU, puede compilar código en C, C++, **Swift**, Objetive-C, Objetive C++, Java y AppleScript.
- Última versión de Xcode es7.2

Xcode


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Triva Game
Trivia
Play Trivia with Your Friends
Label



Swift

- Es un lenguaje de programación creado por Apple para desarrollar aplicaciones para iOs, Mac, Apple TV, Apple Watch.
- Es fácil de usar y de código abierto.
- Última versión Swift 2.1.1

El playground

- El playground nos muestra una ventana que nos va indicando el resultado obtenido de cada una de las líneas que introducimos en el programa y cada vez que alguna de ellas se edita, ésta se re-compila, y nos muestra el resultado.
- El código se ejecuta de arriba hacia abajo, en orden de cada línea escrita.

El playground

El playground

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

print("Hola mundo")
```

"Hola mundo\n"

El lenguaje Swift

- Variables y constantes: Se usa la palabra reservada **let** para declarar constantes y **var** para declarar variables.

```
1  var myVariable = 42
2  myVariable = 50
3  let myConstant = 42
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

var a = 7.7
var b = 42
var c = 20
b = 90
b = b + c
c = a + b

let d = 10
d = 12
```

7.7
42
20
90
110

El lenguaje Swift

- **Asignación implícita y explícita:** En Swift no es necesario especificar el tipo de variable o constante declarado.

```
1  let implicitInteger = 70
2  let implicitDouble = 70.0
3  let explicitDouble: Double = 70
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

var a: Double = 7.7
var b: Int = 42
var c = 20
var d = "Hola"
var e: String = " Swift"

var f = d + e
```

```
7.7
42
20
"Hola"
" Swift"

"Hola Swift"
```

El lenguaje Swift

- **Conversión de tipos:**

```
1  let label = "The width is "
2  let width = 94
3  let widthLabel = label + String(width)
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

let label = "The width is "
let width = 94
let widthLabel = label + String(width)
```

```
"The width is "
94
"The width is 94"
```

El lenguaje Swift

- Esta es una forma simple de incluir valores en los Strings.

```
1  let apples = 3
2  let oranges = 5
3  let appleSummary = "I have \(apples) apples."
4  let fruitSummary = "I have \(apples + oranges) pieces of
           fruit."
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

let apples = 3
let oranges = 5
let appleSummary = "I have \(apples) apples."
let fruitSummary = "I have \(apples + oranges) pieces of fruit."
```

```
3
5
"I have 3 apples."
"I have 8 pieces of fruit."
```

El lenguaje Swift

- **Arreglos y diccionarios:**

```
1  var shoppingList = ["catfish", "water", "tulips", "blue paint"]
2  shoppingList[1] = "bottle of water"
3
4  var occupations = [
5      "Malcolm": "Captain",
6      "Kaylee": "Mechanic",
7  ]
8  occupations["Jayne"] = "Public Relations"
```

El lenguaje Swift

- Para crear un arreglo o diccionario vacío, inicializa de la siguiente forma:

```
1  let emptyArray = [String]()
2  let emptyDictionary = [String: Float]()
```

El lenguaje Swift

- **Estructuras de control:**

```
1   let individualScores = [75, 43, 103, 87, 12]
2   var teamScore = 0
3   for score in individualScores {
4       if score > 50 {
5           teamScore += 3
6       } else {
7           teamScore += 1
8       }
9   }
10  println(teamScore)
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

let individualScores = [75, 43, 103, 87, 12]
var teamScore = 0

for score in individualScores{
    if score > 50 {
        teamScore += 3
    }
    else{
        teamScore += 1
    }
}
print(teamScore)
```

```
[75, 43, 103, 87, 12]
0

(3 times)

(2 times)

"11\n"
```

El lenguaje Swift

- **Valores opcionales:**

```
1  var optionalString: String? = "Hello"
2  println(optionalString == nil)
3
4  var optionalName: String? = "John Appleseed"
5  var greeting = "Hello!"
6  if let name = optionalName {
7      greeting = "Hello, \(name)"
8  }
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

var optionalString: String? = "Hello"
print(optionalString == nil)

var optionalName: String? = "John Applessed"
var greeting = "Hello"

if let name = optionalName{
    greeting = "Hello, \(name)"
}
```

```
"Hello"
"false\n"

