Módulo 4: Elementos Visuales y Flujo de Interacción entre Pantallas

Lección: Introducción al uso de `TextField` y `Autocomplete` en Flutter

Objetivos de la Lección

- Comprender el propósito del widget `TextField` para entrada de texto y conocer sus propiedades principales.
- Aprender a validar datos y personalizar la apariencia de `TextField`.
- Explorar el uso del widget `Autocomplete` para ofrecer sugerencias de entrada de texto dinámicas.
- Implementar ejemplos básicos en Flutter para cada widget y aplicar sus propiedades principales.

Introducción de la Lección

En Flutter, los widgets `TextField` y `Autocomplete` son esenciales para la interacción de los usuarios con la aplicación a través de la entrada de datos. `TextField` permite que los usuarios ingresen texto, mientras que `Autocomplete` sugiere opciones a medida que el usuario escribe, mejorando la experiencia de entrada de datos. Esta lección explica cómo usar estos widgets en Flutter, cubriendo sus propiedades y funcionalidades más importantes, así como ejemplos básicos de implementación.

`TextField`: Entrada de Texto

El widget `TextField` permite que los usuarios ingresen texto en la aplicación. Este widget es personalizable y admite múltiples configuraciones, lo que lo convierte en una herramienta flexible para la entrada de datos.

Propiedades Principales de `TextField`:

- **controller**: Gestiona el texto ingresado en el `TextField`.
- **decoration**: Personaliza el estilo del campo, incluyendo etiquetas y bordes.
- **keyboardType**: Define el tipo de teclado que aparecerá (e.g., numérico, texto).
- **obscureText**: Oculta el texto ingresado, útil para campos de contraseñas.
- **onChanged**: Callback que se ejecuta cuando el usuario cambia el texto.
- **validator**: Ayuda a validar el texto ingresado y es útil cuando se usa junto a un `Form`.

Ejemplo Simple de `TextField`:

En este ejemplo, configuraremos un campo de texto simple que permite personalizar su apariencia y usar un controlador para recuperar el texto ingresado.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
```

```
      home: TextFieldExample(),
    );
  }
}

class TextFieldExample extends StatelessWidget {
  final TextEditingController _controller =
TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Ejemplo de TextField'),
      ),
      body: Padding(
        padding: EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(
              controller: _controller,
              decoration: InputDecoration(
                labelText: 'Nombre',
                hintText: 'Ingresa tu nombre',
                border: OutlineInputBorder(),
              ),
              onChanged: (text) {
                print("Texto ingresado: $text");
              },
            ),
```

```
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                print("Texto final: ${_controller.text}");
              },
              child: Text("Enviar"),
            ),
          ],
        ),
      ),
    );
  }
}
```

En este ejemplo:

- `controller` nos permite obtener el texto ingresado al presionar el botón "Enviar".
- `decoration` configura el estilo del campo de texto con un borde y una etiqueta de texto.
- `onChanged` imprime el texto actual en la consola cada vez que el usuario lo modifica.

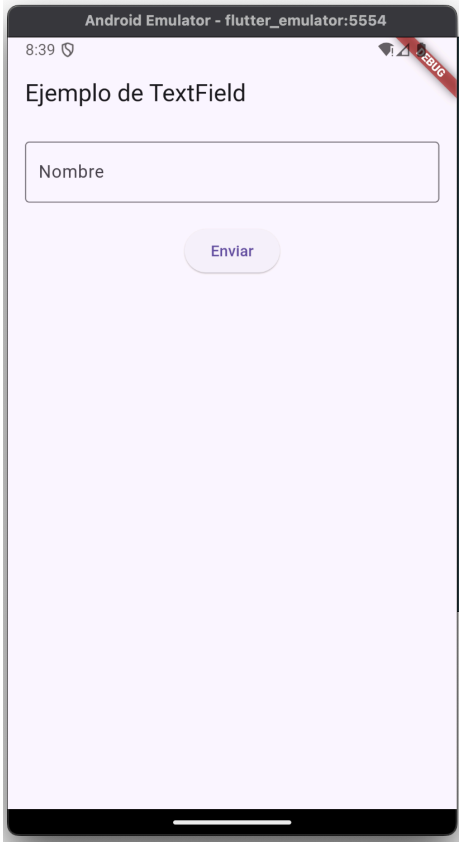

Imagen de la pantalla que incluye el Widget TextField.

Creado por Javier A. Dastas (2024)

Autocomplete: Sugerencias de Entrada de Texto

El widget `Autocomplete` facilita la entrada de datos al usuario al sugerir opciones basadas en el texto que escribe. Esto es útil para campos que requieren selección de opciones específicas, como nombres de ciudades, colores, etc.

Propiedades Principales de `Autocomplete`:

- **optionsBuilder**: Proporciona una lista de sugerencias en función del texto ingresado.
- **onSelected**: Callback que se ejecuta cuando el usuario selecciona una sugerencia.
- **fieldViewBuilder**: Personaliza el campo de entrada (opcional).

- **optionsViewBuilder**: Permite personalizar la lista de sugerencias (opcional).

Ejemplo Simple de `Autocomplete`:

A continuación, se muestra un ejemplo simple que sugiere nombres de colores a medida que el usuario escribe.

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

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

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

class AutocompleteExample extends StatelessWidget {
  final List<String> _colorOptions = ['Rojo', 'Azul', 'Verde',
'Amarillo', 'Naranja', 'Violeta'];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Ejemplo de Autocomplete'),
```

```
      ),
      body: Padding(
        padding: EdgeInsets.all(16.0),
        child: Autocomplete<String>(
          optionsBuilder: (TextEditingValue textEditingValue) {
            if (textEditingValue.text.isEmpty) {
              return const Iterable<String>.empty();
            }
            return _colorOptions.where((String option) {
              return
option.toLowerCase().contains(textEditingValue.text.toLowerCase(
));
            });
          },
          onSelected: (String selection) {
            print('Color seleccionado: $selection');
          },
          fieldViewBuilder: (BuildContext context,
TextEditingController textEditingController, FocusNode
focusNode, VoidCallback onFieldSubmitted) {
            return TextField(
              controller: textEditingController,
              focusNode: focusNode,
              decoration: InputDecoration(
                labelText: 'Color favorito',
                hintText: 'Escribe un color',
                border: OutlineInputBorder(),
              ),
            );
          },
```

```
          ),
        ),
      );
    }
}
```

