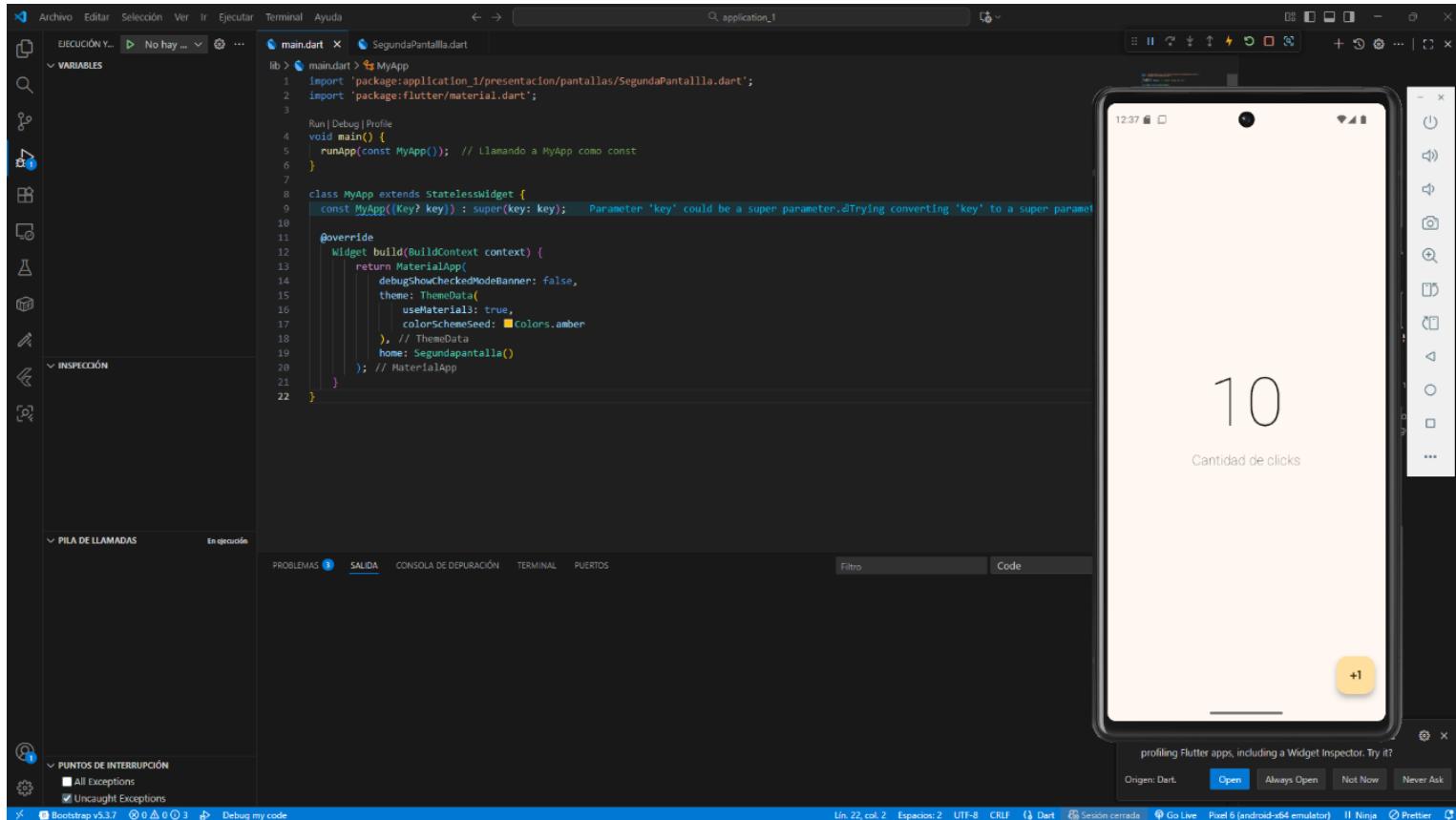


## CURSO DE FLUTTER-DART DESDE CERO | APPBAR , THEME (video 41)

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.

```

1 lib > presentación > pantallas > SegundaPantalla.dart > _SegundapantallaState > build
2
3 // STATEFULLWIDGET - Widget con estado que puede cambiar
4 class Segundapantalla extends StatefulWidget {
5   const Segundapantalla({super.key});
6
7   @override
8     SegundapantallaState createState() => _SegundapantallaState();
9 }
10
11 class _SegundapantallaState extends State<Segundapantalla> {
12   int cantidadClicks = 0; // Variable de estado que puede cambiar
13
14   @override
15   Widget build(BuildContext context) {
16     return Scaffold(
17       appBar: AppBar(
18         title: const Text('Pantalla principal'),
19       ), // AppBar
20       body: Center(
21         child: Column(
22           mainAxisAlignment: MainAxisAlignment.center,
23           children: [
24             Text(
25               'cantidadClicks',
26               style: const TextStyle(
27                 fontSize: 100,
28                 fontFamily: 'Verdana',
29                 fontWeight: FontWeight.w100
30               ), // TextStyle
31             ), // Text
32             const Text(
33               'Cantidad de clicks',
34               style: TextStyle(
35                 fontSize: 20,
36                 fontFamily: 'Verdana',
37                 fontWeight: FontWeight.w100
38               ), // TextStyle
39             ), // Text
40           ],
41         ), // Column
42       ), // Center
43       // ON PRESSED BUTTON - Botón que ejecuta acción al ser presionado
44       floatingActionButton: FloatingActionButton(
45         onPressed: () {
46           setState(() {
47             cantidadClicks++; // Incrementa el contador al presionar el botón
48           });
49         }
50       );
51     );
52   }
53 }

```

PUNTOS DE INTERRUPCIÓN

- All Exceptions
- Uncought Exceptions

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS Filtro Ask ▾

Lin. 31, col. 15 Espacios: 2 UTF-8 CRLF (3 Dart Sesión cerrada Go Live Pixel 6 (android-x64 emulator) If Ninja Prettier ▾

## 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.

```

1 Terminal Ayuda ← →
2 lib > presentación > pantallas > SegundaPantalla.dart > _SegundapantallaState > build
3
