



POLITÉCNICA

ETSIT  
UPM

*dit*  
UPM

# Desarrollo de Apps para iOS Playgrounds

IWEB-LSWC 2014-2015

Santiago Pavón

ver: 2015.02.18

# ¿Qué son los Playgrounds?

- Es un nuevo tipo de documento introducido en Xcode 6
  - Extensión de los ficheros: **.playground**
- Se usa para experimentar y probar fragmentos de código.
  - El código Swift se introduce y se ejecuta cada vez que se hace un cambio.
  - Muestra el resultado de ejecutar las expresiones existentes.
  - Se puede acceder a los recursos del proyecto o del sistema (por ejemplo: imágenes)
  - Tiene un timeline para visualizar el estado de la ejecución en cualquier instante de tiempo.



Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

New ▶  
Add Files... ⌘⇧A  
Open... ⌘O  
Open Recent ▶  
Open Quickly... ⌘⇧O  
Close Window ⌘W  
Close Tab  
Close Document ⌘⇧W  
Close Workspace ⌘⇧W  
Save ⌘S  
Duplicate... ⌘⇧S  
Revert to x  
Unlock...  
Export...  
Show in...  
Open with...  
Save As...  
Workspa...  
Create S...  
Restore S...  
Page Set...  
Print...

Tab ⌘T  
Window ⌘⇧T  
File... ⌘N  
Playground... ⌘⇧N  
Target...  
Project... ⌘⇧N  
Workspace... ⌘⇧N  
Group ⌘⇧N  
Group from Selection



# Welcome to Xcode

Version 6.1.1 (6A2008a)

-  **Get started with a playground**  
Explore new ideas quickly and easily.
-  **Create a new Xcode project**  
Start building a new iPhone, iPad or Mac application.
-  **Check out an existing project**  
Start working on something from an SCM repository.

Show this window when Xcode launches

Open another project...

-  Polares.playground  
...parencias/2014-15 IWEB/028 - Playgrounds
-  Playground Pruebas.playground  
...parencias/2014-15 IWEB/028 - Playgrounds
-  Trayectoria.playground  
...parencias/2014-15 IWEB/028 - Playgrounds
-  MyPlayground.playground  
...parencias/2014-15 IWEB/028 - Playgrounds
-  El Vuelo del Grajo  
...30 - Concurrencia y Usabilidad/Grajo/iOS 8
-  Subir Imagen  
...0 - Web Services/demos/E5 - Subir Imagen
-  Busca Pios  
...14-15 IWEB/131 - Demo Twitter/2014-2015
-  Hola Mundo  
~/tmp/borrarme
-  Hola Mundo  
...o - Hola Mundo/Demos/Hola Mundo/iOS 8
-  Hola Mundo con Swift  
...5 IWEB/012 - Demo - Hola Mundo/Pruebas

Quick Look

Añadir al Timeline

The screenshot shows the Xcode playground interface for a Swift program. The interface is divided into three main sections: code, results, and a timeline.

- Code Section:** Contains Swift code for creating a UIView, setting its bounds, creating a UILabel with the text "hola", and rotating it. A loop rotates the label 158 times.
- Results Section:** Displays the output of the code, including the class hierarchy (UIView, UILabel) and the text "hola" in red.
- Timeline Section:** Shows a visual representation of the label's rotation over time, with a red vertical line indicating the current position in the timeline.

Código con el que experimentamos

Resultados

Timeline

Tiempo extra de ejecución

demo2.playground: Ready | Today at 15:11

demo2.playground

```

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

import Foundation

class Trajectory {...}

var tra = Trajectory(speed: 10, angle: {GRAVITY 9.806...
  0.7})

var a: Double
var d: Double

let tf = tra.timeToTarget()

for var t = 0.0 ; t < tf ; t += 0.1 {
  a = tra.alture(atTime: t)
  d = tra.distance(atTime: t)
}

```

Quick Look

Valor de la expresión

1.31383844072684

(14 times)

(14 times)

Sentencia ejecutada 14 veces

Añadidos al Timeline

GRAVITY 9.80665  
speedX 7.64842187284489  
speedY 6.44217687237691

6,884

1,826

d = tra.distance(atTime: t)

a = tra.alture(atTime: t)

- 30 sec +

# Quick Look

- El botón  muestra gráficamente el valor de una expresión.
- Tipos soportados:
  - Colores, Strings, Arrays, Diccionarios, Clases, Estructuras, Imágenes, Views, Bezier paths, Puntos, Dimensiones, Rectángulos, URLs, ...
- Podemos crear Quick Looks personalizados para subclasses de NSObject.
  - Implementando el método:

```
func debugQuickLookObject() -> AnyObject? {  
    return <Color, String, Imagen o Bezier Path>  
}
```

demo4.playground: Ready | Today at 17:08

demo4.playground

```

import UIKit
import XCPlayground

class Circle: NSObject {
    var radius: Float

    init(radius: Float) {
        self.radius = radius
    }

    func debugQuickLookObject() -> AnyObject? {
        let r = CGFloat(radius)
        var p = UIBezierPath(ovalInRect:
            CGRectMake(0, 0, r, r))
        return p
    }
}

var d = Circle(radius:100)

```

100.0  
7 path elements  
{Some 7 path...

