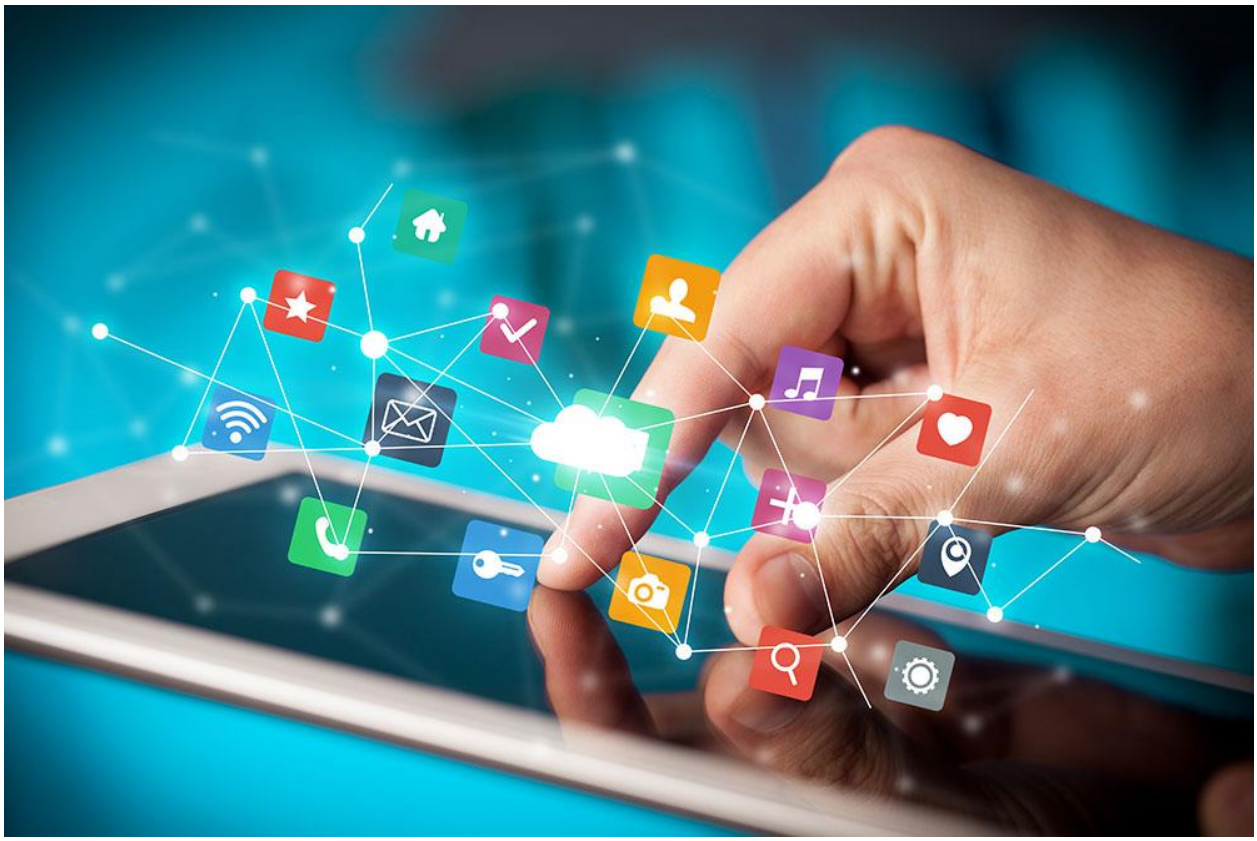
UNIVERSIDAD PONTIFICIA DE SALAMANCA



Manual de Programación

Aplicación en Swift

25/04/2020

Nuria Pacheco Sánchez
Universidad Pontificia de Salamanca
Salamanca

En este documento se recogerá la parte técnica de la aplicación desarrollada en Swift.

El primer fichero que vamos a comenzar es por el fichero ViewController.swift

ImagenPrincipal :Outlet que carga la imagen del muñeco pensando

nickPersona :TextField que almacena el nick de la persona que ha elegido

PalabrasPrincipales : Outlet donde se Almacenan el título de la aplicación

Segmento : Para saber que tipo de color de opción estamos eligiendo

nick :Variable que almacén el nickPersona

```
@IBOutlet weak var IMAGENPRINCIPAL: UIImageView!
@IBOutlet weak var nickPersona: UITextField!
@IBOutlet weak var Segmento: UISegmentedControl!
@IBOutlet weak var PalabrasPrincipales: UILabel!
var nick = " "
override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view.
    PalabrasPrincipales.text = " ¿CUANTAS PALABRAS ERES CAPAZ  DE ADIVINAR "
    IMAGENPRINCIPAL.image = UIImage(named: "Hombre-Pregunta")
}
```

