Lección: Comprendiendo la Estructura del Widget Scaffold en Flutter

Objetivos de la Lección

- Comprender la importancia del widget **Scaffold** como la estructura base para una aplicación en Flutter.
- Identificar los componentes principales que pueden utilizarse dentro del widget **Scaffold**.
- Implementar una aplicación básica utilizando **Scaffold** y personalizar los elementos de la interfaz de usuario, como la **AppBar**, el **body**, el **FloatingActionButton**, y otros widgets.

¿Qué es el Widget Scaffold?

En Flutter, el widget **Scaffold** es una estructura visual que sirve como el **esqueleto básico** o contenedor principal de la interfaz de usuario en una aplicación. Este widget proporciona una **estructura estándar** que incluye elementos comunes en muchas aplicaciones, como una **barra de aplicación (AppBar)**, un **cuerpo (body)** para el contenido principal, un **botón de acción flotante (FloatingActionButton)**, un **drawer** (menú lateral), entre otros.

El **Scaffold** es el widget recomendado para crear aplicaciones que siguen las pautas de diseño de **Material Design**, ya que incluye todo lo necesario para construir una **interfaz coherente** y **funcional** con características modernas.

Componentes Principales del Scaffold

El widget **Scaffold** proporciona varios elementos que se pueden personalizar para construir la estructura principal de la interfaz de la aplicación. A continuación, se describen los componentes más importantes:

1. **AppBar**: Representa la **barra de aplicación** que aparece en la parte superior de la pantalla. Puede incluir un título, iconos de acciones y menús.
2. **Body**: Es el contenido principal de la pantalla, donde se muestran los widgets que componen la interfaz de usuario. Normalmente, se utiliza un `Column`, `Row` o `ListView` para organizar el contenido en el cuerpo.
3. **FloatingActionButton**: Un botón de acción flotante, común en aplicaciones de **Material Design**, que generalmente se utiliza para realizar una acción destacada, como agregar un elemento o abrir un formulario.
4. **Drawer**: Un menú lateral que se puede deslizar desde el borde de la pantalla. Es útil para mostrar opciones de navegación.
5. **BottomNavigationBar**: Una barra de navegación inferior que permite cambiar entre diferentes pantallas o secciones de la aplicación.
6. **SnackBar**: Un mensaje breve que aparece en la parte inferior de la pantalla para mostrar notificaciones o alertas temporales.

Estructura Básica del Widget Scaffold

La estructura del widget **Scaffold** sigue una jerarquía simple. En su forma más básica, un **Scaffold** incluye una **AppBar**en la parte superior y un **body** para el contenido principal. A continuación se muestra un ejemplo básico.

Ejemplo Básico de un Scaffold:

```
import 'package:flutter/material.dart';


void main() {
  runApp(MyApp());
}
```

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Aplicación con Scaffold'),
        ),
        body: Center(
          child: Text('¡Bienvenidos a Flutter!'),
        ),
        floatingActionButton: FloatingActionButton(
          onPressed: () {
            // Acción cuando se presiona el botón
          },
          child: Icon(Icons.add),
        ),
      ),
    );
  }
}
```

Explicación del Código:

1. **MaterialApp**: El widget `MaterialApp` envuelve la aplicación y proporciona temas y navegación.
2. **Scaffold**: Proporciona la estructura base de la pantalla, con varios elementos integrados.

- **AppBar**: Muestra una barra de aplicación con el título "`Aplicación con Scaffold`".
- **Body**: Contiene un widget `Center` que alinea el widget `Text` en el centro de la pantalla.
- **FloatingActionButton**: Un botón de acción flotante con un ícono de "agregar" (`Icons.add`).

Captura de la pantalla del ejemplo presentado con la Estructura Básica de un Scaffold. Creado por Javier A. Dastas (2024)

Personalización de los Componentes del Scaffold

1. AppBar: Personalización y Elementos

El widget **AppBar** permite personalizar la barra de aplicación con elementos como un título, acciones (íconos), y un menú lateral. A continuación, un ejemplo que muestra cómo personalizar la barra de aplicación:

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Mi Aplicación'),
          backgroundColor: Colors.green, // Cambiar el color de
fondo
          actions: <Widget>[
            IconButton(
              icon: Icon(Icons.search),
              onPressed: () {
                // Acción para el ícono de búsqueda
              },
```

```
            ),
            IconButton(
              icon: Icon(Icons.more_vert),
              onPressed: () {
                // Acción para el ícono de menú
              },
            ),
          ],
        ),
        body: Center(
          child: Text('¡Contenido en el cuerpo de la
aplicación!'),
        ),
      ),
    );
  }
}
```

