

UI Diseño e implementación

Protocolos y Clases

- Un protocolo es un componente del lenguaje de programación que funciona como “molde” de una determinada clase, a fin de que dicha clase cumpla con lo necesario para obtener un objeto con un comportamiento.
- Una clase implementa o conforma un protocolo
- En otros lenguajes es llamado *Interfaz* o *Contrato* porque se refiere a esto, se establece un contrato que dicta que quien se conforme a tal protocolo esta obligado a realizar ciertas implementaciones.
- Ejemplos:
 - UITableViewDelegate
 - UITableViewDataSource
 - MKAnnotation
 - MKOverlay

Protocolos y Clases

```
class Triangulo: FiguraGeometrica{
```

```
    var ladoA: Float
    var ladoB: Float
    var ladoC: Float
    var altura: Float
```

```
    func CalcularArea()->Double{
```

```
        return (altura * ladoB) / 2
```

```
    }
```

```
    func CalcularPerimetro()->Double{
```

```
        return ladoA + ladoB + ladoC
```

```
    }
```

```
}
```

```
protocol FiguraGeometrica{
```

```
    func CalcularArea() -> Double
```

```
    func CalcularPerimetro() -> Double
```

```
}
```

```
class Cuadrado: FiguraGeometrica{
```

```
    var lado: Float
```

```
    func CalcularArea()->Double{
```

```
        return lado * lado
```

```
    }
```

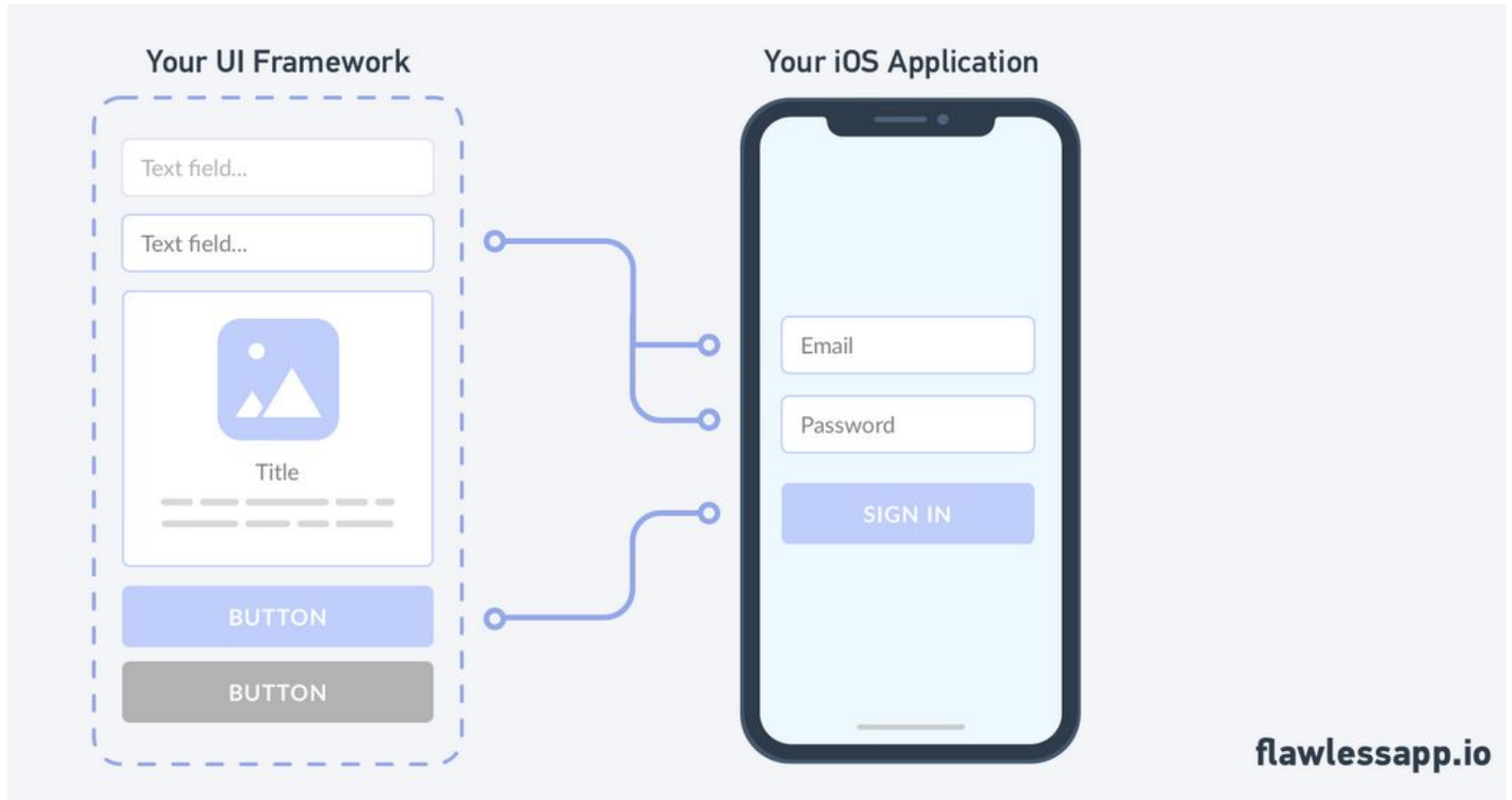
```
    func CalcularPerimetro()->Double{
```

```
        return lado * 4
```

```
    }
```

```
}
```