4 import 'package:flutter/material.dart'; The file name 'Segundapantalla.dart' isn't a lower_case_with_underscores identifier. Try changing it.
5
6 class Segundapantalla extends StatefulWidget {
7   const Segundapantalla({super.key});
8
9   @override
10     SegundapantallaState createState() => _SegundapantallaState();
11 }
12
13 class _SegundapantallaState extends State<Segundapantalla> {
14   int cantidadClicks = 0;
15
16   @override
17   Widget build(BuildContext context) {
18     return Scaffold(
19       appBar: AppBar(
20         title: const Text('Pantalla principal'),
21         leading: IconButton(
22           onPressed: () {
23             setState(() {
24               cantidadClicks = 0; // Botón de reinicio en AppBar
25             });
26           });
27         ),
28         icon: const Icon(Icons.refresh_rounded),
29       ), // IconButton - APPBAR BUTTON // IconButton
30     ), // AppBar // AppBar
31     body: Center(
32       child: Column(
33         mainAxisAlignment: MainAxisAlignment.center,
34         children: [
35           Text(
36             'cantidadClicks',
37             style: const TextStyle(
38               fontSize: 100,
39               fontFamily: 'Verdana',
40               fontWeight: FontWeight.w100
41             ), // TextStyle
42             ), // Text
43             const Text(
44               'Cantidad de clicks',
45             )
46           ],
47         ), // Column
48       ), // Center
49       // ON PRESSED BUTTON - Botón que ejecuta acción al ser presionado
50       floatingActionButton: Row(
51         mainAxisAlignment: MainAxisAlignment.spaceEvenly,
52         children: [
53           FloatingActionButton(
54             onPressed: () {
55               setState(() {
56                 cantidadClicks--;
57               });
58             },
59             child: const Text('-1'),
60           ),
61           FloatingActionButton(
62             onPressed: () {
63               setState(() {
64                 cantidadClicks++;
65               });
66             },
67             child: const Text('+1'),
68           )
69         ],
70       );
71     );
72   }
73 }

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS Filtro (por ejemplo, texto)

Parameter 'key' could be a super parameter. dart[use\_super\_parameters] [Líne. 9, col. 9] ▾

# 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, mantenable y fácil de modificar.

