CCIS كلية علوم الحاسب والمعلومات COLLEGE OF COMPUTER & INFORMATION SCIENCES

Flutter

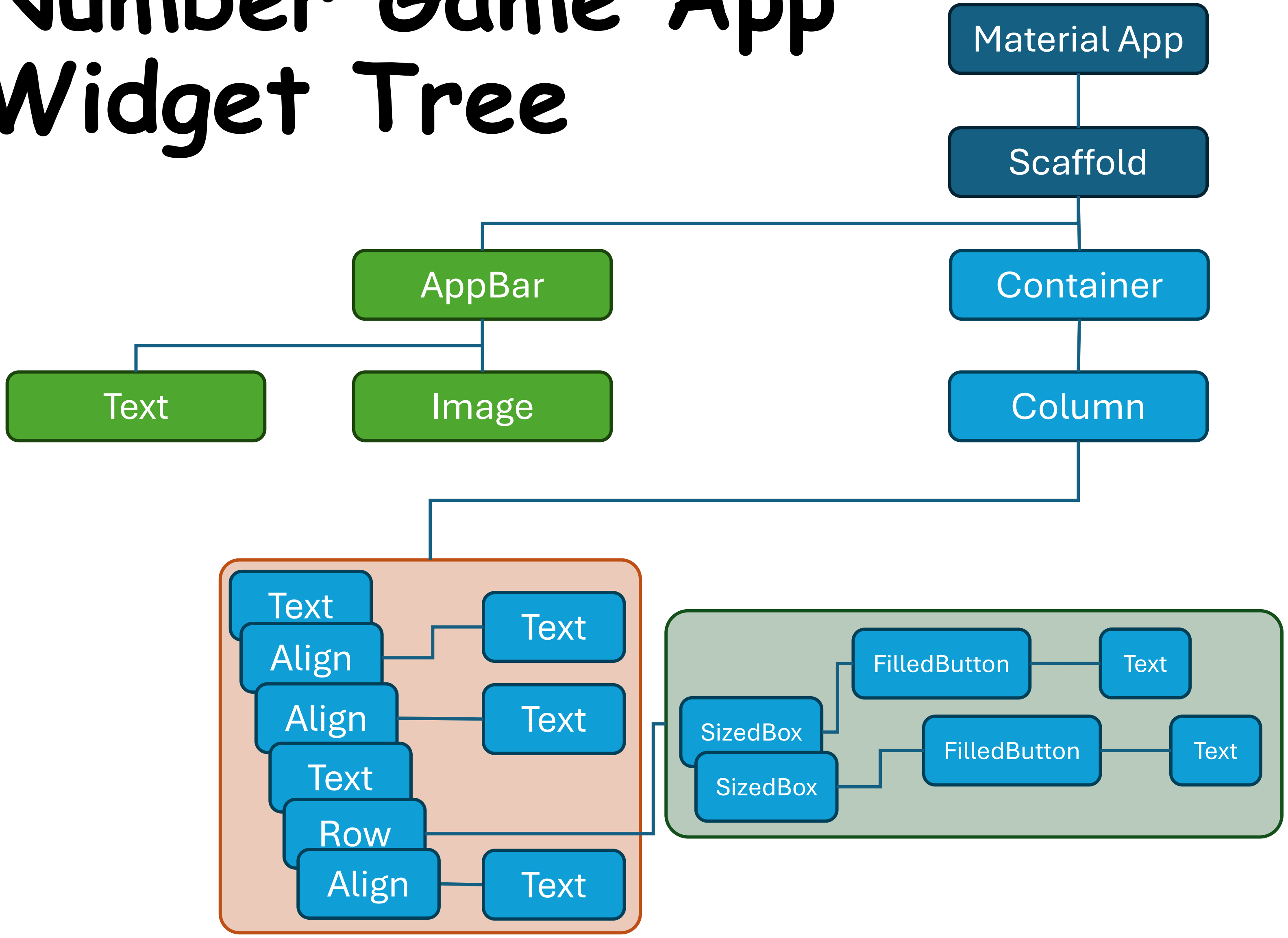
جامعة الأمير سلطان PRINCE SULTAN UNIVERSITY PSU.1998

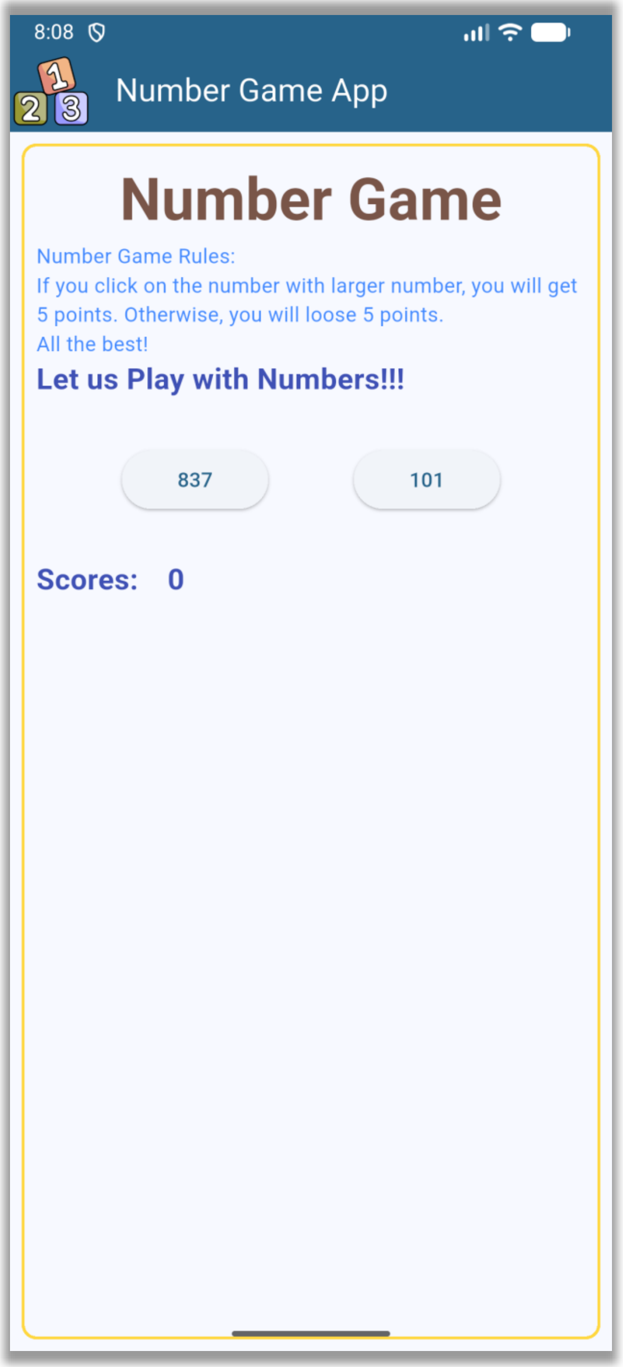
IS487 Emerging Topics in IS

Dr. Abbas Malik

amaalik@psu.edu.sa

Number Game App Widget Tree





Number Game

Number Game Rules:
If you click on the number with larger number, you will get 5 points. Otherwise, you will loose 5 points.
All the best!

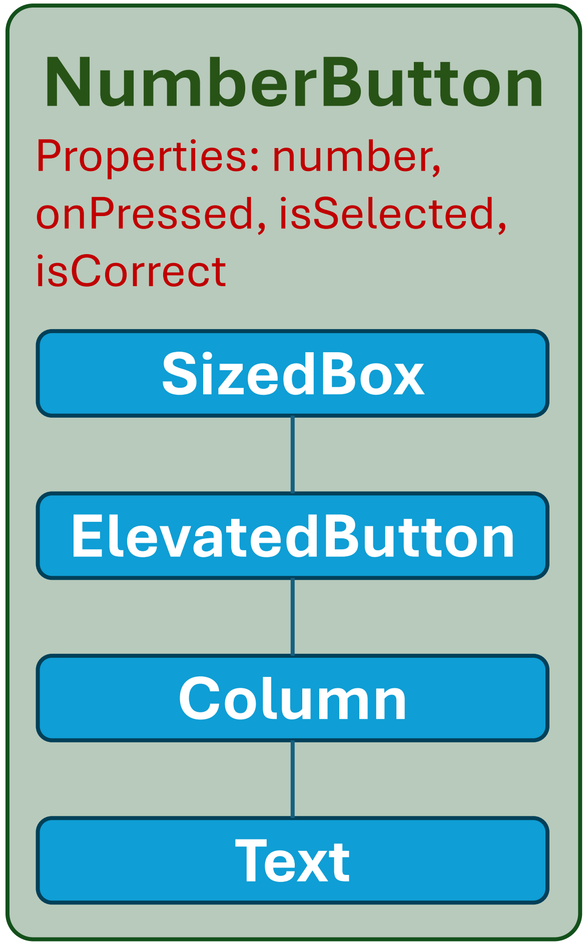
Let us Play with Numbers!!!

