PuntoMovimientoSonido

Aplicación watchOS creada por Daniel Barros López y Yoji Sargent Harada

Requerimientos

Aquí implementamos una aplicación para Apple Watch que se comunica con la app PuntoMovimientoSonido, sirviendo como punto extra de la asignatura.

Tutoriales

Con esta aplicación aprendimos lo más básico del desarrollo para watchOS, además de explorar el framework WatchConnectivity para la comunicación entre aplicaciones watchOS e iOS.

1. Comenzando con watchOS

Para crear una aplicación para Apple Watch abrimos nuestro proyecto con la app para iOS ya desarrollada. Añadimos un nuevo `Target` y seleccionamos `WatchKit App`. Esto creará todos los archivos necesarios en nuestro proyecto para empezar el desarrollo.

Conociendo iOS, watchOS resulta a primera vista fácil de aprender, ya que hace uso de los mismos conceptos de un modo muy simplificado.

En nuestra aplicación creamos una UI muy sencilla con un único label. Lo hacemos todo en Storyboards exactamente igual que haríamos en iOS, salvo que aquí la alineación de los elementos de la interfaz es más simple (alineamos objetos verticalmente arriba, horizontalmente a la derecha, por ejemplo). No hay AutoLayout.
Tenemos un `WKInterfaceController`, que cumple el papel del `UIViewController` en iOS. La inicialización del Interface Controller se hace sobre todo en la función `awakeWithContext(_:)` (algo así como el `viewDidLoad()` de los View Controllers), siendo `willActivate()` y `didDeactivate()` funciones similares a `viewWillAppear()` y `viewWillDisappear()` en iOS (aunque con diferencias).
Sabiendo esto ya tenemos lo que necesitamos para empezar a añadir elementos a nuestra interfaz y manipularlos desde nuestro Interface Controller.

2. Comunicación con aplicación iOS

Haremos uso del framework `WatchConnectivity`, que permite el paso de mensajes entre las dos aplicaciones. Para ello creamos un objeto `WCSession`, asignamos su `delegate` y lo activamos.

```
var session = WCSession.defaultSession()

session.delegate = self
session.activateSession()
```

Debemos declarar `session` como una propiedad de algún objeto que tengamos fuertemente referenciado, ya que necesitaremos acceder a él más allá de su inicialización. Esto lo debemos hacer tanto en la aplicación iOS como en la watchOS. En la aplicación iOS no obstante debemos hacer ciertas comprobaciones antes de usar el session para comunicarnos con el Watch. Debemos asegurarnos de que tenemos un Apple Watch emparejado, con nuestra aplicación instalada y en comunicación con nuestro iPhone.

```
if let validSession = session where validSession.paired &&
validSession.watchAppInstalled && validSession.reachable {
      // La sesión es válida
}
```

`WCSession` nos brinda varias formas de comunicación. Nosotros usamos la más simple de ellas, el paso de mensajes simples enviando objetos dentro de un diccionario. Para enviar un mensaje (en iOS o watchOS) usamos el método `sendMessage(_:replyHandler:errorHandler:)` de `WCSession`

```
validSession?.sendMessage(["mensaje": "hola!"], replyHandler: nil,
errorHandler: nil)
```

y para recibir el mensaje en el otro dispositivo, implementamos la función `session(_:didReceiveMessage:)`

```
func session(session: WCSession, didReceiveMessage message: [String :
AnyObject]) {
      label.setText(message["mensaje"])   // Pondrá "hola!" en el label de
nuestra UI de watchOS
}
```