Asignatura: Diseño de interfaces y principios básicos de UI/UX

Guía de aprendizaje N.º 3

20 de Julio de 2024

Managua, 20 de Julio de 2024

Diseño de interfaces y principios básicos de UI/UX

I. CONTENIDOS

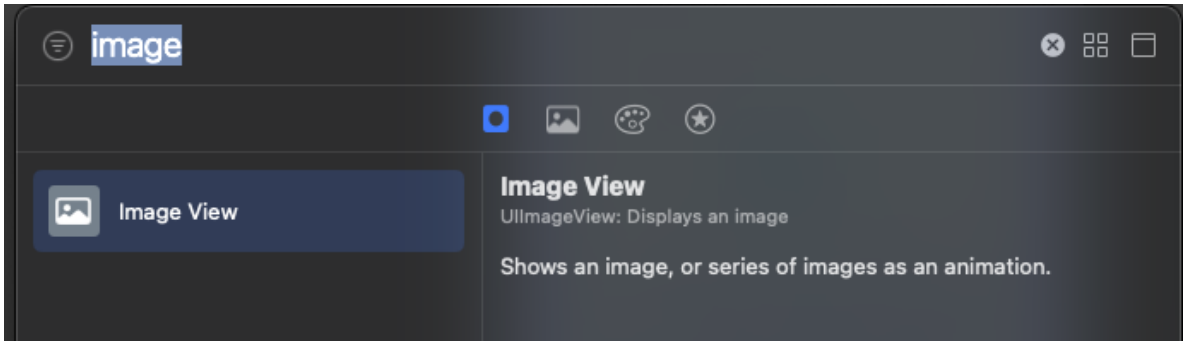
Introducción:

En este documento se creará una serie de procedimientos relacionados a los componentes visuales del toolbox de xcode (**imageView**, **stackView**, **View**), el objetivo es poder comprender las funciones básicas de los componentes más usados en las aplicaciones.

Este documento es una continuación de los componentes visuales de xcode, por lo tanto puedes ocupar el mismo proyecto que empezamos, o crear uno nuevo.

Nota: los componentes a partir de ahora se vuelven más complejos y detallistas.

Abrimos la caja de herramienta de xcode, pero en este caso vamos a ingresar un imageView.



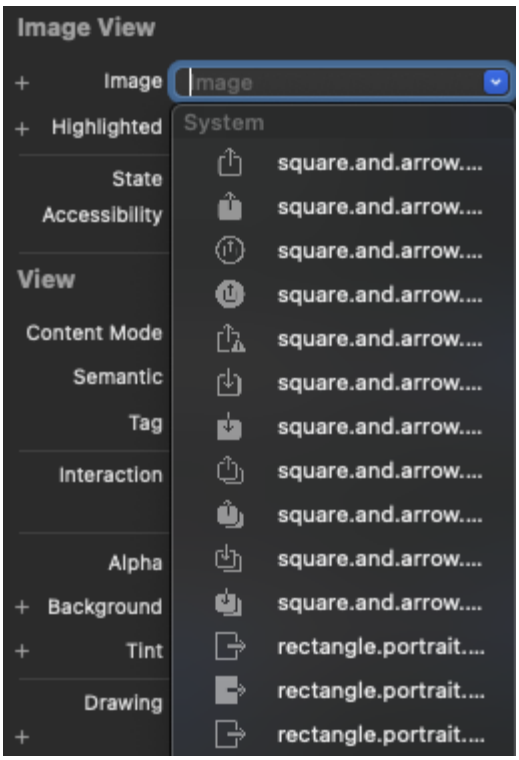
Haz clic y arrastra el Image View desde la biblioteca de objetos a tu storyboard. Colócalo en el lugar donde desees que aparezca en tu vista.

Configurar y personalizar el Image View

Una vez que hayas agregado el Image View a tu vista, puedes configurarlo y personalizarlo según tus necesidades.