Circle

var d...ius:100)

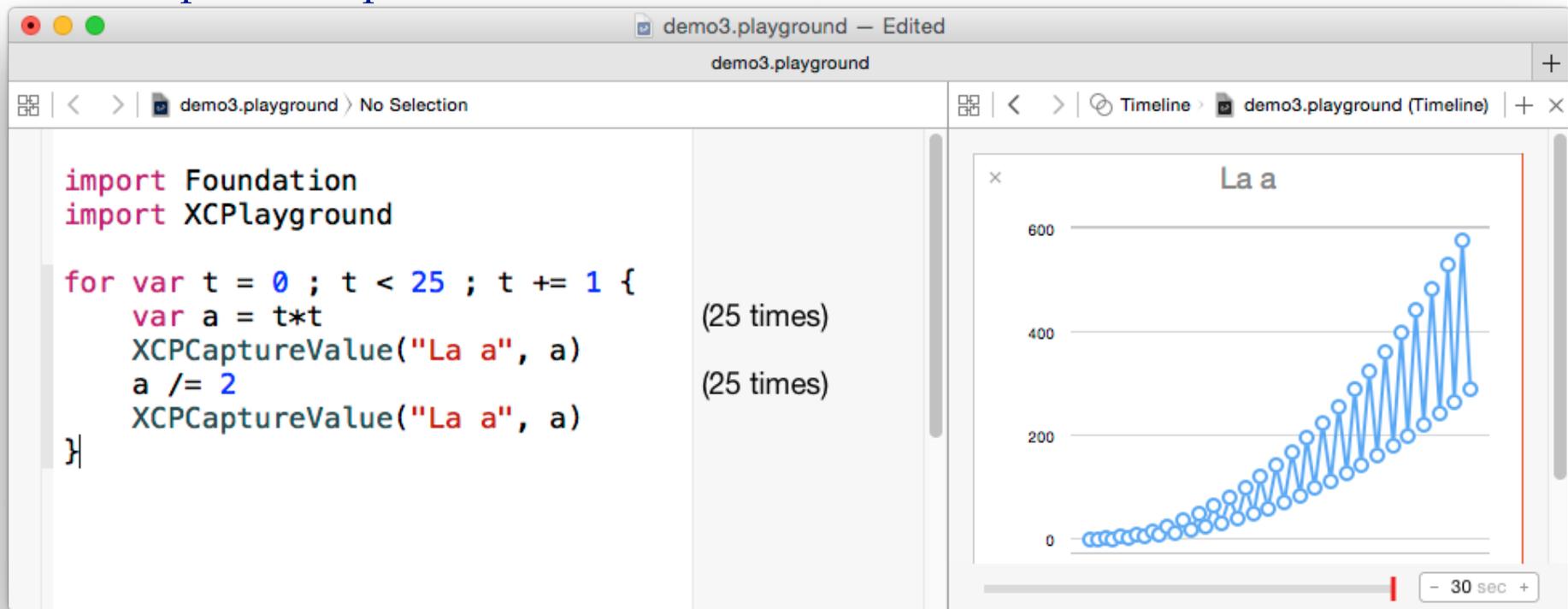
- 30 sec +

# Framework XCPlayground

- Contiene utilidades para usar en los Playgrounds:

```
func XCPCaptureValue<T>(identifier: String, value: T)
```

- Capturar valores en distintas partes del código, asociando los valores capturados con un identificador, y mostrar la evolución de los valores capturados para cada identificador.



