Interfaz de usuario en dispositivos móviles

iOS, sesión 2: Vistas

Puntos a tratar

1. Creación de vistas por código
2. Propiedades de una vista
3. Controles de usuario básicos

Creación de vistas por código

Todo lo que se puede hacer **visualmente** con Xcode se puede hacer también de forma **programática**, ya que lo único que hace el entorno es crear objetos de la API de Cocoa Touch y establecer sus propiedades.

Ventanas

- Las aplicaciones iOS tienen una única ventana, accesible en la propiedad **`window`** del *Application delegate*

```
//En el application:didFinishLaunchingWithOptions: del UIApplicationDelegate
CGRect frame = [[UIScreen mainScreen] bounds];
UIWindow *window = [[UIWindow alloc] initWithFrame:frame];
self.window = window;
[self.window makeKeyAndVisible];
```

Vistas y jerarquía de vistas

- cada vista está asociada a un controller, como ya sabemos, y tiene subvistas

```
UIView* vista = miViewController.view;
UIButton *boton = [[UIButton alloc] init];
[boton setTitle:@"Holaaa" forState:UIControlStateNormal];
[boton setFrame:CGRectMake(0,0,100,50)];
[vista addSubview:boton];
vista.backgroundColor = [UIColor redColor];
```

2. Propiedades de una vista

Algunas propiedades geométricas de las vistas

```
// Limites en coordenadas locales. Su origen siempre es (0,0)
CGRect area = [vista bounds];
// Posición del centro de la vista en coordenadas de su supervista
CGPoint centro = [vista center];
// Marco en coordenadas de la supervista
CGRect marco = [vista frame]
//Transformación afín(escalado y/o rotación y/o traslación)
[vista setTransform:CGAffineTransformMakeScale(2, 1.5)];
```

Color, transparencia, estado...

```
UIButton *boton = [[UIButton alloc] init];
...                  //Habría que establecer el tamaño y contenido
boton.backgroundColor = [UIColor redColor];
boton.alpha = 0.5   //Transparencia del 50%
boton.hidden = YES; //Lo ocultamos, ya no se ve ni recibe eventos
boton.hidden = NO;
boton.enabled = NO; //Lo deshabilitamos
```

3. Controles de usuario básicos

UIAlertView

```
UIAlertView *alert = [[UIAlertView alloc]
       initWithTitle:@"Saludo"
       message:@"Hola usuario"
       delegate:????
       cancelButtonTitle:@"OK"
       otherButtonTitles: nil];
 [alert show]
```

Saber qué ha pulsado el usuario

- Poner como **`delegate`** el objeto que vaya a responder (típicamente el controller)
- Especificar que este objeto sigue el protocolo **`UIAlertViewDelegate`**

```
@interface ViewController : UIViewController <UIAlertViewDelegate>
...
```

Saber qué ha pulsado el usuario (cont.)

- Implementar el método correspondiente del protocolo

```
-(void) alertView:(UIAlertView *)alertView
          didDismissWithButtonIndex:(NSInteger)buttonIndex {
    NSLog(@"Se ha pulsado el boton:%d", buttonIndex);
}
```

Teclado en pantalla

- un problema típico es cómo quitarlo de enmedio. Para quitarlo al pulsar sobre "intro"
  - Crear un *action* con **`Ctrl+Arrastrar`** entre el campo y el **`.m`** del controller.

    En el menú desplegable elegir el evento **`Did end on exit`**
  - En el *action* hacer

```
- (IBAction)pulsadoIntro:(id)sender {
  [sender resignFirstResponder];
  //También valdría esto
  [self.view endEditing:YES;]
 }
```

Teclado sin intro

- El teclado numérico no tiene intro, en este caso lo típico es hacer que se oculte cuando se hace *tap* en el background

```
-(void) touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {
    NSLog(@"touch en la pantalla!!");
    [self.view endEditing:YES];
 }
```