UNLEASHING THE POWER OF FORMS, MEDIA, AND ACCESSIBLITY IN FLUTTER

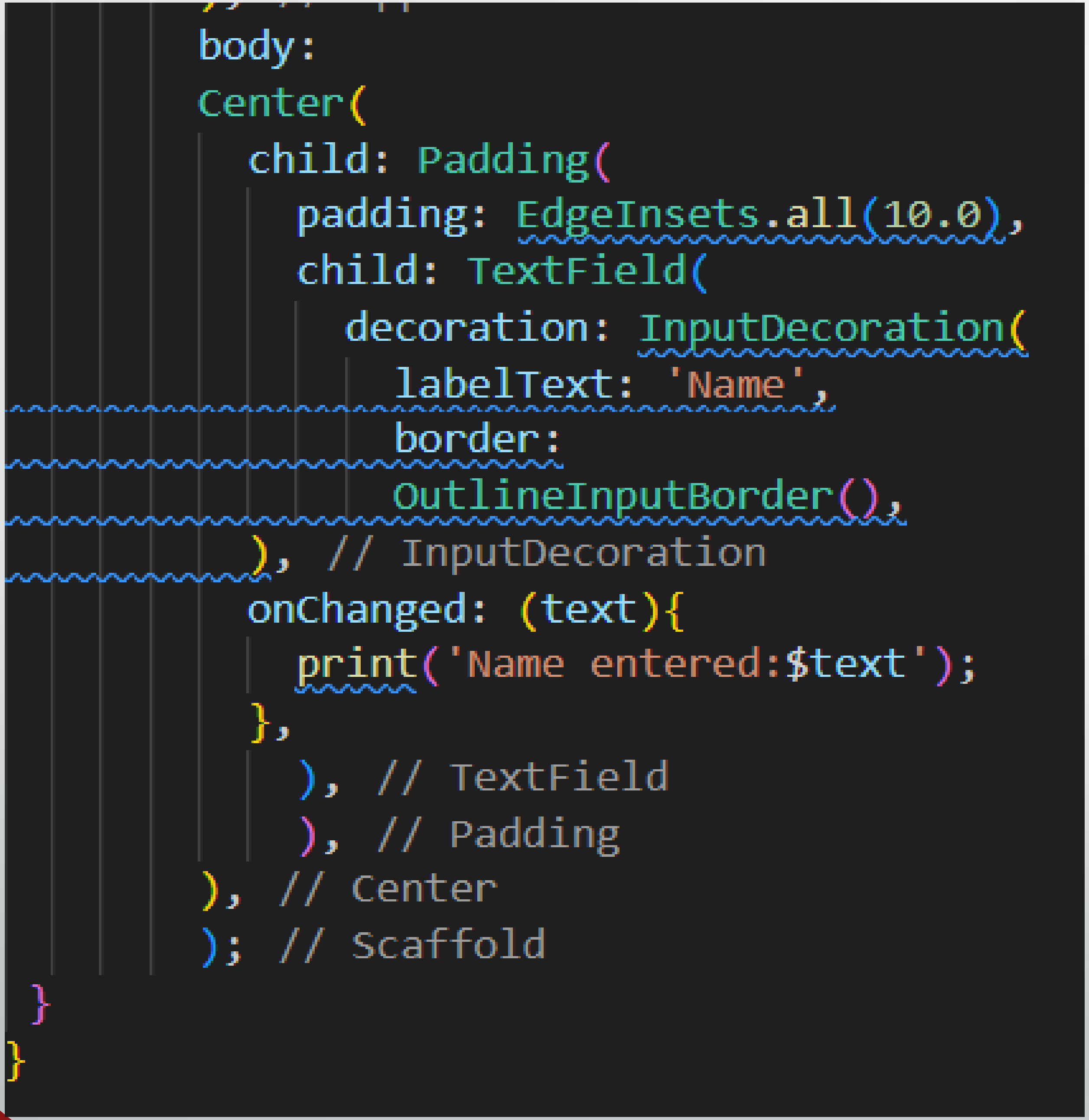
Introduction to forms and User Input

Crafting Effective Forms

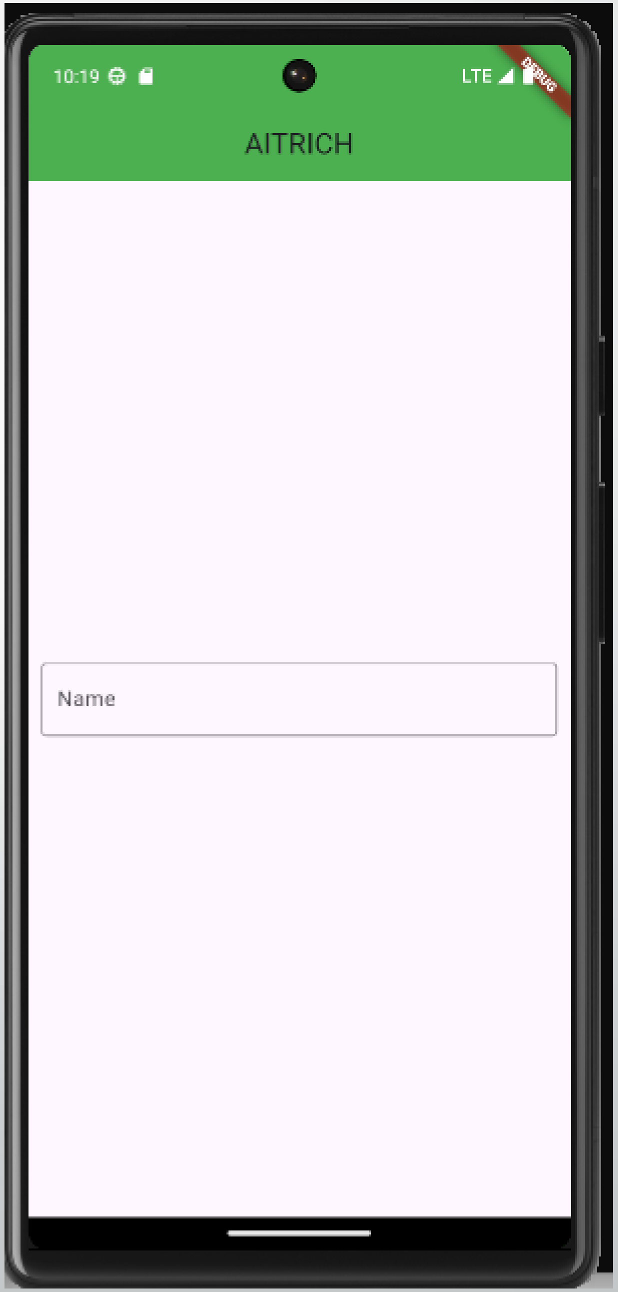
- Forms are essential for gathering user input and driving app functionality.
- Flutter provides a rich collection of form widgets for various input types.
- Building intuitive forms enhances user experience and data collection