En este ejemplo:

- `optionsBuilder` define las sugerencias en función del texto ingresado. Aquí, se filtra la lista de colores según el texto que el usuario escribe.
- `onSelected` muestra el color seleccionado en la consola.
- `fieldViewBuilder` personaliza el campo de entrada del `Autocomplete`.

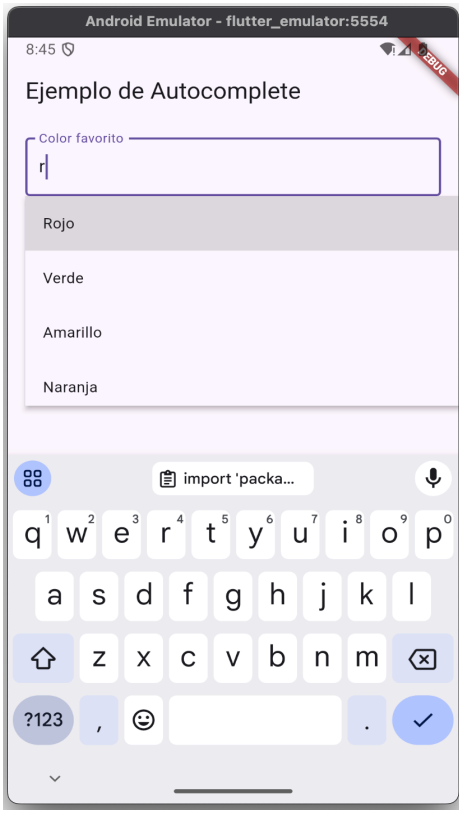

Imagen de la pantalla que incluye el Widget Autocomplete.

Creado por Javier A. Dastas (2024)

Relación con Otros Conceptos

Los widgets `TextField` y `Autocomplete` son esenciales en la creación de formularios en Flutter, donde ambos widgets pueden usarse juntos para mejorar la experiencia de entrada de datos. La integración de `TextField` con validaciones también es clave en aplicaciones que requieren recolección de información precisa del usuario.

`TextField` y `Autocomplete` son herramientas esenciales en Flutter para mejorar la experiencia de usuario (UX) y la interfaz de usuario (UI). Al permitir entradas de texto personalizadas y sugerencias en tiempo real, estos widgets contribuyen a un diseño UX/UI intuitivo, donde el usuario puede completar formularios de forma rápida y precisa. En diseño UX/UI, la facilidad de uso y la interacción fluida son clave para retener a los usuarios, y el uso de estos widgets facilita justamente eso.

Para la Ingeniería de Software, aprender a manejar y personalizar estos widgets es fundamental, ya que los ingenieros a menudo deben crear aplicaciones centradas en el usuario que incluyan validaciones robustas y funcionalidades de autocompletado, elementos que ayudan a prevenir errores de entrada y optimizan la experiencia de interacción con la aplicación.

Resumen de la Lección

En esta lección, exploramos los widgets `TextField` y `Autocomplete` en Flutter. Aprendimos a configurar un campo de entrada de texto con el widget `TextField`, incluyendo propiedades como `controller`, `decoration` y `onChanged`. También revisamos el widget `Autocomplete`, que permite sugerir opciones al usuario mientras escribe, con propiedades como `optionsBuilder` y `onSelected`.

Actividad de la Lección

En esta actividad, aplicaremos lo aprendido sobre los widgets `TextField` y `Autocomplete` en Flutter para crear una aplicación móvil que provea una experiencia de entrada de datos personalizada y dinámica. Al combinar validación en `TextField` con sugerencias en `Autocomplete`, los estudiantes podrán desarrollar campos de entrada más interactivos y amigables para el usuario.

La actividad tiene dos partes:

1. **Personalización y Validación con `TextField`**: Crear un campo que valide la entrada de una dirección de correo electrónico, mostrando un mensaje de error si no cumple con el formato adecuado. Esta parte ayudará a reforzar la capacidad de validar datos y personalizar el comportamiento de `TextField`.
2. **Sugerencias Dinámicas con `Autocomplete`**: Configurar un campo que sugiera nombres de frutas mientras el usuario escribe, aprovechando el widget `Autocomplete` para ofrecer opciones en tiempo real.

Entrega de la Actividad:

- Desarrolla un documento en formato PDF que incluya el código desarrollado para esta actividad, capturas de pantalla de la aplicación en ejecución, y una breve descripción de cómo cada widget ayuda a mejorar la experiencia de usuario en la selección de opciones.

Con esta actividad, se busca consolidar el conocimiento en la creación de interfaces de entrada que combinen facilidad de uso y control de calidad en los datos, mejorando la experiencia del usuario.