Módulo 4: Elementos Visuales y Flujo de Interacción entre Pantallas

Lección: Uso de Strings y Clases con el widget `ListTile` en Flutter

Objetivos de la Lección

- Comprender cómo utilizar listas de tipo `String` y listas de tipo Clase como fuente de datos para el widget `ListTile`.
- Aprender a crear y mostrar listas de datos en una aplicación Flutter usando `ListTile`.
- Explorar un ejemplo de lista de Clase que presenta información en una estructura visual no tradicional.

Introducción de la Lección

En Flutter, el uso de listas como fuente de datos es una técnica común para presentar múltiples elementos en una interfaz de usuario. Las listas pueden contener datos simples, como `String`, o datos más complejos organizados en Clases. Al utilizar el widget `ListTile`, podemos mostrar estos elementos en listas organizadas y personalizadas. En esta lección, explicaremos cómo utilizar listas de `String` y listas de Clase en `ListTile`, así como técnicas de personalización para mostrar datos de manera no convencional.

Uso de Listas de `String` como Fuente de Datos

Las listas de `String` son adecuadas para mostrar datos simples en una lista, como nombres, títulos o etiquetas. En el siguiente ejemplo, implementaremos una lista de `String` que se muestra en `ListTile`.

Ejemplo de Código con Listas de `String`:

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: StringListExample(),
    );
  }
}

class StringListExample extends StatelessWidget {
  final List<String> items = ['Manzanas', 'Naranjas', 'Bananas',
'Peras'];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
```

```
      appBar: AppBar(
        title: Text('Lista de Frutas'),
      ),
      body: ListView.builder(
        itemCount: items.length,
        itemBuilder: (context, index) {
          return ListTile(
            title: Text(items[index]),
          );
        },
      ),
    );
  }
}
```

En este ejemplo:

- Se utiliza una lista de `String` llamada `items`, que contiene los nombres de las frutas.
- `ListView.builder` recorre la lista y crea un `ListTile` para cada fruta, mostrando su nombre en el título (`title`).

Imagen de la pantalla del código de ejemplo anterior
utilizando el widget ListTile con Listas de String.
Creado por Javier A. Dastas (2024)

Uso de Listas de Clase como Fuente de Datos

Cuando se necesita mostrar datos más complejos, una lista de objetos de Clase es una opción adecuada. Las Clases permiten organizar varios atributos en una sola estructura, proporcionando flexibilidad para personalizar la apariencia del `ListTile`.

Ejemplo de Código con Listas de Clase y Estructura No Tradicional:

En este ejemplo, crearemos una clase `Producto` con varios atributos y personalizamos el `ListTile` para mostrar la información de forma no tradicional.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: ProductListExample(),
    );
  }
}

class Producto {
  final String nombre;
  final double precio;
  final int cantidad;
  final bool enStock;

  Producto({
    required this.nombre,
    required this.precio,
    required this.cantidad,
    required this.enStock,
  });
}

class ProductListExample extends StatelessWidget {
  final List<Producto> productos = [
```

```
    Producto(nombre: 'Laptop', precio: 1200.00, cantidad: 5,
enStock: true),
    Producto(nombre: 'Tablet', precio: 350.00, cantidad: 0,
enStock: false),
    Producto(nombre: 'Celular', precio: 800.00, cantidad: 3,
enStock: true),
    Producto(nombre: 'Monitor', precio: 220.00, cantidad: 2,
enStock: true),
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Lista de Productos'),
      ),
      body: ListView.builder(
        itemCount: productos.length,
        itemBuilder: (context, index) {
          return ListTile(
            leading: Icon(
              productos[index].enStock ? Icons.check_circle :
Icons.cancel,
              color: productos[index].enStock ? Colors.green :
Colors.red,
            ),
            title: Text(productos[index].nombre),
            subtitle: Text('Precio:
\$${productos[index].precio.toStringAsFixed(2)}'),
            trailing: Column(
```

```
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Text('Cant: ${productos[index].cantidad}'),
                productos[index].enStock
                    ? Text('En stock', style: TextStyle(color:
Colors.green))
                    : Text('Sin stock', style: TextStyle(color:
Colors.red)),
              ],
            ),
            onTap: () {
              print("Seleccionaste el producto:
${productos[index].nombre}");
            },
          );
        },
      ),
    );
  }
}
```