Explicación del Código:

- **backgroundColor**: Cambia el color de fondo de la barra de aplicación.
- **actions**: Permite añadir íconos a la derecha de la barra de aplicación. En este ejemplo, se agregan un ícono de **búsqueda** y un ícono de **menú**.

Captura de la pantalla del ejemplo presentado del Widget Scaffold con elementos personalizados. Creado por Javier A. Dastas (2024)

2. Body: Estructura y Organización del Contenido

El `body` es el área principal donde se coloca el contenido de la aplicación. Puedes organizar este contenido utilizando widgets como `Column`, `Row` o `ListView`.

Ejemplo con un Column en el Body:

```
import 'package:flutter/material.dart';


void main() {
  runApp(MyApp());
}
```

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Estructura del Body'),
        ),
        body: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text('Primera Línea'),
            SizedBox(height: 20), // Espacio entre los elementos
            Text('Segunda Línea'),
            SizedBox(height: 20),
            Text('Tercera Línea'),
          ],
        ),
      ),
    );
  }
}
```

Explicación del Código:

- **Column**: Organiza los widgets en una columna vertical, más adelante en presentaremos una lección relacionada a este Widget.

- **mainAxisAlignment**: Centra los elementos en el eje principal (vertical).
- **SizedBox**: Añade espacio vertical entre las líneas de texto.

3. FloatingActionButton: Botón de Acción Flotante

El **FloatingActionButton** es un botón que flota sobre el contenido principal de la aplicación y generalmente se utiliza para ejecutar la acción principal de la pantalla.

Ejemplo de FloatingActionButton con Acción:

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Botón de Acción Flotante'),
        ),
        body: Center(
          child: Text('Presiona el botón flotante'),
        ),
        floatingActionButton: FloatingActionButton(
          onPressed: () {
            // Acción que se ejecuta al presionar el botón
```

```
            print('Botón presionado');
          },
          child: Icon(Icons.thumb_up),
          backgroundColor: Colors.blue,
        ),
      ),
    );
  }
}
```

Explicación del Código:

- **onPressed**: Define la acción que se ejecuta cuando se presiona el botón.
- **Icon**: El ícono del botón es un pulgar arriba (`Icons.thumb_up`).
- **backgroundColor**: Cambia el color de fondo del botón a azul.

Captura de la pantalla del ejemplo presentado con Scaffold y un Widget de Botón. Creado por Javier A. Dastas (2024)

Otros Componentes del Scaffold

4. Drawer: Menú Lateral

El **Drawer** es un menú lateral que se desliza desde el borde de la pantalla, útil para proporcionar opciones de navegación adicionales.