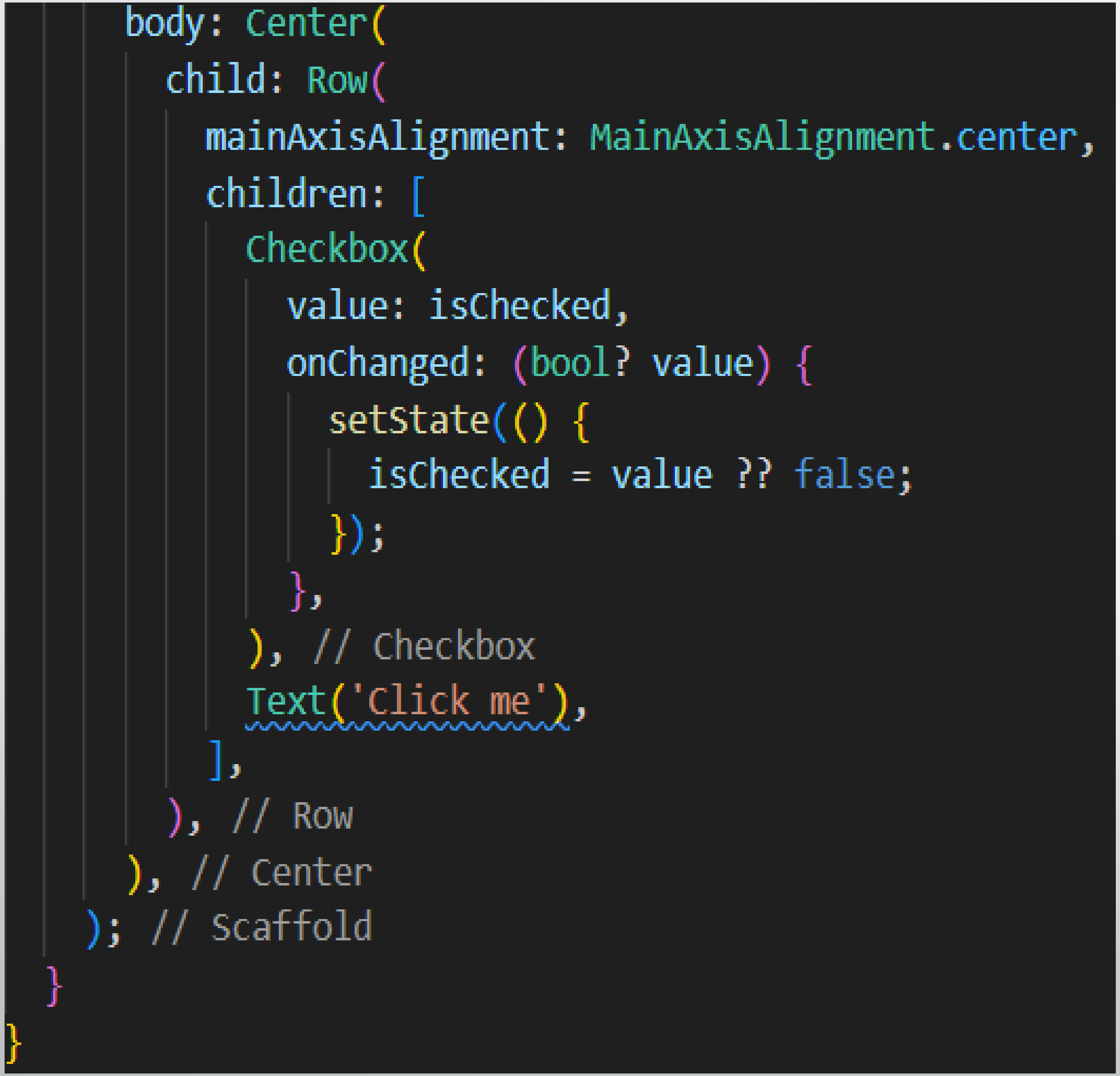
Essential Form Widgets

- Text Field
- Checkbox
- Radio Button
- Slider
- Dropdown Button

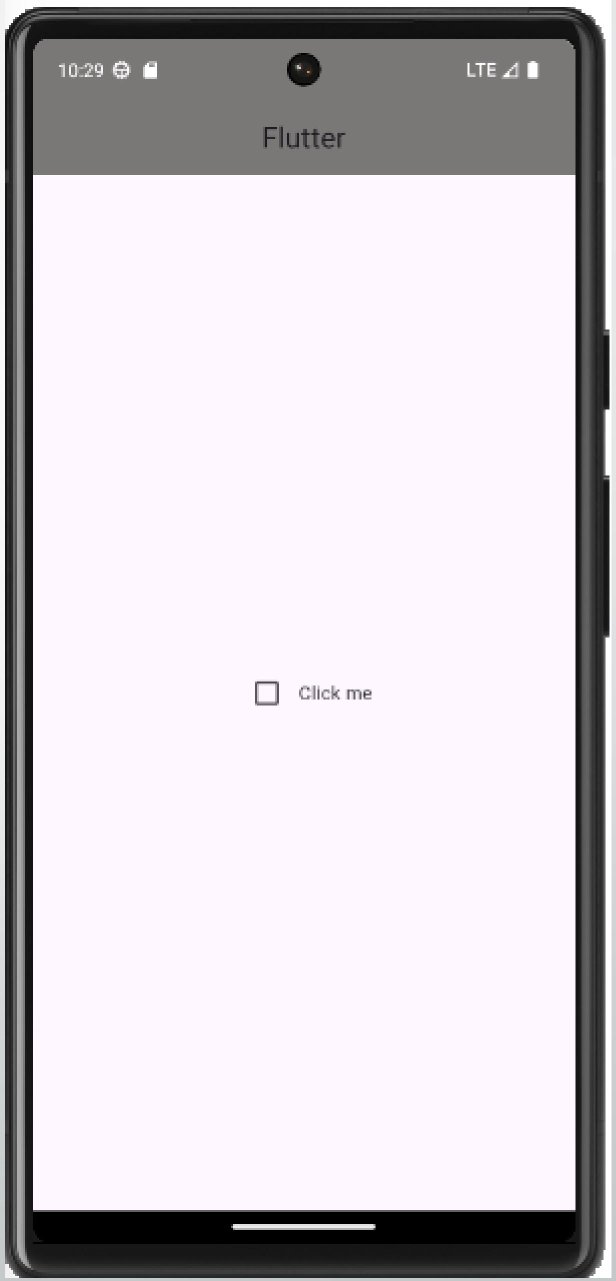


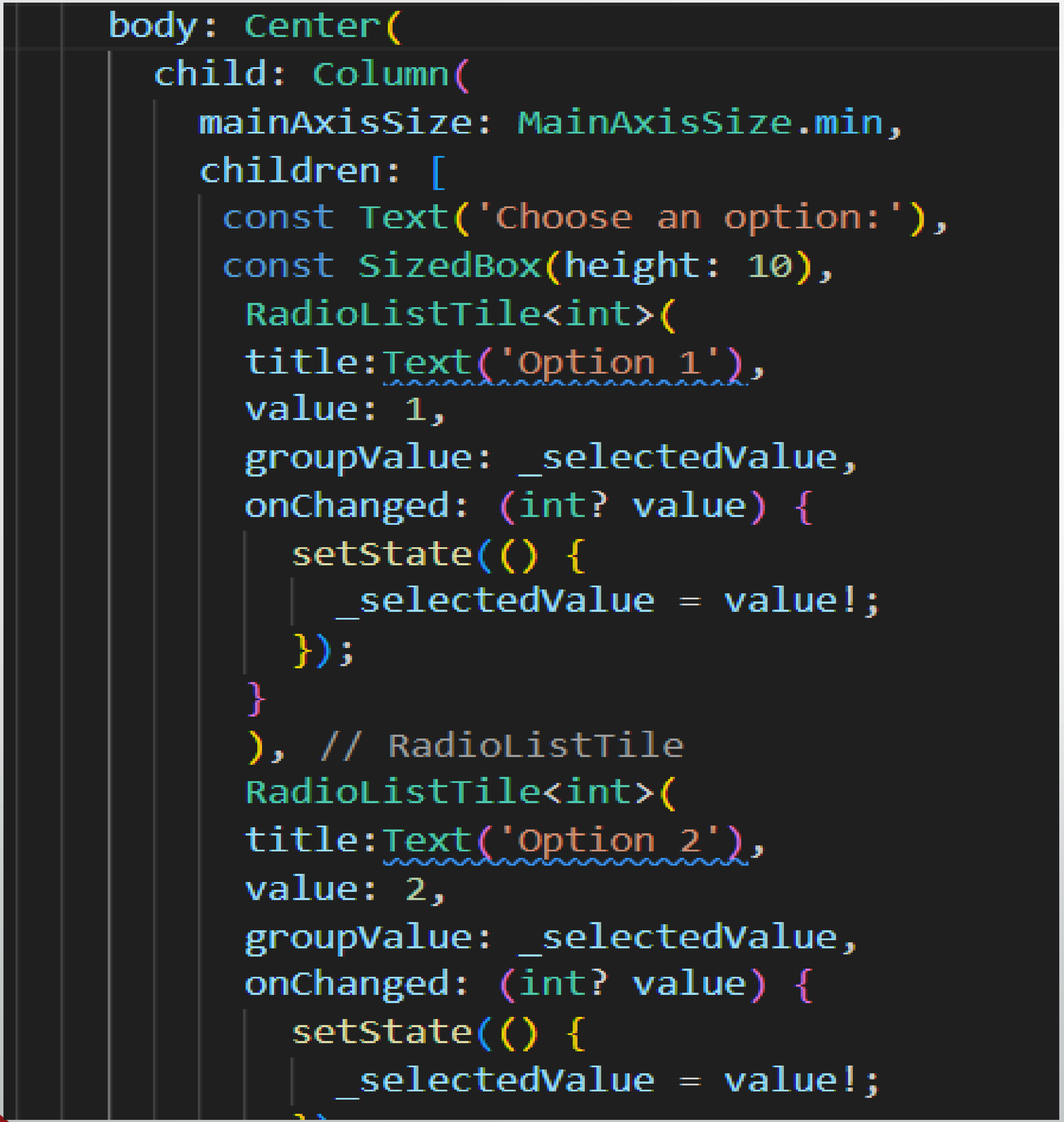
```
      body:
      Center(
        child: Padding(
          padding: EdgeInsets.all(10.0),
          child: TextField(
            decoration: InputDecoration(
              labelText: 'Name',
              border:
              OutlineInputBorder(),
          ), // InputDecoration
        onChanged: (text){
          print('Name entered:$text');
        },
          ), // TextField
          ), // Padding
      ), // Center
      ); // Scaffold
  }
}
```



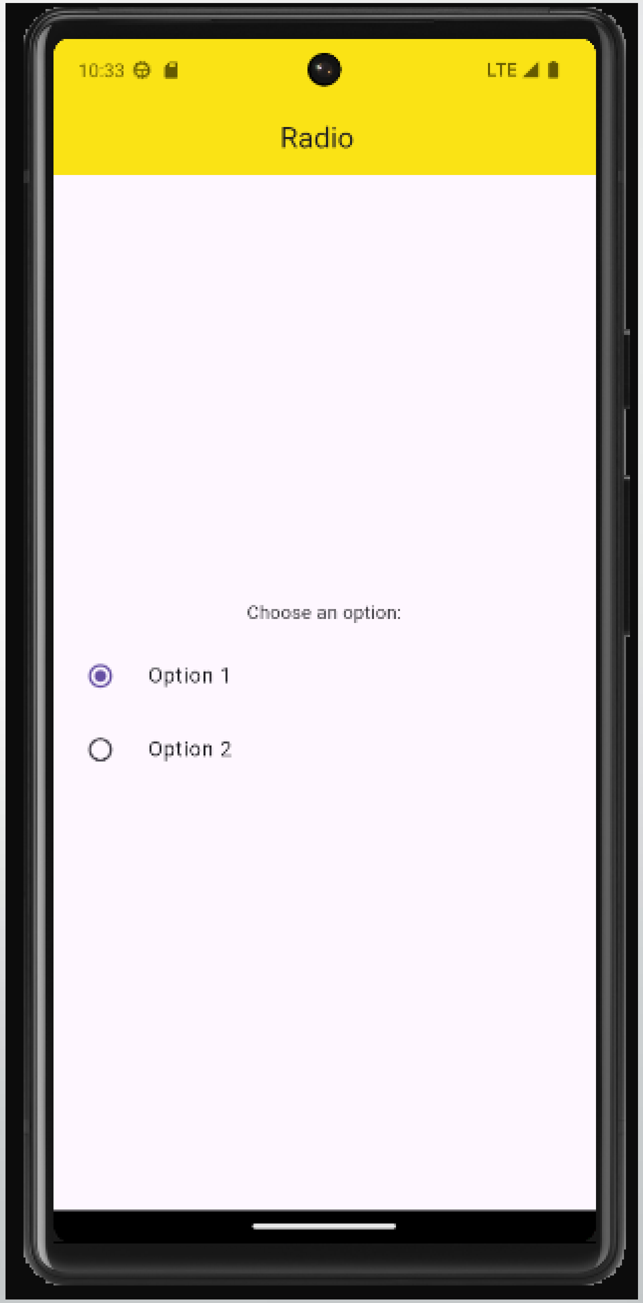


```
      body: Center(
        child: Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Checkbox(
              value: isChecked,
              onChanged: (bool? value) {
                setState(() {
                  isChecked = value ?? false;
                });
              },
            ), // Checkbox
            Text('Click me'),
          ],
        ), // Row
      ), // Center
    ); // Scaffold
  }
}
```





```
body: Center(
  child: Column(
    mainAxisSize: MainAxisSize.min,
    children: [
     const Text('Choose an option:'),
     const SizedBox(height: 10),
      RadioListTile<int>(
      title:Text('Option 1'),
      value: 1,
      groupValue: _selectedValue,
      onChanged: (int? value) {
        setState(() {
          _selectedValue = value!;
        });
      }
      ), // RadioListTile
      RadioListTile<int>(
      title:Text('Option 2'),
      value: 2,
      groupValue: _selectedValue,
      onChanged: (int? value) {
        setState(() {
          _selectedValue = value!;
```

