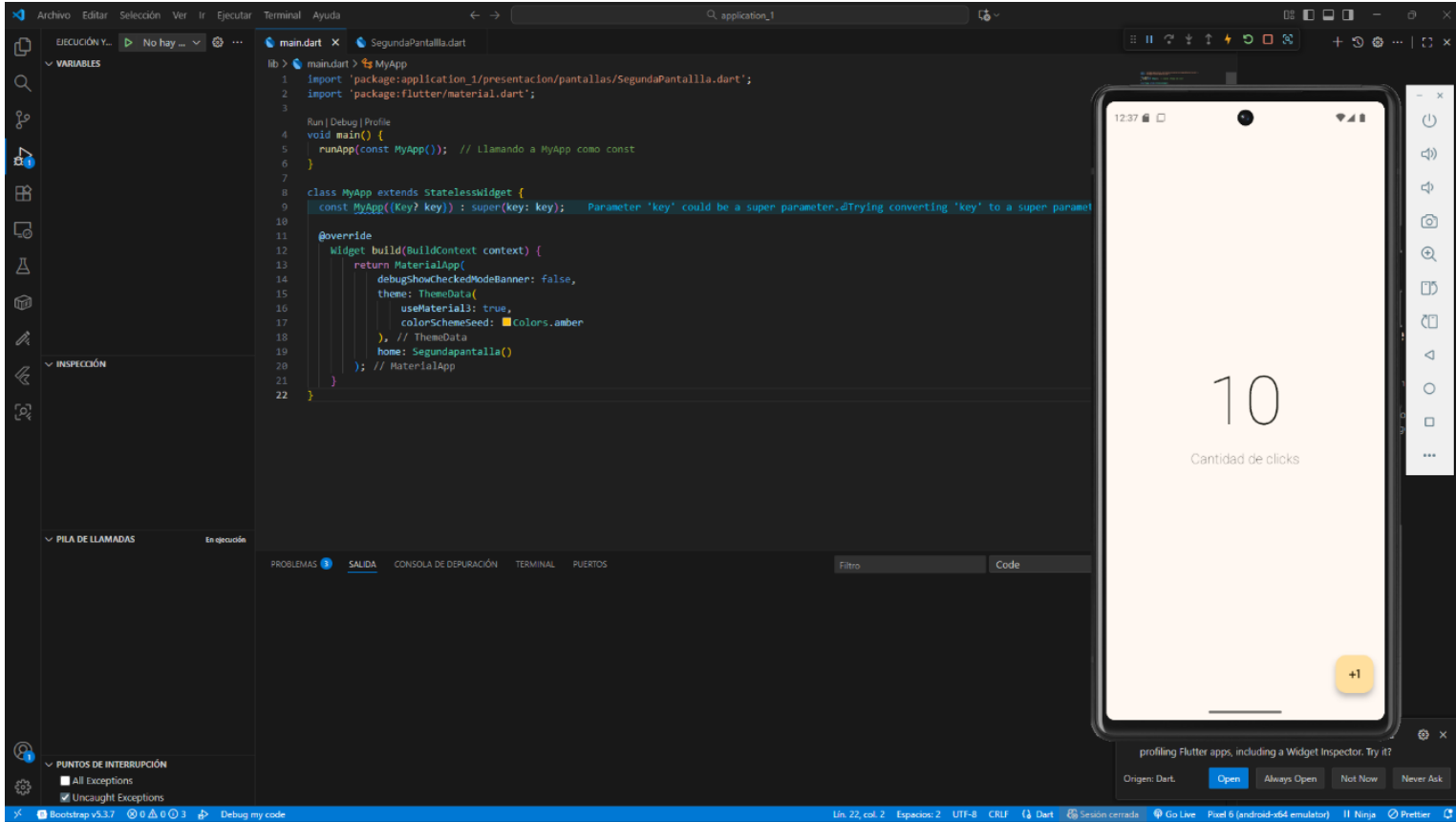


CURSO DE FLUTTER-DART DESDE CERO | APPBAR , THEME (video 41)