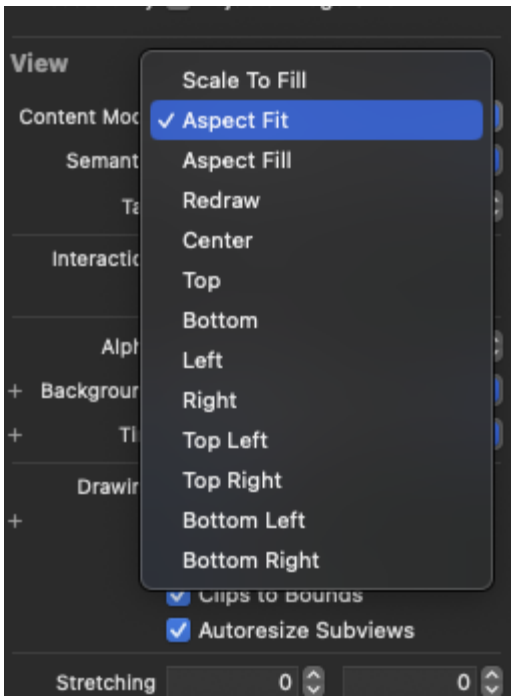
Configuración básica

1. **Seleccionar el Image View**: Haz clic en el Image View en tu storyboard para seleccionarlo.
2. **Establecer la imagen**: En el panel de atributos (Inspector de atributos) en el lado derecho, verás una opción para seleccionar una imagen. Haz clic en el menú desplegable y selecciona la imagen que deseas mostrar.



3. Ejemplo de configuración en el Inspector de atributos

- **Content Mode**: Ajusta el modo de contenido para determinar cómo se muestra la imagen dentro del Image View. Las opciones comunes son:
    - **Aspect Fit**: La imagen se ajusta dentro del Image View manteniendo su proporción.
    - **Aspect Fill**: La imagen llena el Image View, pero puede recortarse para mantener su proporción.
    - **Scale To Fill**: La imagen se escala para llenar todo el Image View, pero puede distorsionarse.



Ahora probaremos algo de código

Primero, asegúrate de que tu Image View esté conectado a tu ViewController. Para hacer esto:

1. **Crear una referencia**: Control-drag desde el Image View en el storyboard a tu ViewController para crear un IBOutlet.

```swift
import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var imageView: UIImageView!

    override func viewDidLoad() {
        super.viewDidLoad()
        // Configuración del ImageView
        setupImageView()
    }

    func setupImageView() {
        // Establecer una imagen programáticamente
        imageView.image = UIImage(named: "exampleImage")

        // Configurar el modo de contenido
        imageView.contentMode = .scaleAspectFit

        // Aplicar otras personalizaciones
        imageView.layer.cornerRadius = 10
        imageView.clipsToBounds = true
    }
}
```

Aquí hay una explicacion mas a detalle.

`imageView.image` establece la imagen que se mostrará. Asegúrate de que "exampleImage" sea el nombre de una imagen en tu directorio de Assets.

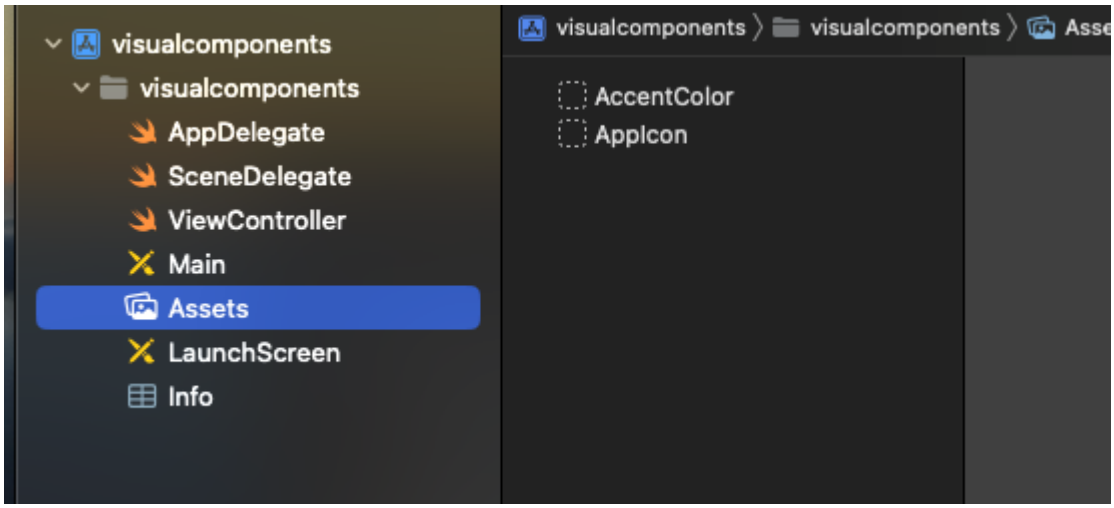
`imageView.contentMode` ajusta cómo se muestra la imagen dentro del Image View.