Scores: 0

Modular Design

NumberButton

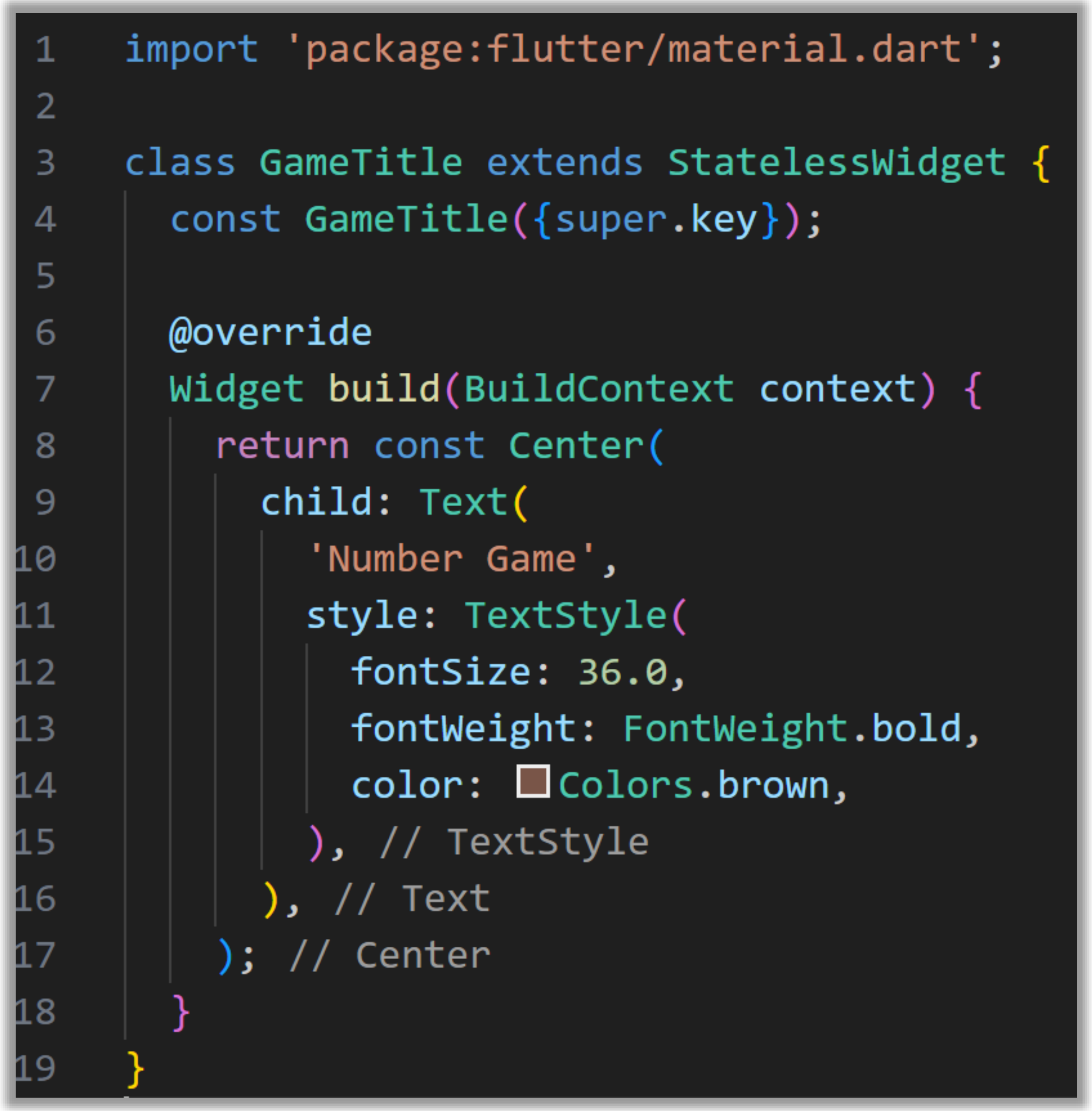
- StatelessWidget
- Write only one time and used twice
- Type of both Left and Right buttons
- State is provided by the Page/Screen



Modular Design

Game Title

- Stateless
- Small and manageable

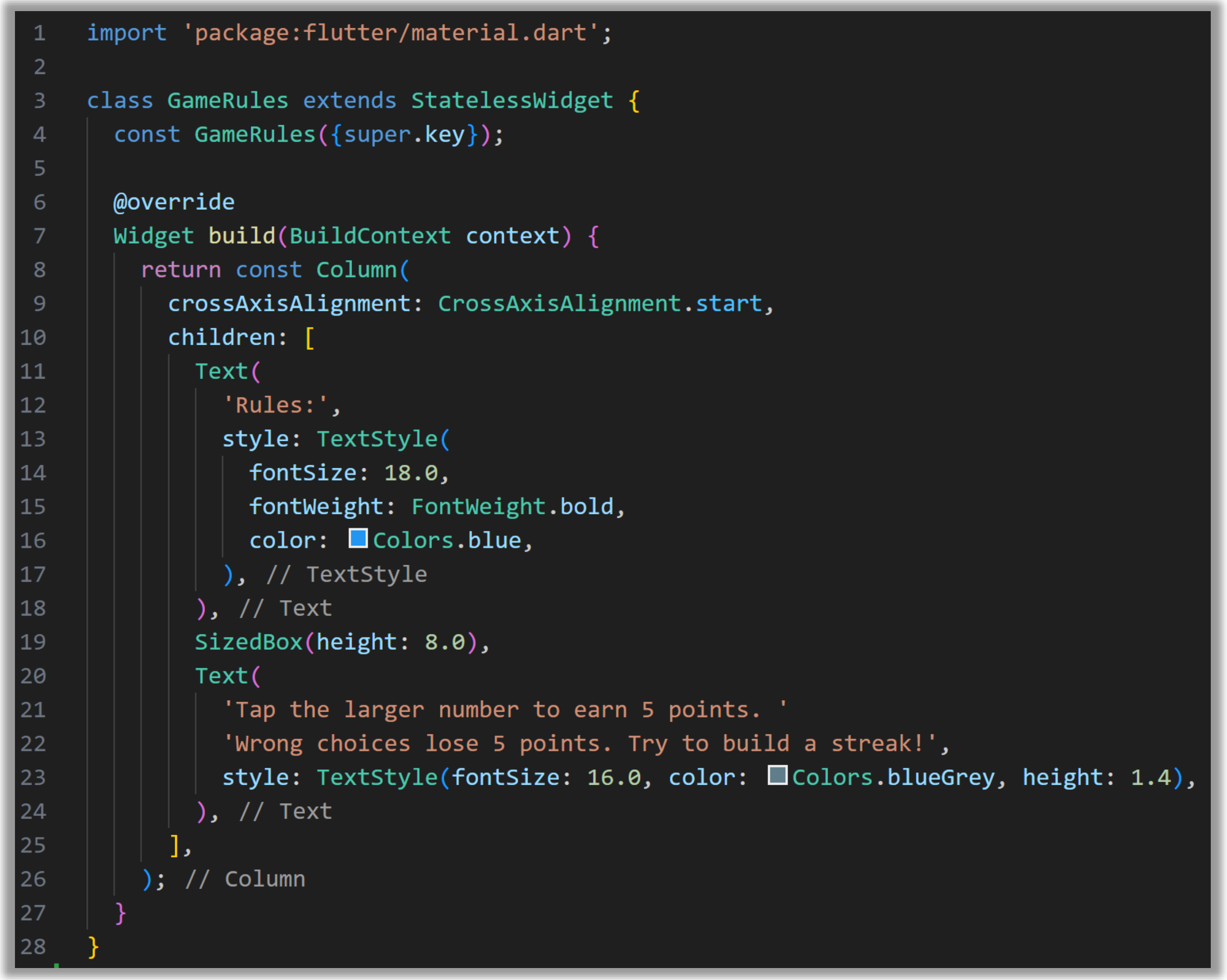


```dart
import 'package:flutter/material.dart';

class GameTitle extends StatelessWidget {
  const GameTitle({super.key});

  @override
  Widget build(BuildContext context) {
    return const Center(
      child: Text(
        'Number Game',
        style: TextStyle(
          fontSize: 36.0,
          fontWeight: FontWeight.bold,
          color: Colors.brown,
        ), // TextStyle
      ), // Text
    ); // Center
  }
}
```