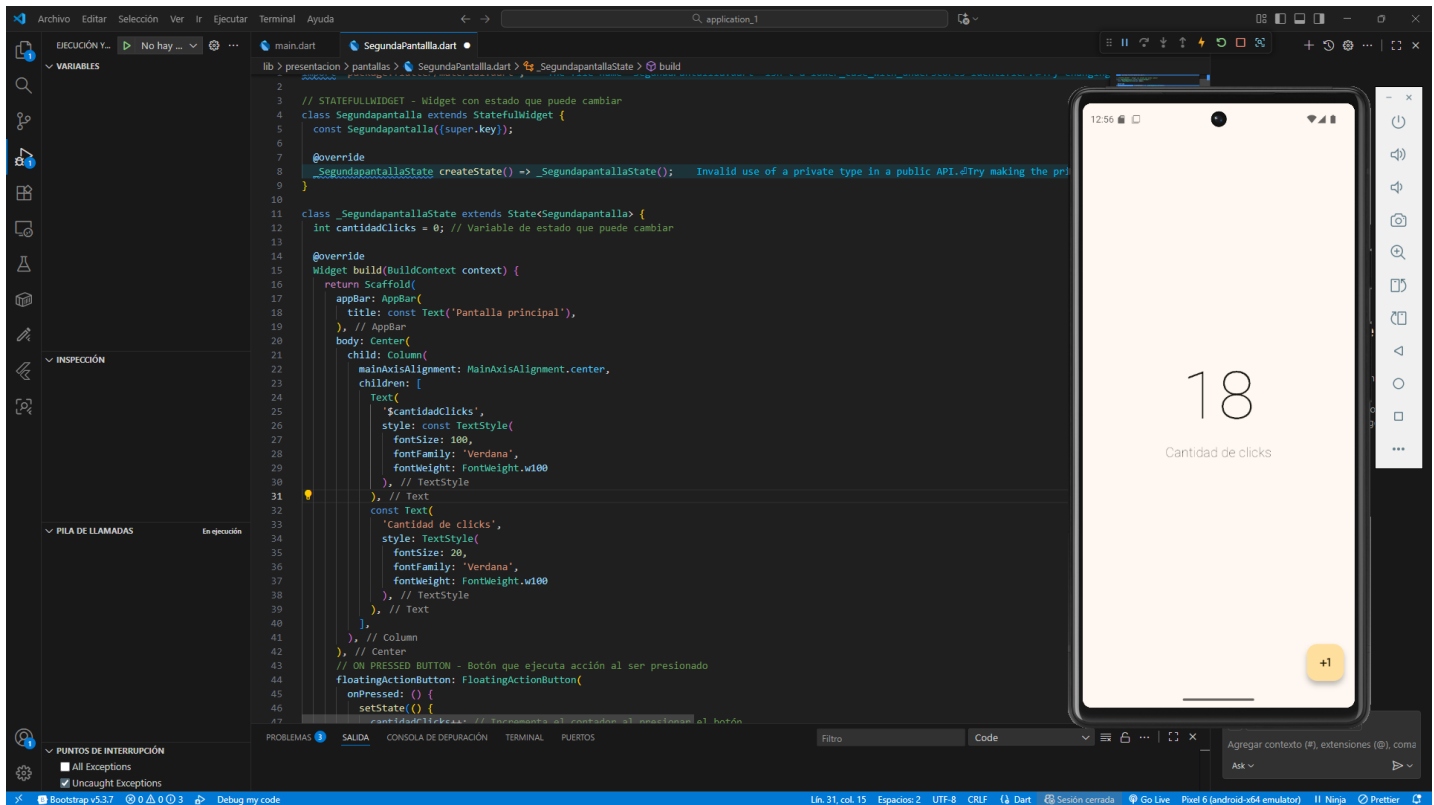
En este video simplemente se implementaron temas, en los cuales se aplicaron estilos y colores en la app.