Ejemplo de Scaffold con Drawer:

```
import 'package:flutter/material.dart';
```

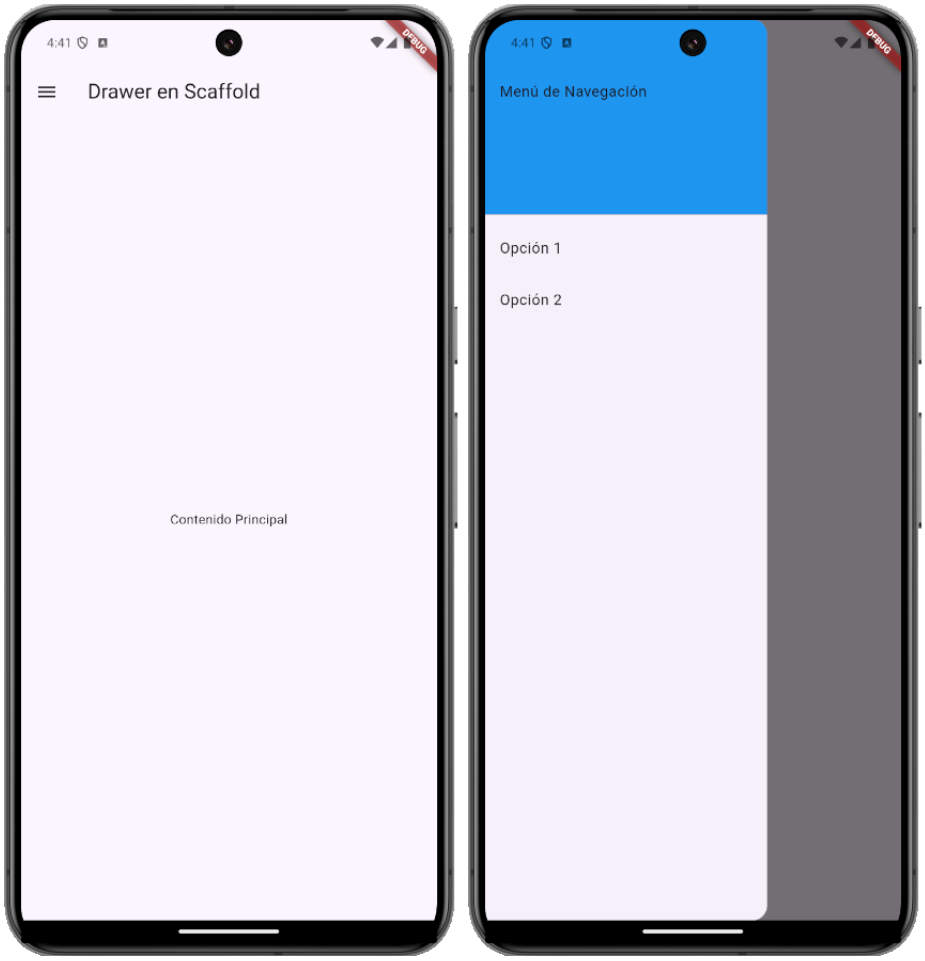
```
void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Drawer en Scaffold'),
        ),
        body: Center(
          child: Text('Contenido Principal'),
        ),
        drawer: Drawer(
          child: ListView(
            padding: EdgeInsets.zero,
            children: <Widget>[
              DrawerHeader(
                child: Text('Menú de Navegación'),
                decoration: BoxDecoration(
                  color: Colors.blue,
                ),
              ),
              ListTile(
                title: Text('Opción 1'),
```

```
                onTap: () {
                  // Acción al seleccionar la Opción 1
                },
              ),
              ListTile(
                title: Text('Opción 2'),
                onTap: () {
                  // Acción al seleccionar la Opción 2
                },
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

Explicación del Código:

- **Drawer**: Implementa un menú lateral.
- **DrawerHeader**: Encabezado del menú, en este caso con un fondo azul y texto.
- **ListTile**: Crea opciones de navegación en el menú.



Captura de la pantalla del ejemplo presentado con Scaffold y el Widget Drawer.
Creado por Javier A. Dastas (2024)

Resumen de la Lección

En esta lección, hemos aprendido que el widget **Scaffold** proporciona la **estructura principal** para construir aplicaciones en Flutter que siguen el diseño **Material Design**. Hemos visto los componentes clave del **Scaffold**, como la **AppBar**, el **body**, el **FloatingActionButton**, y el **Drawer**, que permiten crear aplicaciones funcionales y bien estructuradas.

El **Scaffold** es fundamental en Flutter porque facilita la creación de interfaces coherentes y bien organizadas, proporcionando una base sólida para agregar elementos adicionales a medida que se construyen aplicaciones más complejas.

Actividad de la Lección

Esta actividad te ayudará a aplicar los conceptos aprendidos sobre el widget **Scaffold** y a familiarizarse con la creación de aplicaciones estructuradas y funcionales en Flutter.

Instrucciones:

1. Crea una aplicación en Flutter utilizando **Scaffold** que incluya una **AppBar** con un menú de acciones, un **FloatingActionButton**, y un **Drawer** con al menos tres opciones de navegación.
2. Personaliza la **AppBar** para que tenga un color de fondo diferente y añade un **SnackBar** que se muestre cuando se presiona el **FloatingActionButton**.
3. Entrega un documento en formato PDF con copia de tu código y copia de imágenes o capturas de pantalla demostrando que tu código funciona.