`imageView.layer.cornerRadius` redondea las esquinas del Image View.

`imageView.clipsToBounds` asegura que la imagen no se salga de los límites del Image View, especialmente útil cuando se aplican esquinas redondeadas.

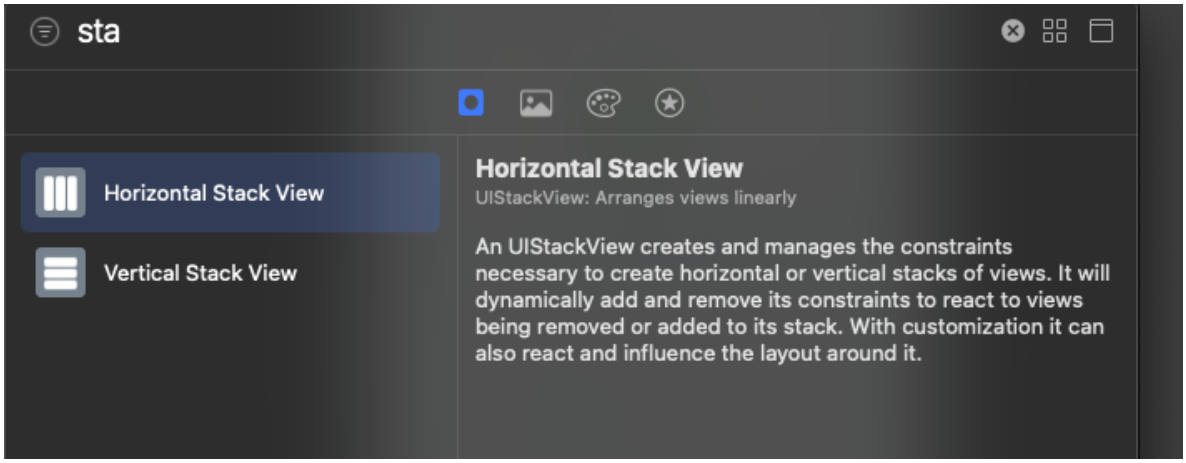
Agregar imágenes al proyecto

Para agregar imágenes a tu proyecto, arrastra y suelta los archivos de imagen en la carpeta de Assets de tu proyecto en Xcode. Asegúrate de nombrar correctamente cada imagen para poder referenciarla en tu código.



Listo ahora probemos otro componente, en este caso le toca el turno a StackView

En la barra de búsqueda de la biblioteca de objetos, escribe "Stack View". Verás las opciones para `Horizontal Stack View` y `Vertical Stack View`.

