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

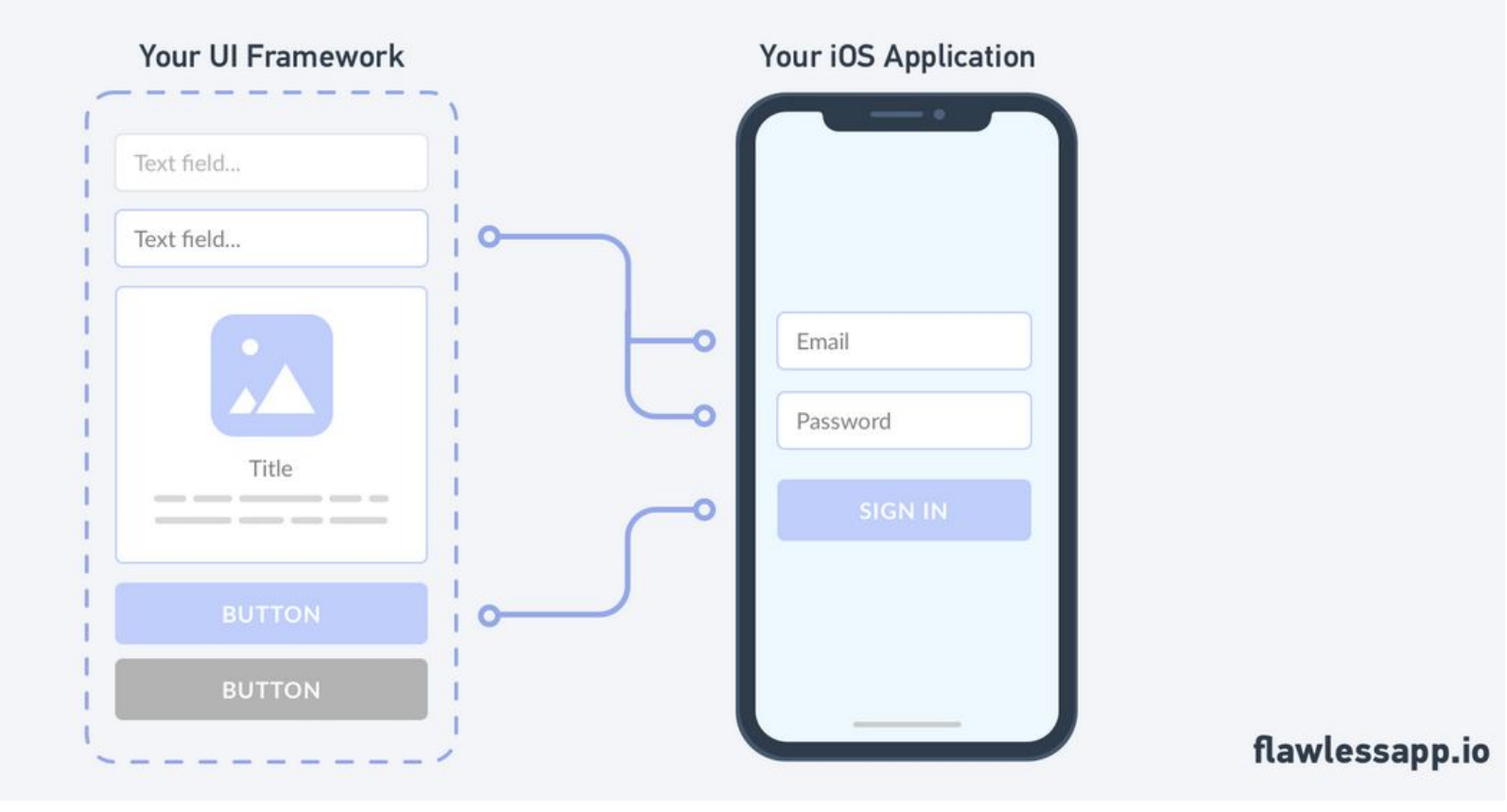
Protocolos y Clases

```
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{

                returen 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
            }
    }
```