Modular Design

Game Rules

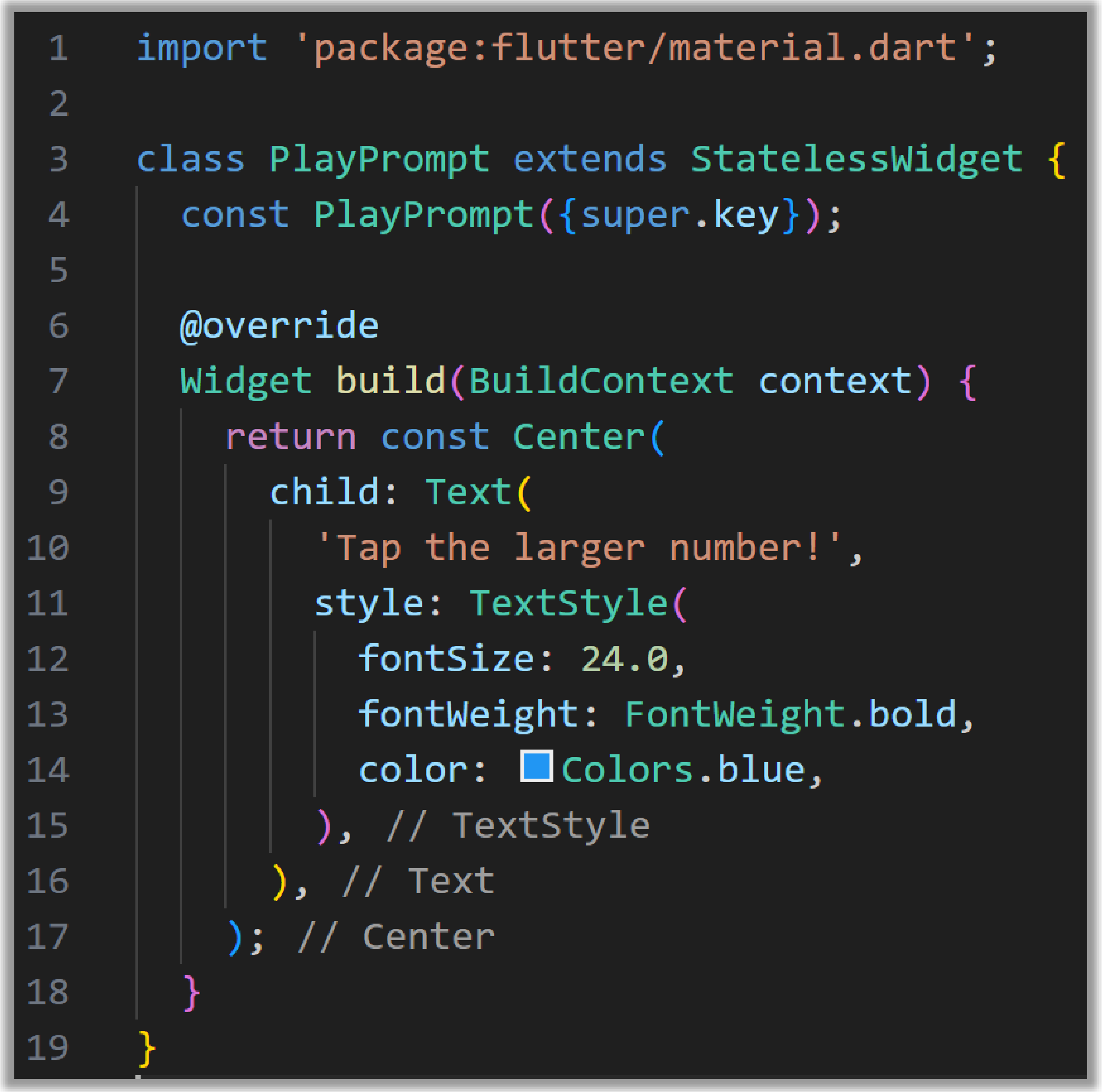
- Stateless
- Small and manageable



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

class GameRules extends StatelessWidget {
  const GameRules({super.key});

  @override
  Widget build(BuildContext context) {
    return const Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Text(
          'Rules:',
          style: TextStyle(
            fontSize: 18.0,
            fontWeight: FontWeight.bold,
            color: Colors.blue,
          ), // TextStyle
        ), // Text
        SizedBox(height: 8.0),
        Text(
          'Tap the larger number to earn 5 points. '
          'Wrong choices lose 5 points. Try to build a streak!',
          style: TextStyle(fontSize: 16.0, color: Colors.blueGrey, height: 1.4),
        ), // Text
      ],
    ); // Column
  }
}
```

Modular Design
Play Prompt

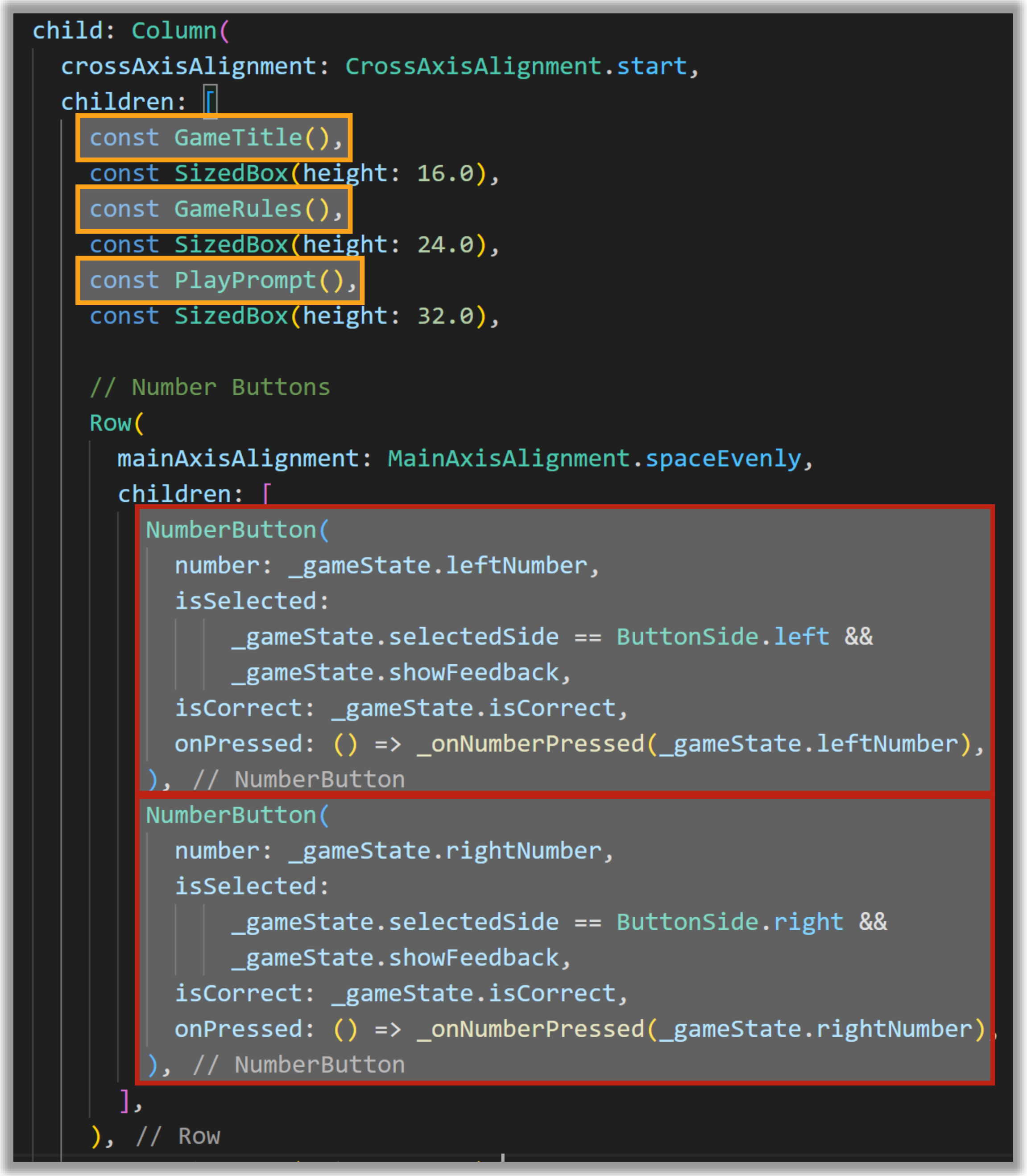
- Stateless
- Small and manageable



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

class PlayPrompt extends StatelessWidget {
  const PlayPrompt({super.key});

  @override
  Widget build(BuildContext context) {
    return const Center(
      child: Text(
        'Tap the larger number!',
        style: TextStyle(
          fontSize: 24.0,
          fontWeight: FontWeight.bold,
          color: 🟦Colors.blue,
        ), // TextStyle
      ), // Text
    ); // Center
  }
}
```