CURSO DE FLUTTER-DART DESDE CERO | STATEFULLWIDGET, ON PRESSED BUTTON (video 42)

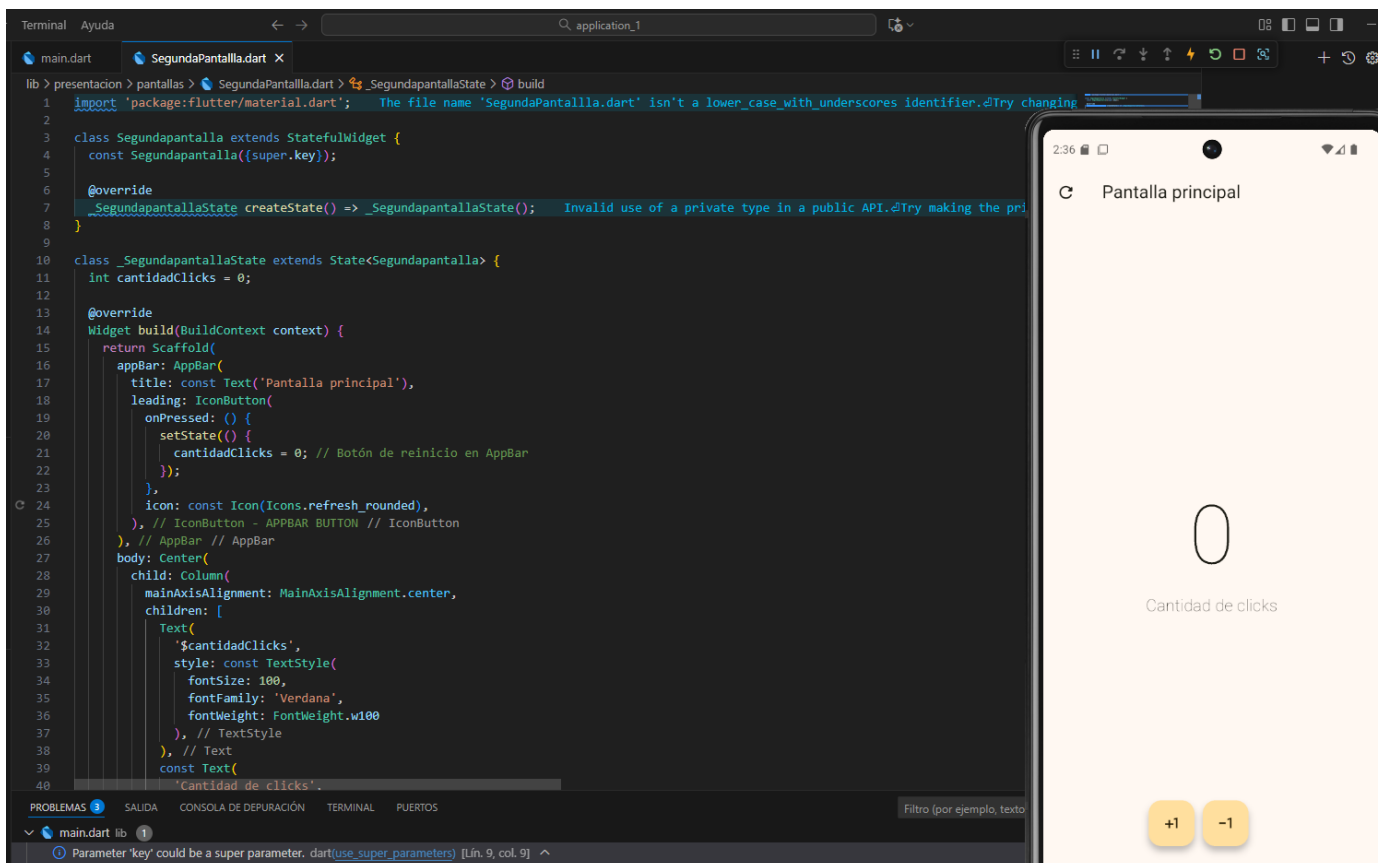
Lo que se implementó en este video fue:

1. **StatefulWidget**: Permite que la pantalla cambie (el número puede aumentar).
2. **cantidadClicks**: Variable que guarda el número del contador.
3. **onPressed**: Detecta cuando se toca el botón.
4. **setState**: Actualiza la pantalla con el nuevo número.



CURSO DE FLUTTER-DART DESDE CERO | APPBAR BUTTONS, ROW WIDGET (video 43)

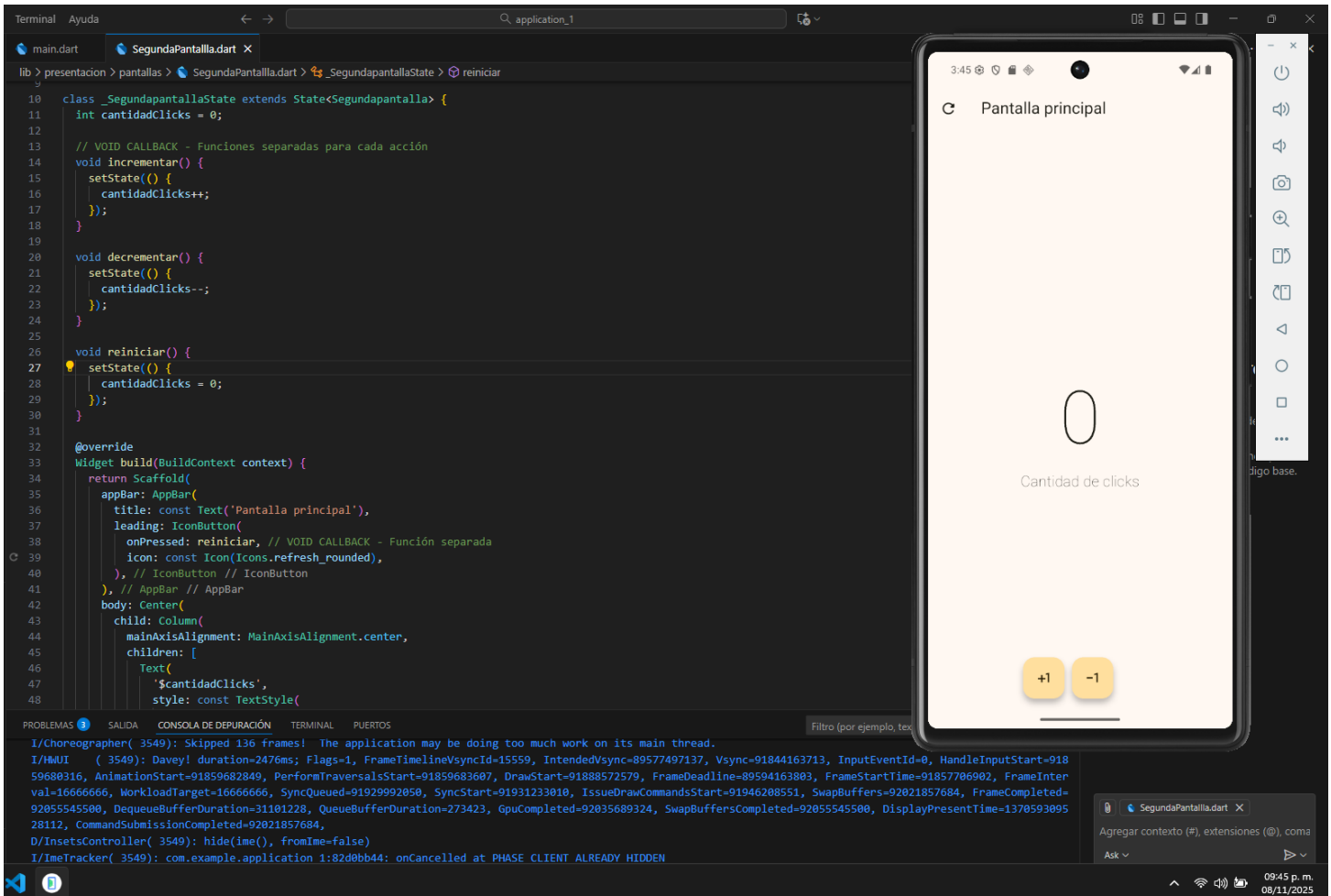
Se modificó la aplicación para agregar tres botones de control: un botón de reinicio en la AppBar que restablece el contador a cero, y dos botones flotantes organizados en fila (usando Row Widget) que permiten incrementar y decrementar el contador. Ahora los usuarios pueden aumentar, disminuir y reiniciar el número fácilmente, mejorando la interactividad de la interfaz.