```
Terminal Ayuda ↗ SegundaPantalla.dart × 🔍 application_1
```

```
lib > presentacion > pantallas > SegundaPantalla.dart > _SegundapantallaState > reiniciar
```

```
10 class _SegundapantallaState extends State<Segundapantalla> {
11   int cantidadClicks = 0;
12
13   // VOID CALLBACK - Funciones separadas para cada acción
14   void incrementar() {
15     setState(() {
16       cantidadClicks++;
17     });
18   }
19
20   void decrementar() {
21     setState(() {
22       cantidadClicks--;
23     });
24   }
25
26   void reiniciar() {
27     setState(() {
28       cantidadClicks = 0;
29     });
30   }
31
32   @override
33   Widget build(BuildContext context) {
34     return Scaffold(
35       appBar: AppBar(
36         title: const Text('Pantalla principal'),
37         leading: IconButton(
38           onPressed: reiniciar, // VOID CALLBACK - Función separada
39           icon: const Icon(Icons.refresh_rounded),
40         ), // IconButton // AppBar
41       ),
42       body: Center(
43         child: Column(
44           mainAxisAlignment: MainAxisAlignment.center,
45           children: [
46             Text(
47               '$cantidadClicks',
48               style: const TextStyle(

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS Filtro (por ejemplo, te)

```
I/Choreographer( 3549): Skipped 136 frames! The application may be doing too much work on its main thread.
I/HMUI    ( 3549): Davey! duration=2476ms; Flags=1, FrameTimelineVsyncId=15559, IntendedVsync=89577497137, Vsync=91844163713, InputEventId=0, HandleInputStart=91859682849, AnimationStart=91859683607, DrawStart=91888572579, FrameDeadline=89594163803, FrameStartTime=91857706902, FrameInterval=16666666, WorkLoadTarget=16666666, SyncQueued=91929992050, SyncStart=91931233010, IssueDrawCommandsStart=91946208851, SwapBuffers=92021857684, FrameCompleted=92055545500, DequeueBufferDuration=31101228, QueueBufferDuration=273423, GpuCompleted=92035689324, SwapBuffersCompleted=92055545500, DisplayPresentime=1370593095
D/InsetsController( 3549): hide(ime(), fromIme=false)
I/ImeTracker( 3549): com.example.application 1:82d0bb44: onCancelled at PHASE CLIENT ALREADY HIDDEN
```

SegundaPantalla.dart × Agregar contexto (#), extensiones (@), coma Ask ▾

09:45 p.m. 08/11/2025

## 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.

The screenshot shows the Flutter development environment. On the left, the code for 'ingresodetexto.dart' is displayed in a code editor. The code defines a stateful widget 'IngresoTextoState' that contains a TextField and two ElevatedButtons ('Mostrar' and 'Limpiar'). It also includes logic to handle text changes and button presses. On the right, an Android emulator displays the resulting application titled 'Registro de nombre'. It features a text input field with placeholder text 'Ingrese texto...', a blue 'Mostrar' button, and a red 'Limpiar' button. Below the screen, descriptive text explains the functions of each button: 'Mostrar: muestra texto' and 'Limpiar: limpia el texto colocado'.

```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState > build
10 class IngresoTextoState extends State<IngresoTexto> {
11
12     TextEditingController _controller = TextEditingController();
13
14     String textomostrado = '';
15
16     void limpiaTexto() {
17         setState(() {
18             _controller.clear();
19             textomostrado = '';
20         });
21     }
22
23     @override
24     Widget build(BuildContext context) {
25         return Scaffold(
26             appBar: AppBar(title: const Text("Registro de nombre")),
27             body: Padding(
28                 padding: const EdgeInsets.all(20.0),
29                 child: Column(
30                     crossAxisAlignment: CrossAxisAlignment.start,
31                     children: [
32                         TextField(
33                             controller: _controller,
34                             decoration: const InputDecoration(
35                                 labelText: 'Ingrese texto...',
36                                 border: OutlineInputBorder(),
37                             ), // InputDecoration // InputDecoration
38                         ), // TextField // TextField
39                         const SizedBox(height: 20),
40                         Row(
41                             mainAxisAlignment: MainAxisAlignment.spaceEvenly,
42                             children: [
43                                 ElevatedButton(
44                                     onPressed: muestraTexto,
45                                     style: ElevatedButton.styleFrom(backgroundColor: Colors.blue),
46                                     child: const Text('Mostrar'),
47                                 ), // ElevatedButton // ElevatedButton
48                                 ElevatedButton(
49                                     onPressed: limpiaTexto,
50                                     style: ElevatedButton.styleFrom(backgroundColor: const Color.fromARGB(255, 219, 12, 12)),
51                                     child: const Text('Limpiar'),
52                                 ), // ElevatedButton // ElevatedButton
53                             ],
54                         ), // Row // Row
55                         const SizedBox(height: 30),
56                         Text(
57                             textomostrado.isEmpty
58                             ? 'Mostrar: muestra texto\nLimpiar: limpia el texto colocado'
59                             : 'Texto ingresado: $textomostrado',
60                             style: const TextStyle(fontSize: 18),
61                         ), // Text // Text
62                     ],
63                 ), // Column // Column
64             ), // Padding // Padding
65         );
66     }
67 }
68 
```