Modular Design
NumberGame Page

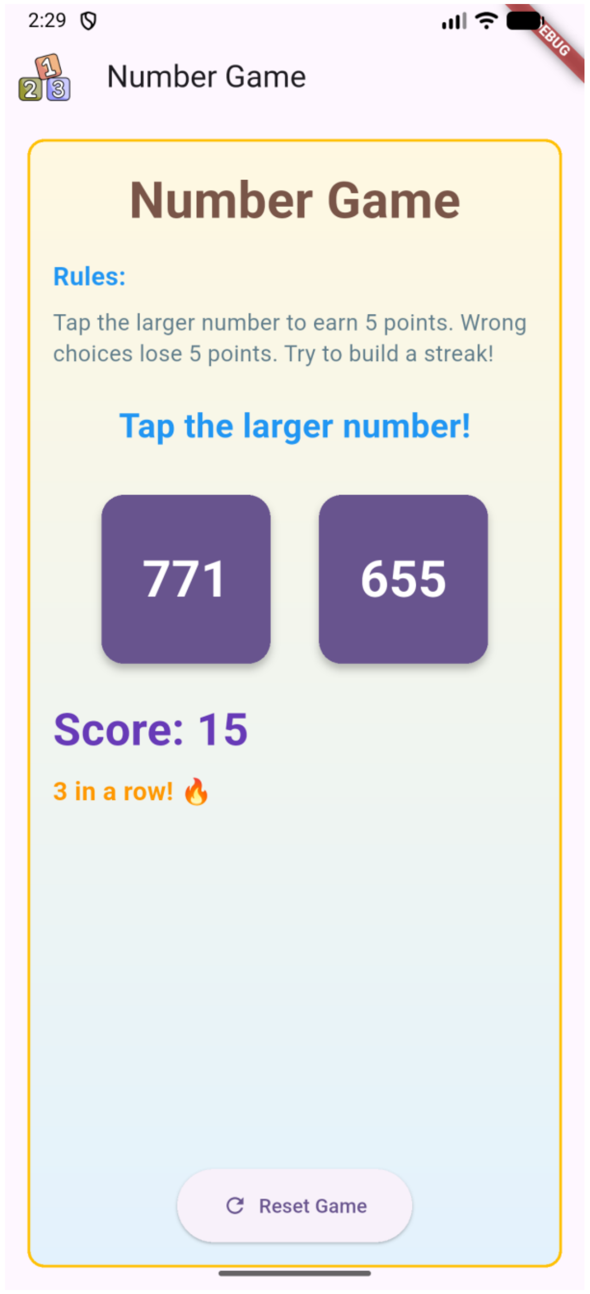
- Simpler Code
- Manageable
- Easily changeable
- Reusing NumberButton



```
child: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    const GameTitle(),
    const SizedBox(height: 16.0),
    const GameRules(),
    const SizedBox(height: 24.0),
    const PlayPrompt(),
    const SizedBox(height: 32.0),

    // Number Buttons
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: [
        NumberButton(
          number: _gameState.leftNumber,
          isSelected:
              _gameState.selectedSide == ButtonSide.left &&
              _gameState.showFeedback,
          isCorrect: _gameState.isCorrect,
          onPressed: () => _onNumberPressed(_gameState.leftNumber),
        ), // NumberButton
        NumberButton(
          number: _gameState.rightNumber,
          isSelected:
              _gameState.selectedSide == ButtonSide.right &&
              _gameState.showFeedback,
          isCorrect: _gameState.isCorrect,
          onPressed: () => _onNumberPressed(_gameState.rightNumber),
        ), // NumberButton
      ],
    ), // Row
```

Modular Design
Game

- flutter_lec3_number_game_modular
- Code



2:29

Number Game

Number Game

Rules:

Tap the larger number to earn 5 points. Wrong choices lose 5 points. Try to build a streak!

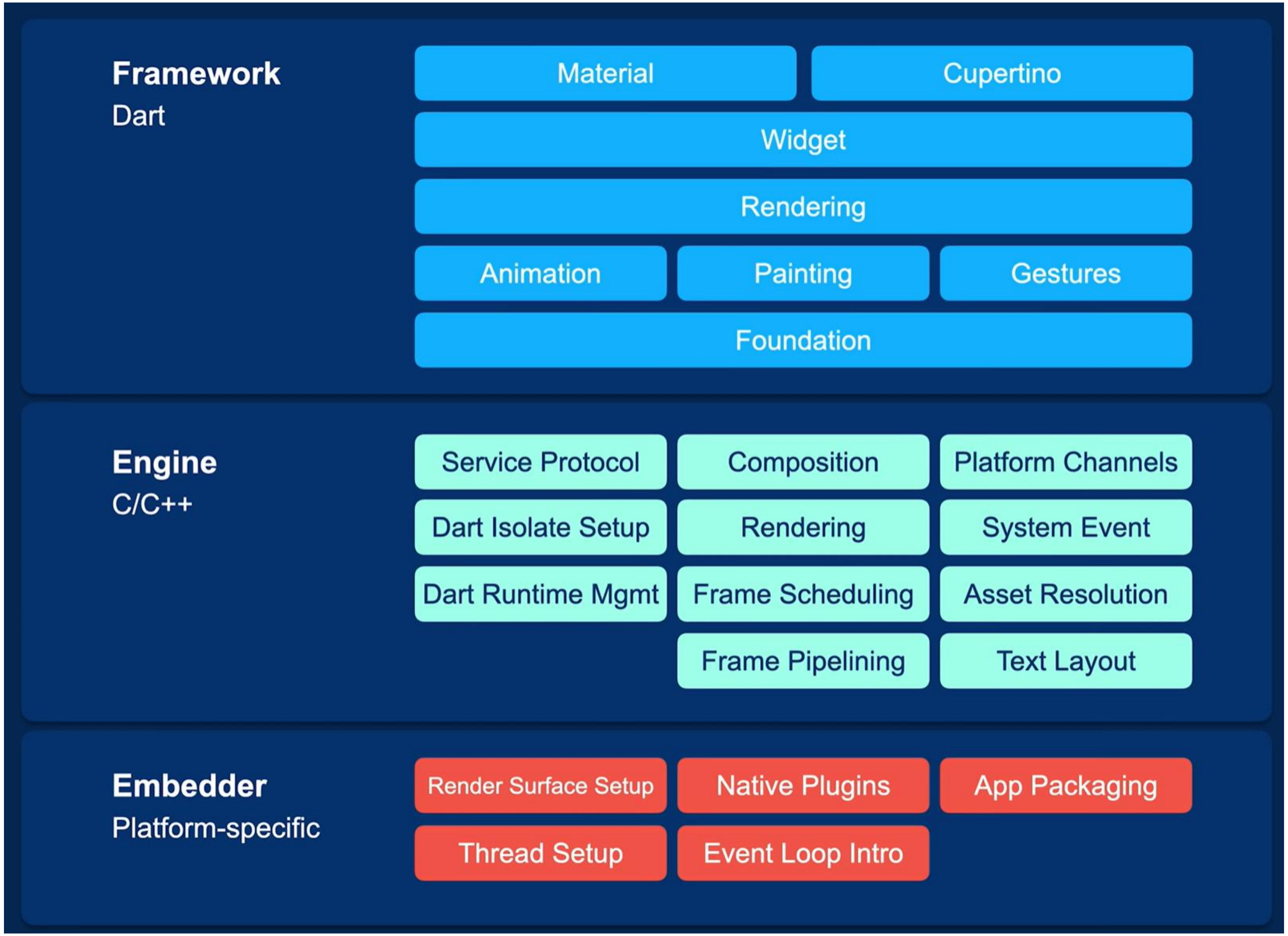
Tap the larger number!