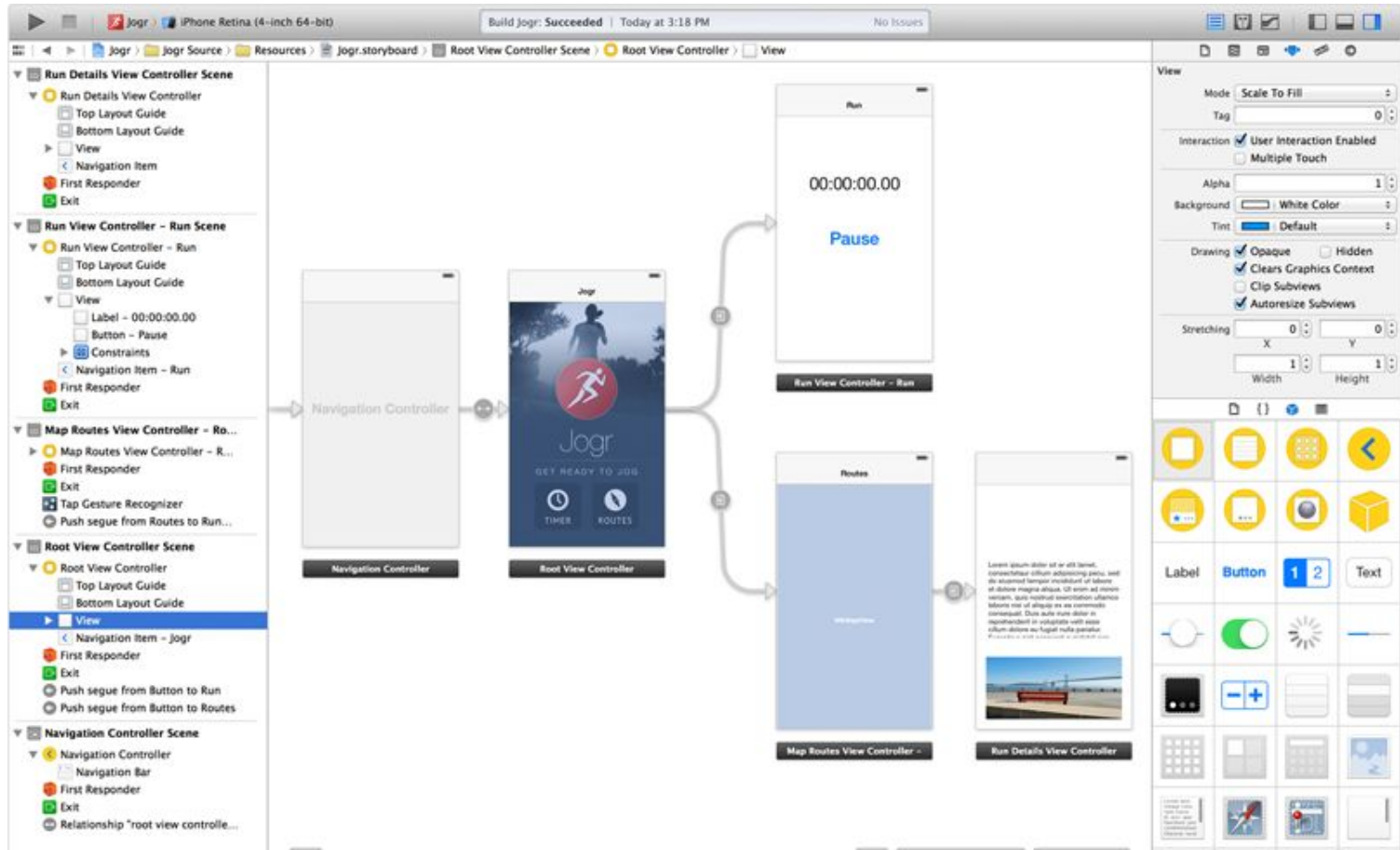
*U*ser *I*nterface



Cocoa Touch

- Originalmente escrito en Objective-C
- Framework para iOS
- Cocoa es el “ayudante” de Swift que provee diferentes esquemas y bibliotecas para desarrollar aplicaciones con interfaces gráficas, a diferencia de éste, Cocoa Touch incluye más funcionalidades y nuevos componentes para el ambiente iOS (sensores).

Interface Builder y UIKit



Interface Builder y UIKit

```
// FoodImageView.swift

import UIKit

class FoodImageView: UIImageView {

    func shake() {
        let animation = CABasicAnimation(keyPath: "position")
        animation.duration = 0.05
        animation.repeatCount = 5
        animation.autoreverses = true
        animation.fromValue = NSValue(CGPoint: CGPointMake(self.center.x - 4.0, self.center.y))
        animation.toValue = NSValue(CGPoint: CGPointMake(self.center.x + 4.0, self.center.y))
        layer.addAnimation(animation, forKey: "position")
    }
}
```



Java™



Objective-C



Kotlin



Swift

© ScienceSoft USA Corporation

Canvas y Swift UI



SwiftUISample

SwiftUISample

AppDelegate

Extensions

Subclasses

Protocols

Models

Services

Utils

Views

TVC

PlaceholderTF

LoginView.swift

HorizontalLineShape.swift

HorizontalLine.swift

TextFieldBottomLine.swift

SecureFieldBottomLine.swift

KeyboardGuardian.swift

GeometryGetter.swift

Storyboards

ViewControllers

Resources

Assets

Assets.xcassets

Fonts

Localization

Plists

Frameworks

Products

Pods

Frameworks

Pods

```
11 struct LoginView: View {
12
13     // MARK: - Body
14
15     var body: some View {
16
17         VStack() {
18             ScrollView {
19                 Image("login").resizable().frame(height: CGFloat(400))