En este ejemplo:

- Creamos una clase `Producto` que incluye atributos como `nombre`, `precio`, `cantidad` y `enStock`.
- La propiedad `leading` utiliza un ícono que cambia según el valor de `enStock`, verde para productos disponibles y rojo para los que están fuera de stock.
- **`trailing`** muestra una columna con la cantidad y el estado de inventario del producto, personalizando la apariencia para que no sea solo un subtítulo estándar.

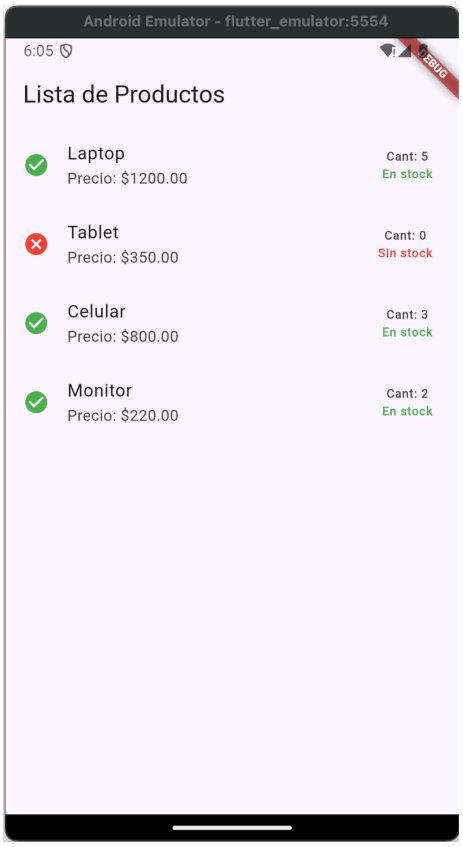

Imagen de la pantalla del código de ejemplo anterior
utilizando el widget ListTile con Listas de estructuras de Clase.
Creado por Javier A. Dastas (2024)

Relación con Otros Conceptos

El uso de listas como fuente de datos para `ListTile` permite crear interfaces organizadas que muestran información estructurada, desde elementos simples hasta datos complejos. La integración de clases y listas en el desarrollo de aplicaciones móviles en Flutter permite manejar datos en estructuras eficientes, importantes en aplicaciones de e-commerce, inventarios o sistemas de gestión.

Resumen de la Lección

En esta lección, exploramos cómo utilizar listas de tipo `String` y listas de Clase como fuente de datos para el widget `ListTile`. Aprendimos cómo personalizar los

elementos visuales para mostrar datos complejos y cómo estructurar la lista de manera no tradicional para incluir múltiples atributos. Estas técnicas son clave en el desarrollo de aplicaciones interactivas y centradas en el usuario.

Actividad de la Lección

En esta actividad, aplicaremos lo aprendido para crear listas que presenten elementos simples y complejos con `ListTile`, permitiendo mostrar datos y opciones en la interfaz.

La actividad tiene dos partes:

1. **Lista de Elementos Simples con `String`**:
    - Implementa una lista de `ListTile` con nombres de ciudades.
    - Cada `ListTile` debe mostrar el nombre de una ciudad y reaccionar al toque mostrando un mensaje en la consola.
2. **Lista de Elementos Complejos con Clase Personalizada**:
    - Crea una clase `Evento` con atributos como `nombre`, `ubicación`, `fecha` y `estado`.
    - Muestra cada `Evento` en un `ListTile`, con un ícono que indique si el evento es "Activo" o "Cancelado".
    - Personaliza el `trailing` para mostrar tanto la fecha como la ubicación del evento, con un color distintivo para cada estado.

Entrega de la Actividad:

- Desarrolla un documento en formato PDF que incluya el código, capturas de pantalla de la aplicación en ejecución y una breve explicación de cómo el uso de listas y `ListTile` permite mostrar datos estructurados en Flutter.