CURSO DE FLUTTER-DART DESDE CERO | VOID CALLBACK

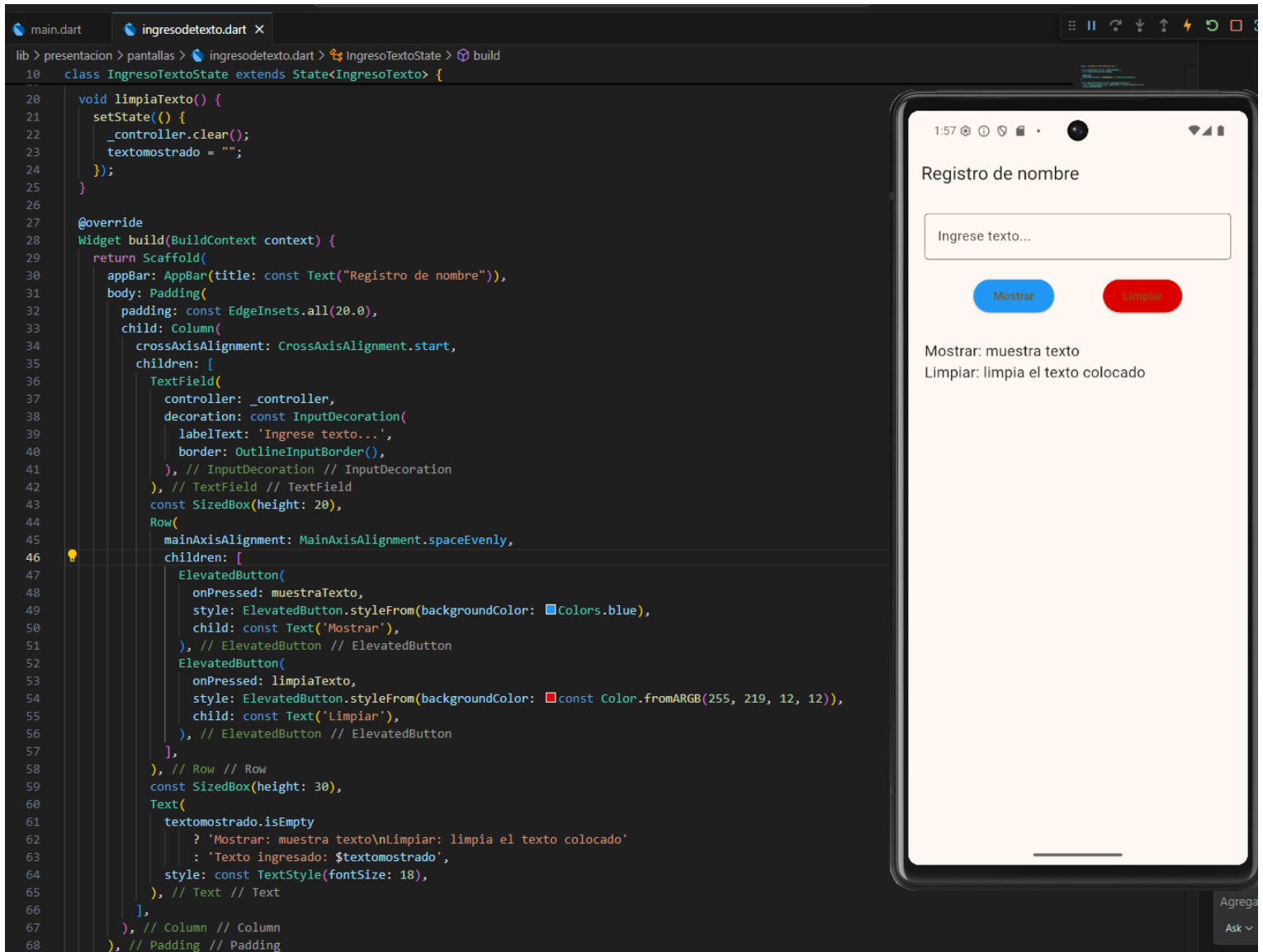
(video 44)