## 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 ingresan 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.

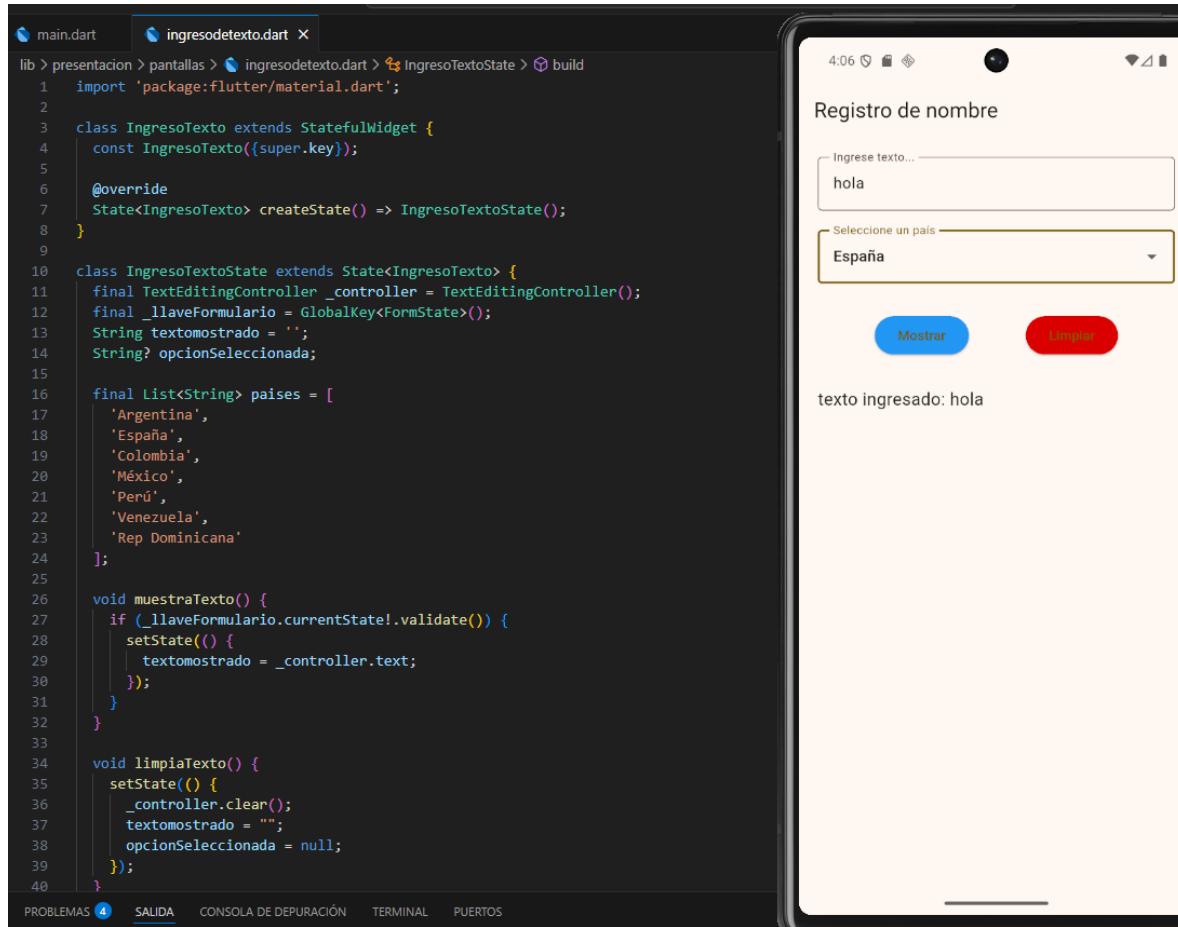


```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState > limpiaTexto
1 import 'package:flutter/material.dart';
2
3 class IngresoTexto extends StatefulWidget {
4   const IngresoTexto({super.key});
5
6   @override
7   State<IngresoTexto> createState() => IngresoTextoState();
8 }
9
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                 }
53               )
54             ],
55           ),
56         ),
57       ),
58     );
59   }
60 }
```