Score: 15

3 in a row! 🔥

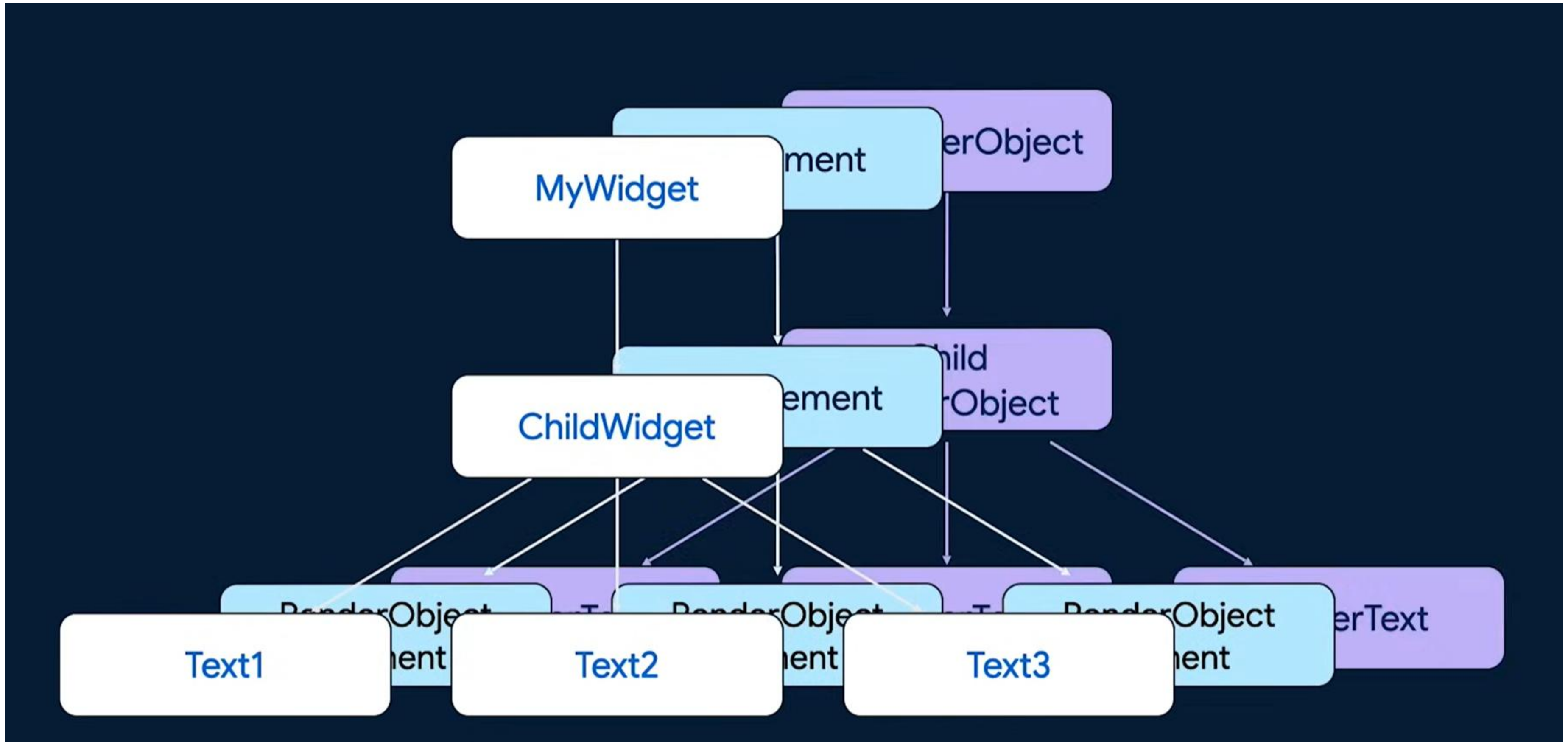
How Flutter Works?

How it works

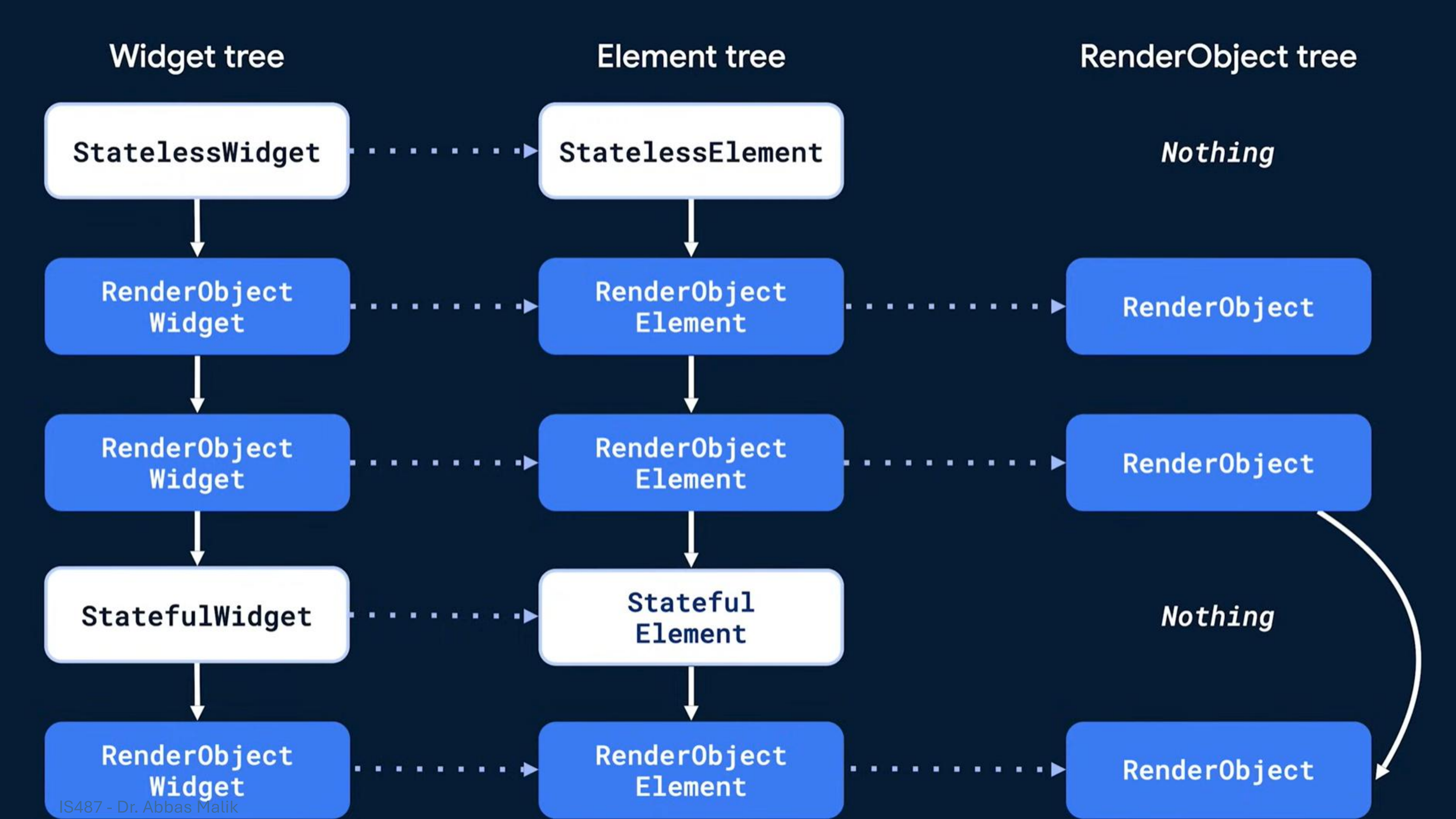

Framework
Dart
Material
Cupertino
Widget
Rendering
Animation
Painting
Gestures
Foundation
Engine
C/C++
Service Protocol
Composition
Platform Channels
Dart Isolate Setup
Rendering
System Event
Dart Runtime Mgmt
Frame Scheduling
Asset Resolution
Frame Pipelining
Text Layout
Embedder
Platform-specific
Render Surface Setup
Native Plugins
App Packaging
Thread Setup
Event Loop Intro

Three Trees

Widget Tree   Element Tree   Render Object Tree



Learn More


Widget tree
Element tree
RenderObject tree
StatelessWidget
StatelessElement
Nothing
RenderObject Widget
RenderObject Element
RenderObject
RenderObject Widget
RenderObject Element
RenderObject
StatefulWidget
Stateful Element
Nothing
RenderObject Widget
RenderObject Element
RenderObject

States

Ephemeral State Vs. **App State**

State contain in a single widget

- Current Page
- Progress of complex animation
- Current selected tab

Application state

- User preferences
- Authentication info
- Notifications
- Shopping cart info
- Read/Unread info

Widget Lifecycle and Methods

initState()

Rendering Widget
didChangeDependencies()
didUpdateWidget()
Build()

dispose()