Se implementaron funciones void callback separadas para cada acción del contador: incrementar(), decrementar () y reiniciar(), reemplazando el código directo en los botones. Esto organiza mejor la lógica de la aplicación, haciendo que las acciones queden independientes de la interfaz gráfica y permitiendo un código más limpio, mantenible y fácil de modificar.



CURSO DE FLUTTER-DART DESDE CERO | TEXTFIELD (video 45)

Se implementó un TextField que permite a los usuarios ingresar texto, complementado con dos botones: "Mostrar" (azul) que toma el texto del TextField y lo muestra en pantalla, y "Limpiar" (rojo) que borra tanto el contenido del TextField como el texto mostrado, utilizando un TextEditingController para gestionar el input y setState para actualizar la interfaz dinámicamente.



CURSO DE FLUTTER-DART DESDE CERO | VALIDACIONES - GLOBALKEY-TEXTFORMFIELD (video 46)

Ahora se implementó un sistema de validaciones usando GlobalKey y TextFormField, donde se agregó un validador que verifica si el campo de texto está vacío, mostrando el mensaje "Por favor ingrese algún texto" cuando no se ingresa datos, y se encapsuló el formulario dentro de un widget Form con una llave global para gestionar el estado de validación antes de permitir mostrar el texto ingresado.