20                 Spacer()
21
22                 VStack(alignment: .leading, spacing: 20) {
23
24                     Text("EMAIL ADDRESS*").font(.callout).foregroundColor(Color.white)
25                     TextFieldBottomLine(placeholder: "").foregroundColor(Color.white)
26
27                     Text("PASSWORD*").font(.callout).foregroundColor(Color.white)
28                     SecureFieldBottomLine(placeholder: "").foregroundColor(Color.white)
29
30                     Button(action: {
31                         // your action here
32                     }) { Text("Forgot password").foregroundColor(Color.purple) }
33
34                     VStack(alignment: .center) {
35                         Button(action: {
36                             // your action here
37                         }) { Text("SIGN IN").foregroundColor(Color.white)
38                             .frame(minWidth: 0, maxWidth: .infinity, alignment:
39                                 .center).padding()
40
41                         }.background(Color.purple).cornerRadius(10)
42                     }
43                     Divider()
44
45                     HStack(spacing: 10) {
46                         Text("Don't have an account?").foregroundColor(Color.white)
47
48                         Button(action: {
49                             // your action here
50                         }) { Text("Sign Up").foregroundColor(Color.purple) }
51                     }
52                     Spacer()
53
54                 }.padding(20).background(Color.black)
55             }.background(Color.black).edgesIgnoringSafeArea(.all)
56         }
57     }
```