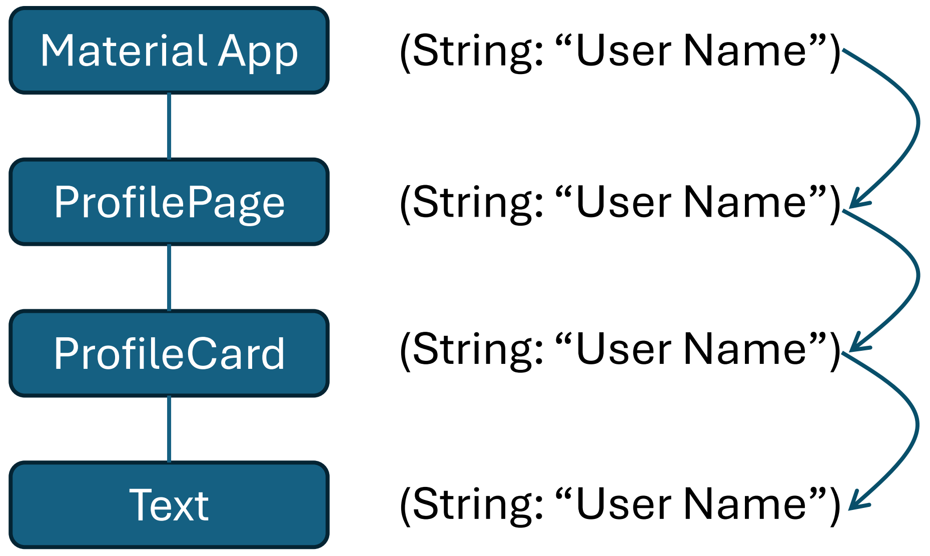
How Flutter Works

Learn More

Stateful Widget

Passing Data Down Tree

- Not a good way to pass data from top of tree down to the widget that need the data
- Need to find a better way to do it.



Inherited Widgets

Themes

Theme ThemeData

InheritedWidget

- Efficiently propagate data down the widget tree

```
theme: ThemeData(colorScheme: .fromSeed(seedColor: Colors.deepPurple)),
```

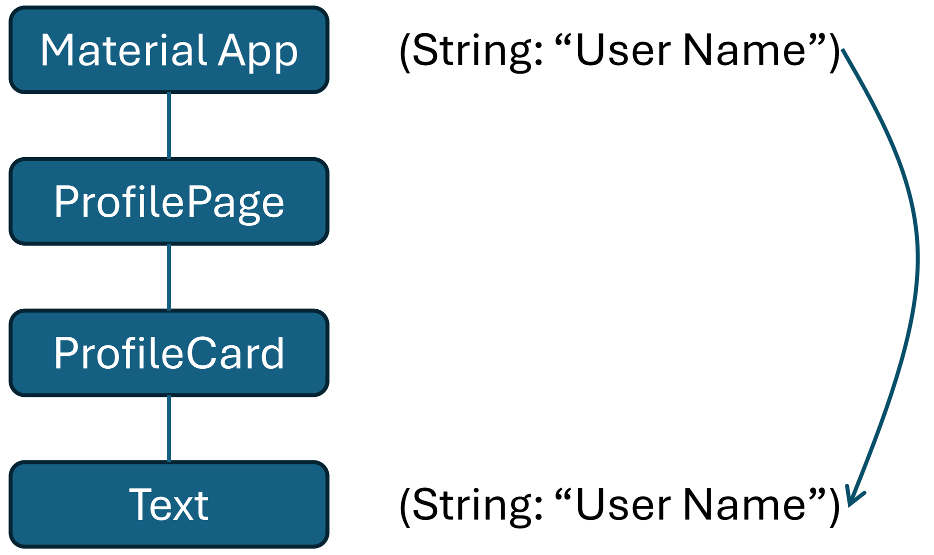
```
Color backgroundColor = Theme.of(context).colorScheme.primary;
```

```
width: MediaQuery.of(context).size.width,
```

MediaQuery

Inherited Widgets

- Pass data where it is needed



Inherited Widgets

- Two ways to create Inherited Widget

1

Direct: Create a child of InheritedWidget class and implement required methods

2

Package: use provider package to make an InheritedWidget.

Counter App with InheritedWidget

Direct

@imutable class

```
class MyCounter extends InheritedWidget {
  int _count = 0;
  int get count => _count;
  MyCounter({Key? key, required Widget child}) : super(key: key, child: child);
  static MyCounter of(BuildContext context) {
    return context.dependOnInheritedWidgetOfExactType<MyCounter>()!;
  }
  @override
  bool updateShouldNotify(MyCounter oldWidget) {
    return oldWidget.count != count;
  }
}
```

Counter App with InheritedWidget

Direct

- Pair it with a StatefulWidget.
- StatefulWidget maintain the state of data
- While InheritedWidget is recreated everytimg data is changed
- More Complex: InheritedNotifier combines InheritedWidget and ChangeNotifier

Counter App with InheritedWidget

Provider Package

- No need to reinvent the wheel, just use it

1. Create a new project
2. Add provider package in your project

```
flutter pub add provider
```

Counter App with InheritedWidget

3. Create a class to hold data with ChangeNotifier

```
import 'package:flutter/material.dart';
class ProviderCount extends ChangeNotifier {
  int _count = 0;
  int get count => _count;

  void incrementCount() {
    _count++;
    notifyListeners();
  }
}
```

Counter App with InheritedWidget

4. Change main() method

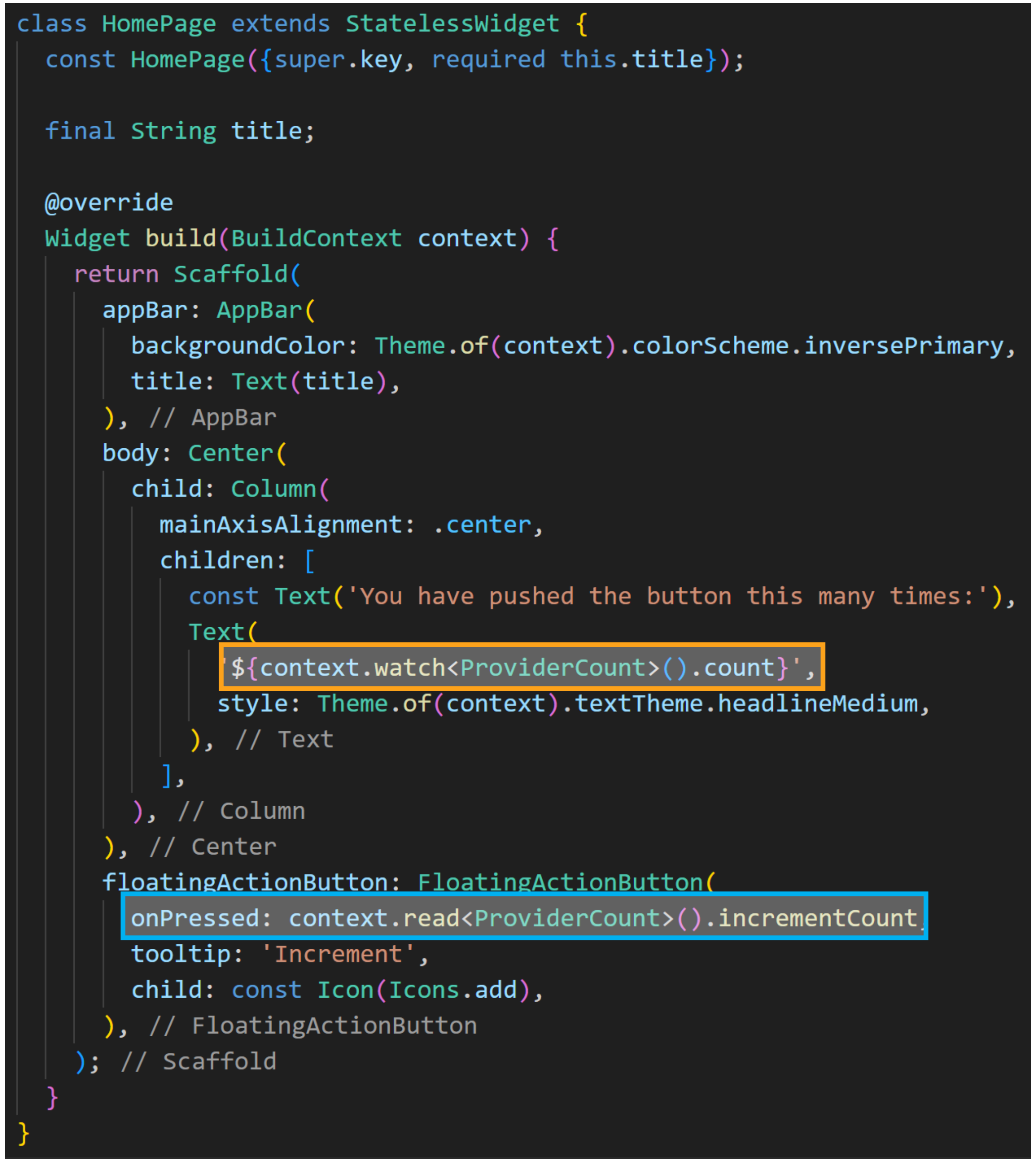
```
void main() {
  runApp(
    MultiProvider(
      providers: [ChangeNotifierProvider(create: (_) => ProviderCount())],
      child: const MyApp(),
    ),
  );
}
```