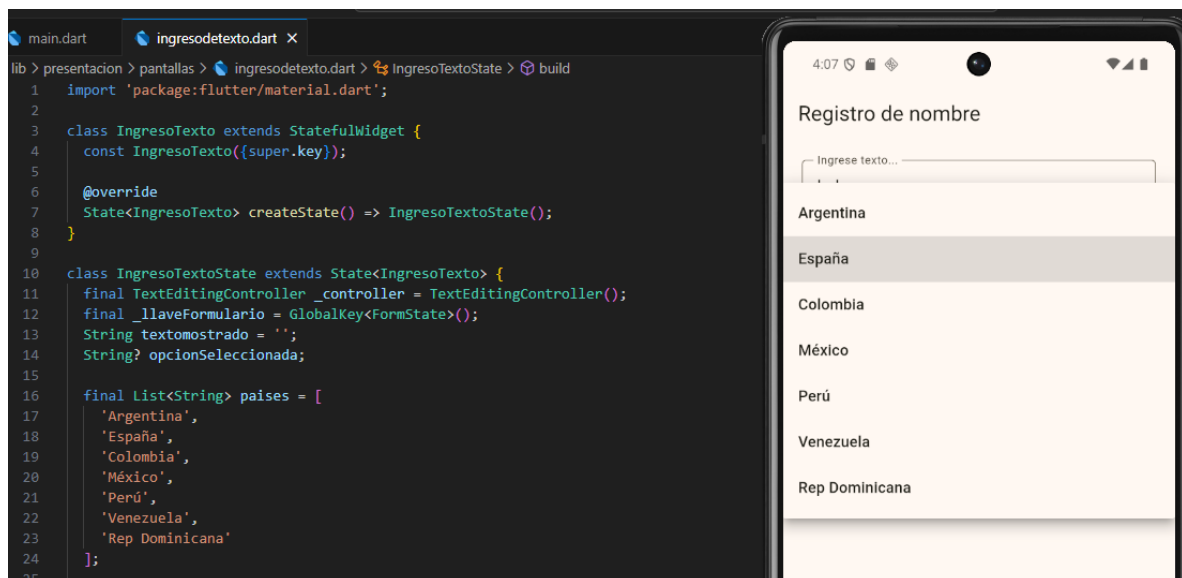
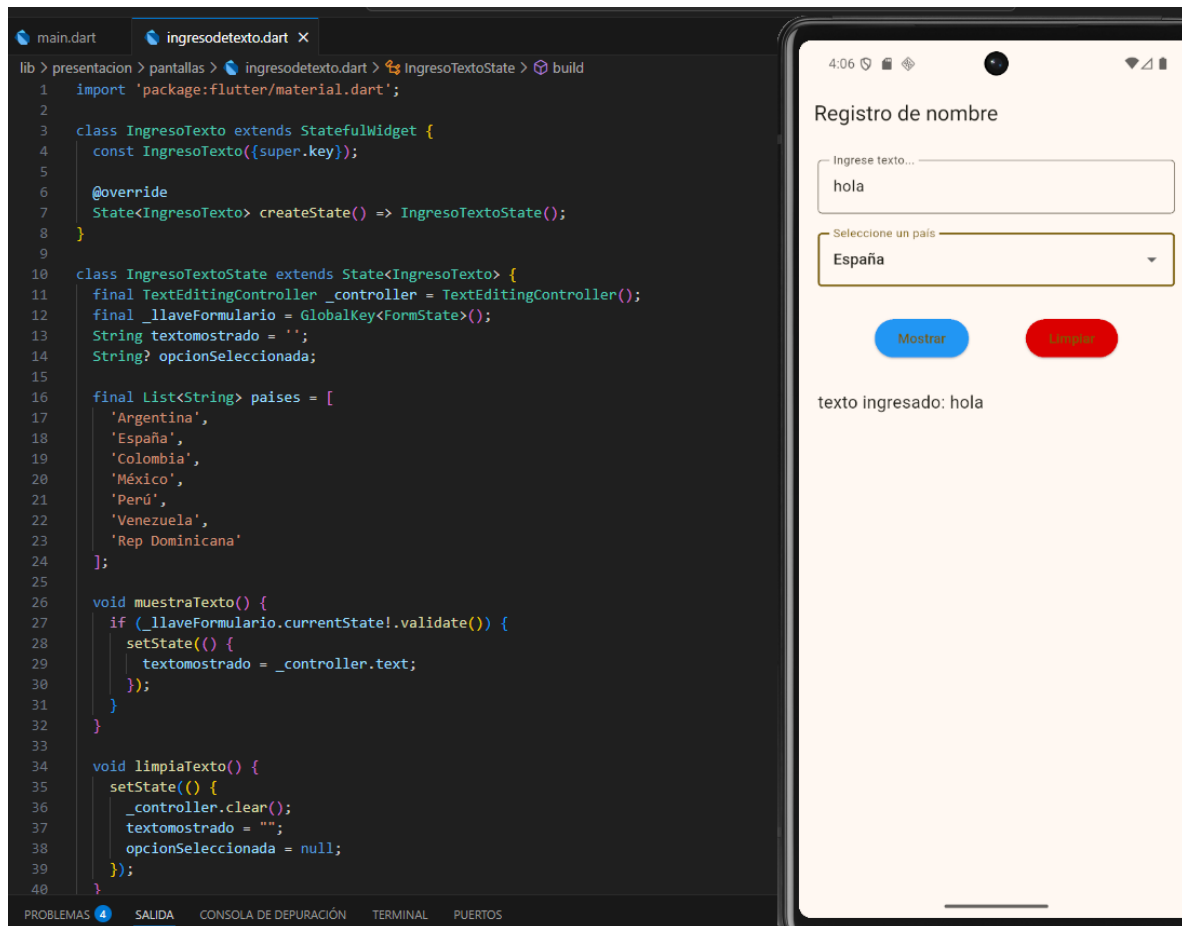
The image shows a code editor with the following Dart code for `ingresodetexto.dart`:

```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState > limpiaTexto
1 import 'package:flutter/material.dart';
2
3 class IngresoTexto extends StatefulWidget {
4   const IngresoTexto({super.key});
5
6   @override
7   State<IngresoTexto> createState() => IngresoTextoState();
8 }
9
10 class IngresoTextoState extends State<IngresoTexto> {
11   final TextEditingController _controller = TextEditingController();
12   final _llaveFormulario = GlobalKey<FormState>(); // GLOBALKEY
13   String textomostrado = '';
14
15   void muestraTexto() {
16     if (_llaveFormulario.currentState!.validate()) { // VALIDACIÓN
17       setState(() {
18         textomostrado = _controller.text;
19       });
20     }
21   }
22
23   void limpiaTexto() {
24     setState(() {
25       _controller.clear();
26       textomostrado = '';
27     });
28   }
29
30   @override
31   Widget build(BuildContext context) {
32     return Scaffold(
33       appBar: AppBar(title: const Text("Registro de nombre")),
34       body: Padding(
35         padding: const EdgeInsets.all(20.0),
36         child: Form( // FORM WIDGET
37           key: _llaveFormulario, // GLOBALKEY ASIGNADA
38           child: Column(
39             crossAxisAlignment: CrossAxisAlignment.start,
40             children: [
41               TextFormField( // TEXTFORMFIELD CON VALIDACIÓN
42                 controller: _controller,
43                 decoration: const InputDecoration(
44                   labelText: 'Ingrese texto...',
45                   border: OutlineInputBorder(),