09:41

EMAIL ADDRESS*

zaitseva.alina@steelkiwi.com

PASSWORD*

Forgot password

SIGN IN

Don't have an account? Sign Up

Preview



VS



Human Interface Guidelines

- Diseño estructural mínimo de la App
- Interacción del Usuario
- Acoplamiento con el sistema operativo (iOS)
- Iconografía
- Barras, vistas y controles
- Extensiones de Cocoa
- Diseño Visual

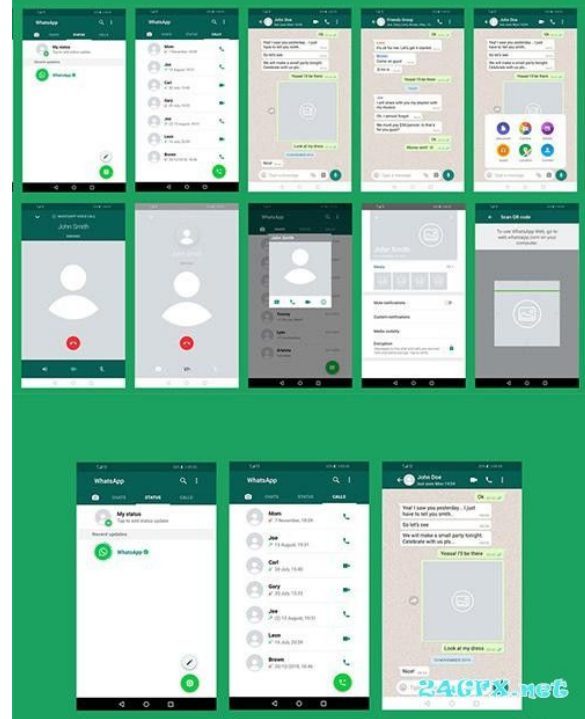
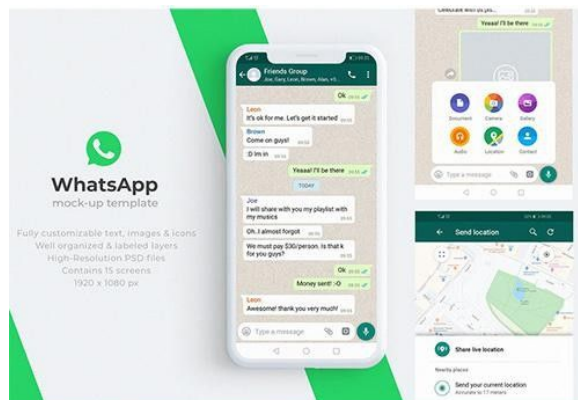


<https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/>

Mock-up

-Prototipo

-Storyboard





[FEATURES](#)

[DEMOS](#)