Counter App with InheritedWidget

5. Create a StatelessWidget HomePage

```
Provider.of<ProviderCount>(
  context,
  listen: true
).count
```

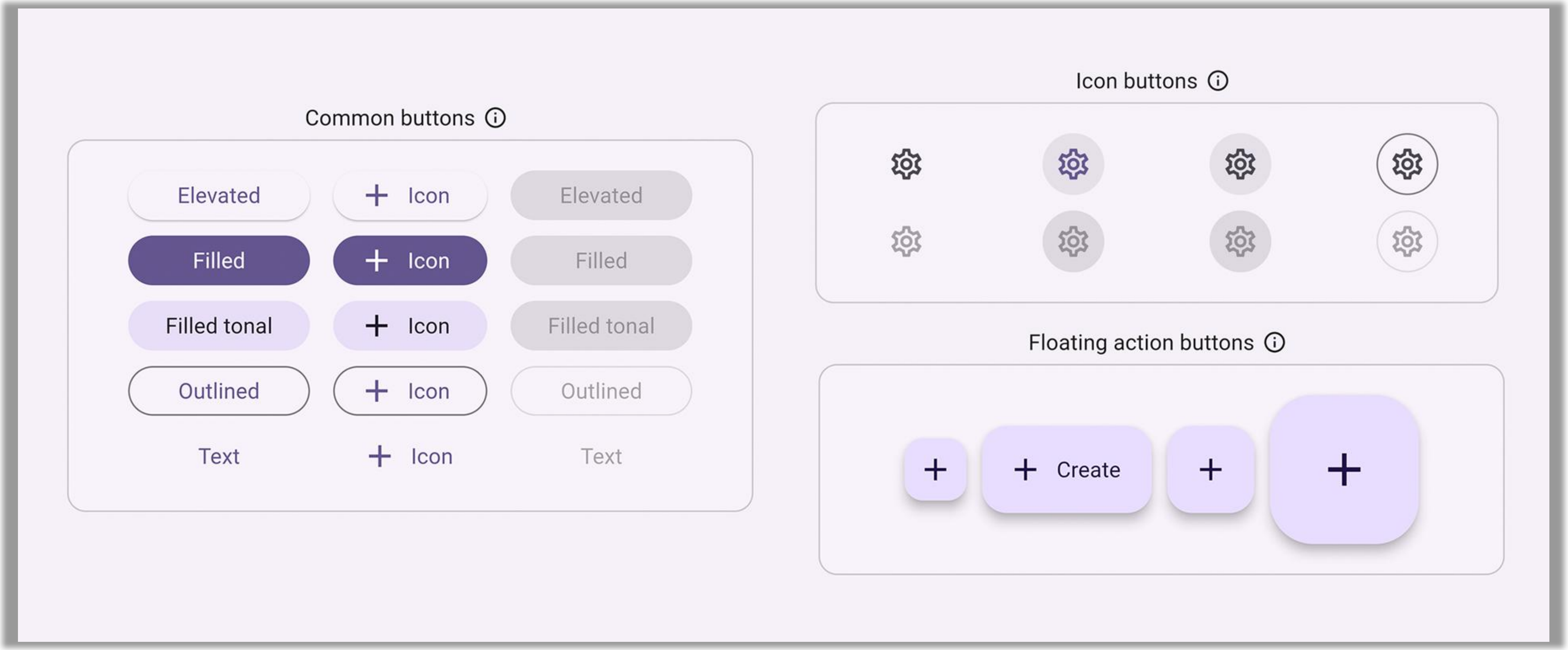
```
onPressed: Provider.of<ProviderCount>(
  context,
  listen: false,
).incrementCount,
```



```
class HomePage extends StatelessWidget {
  const HomePage({super.key, required this.title});

  final String title;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(title),
      ), // AppBar
      body: Center(
        child: Column(
          mainAxisAlignment: .center,
          children: [
            const Text('You have pushed the button this many times:'),
            Text(
              '${context.watch<ProviderCount>().count}',
              style: Theme.of(context).textTheme.headlineMedium,
            ), // Text
          ],
        ), // Column
      ), // Center
      floatingActionButton: FloatingActionButton(
        onPressed: context.read<ProviderCount>().incrementCount,
        tooltip: 'Increment',
        child: const Icon(Icons.add),
      ), // FloatingActionButton
    ); // Scaffold
  }
}
```

User Input

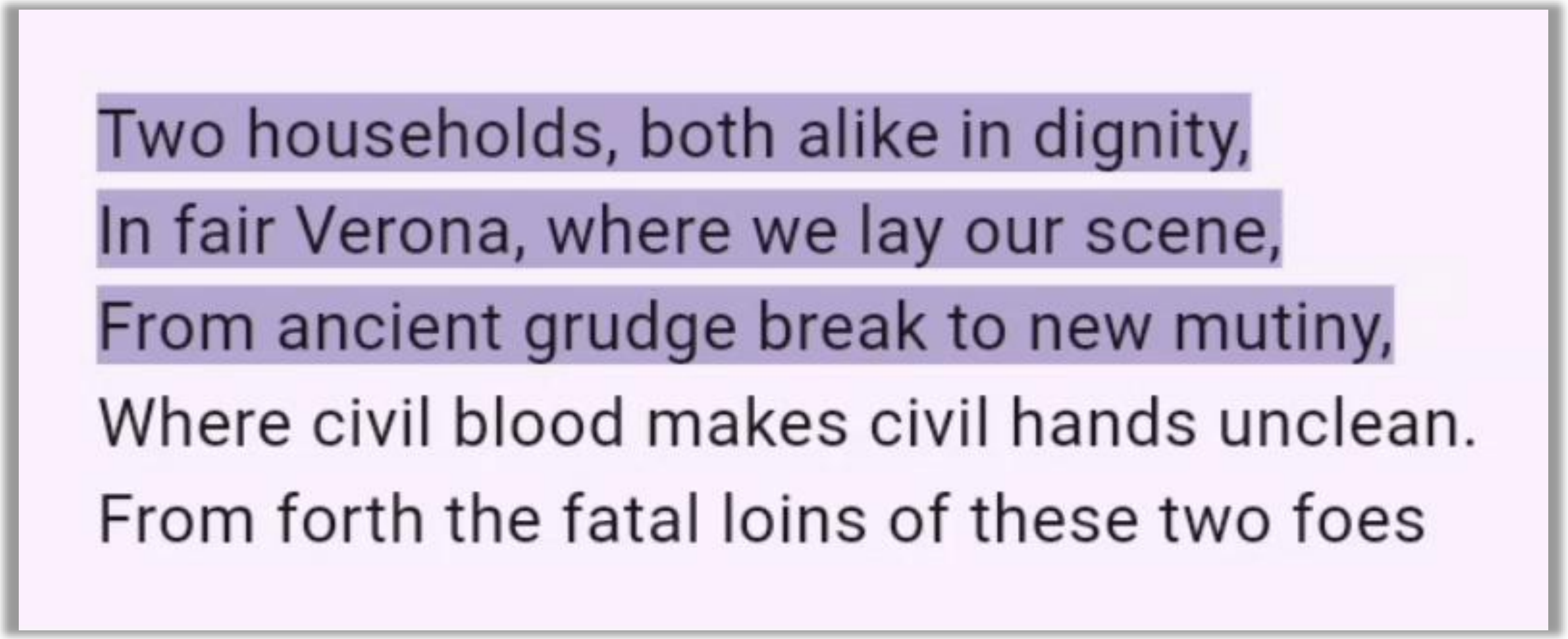
Buttons


Common buttons
Elevated
Icon
Elevated
Filled
Icon
Filled
Filled tonal
Icon
Filled tonal
Outlined
Icon
Outlined
Text
Icon
Text
Icon buttons
Floating action buttons
Create

SelectableText

```
SelectableText('''
Two households, both alike in dignity,
In fair Verona, where we lay our scene,
From ancient grudge break to new mutiny,
Where civil blood makes civil hands unclean.
From forth the fatal loins of these two
foes''')
```

- User can select the text


Two households, both alike in dignity,
In fair Verona, where we lay our scene,
From ancient grudge break to new mutiny,
Where civil blood makes civil hands unclean.
From forth the fatal loins of these two foes