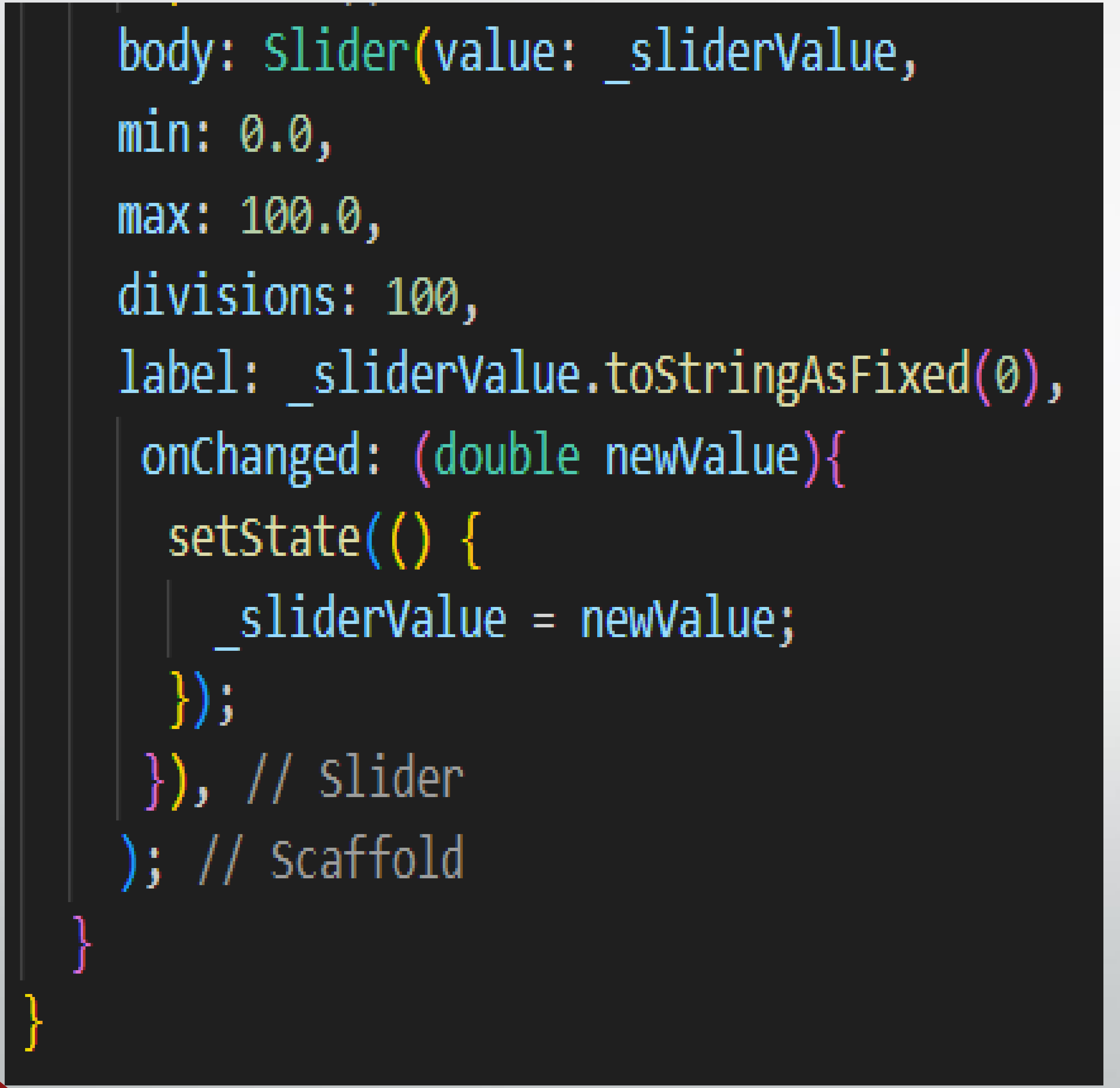


Radio

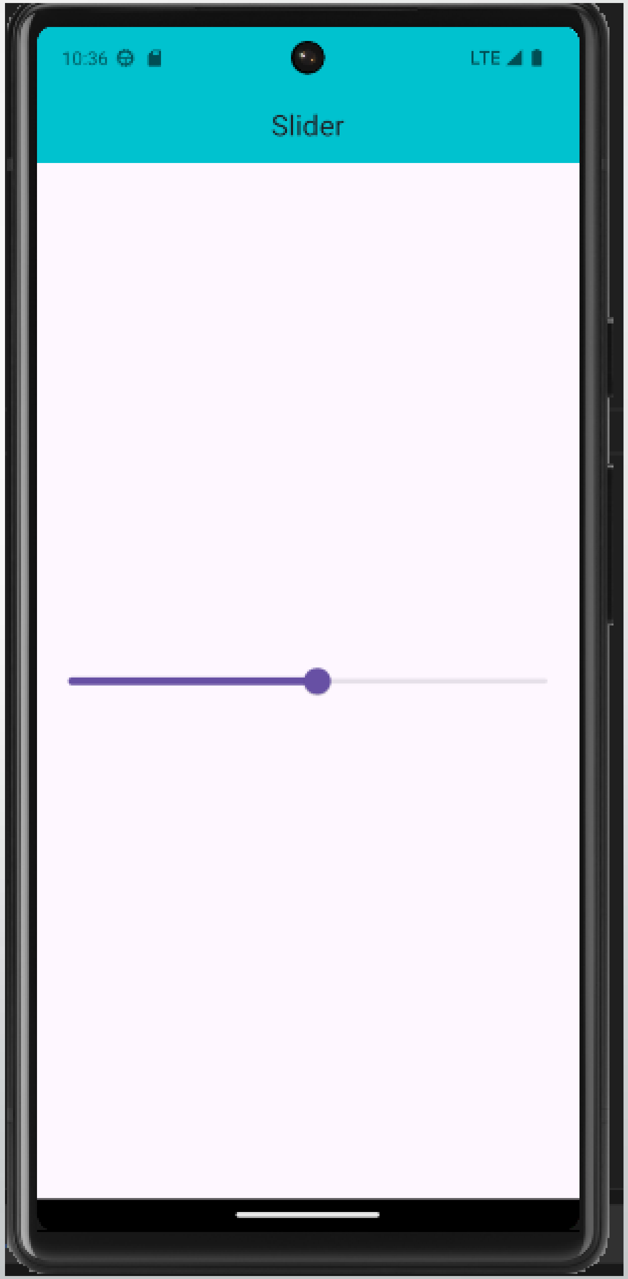
Choose an option:

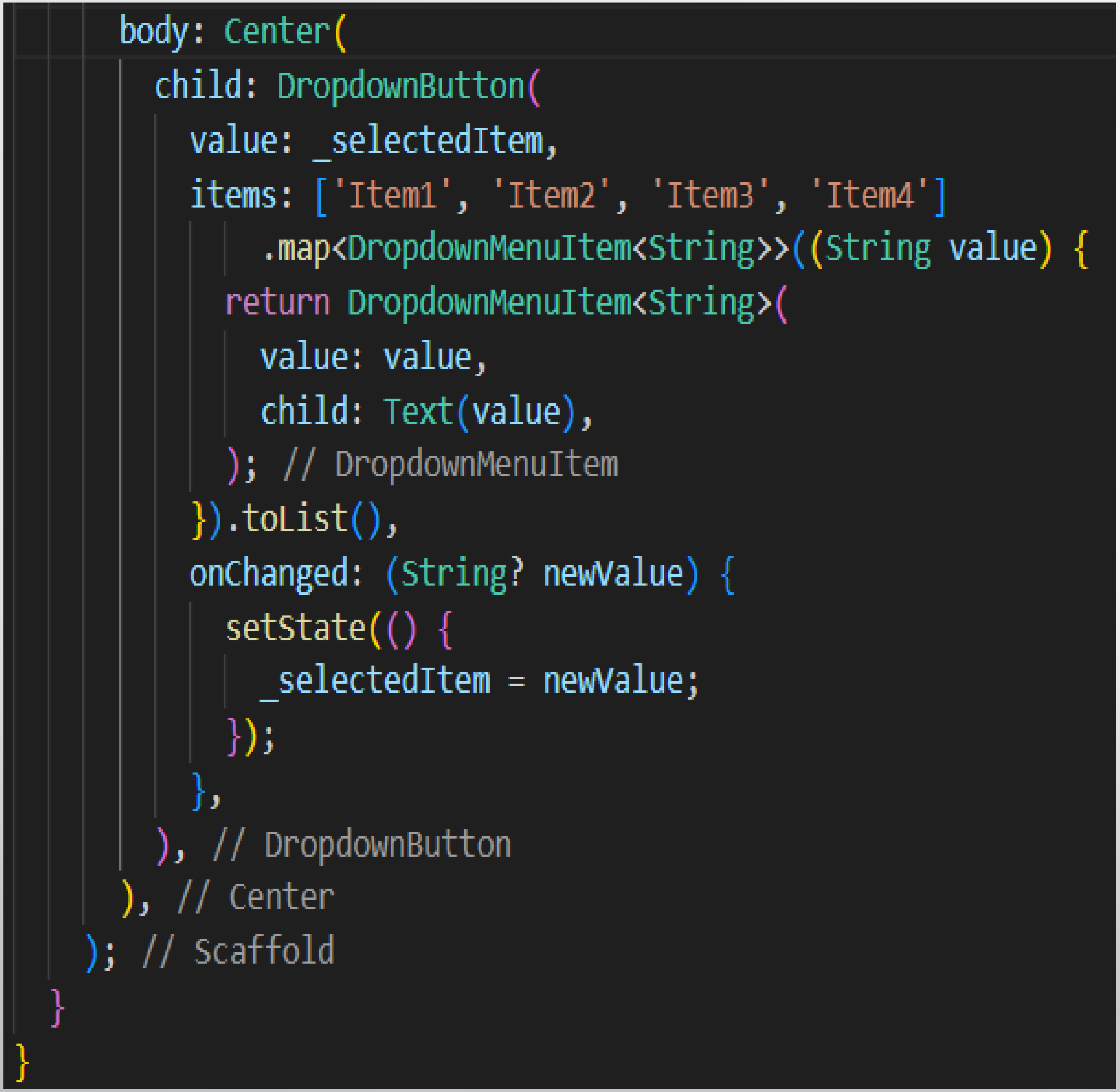
● Option 1

○ Option 2

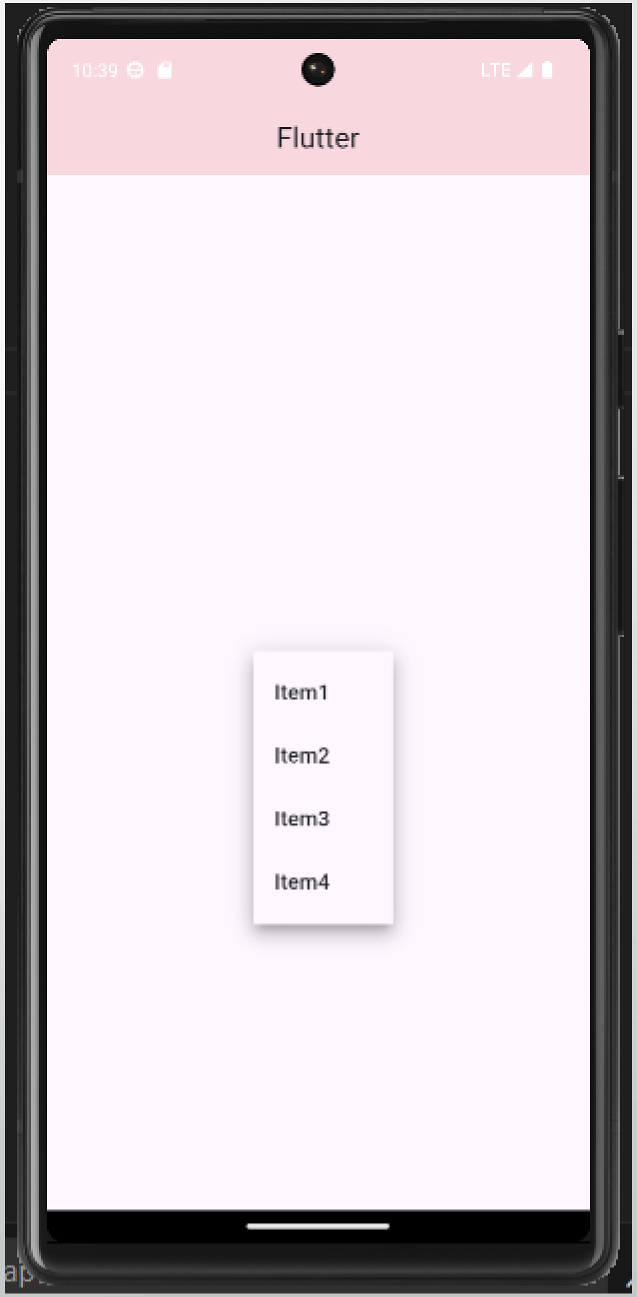


```
    body: Slider(value: _sliderValue,
    min: 0.0,
    max: 100.0,
    divisions: 100,
    label: _sliderValue.toStringAsFixed(0),
     onChanged: (double newValue){
      setState(() {
        _sliderValue = newValue;
      });
     }), // Slider
    ); // Scaffold
  }
}
```





```
      body: Center(
        child: DropdownButton(
          value: _selectedItem,
          items: ['Item1', 'Item2', 'Item3', 'Item4']
              .map<DropdownMenuItem<String>>((String value) {
            return DropdownMenuItem<String>(
              value: value,
              child: Text(value),
            ); // DropdownMenuItem
          }).toList(),
          onChanged: (String? newValue) {
            setState(() {
              _selectedItem = newValue;
            });
          },
        ), // DropdownButton
      ), // Center
    ); // Scaffold
  }
}
```