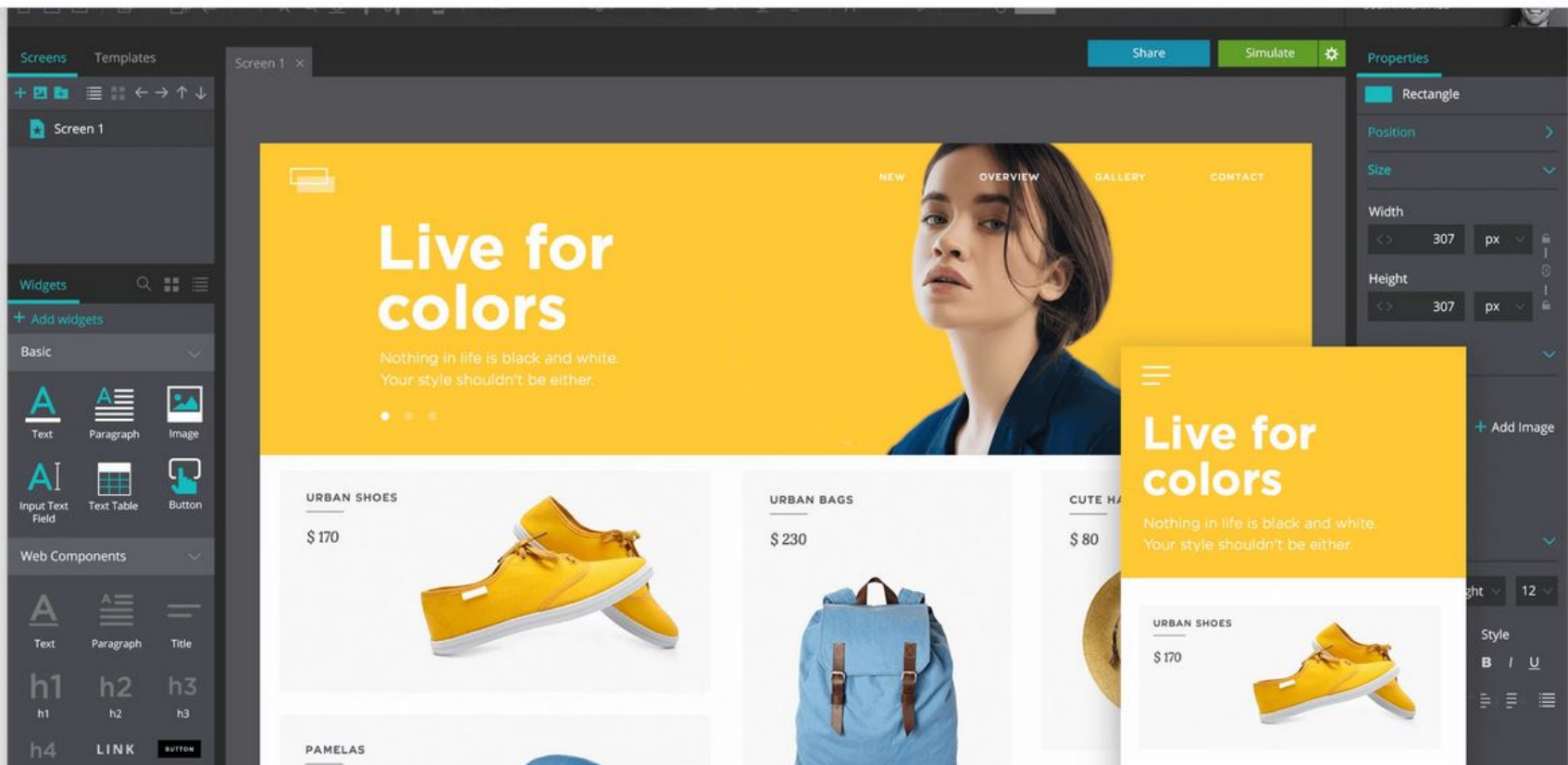
[PLANS](#)

[SIGN IN](#)

[CREATE ACCOUNT](#)

Create Web and Mobile Prototypes in Minutes

[CREATE YOUR FREE LIFETIME ACCOUNT HERE](#)



El FIN :(... .. Recuento de los daños

¿Qué NO vimos ?

- Flujo de datos
- Persistencia de datos completo
 - Core data
 - Implementación de mecanismos Firebase, Realm y AWS de una App Móvil
- Mapas, Safari
- Patrones de Diseño de Cocoa
- Manejo de Errores
- Unit Testing
- Seguridad

Últimas palabras

UIKit puro

<https://youtu.be/1egNops2wrM>

Master Interface Builder

<https://youtu.be/htQATVxILng>

SwiftUI

<https://youtu.be/AWPiup9fE2c>