Lín. 27, col. 8 Espacios: 2 UTF-8 CRLF (À Dart Sesión cerrada Go Live Pixel 6 (android-x64) Ask ✓

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

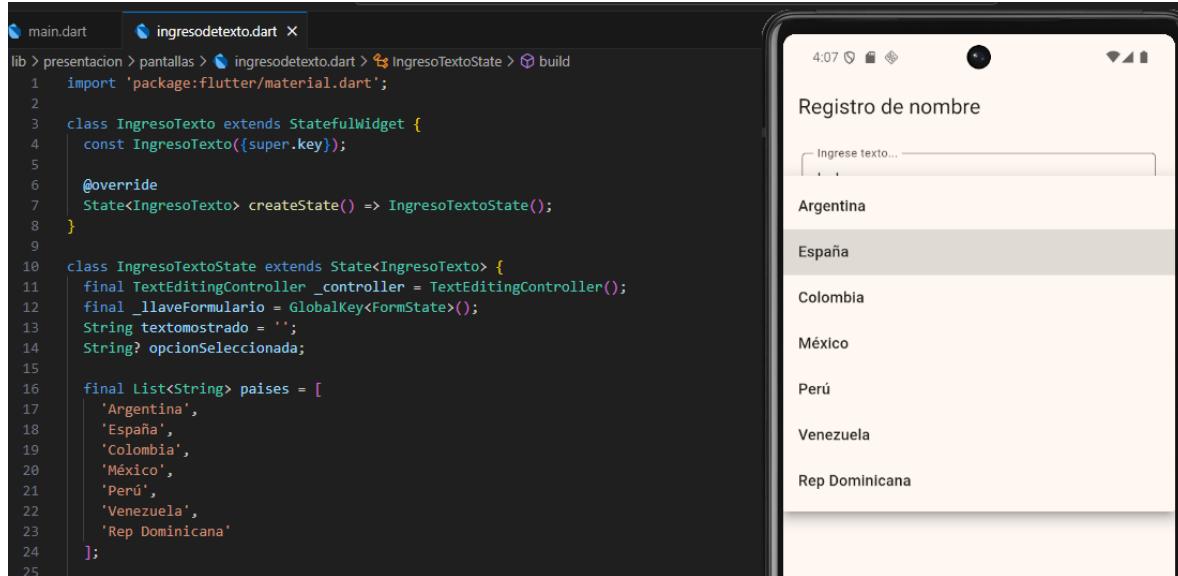
Lo que se realizó ahora es que se implementó un DropdownButtonFormField para permitir al usuario seleccionar un país de una lista desplegable de opciones predefinidas, utilizando un menú que muestra países latinoamericanos y almacena la selección en una variable para su posterior uso o validación en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



The image shows the Android Studio interface. On the left, there are two tabs: 'main.dart' and 'ingresodetexto.dart'. The 'ingresodetexto.dart' tab is active, displaying the following Dart code:

```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState > build
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 GlobalKey<FormState>() _llaveFormulario = GlobalKey<FormState>();
13   String textomostrado = '';
14   String? opcionSeleccionada;
15
16   final List<String> paises = [
17     'Argentina',
18     'España',
19     'Colombia',
20     'México',
21     'Perú',
22     'Venezuela',
23     'Rep Dominicana'
24   ];
25
26   void muestraTexto() {
27     if (_llaveFormulario.currentState!.validate()) {
28       setState(() {
29         textomostrado = _controller.text;
30       });
31     }
32   }
33
34   void limpiaTexto() {
35     setState(() {
36       _controller.clear();
37       textomostrado = "";
38       opcionSeleccionada = null;
39     });
40   }
41 }
```