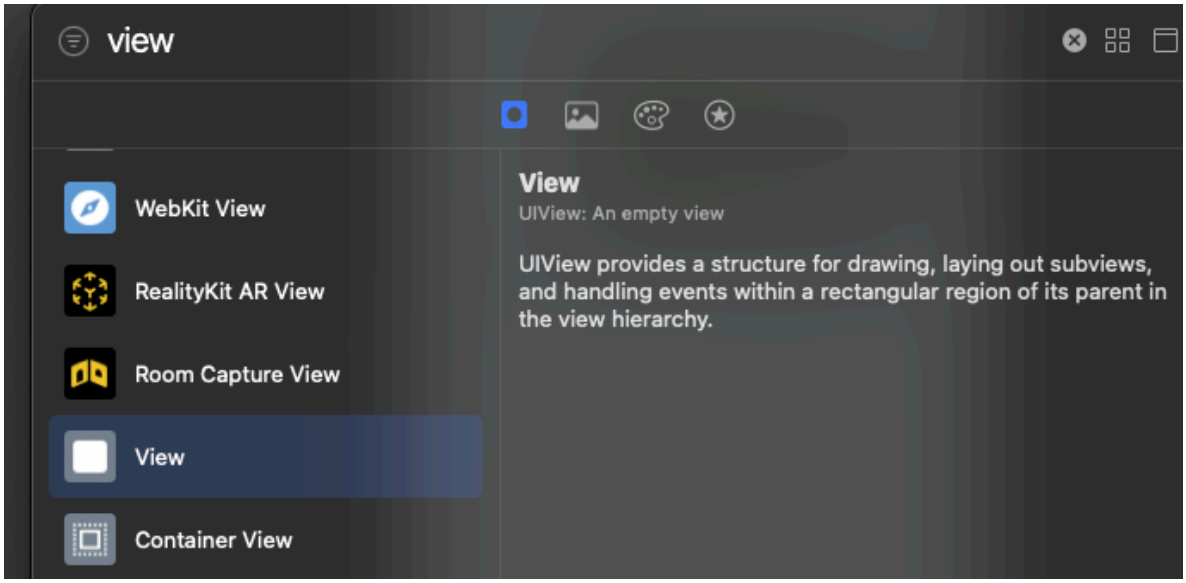


Configurar y personalizar el Stack View

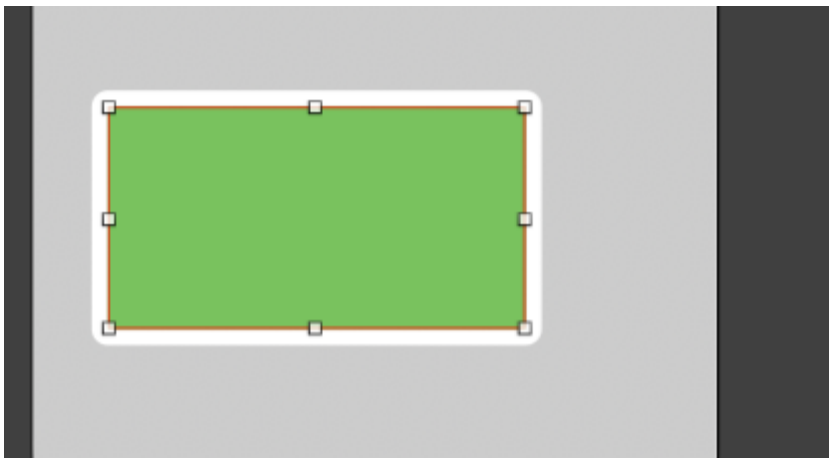
Configuración básica

1. **Seleccionar el Stack View**: Haz clic en el Stack View en tu storyboard para seleccionarlo.
2. **Agregar elementos al Stack View**: Puedes arrastrar y soltar otros componentes (como `UILabel`, `UIImageView`, `UIButton`, etc.) dentro del Stack View para que se alineen de forma automática según la orientación del Stack View.

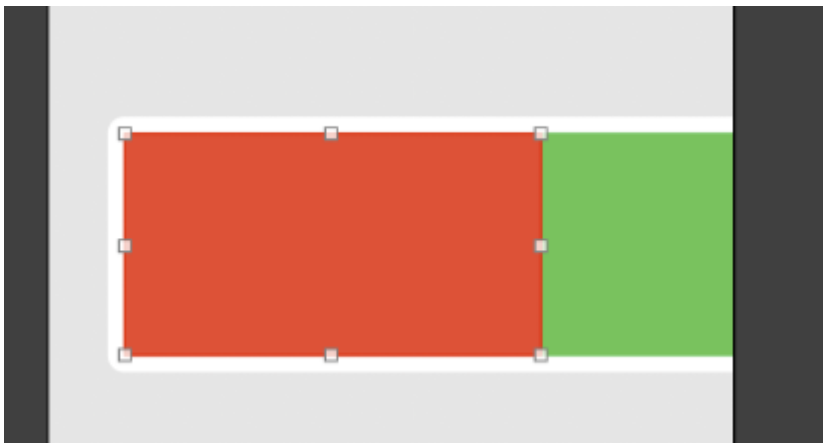
Para este ejemplo voy a cambiar el background para poder distinguir mi stackview, y lo que voy a hacer es agregar el componente View dentro de mi stackView, teniendo en cuenta que cada View debo cambiar el color para poder diferenciarlo.



Arrastramos y soltamos dentro de nuestro stackView en este caso el stack es horizontal. a mi primer view le di un color de verde.



Ahora ingresará otro view dentro de mi stackView, y le daré un color de rojo, quedando de la siguiente manera.