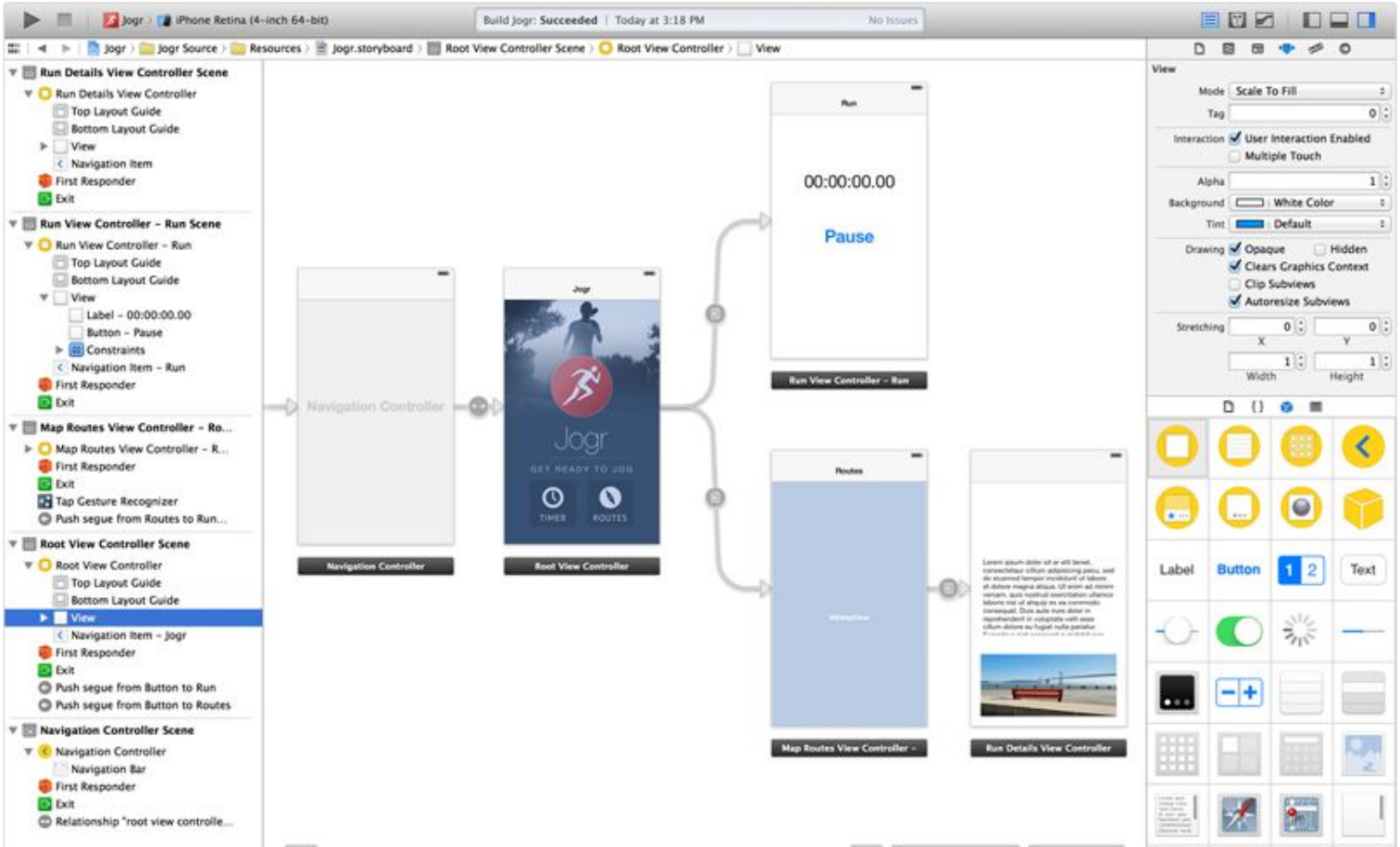
User Interface



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).