Below the code editor, there are tabs for 'PROBLEMAS', 'SALIDA', 'CONSOLA DE DEPURACIÓN', 'TERMINAL', and 'PUERTOS'. The 'SALIDA' tab is selected. To the right of the code editor is a screenshot of the app running on an Android device. The screen title is 'Registro de nombre'. It contains a text input field with the placeholder 'Ingrese texto...' and the value 'hola'. Below it is a dropdown menu labeled 'Seleccione un país' with the option 'España' selected. At the bottom are two buttons: 'Mostrar' (blue) and 'Limpiar' (red). Below the screen, the text 'texto ingresado: hola' is displayed.



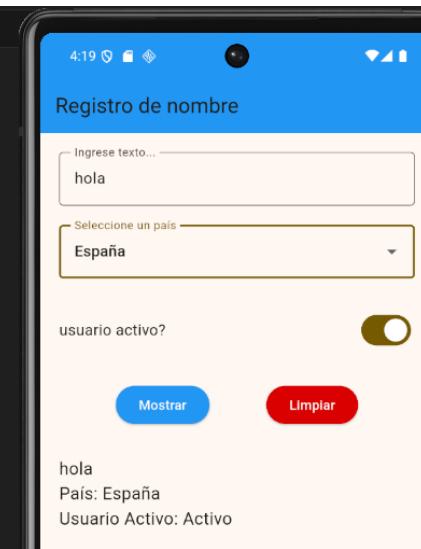
The image shows the Android Studio interface again. The 'ingresodetexto.dart' tab is active, displaying the same Dart code as the previous screenshot. The 'SALIDA' tab is selected below the code editor. To the right is a screenshot of the app running on an Android device. The screen title is 'Registro de nombre'. It shows a dropdown menu with a list of countries: Argentina, España, Colombia, México, Perú, Venezuela, and Rep Dominicana. The option 'Argentina' is currently selected. The rest of the interface is identical to the one shown in the first screenshot.

## 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.



```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState
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 GlobalKey<FormState> _llaveFormulario = GlobalKey<FormState>();
13   String textomostrado = '';
14   String? opcionSeleccionada;
15   bool usuarioActivo = false;
16
17   final List<String> paises = [
18     'Argentina',
19     'España',
20     'Colombia',
21     'México',
22     'Perú',
23     'Venezuela',
24     'Rep Dominicana'
25   ];
26
27   void muestraTexto() {
28     if (_llaveFormulario.currentState!.validate()) {
29       setState(() {
30         textomostrado = '${_controller.text}\nPaís: $opcionSeleccionada\nUsuario Activo: ${usuarioActivo ? 'Activo' : 'Inactivo'}';
31       });
32     }
33   }
34
35   void limpiaTexto() {
36     setState(() {
37       _controller.clear();
38       textomostrado = '';
39       opcionSeleccionada = null;
40       usuarioActivo = false;
41     });
42   }
43 }
```



```
lib > presentacion > pantallas > ingresodetexto.dart > IngresoTextoState
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 GlobalKey<FormState> _llaveFormulario = GlobalKey<FormState>();
13   String textomostrado = '';
14   String? opcionSeleccionada;
15   bool usuarioActivo = false;
16
17   final List<String> paises = [
18     'Argentina',
19     'España',
20     'Colombia',
21     'México',
22     'Perú',
23     'Venezuela',
24     'Rep Dominicana'
25   ];
26
27   void muestraTexto() {
28     if (_llaveFormulario.currentState!.validate()) {
29       setState(() {
30         textomostrado = '${_controller.text}\nPaís: $opcionSeleccionada\nUsuario Activo: ${usuarioActivo ? 'Activo' : 'Inactivo'}';
31       });
32     }
33   }
34
35   void limpiaTexto() {
36     setState(() {
37       _controller.clear();
38       textomostrado = '';
39       opcionSeleccionada = null;
40       usuarioActivo = false;
41     });
42   }
43 }
```