Función Empezar: Cuando damos al botón de empezar, comprobamos que el textfield no es vacío, si es vacío avisamos a través de una alert indicamos que no es un nombre valido. Si este no esta vacío , asignamos el nickPersona a la variable nick y llamamos al segue Prueba para que salte la función prepare , para que pase este nick a ViewController2

```
@IBAction func Empezar(_ sender: UIButton) {
    if !(nickPersona.text?.isEmpty ?? true){
            nick = nickPersona.text!
            performSegue(withIdentifier: "Prueba", sender: self)
    }else{
        let al = UIAlertController (title: "Error", message :"Nombe no valido" , preferredStyle:
.alert)
        al.addAction(UIAlertAction (title : "OK" , style : .default , handler: nil))
```

```
self.present(al,animated: true)
    }
  }
```

Función prepare : La usamos para pasar información de ViewController a ViewController2 , en este caso será el nick del usuario

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    let DVC = segue.destination as! ViewContoller2
    DVC.ni = nick
    DVC.color = color
}
```

Función textFieldShouldReturn : Función para que del teclado intro , la información se quede almacenada.

Se llama resignFirstResponde , para que renuncie a su estado como primer respondedor en su ventana

```
func textFieldShouldReturn(_ textField: UITextField) -> Bool {
      return textField.resignFirstResponder()
    }
```

función touchesBegan : Reconocer el gesto cuando uno o más dedos tocan la vista asociada y que se esconda el teclado con los valores del nick

```
override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
    self.view.endEditing(true)
}
```

Función Color : Esta función sirve para que el usuario cambie el color de la interfaz

Si el segmento elegido es 0 -> Entonces elige la opcion Clara (white) y cambiamos el color de navigation y view a este y el color blanco

Si el segmento elegido es 1 -> Entonces elegimos la opción oscura (black) y cambiamos el color navigation , view y pasamos el color a la variable color (Esta variable pasara el nombre del color al Viewcontroller2)

```
@IBAction func Color(_ sender: UISegmentedControl) {
    if Segmento.selectedSegmentIndex == 0{
        navigationController?.navigationBar.barTintColor = UIColor.white
        self.view.backgroundColor = UIColor.white
        self.color = "white"
```

```
    }else{
        self.view.backgroundColor = UIColor.black
        navigationController?.navigationBar.barTintColor = UIColor.black
        self.color = "black"
    }
}
```

Ahora vamos a pasar a ver el fichero ViewController2.swift

var second : Los 3 minutos dejamos para acertar las palabras

var Tim : La variable que señala al temporizador

var Oculta : Variable que almacena la palabra a Adivinar

var Acierto : Variable que almacena el número de aciertos del usuario

var temp : Variable que almacena el estado anterior de la palabra

var pausar : La variable que se usa si esta en true para pausar el temporizador

```
var second = 180        var Tim = Timer()        var palabra = Palabra()
var Oculta = ""         var s = ""               var acierto = 0
var temp = ""           var pausar = false
```

tiempo :Label que recoge el temporizador en la pantalla

nick : Label que recoge el nick del usuario

palabraOculta : Label donde ira apareciendo la palabra Oculta

NumAcierto : Label donde ira sumandose los aciertos de las palabras adivinadas

Ni : Esta variable almacenara el Nick del ViewController

```
@IBOutlet weak var tiempo: UILabel!
@IBOutlet weak var nick: UILabel!
@IBOutlet weak var Imagen: UIImageView!
```

```
@IBOutlet weak var palabraOculta: UILabel!

@IBOutlet weak var NumAcierto: UILabel!

var ni = ""
```

función ViewDidLoad : Cambiamos el color de la interfaz dependiendo del color que nos hayan pasado por el segue , llamamos a la función time para que inicie el temporizador , pasamos a la label nick el valor del nombre del usuario que está almacenado en la variable ni. Ponemos el número de Acierto a 0 de momento. Llamamos al método CrearRandom de la clase File , para que crear la primera palabra a adivinar y al crearLinea para crear los guiones , que es la longitud de la palabraOculta a adivinar