46                 ), // InputDecoration
47                 validator: (value) { // VALIDADOR
48                   if (value == null || value.isEmpty) {
49                     return 'Por favor ingrese algún texto';
50                   }
51                   return null;
52                 },
43               ),
44             ],
45           ),
46         ),
47       ),
48     );
49   }
50 }
```

The mobile app preview on the right shows a screen titled "Registro de nombre". It features a text input field with the placeholder "Ingrese texto...". Below the field is a red error message: "Por favor ingrese algún texto". There are two buttons: "Mostrar" (blue) and "Limpiar" (red). Below the buttons, the text "Mostrar: muestra texto" and "Limpiar: limpia el texto colocado" is displayed. A bottom sheet is open, showing icons for voice search, close, check, smiley face, and a menu.

CURSO DE FLUTTER-DART DESDE CERO | DROPDOWN (video 47)

Lo que se realizó ahora es que se implementó un `DropdownButtonFormField` para permitir al usuario seleccionar un país de una lista desplegable de opciones predefinidas, utilizando un menú que muestra países latinoamericanos y almacena la selección en una variable para su posterior uso o validación en el formulario.



CURSO DE FLUTTER-DART DESDE CERO | WIDGET SWITCH (video 48)

En este ultimo lo que se realizo fue implementar un widget switch que permite al usuario activar o desactivar el estado de "usuario activo" mediante un interruptor visual, el cual actualiza dinámicamente la interfaz para mostrar el estado seleccionado (Activo/Inactivo) junto con el texto ingresado y el país elegido en el formulario completo.