"John Applessed"
"Hello"

"Hello, John Applessed"
```

El lenguaje Swift

- **Switch:**

```
1   let vegetable = "red pepper"
2   switch vegetable {
3   case "celery":
4       let vegetableComment = "Add some raisins and make ants on a
            log."
5   case "cucumber", "watercress":
6       let vegetableComment = "That would make a good tea
            sandwich."
7   case let x where x.hasSuffix("pepper"):
8       let vegetableComment = "Is it a spicy \(x)?"
9   default:
10      let vegetableComment = "Everything tastes good in soup."
11  }
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

let vegetable = "red pepper"

switch vegetable{
    case "celery": let vegetableComment = "Add some raisins ann make
        ants        on a log."
    case "cucumber", "watercrees": let vegerableComment = "That would
        make a good tea sandwich"
    case let x where x.hasSuffix("pepper"): let vegetableComment = "Is
        it a spicy \(x)?"
    default:
    let vegetableComment = "Everything tastes good in soup."
}
```

"red pepper"

"Is it a spicy red pepper?"

El lenguaje Swift

- **Diccionarios:**

```
1   let interestingNumbers = [
2       "Prime": [2, 3, 5, 7, 11, 13],
3       "Fibonacci": [1, 1, 2, 3, 5, 8],
4       "Square": [1, 4, 9, 16, 25],
5   ]
6   var largest = 0
7   for (kind, numbers) in interestingNumbers {
8       for number in numbers {
9           if number > largest {
10              largest = number
11          }
12      }
13  }
14  println(largest)
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

let interestingNumbers = [
    "Prime": [2,3,5,7,11,13],
    "Fibonacci": [1,1,2,3,5,8],
    "Square": [1,4,9,16,25],
]

var largest = 0
for (kind, numbers) in interestingNumbers{
    for number in numbers{
        if number > largest{
            largest = number
        }
    }
}

print(largest)
```

```
["Prime": [2, 3, 5, 7, 11, 13], "Fibonacci": [1, 1, 2, 3, 5, 8], "Square": [1, 4, 9, 16, 25]]

0

(8 times)

"25\n"
```

EL lenguaje Swift

- **while y do_while:**

```
1   var n = 2
2   while n < 100 {
3       n = n * 2
4   }
5   println(n)
6
7   var m = 2
8   do {
9       m = m * 2
10  } while m < 100
11  println(m)
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

var n = 2
while n < 100{
    n = n * 2
}
print(n)


var m = 2
repeat{
    m = m * 2
}while m < 100
print(m)
```

2

(6 times)

"128\n"

2

(6 times)

"128\n"

El lenguaje Swift

```
1   var firstForLoop = 0
2   for i in 0..<4 {
3       firstForLoop += i
4   }
5   println(firstForLoop)
6
7   var secondForLoop = 0
8   for var i = 0; i < 4; ++i {
9       secondForLoop += i
10  }
11  println(secondForLoop)
```

Codigo1

```
//: Playground - noun: a place where people can play

import UIKit

var firstForLoop = 0                                    0
for i in 0..<4{
    firstForLoop += i                                   (4 times)
}
print(firstForLoop)                                     "6\n"


var secondForLoop = 0                                   0
for var i = 0; i < 4; ++i {
    secondForLoop += i                                  (4 times)
}
print(secondForLoop)                                    "6\n"
```

Ejercicios:

1. Instala Xcode.
2. Captura y verifica el funcionamiento de los códigos contenidos en la presentación.
3. Declara de forma explícita una constante de tipo Float con un valor de 4.
4. Intenta quitar la conversión a String de la última línea. ¿Qué tipo de error se obtiene? (Diapositiva 9).
5. Cambia optionalName a nil. ¿Qué saludo obtienes? Agrega una sentencia else que ponga un saludo diferente si optionalName es nil. (Diapositiva 14).
6. Elimina el default. ¿Qué error obtienes? (Diapositiva 15).