Form Validation

Form validation in Flutter involves checking whether the information entered by the user in a form is correct and meets certain criteria, such as filling all required fields or providing valid data

Ensuring Data Integrity

Ensuring data integrity in Flutter means validating, securing, and maintaining consistent data across the app, using proper storage, error handling, and synchronization with backends.

Custom Validation Methods

Custom Validation Methods in Flutter are user-defined functions used to validate form input based on specific rules.

Displaying Error Using UI

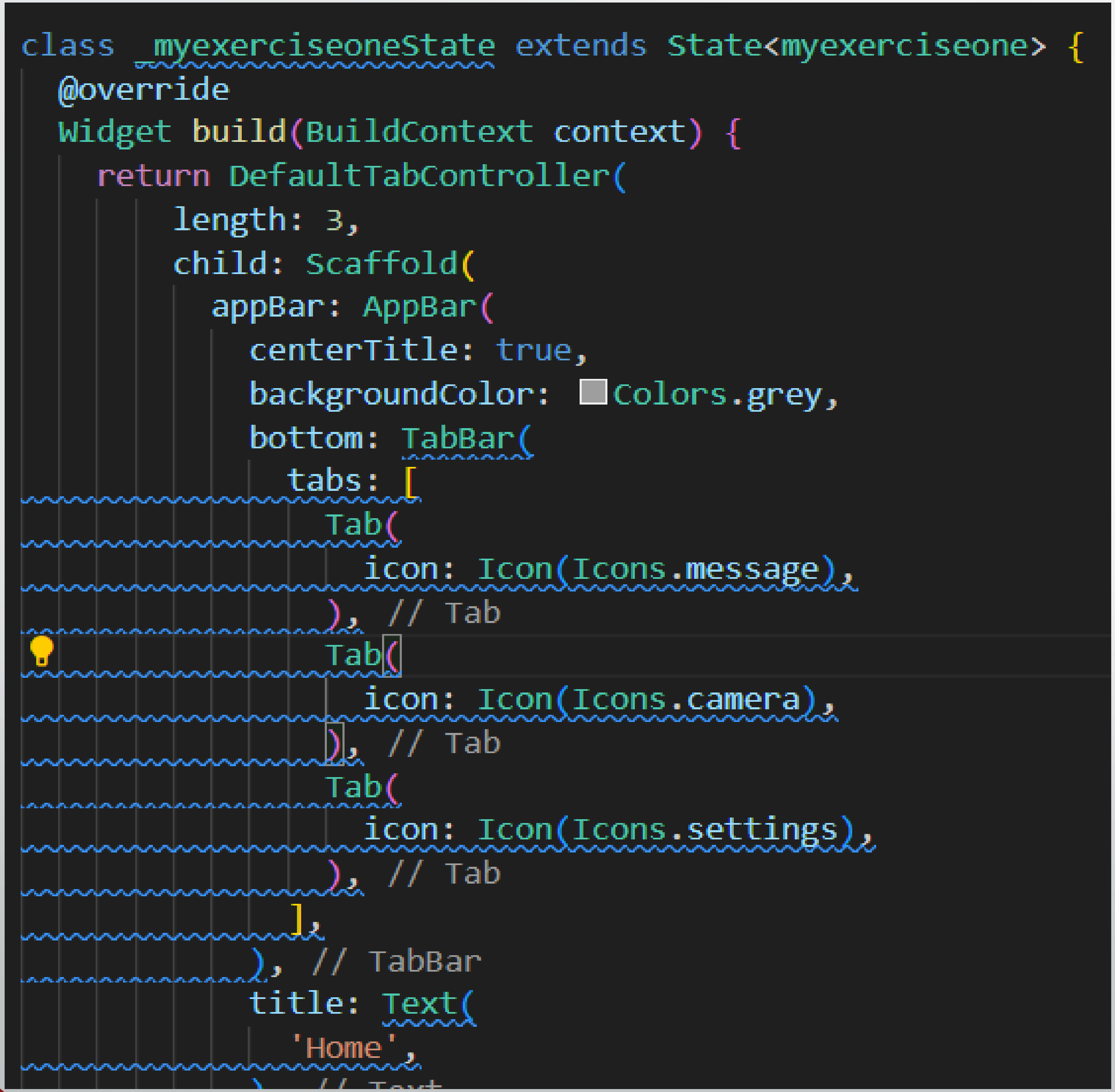
In Flutter, to display an error, you can use widgets like Text, Alert Dialog, to show error messages on the UI.