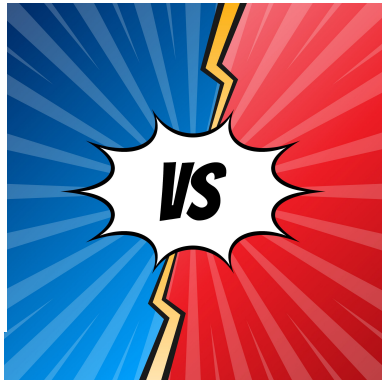
Interface Builder y UIKit



Interface Builder y UIKit

```swift
//  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")
    }
}
```

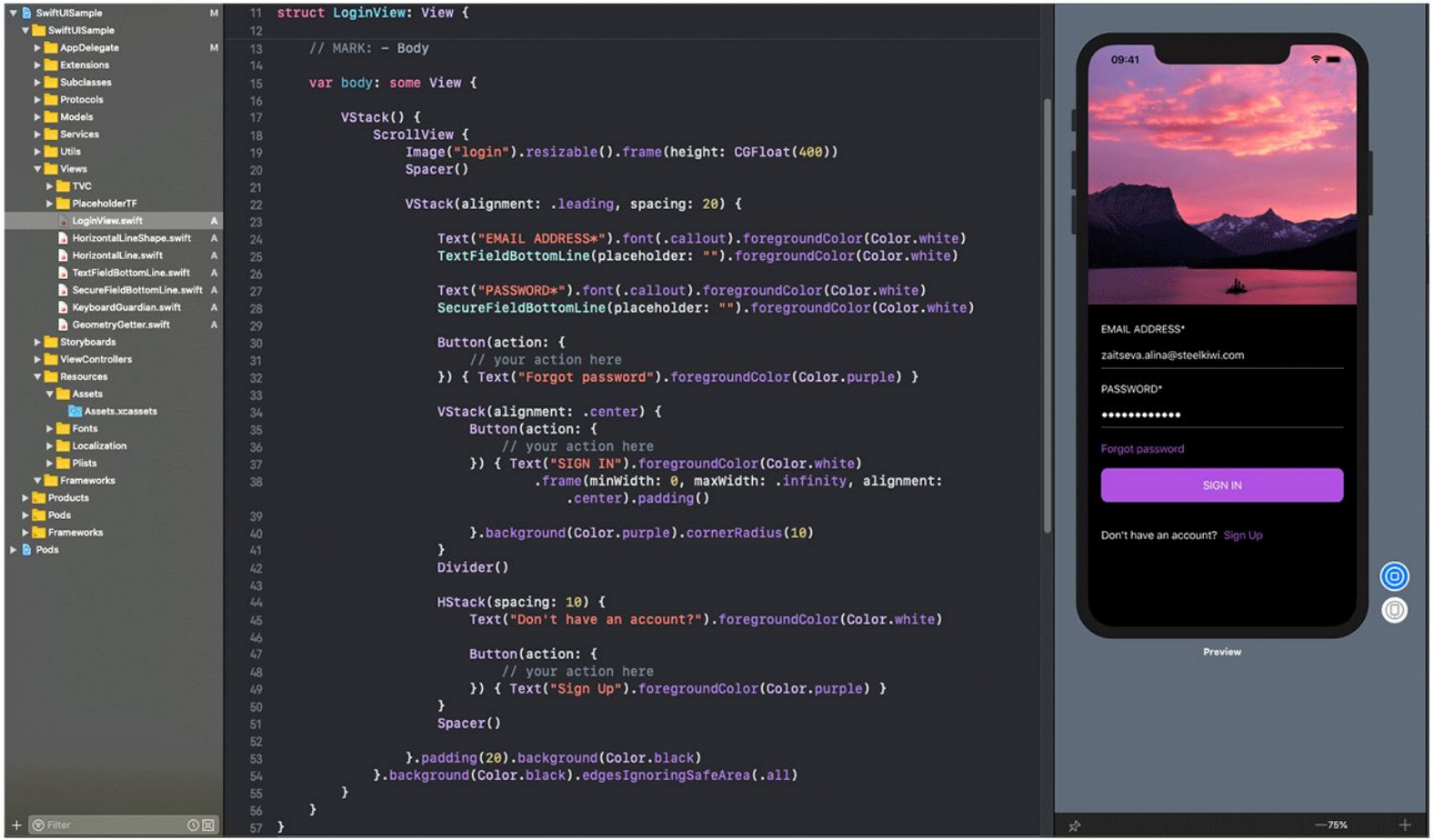

Java™