```
override func viewDidLoad(){
    super.viewDidLoad()
 if color == "black"{
        self.view.backgroundColor = UIColor.black
        navigationController?.navigationBar.barTintColor = UIColor.black
    }else{
        self.view.backgroundColor = UIColor.white
        navigationController?.navigationBar.barTintColor = UIColor.white
    }
     time()
    nick.text = ni
    NumAcierto.text = "Aciertos : " + "\(acierto)"
    //print (ni)
    let (oculta) = palabra.CrearRandom(palabra:"")
    palabraOculta.text = palabra.crearLineasPalabras()
    Oculta = oculta
}
```

Función time : Esta función crea un temporizador. Para eso usamos un método de clase , su valor de retorno se asigna a la constate Tim.

Este tiene un valor de retorno que se le asigna a la constante Tim. Esta constante ahora contiene una referencia al temporizador, el selector llama a la función Updatetime, y el time interval es el intervalo de tiempo que se llama frecuencia al método

```
func time (){
    Tim = Timer.scheduledTimer(timeInterval: 1, target: self, selector:
(#selector(ViewContoller2.updateTime)), userInfo: nil, repeats: true)   }
```

Función PulsaLetra : Esta función está asociada a todos los botones con las letras del abecedario. En primer lugar asociamos la palabra anterior a la variable temp.

En esta función llamamos a la función del fichero File comprobarLetra, para que compruebe si esta letra está o no contenida en la palabra. Si la variable va nos da una nueva string , eso significa que es distinta a la variable temp, entonces esta letra está en la palabra .

Ahora comprobamos si contiene todavía guiones la variable va , si todavía tiene guiones , eso quiere decir que todavía nos falta por acertar más letras de la palabra. Con lo cual asociamos la variable va a la label palabraOculta.

En cambio si no contiene ningún guion más , eso quiere decir que esta la palabra completa y hemos acertado la palabra , con lo cual sumamos el acierto al marcador, paramos el tiempo del temporizador, y salta la alerta como que hemos acertado la palabra , con dos botones Ok

Ok -> vuelve a crear otra palabra nueva para acertar, comprobamos si es la última palabra del array. Si no se crea de nuevo la pantalla de ViewController2, empieza a contar el contador del tiempo desde el donde se dejo pausado.

Cancel-> Antes de volver al ViewController , Nos salta una alerta diciendo Adios y el numero de palabras que el usuario hay acertado

Si por el contrario si la variable va es igual a la variable temp, significa que esa letra no esta contenida en la palabra oculta, por lo tanto llamamos a la función cargarImagen()

```
@IBAction func PulsaLetra(_ sender: UIButton) {
      let le = sender.currentTitle!
       temp = palabraOculta.text!
      var va = palabra.comprobarLetra(letra: le  ,anterior: temp)
    ultimaPalabra(ultima: va)
      if va != temp{

   i
f va.contains("-"){
          palabraOculta.text = va
          palabraOculta.adjustsFontSizeToFitWidth = true
           print ("La letra que ha sido pausada" , va)

      }else{
```

```
palabraOculta.text=va
            acierto += 1
            NumAcierto.text = "Aciertos : " +  "\(acierto)"
            Tim.invalidate()
            self.pausar = true

            let alert = UIAlertController(title: "Acertates" , message : nil , preferredStyle: .alert)
            let imagen = UIImage(named: "HOMERpng")

            let imagView = UIImageView (frame : CGRect (x : 10 , y : 50 , width: 250 , height: 230 ))
             imagView.image = imagen
            alert.view.addSubview(imagView)

          let height = NSLayoutConstraint (item : alert.view , attribute: .height , relatedBy: .equal ,
toItem: nil,attribute: .notAnAttribute , multiplier: 1 , constant: 320)
            let width = NSLayoutConstraint (item : alert.view , attribute: .width , relatedBy: .equal ,
toItem: nil,attribute: .notAnAttribute , multiplier: 1 , constant:250)

            alert.view.addConstraint(height)
            alert.view.addConstraint(width)
            //alert.addAction(action)
            let action1 = UIAlertAction (title: "Ok", style: .default, handler: {
               action in
               self.time()
               self.temp = va
               self.palabraOculta.text = ""
               let (pala) = self.palabra.CrearRandom(palabra: self.temp)
               self.ultimaPalabra(ultima: pala)
               self.palabraOculta.text=self.palabra.crearLineasPalabras()
               self.Oculta = pala
               self.Imagen.image = nil
```

