BrújulaVoz

Aplicación iOS creada por Daniel Barros López y Yoji Sargent Harada

Requerimientos

Con la appFotoVoz se debe identificar mediante interfaz vocal un punto cardinal (Norte, Sur, Este y Oeste) y el porcentaje de error (cantidad entera) en la detección del punto, solo el número, por ejemplo “Norte diez”. Una vez reconocido el patrón vocal se debe mostrar una brújula para que el usuario se oriente en la dirección indicada, cuando lo realice se le debe indicar que ya se ha conseguido.

Tutoriales

Ya que en iOS no existe un framework oficial para reconocimiento de voz, decidimos valernos del dictado por voz del teclado del sistema y un UITextField para obtener el resultado deseado. Existen varias librerías de terceros para reconocimiento de voz pero decidimos no optar por esa vía. También exploramos un poco la librería CoreLocation para la obtención de la orientación de nuestro dispositivo.

1. Reconocimiento de voz (nuestro arreglo)

Usamos un `UITextField` oculto (alpha 0) y un botón que inicia la introducción de texto en el text field. Esto se consigue haciendo al text field nuestro First Responder.

```
textField.becomeFirstResponder()
```

Una vez hecho esto se mostrará el teclado. Para evitar que el usuario pueda introducir la información a mano y forzarlo a usar el dictado por voz implementaremos la función `textField(_:shouldChangeCharactersInRange:replacementString:)` en el `delegate` de nuestro text field.
En ella devolveremos siempre `false`, lo que evitará que el usuario pueda componer una cadena válida manualmente mediante el teclado, ya que la función se llama por cada caracter introducido. Sí permitirá introducir cadenas creadas mediante el dictado, ya que en ese caso la función solo es llamada una vez al finalizar el dictado. Así, lo único que debemos hacer en la función será comprobar si la cadena introducida tiene el formato correcto (punto cardinal en español seguido de un número entero), y devolver false.

```
func textField(textField: UITextField, shouldChangeCharactersInRange range:
NSRange, replacementString string: String) -> Bool {
      // comprobar si string tiene el formato correcto
      // ...
      return false
}
```

2. Obtención de la orientación del dispositivo

Para la orientación usaremos la librería `CoreLocation`. Primero se necesita un objeto `CLLocationManager` para recibir actualizaciones de la orientación del dispositivo:

```
let locationManager = CLLocationManager()
```

Asignamos a nuestro view controller como `delegate` y ya podemos empezar a recibir actualizaciones llamando al método `startUpdatingHeading()`

```
locationManager.delegate = self
locationManager.startUpdatingHeading()
```

e implementamos la función `locationManager(_: didUpdateHeading:)` declarada en el protocolo `CLLocationManagerDelegate`

```
func locationManager(manager: CLLocationManager, didUpdateHeading newHeading:
CLHeading) {
      let degrees = newHeading.magneticHeading
      // ...
}
```

En esta función podemos acceder a la propiedad `magneticHeading` del objeto `newHeading` que nos indica la orientación del dispositivo en grados relativa al norte.