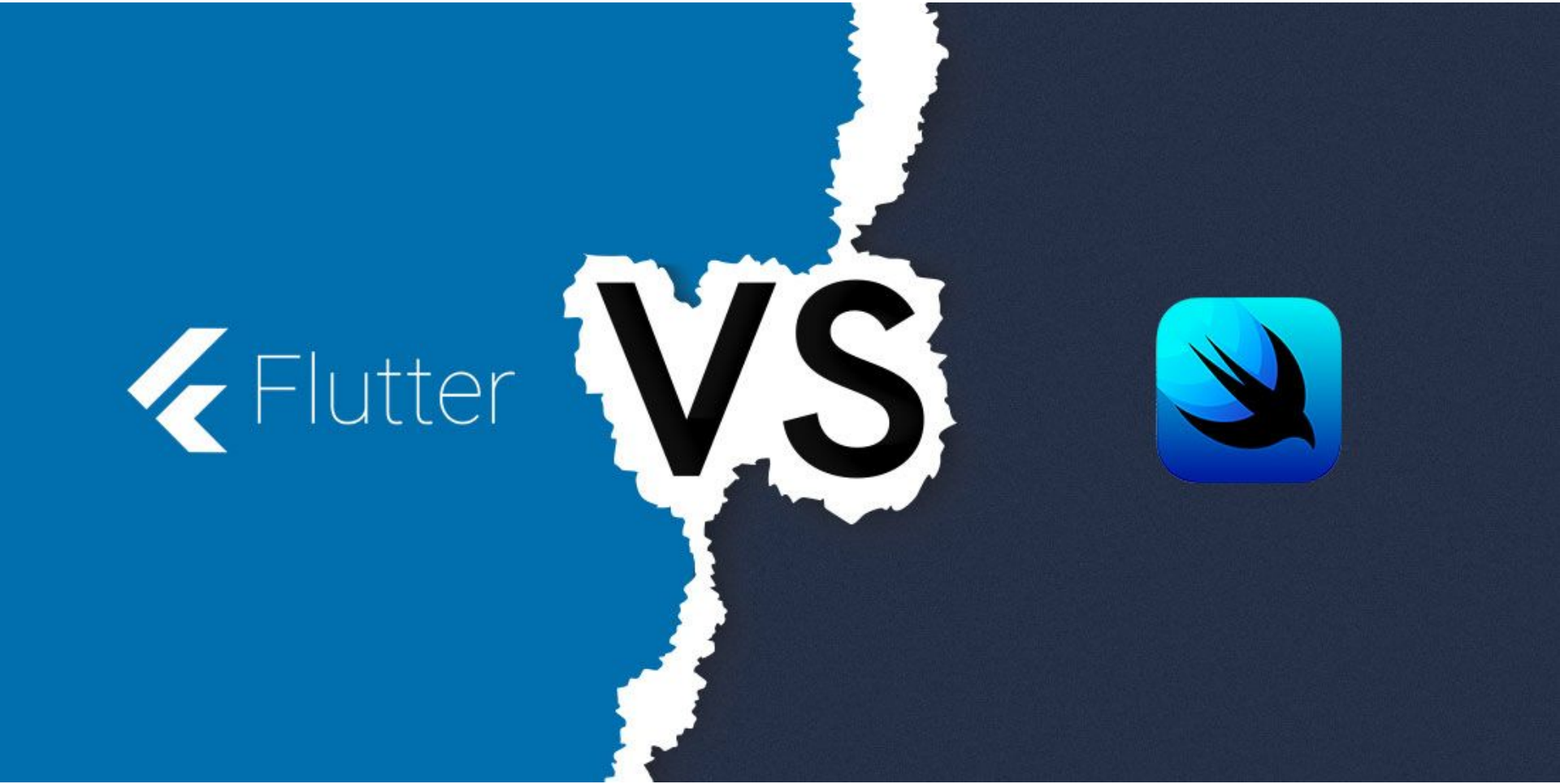

VS


Objective-C


Kotlin
VS
Swift
© ScienceSoft USA Corporation

Canvas y Swift UI




Flutter
VS

Human Interface Guidelines

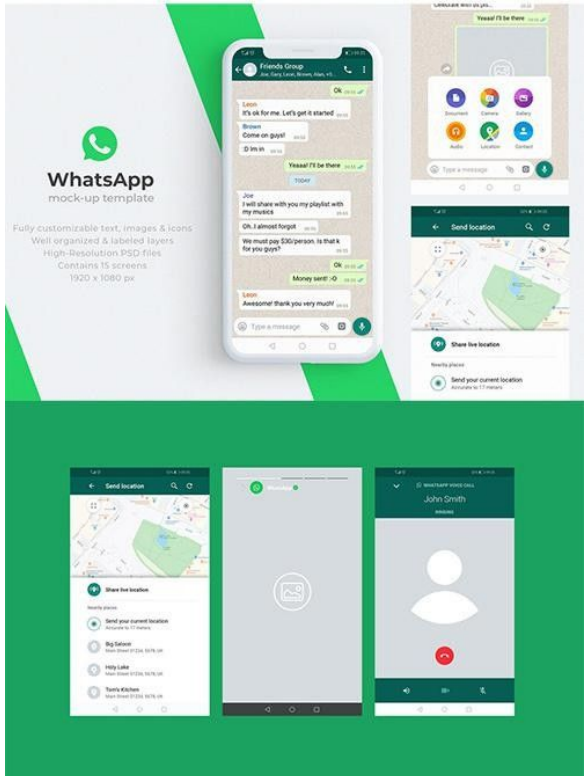
- Diseño estructural mínimo de la App