Propiedades importantes del Stack View

- **Axis**: Determina si el Stack View es horizontal (`.horizontal`) o vertical (`.vertical`).
- **Alignment**: Establece cómo se alinean los elementos a lo largo del eje perpendicular. Las opciones comunes incluyen `.fill`, `.leading`, `.center`, `.trailing` para Stack Views horizontales, y `.fill`, `.top`, `.center`, `.bottom` para Stack Views verticales.
- **Distribution**: Determina cómo se distribuyen los elementos a lo largo del eje principal. Las opciones comunes son `.fill`, `.fillEqually`, `.fillProportionally`, `.equalSpacing`, y `.equalCentering`.
- **Spacing**: Define el espacio entre los elementos en el Stack View.

Ejemplo de configuración mediante código

Primero, crea una referencia al Stack View en tu `ViewController`. Luego, configura sus propiedades y agrega subviews en el código.

```
class ViewController: UIViewController {

    @IBOutlet weak var horizontalStackView: UIStackView!
    @IBOutlet weak var verticalStackView: UIStackView!

    override func viewDidLoad() {
        super.viewDidLoad()
        // Configuración del Horizontal Stack View
        setupHorizontalStackView()
        // Configuración del Vertical Stack View
        setupVerticalStackView()
    }

    func setupHorizontalStackView() {
        horizontalStackView.axis = .horizontal
        horizontalStackView.alignment = .fill
        horizontalStackView.distribution = .fillEqually
        horizontalStackView.spacing = 10

        // Añadir elementos al Horizontal Stack View
        let label1 = UILabel()
        label1.text = "Label 1"
        label1.backgroundColor = .red

        let label2 = UILabel()
        label2.text = "Label 2"
        label2.backgroundColor = .green

        let label3 = UILabel()
        label3.text = "Label 3"
        label3.backgroundColor = .blue

        horizontalStackView.addArrangedSubview(label1)
        horizontalStackView.addArrangedSubview(label2)
        horizontalStackView.addArrangedSubview(label3)
    }
```

```swift
func setupVerticalStackView() {
    verticalStackView.axis = .vertical
    verticalStackView.alignment = .fill
    verticalStackView.distribution = .equalSpacing
    verticalStackView.spacing = 10

    // Añadir elementos al Vertical Stack View
    let button1 = UIButton()
    button1.setTitle("Button 1", for: .normal)
    button1.backgroundColor = .purple

    let button2 = UIButton()
    button2.setTitle("Button 2", for: .normal)
    button2.backgroundColor = .orange

    let button3 = UIButton()
    button3.setTitle("Button 3", for: .normal)
    button3.backgroundColor = .cyan

    verticalStackView.addArrangedSubview(button1)
    verticalStackView.addArrangedSubview(button2)
    verticalStackView.addArrangedSubview(button3)
}
```

- `horizontalStackView.axis = .horizontal` establece la orientación del Stack View como horizontal.
- `horizontalStackView.alignment = .fill` establece la alineación de los elementos.
- `horizontalStackView.distribution = .fillEqually` distribuye los elementos de manera equitativa.
- `horizontalStackView.spacing = 10` define un espacio de 10 puntos entre los elementos.

Lo mismo se aplica para el Stack View vertical, con propiedades ajustadas según la necesidad.

Ejercicios

Descripción: Los estudiantes realizarán una serie de ejercicios prácticos que cubrirán todos los temas tratados en la clase. Los ejercicios permitirán a los estudiantes poner en práctica lo aprendido y recibir retroalimentación inmediata sobre su comprensión.

II. LOGROS DE APRENDIZAJES

Al finalizar el encuentro, los estudiantes serán capaces de:

- Conocer los componentes visuales del toolbox de xcode view, imageView, StackView.
- Poder implementar componentes visuales, detallando colores y funcionamiento.
- Identificar la caja de herramientas, storyboard y menú de implementación

III. ORIENTACIONES METODOLÓGICAS

Para alcanzar los logros de aprendizaje, luego de la evaluación trabajaremos con las siguientes estrategias y recursos:

⏩ **Actividad de Aprendizaje No.3**

Modalidad: Individual
Tiempo aproximado: 20 minutos
Periodo de realización: 20 de Julio, en el aula de clases
Puntaje: 20 pts
Descripción de la actividad: Es momento de poner en práctica lo aprendido, en este ejercicio, lo que quiero que hagas, es un nuevo proyecto y añadas una imagen al centro de la interfaz, hazla circular y abajo con un label ingresa alguna información de tu interés.