```
            self.view.layoutIfNeeded()
        })
        let action2 = UIAlertAction (title: "Cancel", style: .default, handler: {
            action in
            let al = UIAlertController (title: "ADIOS", message : self.ni + " Has acertado " +
"\(self.acierto) " + "palabras", preferredStyle: .alert)

                al.addAction(UIAlertAction (title : "OK" , style : .default , handler: {
                  action in
                   self.Tim.invalidate()
                    self.pausar = true
                   self.navigationController?.popViewController(animated: true)
                }))
                self.present(al, animated : true)
        })
        alert.addAction(action1)
        alert.addAction(action2)
       //  pruneNegativewidthConstraints()
        self.present (alert , animated : false)
    }
    }else{
   //     print("Esa letra no esta en la palabra", le)
        cargarImagen()
    }
}
```

Función updateTime : Esta función la uso para que vaya restando uno al timer , si el timer llega a ser menor de 0 . Este se para , para que no siga contando y salta una alerta de tiempo terminado , con las palabras que ha acertado el usuario y con un boton Ok que nos devuelve a la pantalla Viewcontrolller

```
  @objc func updateTime () {
```

```
if second < 1{

Tim.invalidate()

            let al = UIAlertController (title: "Se acabo el tiempo", message : ni + " Has acertado "
+ "\(acierto) " + "palabras" , preferredStyle: .alert)

            al.addAction(UIAlertAction (title : "OK" , style : .default , handler: {

               action in

             self.Tim.invalidate()

              self.pausar = true

               self.navigationController?.popViewController(animated: true)

            }))

            self.present(al, animated : true)

          }else {

          second -= 1

        tiempo.text = formato (time : TimeInterval (second))

             }

       }
```

función Formato : Esta función formatea el timer , para que salga de la manera 00:00

```
func formato(time : TimeInterval) -> String {

   let minutes = Int (time) / 60 % 60

   let seconds = Int (time) % 60

   return String(format : "%02i : %02i" , minutes , seconds)

}
```

Función ultimaPalabra : Esta función se usa si hemos acertado todas las palabras que contiene la Array en el tiempo estimado, entonces hemos quedado a la array sin palabras para

adivinar . Esta función comprueba si la variable ultima que nos pasa es igual "" , si es así salta una alerta de todas la palabra acertadas, con un boton Ok , que nos devuelve a la primera pantalla de ViewController

```
func ultimaPalabra (ultima:String){
    if ultima == ""{

        let alert = UIAlertController(title: "Acertates TODAS" , message :ni + "Has acertado " +
"\(acierto) " + "palabras", preferredStyle: .alert)

                let action1 = UIAlertAction (title: "Ok", style: .default, handler: {
                 action in
                        self.navigationController?.popViewController(animated: true)


                })
                alert.addAction(action1)

                self.present (alert , animated : false)

}
}
```

Función cagarImagen : Va cargando la imagen del muñeco del ahorcado en la UIImageView . Depende de qué estado de la imagen estemos , va cargando la siguiente de este Si creamos todo el muñeco del ahorcado, salta una alerta de Game Over, con las palabras que ha acertado el jugador y un botón ok que nos devolverá a la pantalla ViewController

```
func cargarImagen (){
    if Imagen.image == nil{
        Imagen.image = UIImage (named : "base1")
    }else{
        if Imagen.image == UIImage (named :"base1"){
            Imagen.image = UIImage(named: "base2")
        }else if Imagen.image == UIImage (named : "base2"){
```

```
UIImage (named: Imagen.image = "base3")
        }else if Imagen.image == UIImage (named : "base3"){
            Imagen.image = UIImage (named: "base4")
        }else if Imagen.image == UIImage (named : "base4"){
            Imagen.image = UIImage (named: "base5")
        }else if Imagen.image == UIImage (named : "base5"){
            Imagen.image = UIImage (named: "base6")
        }else if Imagen.image == UIImage (named : "base6"){
            Imagen.image = UIImage (named: "base7")
        }else if Imagen.image == UIImage (named : "base7"){
            Imagen.image = UIImage (named: "base8")
        }else if Imagen.image == UIImage (named : "base8"){
            Imagen.image = UIImage (named: "base9")
        }else if Imagen.image == UIImage (named : "base9"){
            Imagen.image = UIImage (named: "base10")
            let al = UIAlertController (title: "GAME OVER", message : ni + " Has acertado " +
"\(acierto) " + "palabras", preferredStyle: .alert)

            al.addAction(UIAlertAction (title : "OK" , style : .default , handler: {
              action in
               self.Tim.invalidate()
                self.pausar = true

               self.navigationController?.popViewController(animated: true)



            }))
            self.present(al, animated : true)
        }
    }
}
```

```
}
```

Por último, Comentaremos el fichero File.swift

VARIABLES:

Oculta: Esta variable almacena la palabra que hay que almacenar

array palabraOculta: Array donde esta almacenada las posibles palabras Ocultas del juego

s: Variable String donde almacenaremos tantos guiones como caracteres tenga la palabra Oculta