- Interacción del Usuario
- Acoplamiento con el sistema operativo (iOS)
- Iconografía
- Barras, vistas y controles
- Extensiones de Cocoa
- Diseño Visual

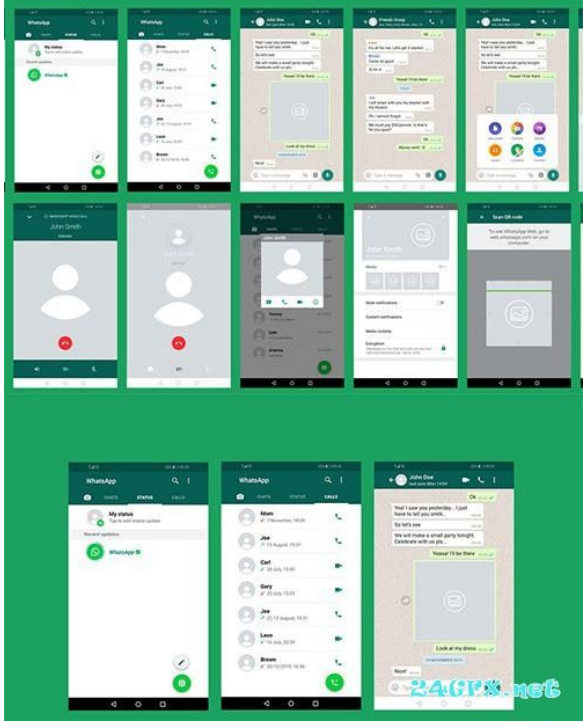
https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/