Flutter – Building Adaptive Apps

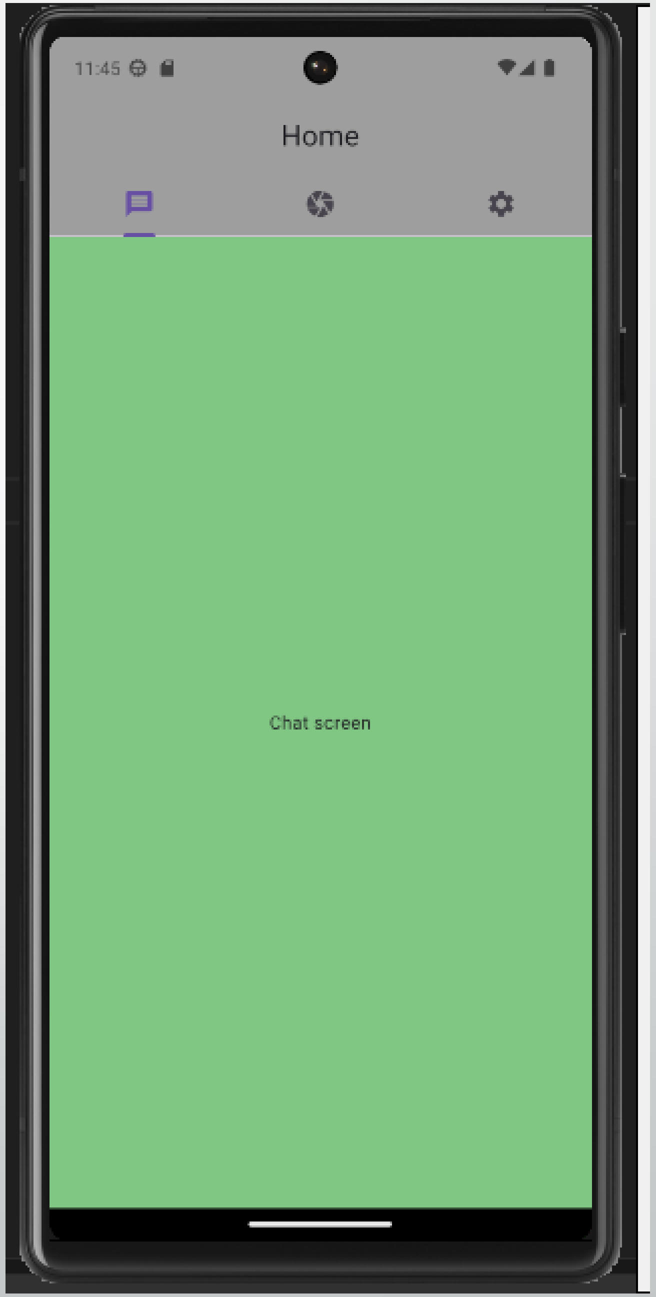
Building adaptive apps in Flutter means creating apps that adjust to different screen sizes, platforms, and orientations for a consistent user experience.

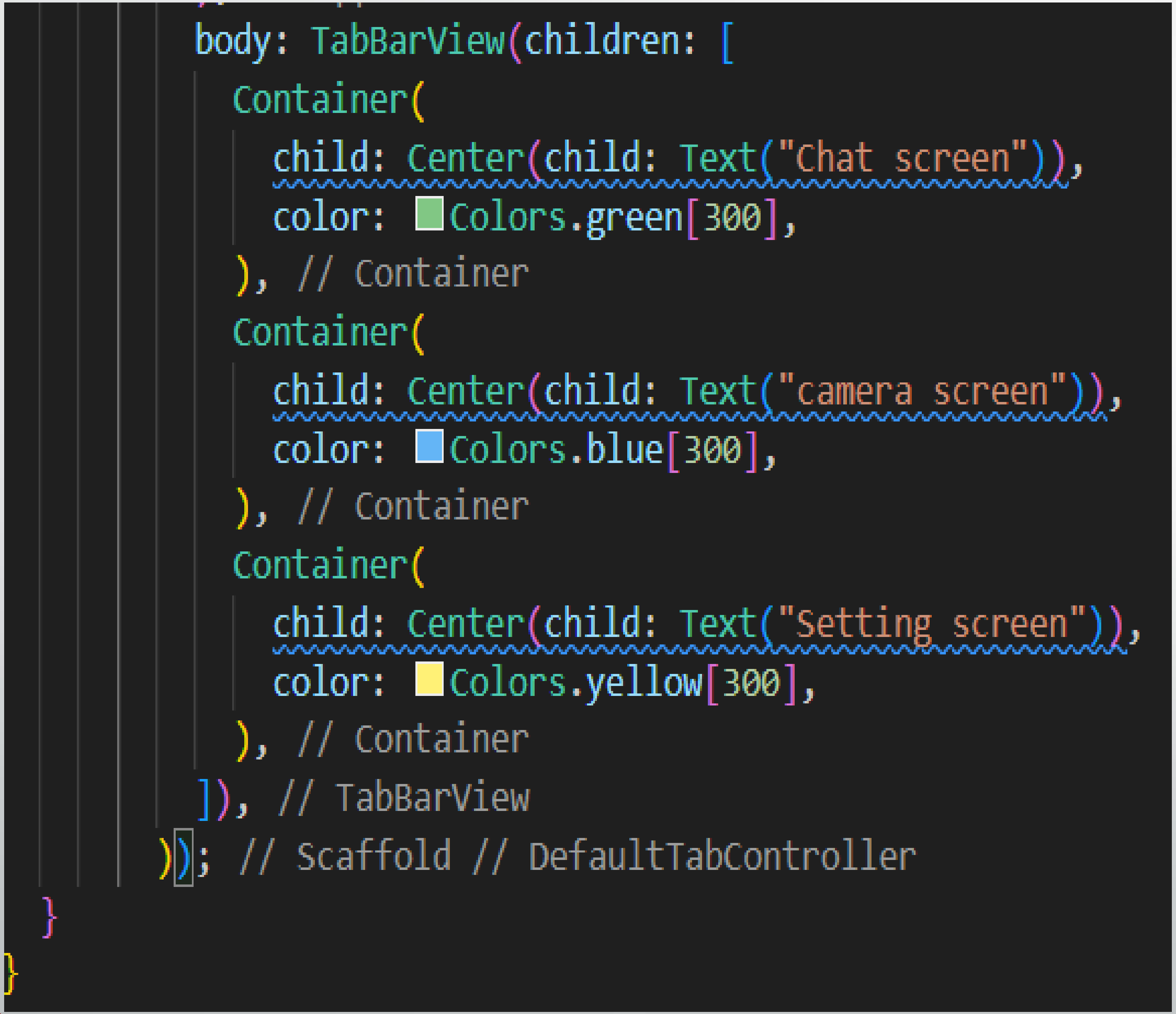
Flutter - Accessibility In Apps

Accessibility makes Flutter apps inclusive and usable for people with disabilities. Employ semantic widgets like Text, Image, and Button for accessibility