The screenshot shows an XCPlayground window titled "demo3.playground - Edited". The code editor displays the following Swift code:

```
import Foundation
import XCPlayground

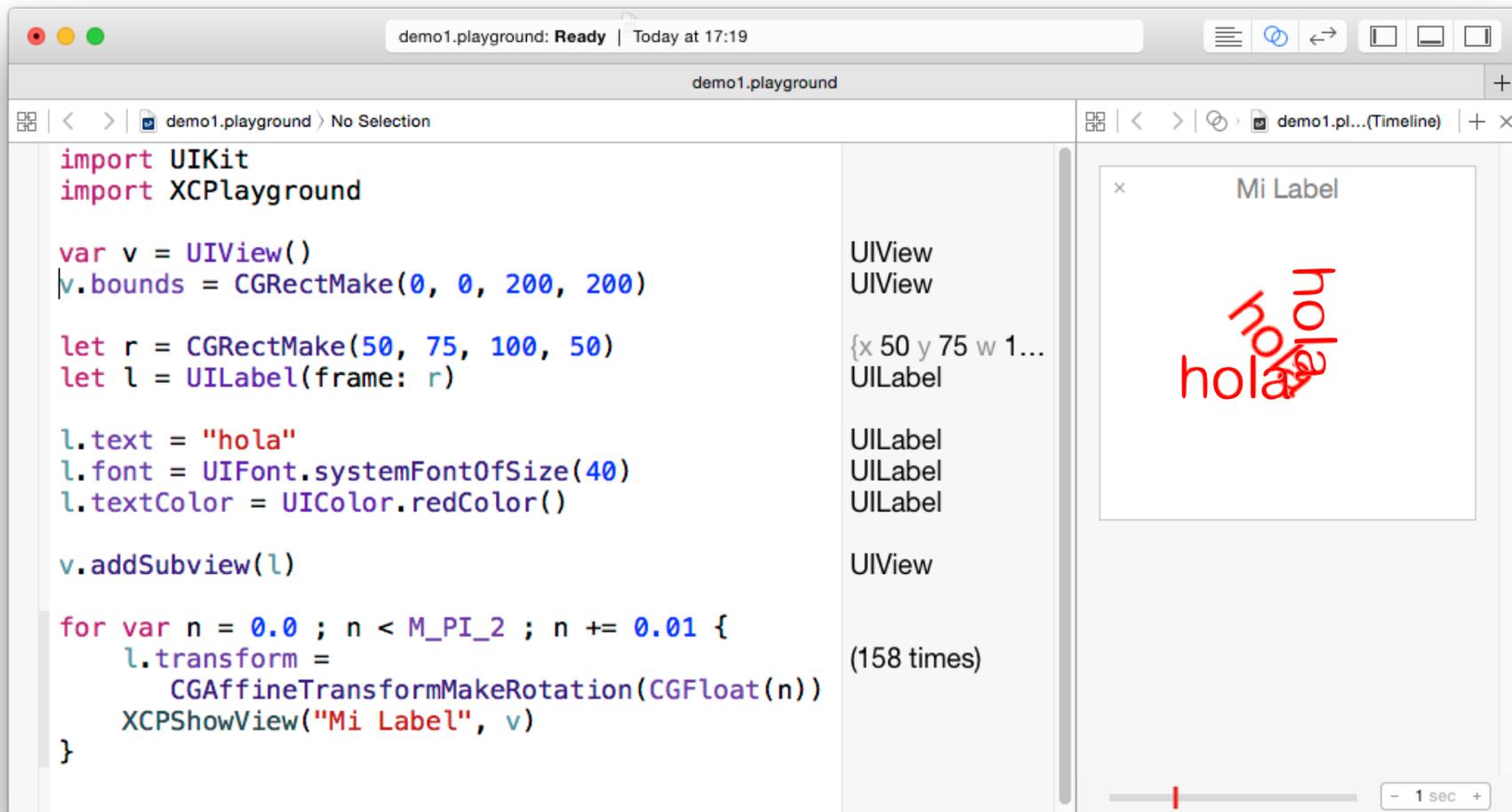
for var t = 0 ; t < 25 ; t += 1 {
    var a = t*t
    XCPCaptureValue("La a", a)
    a /= 2
    XCPCaptureValue("La a", a)
}
```

To the right of the code, there are two annotations: "(25 times)" next to the first `XCPCaptureValue` call and another "(25 times)" next to the second `XCPCaptureValue` call.

On the right side of the window, a timeline view titled "demo3.playground (Timeline)" shows a graph titled "La a". The graph plots the value of 'a' over time, showing a series of blue circles connected by a line, representing the values of 'a' at each iteration of the loop. The y-axis ranges from 0 to 600, and the x-axis represents time. A red vertical line indicates the current position in the timeline, and a slider at the bottom shows a duration of 30 seconds.

```
func XCPShowView(identificador: String, view: UIView)
```

- Mostrar los cambios de una view, asociándolos con un identificador. Se crea una animación con la evolución de la view.
- La view a capturar no debe tener una superview.
  - La view se añade automáticamente a una superview para mostrarse en el timeline.



```
func XCPExecutionShouldContinueIndefinitely() -> Bool
```

```
func XCPSetExecutionShouldContinueIndefinitely(  
    continueIndefinitely: Bool = default)
```

- La ejecución del playground dura mientras se está ejecutando el código introducido.
  - Sin tener en cuenta si existen tareas en background, planificadas para ejecutarse en un futuro, o esperando respuestas de forma asíncrona.
  - Si se edita el código, se detiene la ejecución y se empieza a ejecutar todo de nuevo.
- Adicionalmente, la ejecución se prolonga por el tiempo indicado por el slider de timeout existente en el timeline. Por defecto son 30 segundos adicionales.
  - Estas funciones se usan para indicar (o consultar) si este tiempo adicional de ejecución debe concederse o no.
- Nota: la función **XCPShowView** llama implícitamente a la función **XCPSetExecutionShouldContinueIndefinitely** para habilitar este tiempo adicional.

# Limitaciones

- No es válido para analizar las prestaciones del código.
- No puede usarse para tareas que requieran:
  - interacciones del usuario.
  - entitlements.
  - ejecución en un terminal real.
  - usar código existente en la app/framework.
    - (copiarlo al playground)