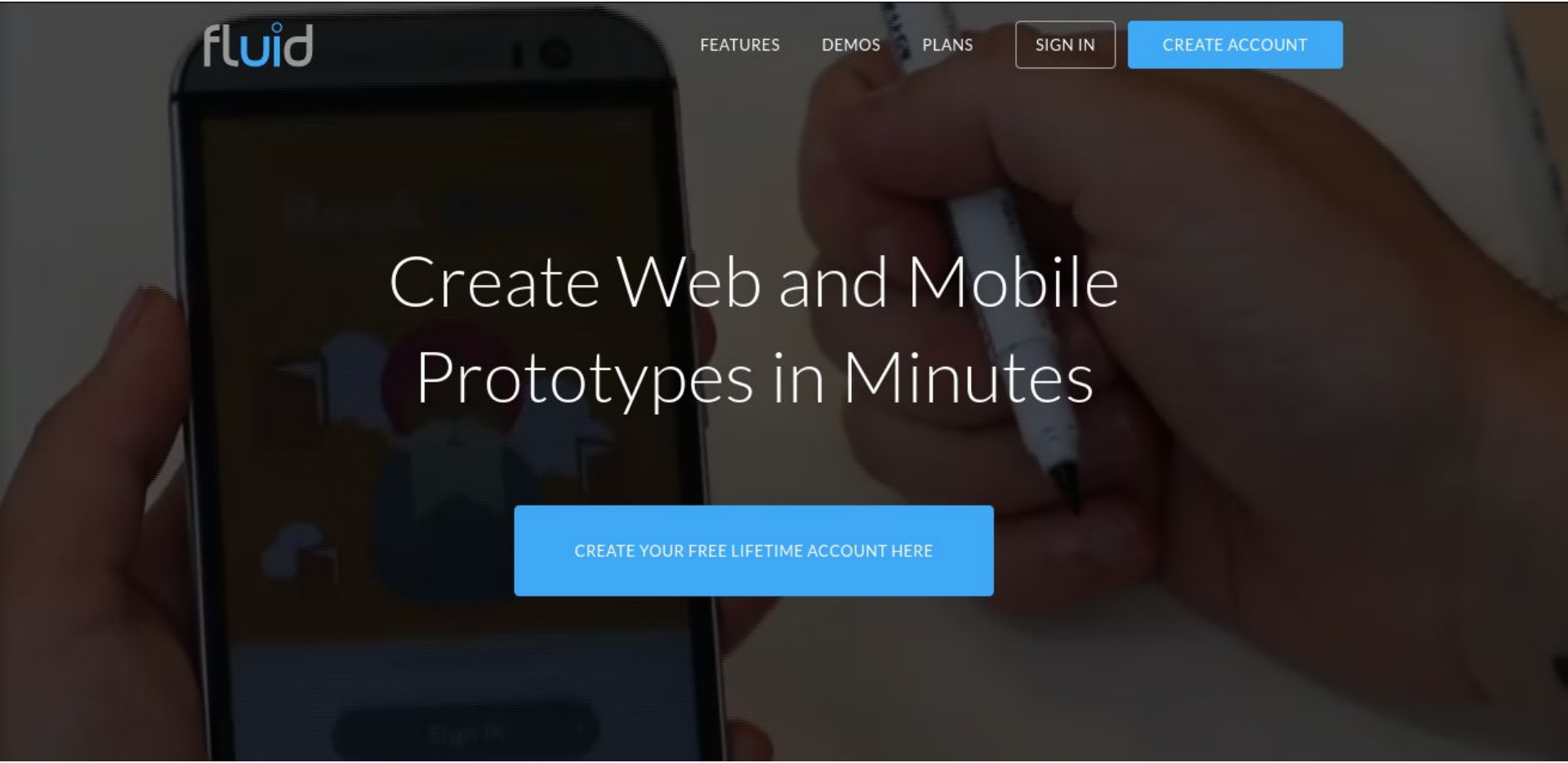
Mock-up

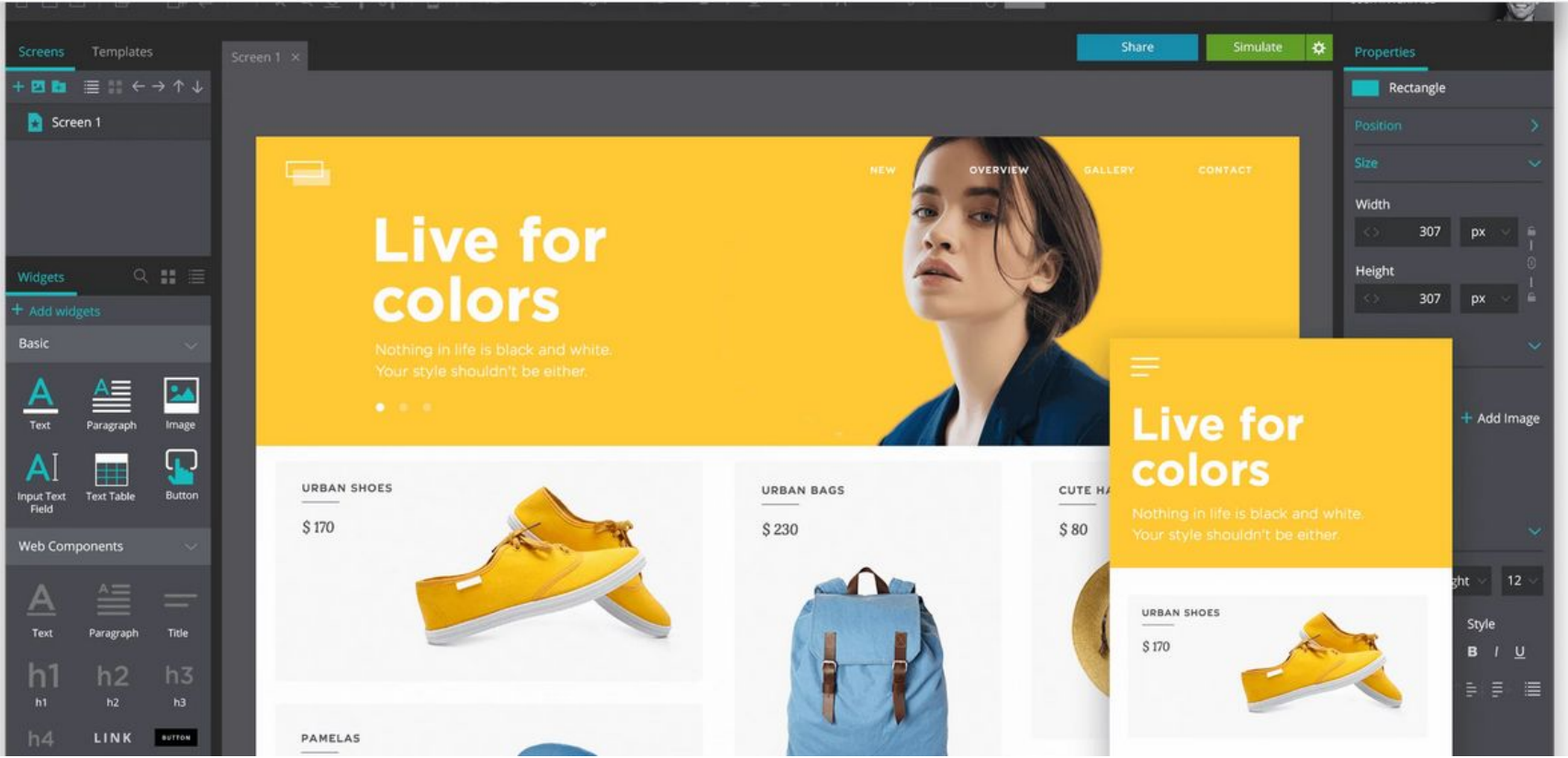
-Prototipo

-Storyboard






fluid
FEATURES
DEMOS
PLANS
SIGN IN
CREATE ACCOUNT
Create Web and Mobile Prototypes in Minutes
CREATE YOUR FREE LIFETIME ACCOUNT HERE


Screens
Templates
Screen 1
Widgets
Add widgets
Basic
Text
Paragraph
Image
Input Text Field
Text Table
Button
Web Components
Text
Paragraph
Title
h1
h2
h3
h4
LINK
BUTTON
Screen 1
Share
Simulate
NEW
OVERVIEW
GALLERY
CONTACT
Live for colors
Nothing in life is black and white.
Your style shouldn't be either.
URBAN SHOES
$ 170
URBAN BAGS
$ 230
CUTE HA
$ 80
PAMELAS
Properties
Rectangle
Position
Size
Width
307
px
Height
307
px
Add Image
Style
Live for colors
Nothing in life is black and white.
Your style shouldn't be either.
URBAN SHOES
$ 170

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/htQATVxlLng

SwiftUI

https://youtu.be/AWPiup9fE2c