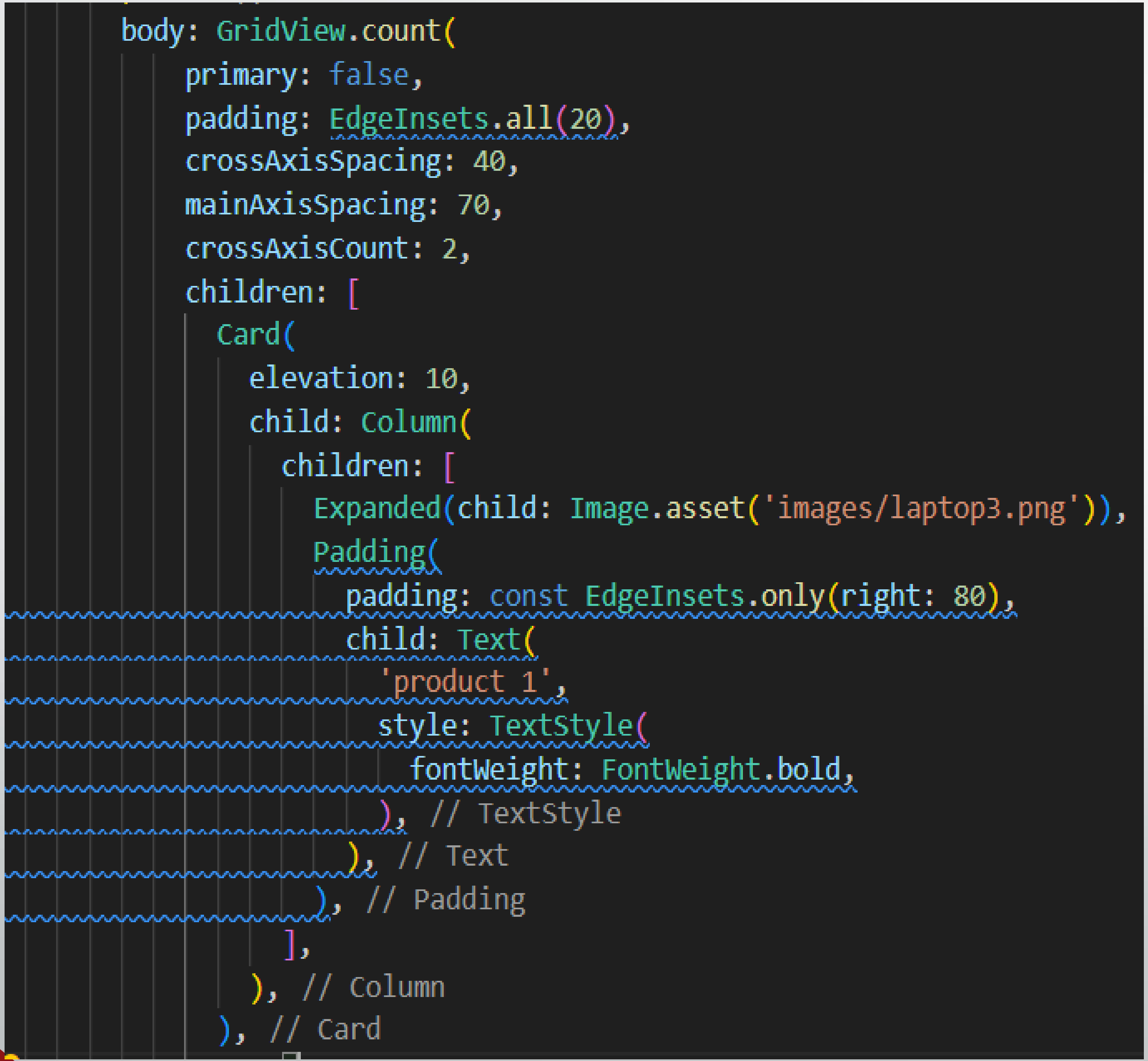


```
class _myexerciseoneState extends State<myexerciseone> {
  @override
  Widget build(BuildContext context) {
    return DefaultTabController(
        length: 3,
        child: Scaffold(
          appBar: AppBar(
            centerTitle: true,
            backgroundColor: Colors.grey,
            bottom: TabBar(
              tabs: [
                Tab(
                  icon: Icon(Icons.message),
                ), // Tab
                Tab(
                  icon: Icon(Icons.camera),
                ), // Tab
                Tab(
                  icon: Icon(Icons.settings),
                ), // Tab
              ],
            ), // TabBar
            title: Text(
              'Home',
            ), // Text
```

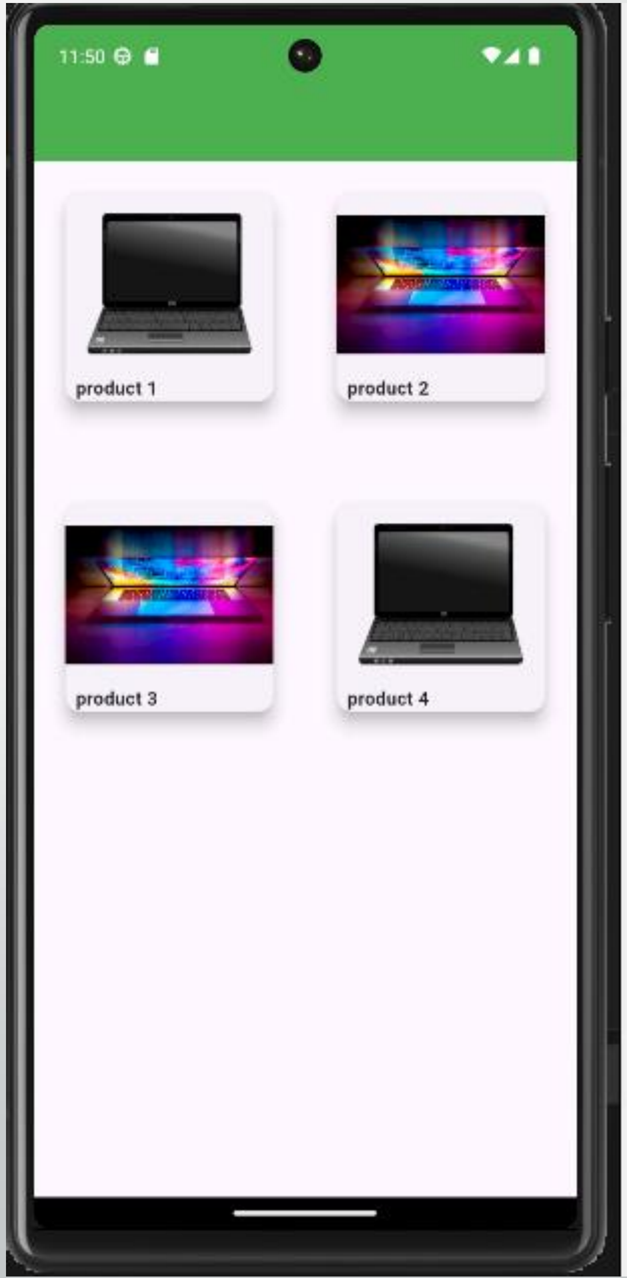




```
          body: TabBarView(children: [
            Container(
              child: Center(child: Text("Chat screen")),
              color: Colors.green[300],
            ), // Container
            Container(
              child: Center(child: Text("camera screen")),
              color: Colors.blue[300],
            ), // Container
            Container(
              child: Center(child: Text("Setting screen")),
              color: Colors.yellow[300],
            ), // Container
          ]), // TabBarView
        )); // Scaffold // DefaultTabController
  }
}
```



```
body: GridView.count(
  primary: false,
  padding: EdgeInsets.all(20),
  crossAxisSpacing: 40,
  mainAxisSpacing: 70,
  crossAxisCount: 2,
  children: [
    Card(
      elevation: 10,
      child: Column(
        children: [
          Expanded(child: Image.asset('images/laptop3.png')),
          Padding(
            padding: const EdgeInsets.only(right: 80),
            child: Text(
              'product 1',
              style: TextStyle(
                fontWeight: FontWeight.bold,
              ), // TextStyle
            ), // Text
          ), // Padding
        ],
      ), // Column
    ), // Card
```



THANK YOU