```
var oculta = ""
var palabraOculta = ["ESPAÑA", "BRASIL", "TURQUIA" , "ETIOPIA", “COLOMBIA”, “ARGENTINA”, “FRANCIA”, “ARGELIA” ]
var s = ""
```

Funcion CrearRandom : Esta función la usare para crear de forma aleatoria la palabra Oculta. Esto se hace mediante el método randomElement, que elige de manera aleatoria una palabra de la array PalabraOculta. Esta palabra se guarda en la variable oculta. Si la array Palabraoculta se queda vacía , en la variable oculta se almacena como vacía.

En la función nos pasan un String palabra . Esta variable la usamos en la función EliminarPalabra , que comentaremos más abajo para que lo uso. Y siempre devolvemos el valor de la variable oculta

```
func CrearRandom(palabra : String)->(String){
    var randomName = ""
    //var item = 0
    //let longi = palabraOculta.count
    palabraOculta = EliminarPalabra(elimina: palabra, lista: palabraOculta)
      if !palabraOculta.isEmpty {
      randomName = palabraOculta.randomElement()!
      oculta = randomName
      }else{
        oculta = ""
    }

    return (oculta)
```

```
}
```

Función crearLineasPalabras() : Esta función crea los guiones con la longitud de la palabra a adivinar.

s es vacía, entra en el while y rellena s con los guiones de la palabra Oculta.

Si s no esta vacía, la volvemos vacía (Esto es porque vienes de una palabra anterior acertada)

```
func crearLineasPalabras ()->String{
var i = 0
   if !s.isEmpty{
      s = ""
   }
   while (i < oculta.count){
   s += "-"
   i += 1
 }
   return s
}
```

Función comprobar Letras : Esta función la uso para ir comprobando las letras de la palabra , con las letras que ha pausado el usuario. Si esta está en la palabra llamamos a la fucion remplaza , si no devolvemos el estado anterior de la palabra.

```
func comprobarLetra ( letra : String, anterior : String)->(String){
    var gs = anterior
   for item in 0..<oculta.count{
      let pal = oculta.index(oculta.startIndex, offsetBy: item)
      var r = oculta[pal]
      if  r == Character.init(letra) {
   // if let index = oculta.index (of: Character.init(letra)){
    //if  let firstIndex : Range <String.Index> = oculta.range(of: letra)  {
    //    let index = oculta.distance(from: oculta.startIndex, to: firstIndex.lowerBound)
          gs = remplaza(offset: item , le: letra)
          // print (index)
```

```
    //  }
      }
    }
      return gs
  }
```

Función remplaza : Esta función la uso cuando una letra está en la palabra, para remplazar los guiones por la letra que nos han pasado en la posición que le corresponde. Devolveremos la variable s con los cambios

```
  func remplaza (offset : Int, le : String ) -> String{
    // var temp = anterior
     let index = s.index (s.startIndex , offsetBy: offset)
     let range = index...index
     s.replaceSubrange(range, with: le)

     return s
  }
```

Funcion EliminarPalabra : Esta función la uso para eliminar la palabra que ya hemos usado en el juego. Para ello usamos el metodo filter , creando una nueva array con las palabras que son distintas a la palabra ya utilizada. Devolvemos la nuevo array. Esta función la usamos más arriba en la función crearPalabra.

```
  func EliminarPalabra (elimina : String ,lista:[String])->[String]{

     let pal = lista.filter{ $0 != elimina}
         return pal

     }
```

Función Activar Tiempo : Sirve para activar o desactivar el tiempo arriba. Jugar sin temporizador o con él.

si el Switch esta activado -> el tiempo esta corriendo y la label la tenemos activada

Si el switch esta desactivado -> el tiempo se para y la label esta desactivada

```
@IBAction func ActivarTiempo(_ sender: UISwitch) {
    if sender.isOn{
        tiempo.isEnabled = true
        time()
    }else {
        tiempo.isEnabled = false
        Tim.invalidate()
        pausar = true
    }

}
```

En el siguiente enlace he colgado el manual del usuario de la aplicación , porque contiene imágenes gif para su mejor explicación.

Enlace Manual Usuario

Video usando aplicación

Documentación utilizada:

https://developer.apple.com/documentation

https://stackoverflow.com/

https://learnappmaking.com/timer-swift-how-to/

https://medium.com/ios-os-x-development/build-an-stopwatch-with-swift-3-0-c7040818a10f

https://learnappmaking.com/uialertcontroller-alerts-swift-how-to/

https://stackoverflow.com/questions/28340836/add-image-to-alert-view/28341504