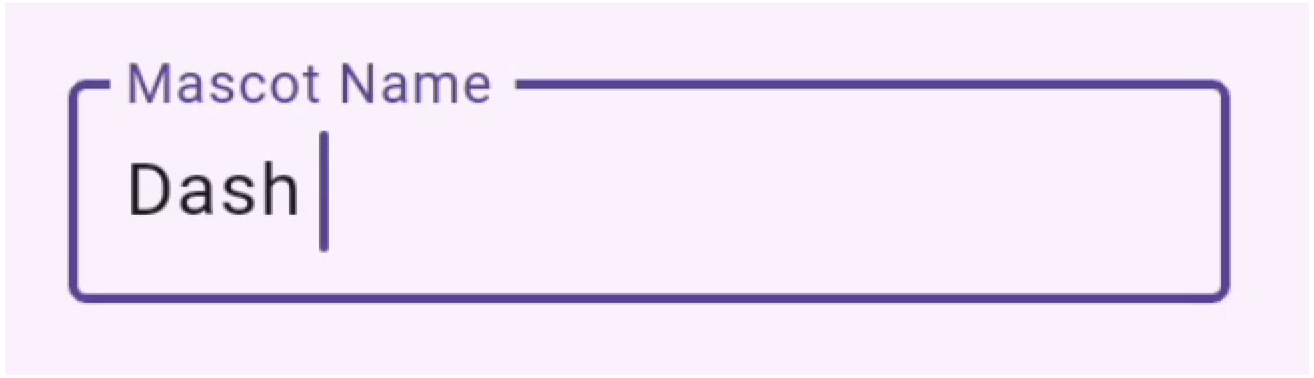
Formated Text - Rich Text

- RichText
- TextSpan

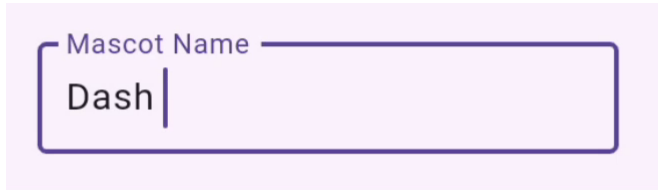
https://github.com/flutter/samples/tree/main/simplistic_editor

TextField - Text Input

- Let user enter text in text box using a hardware or onscreen keyboard

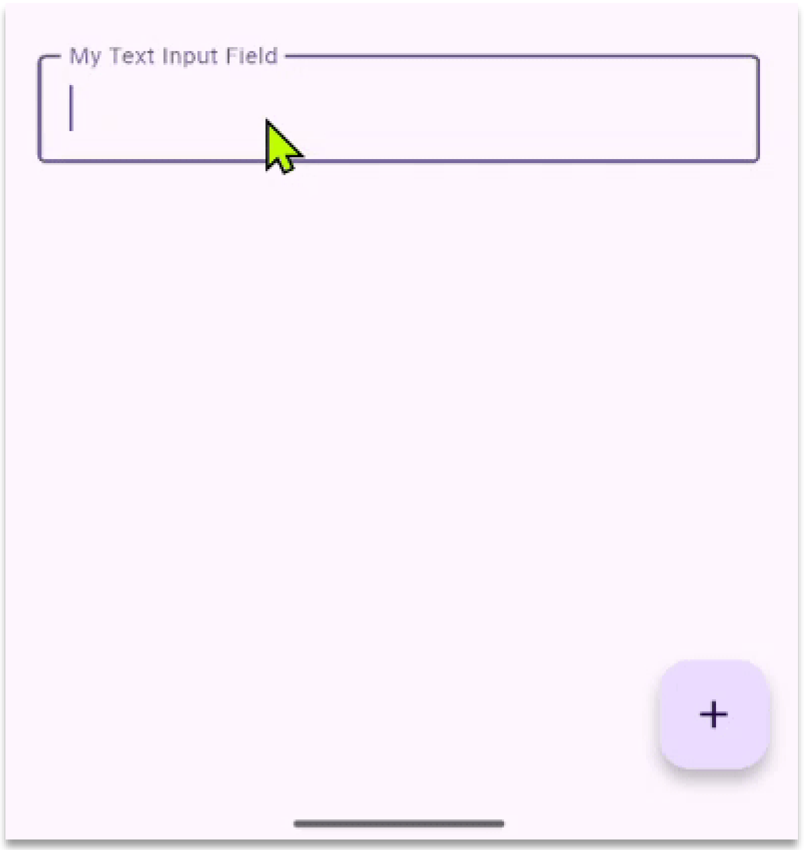


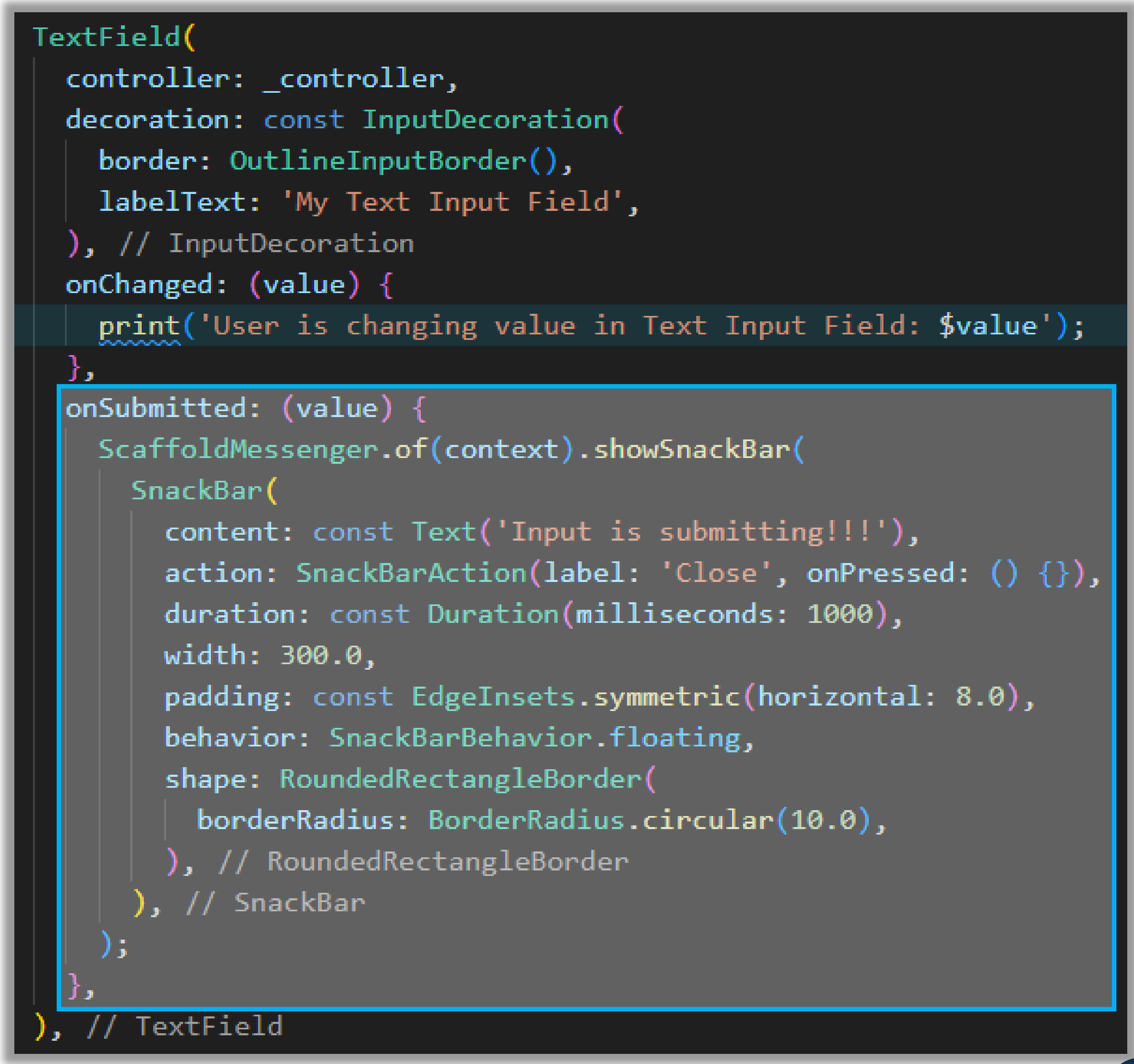
TextField - Text Input



- decoration: InputDecoration - determines text field's appearance
- controller: TextEditingController - control text input programmatically
- onChanged: callback triggers when user changes value in the text field
- onSubmitted: callback triggers indicating user is done editing, e.g. tapping the enter key

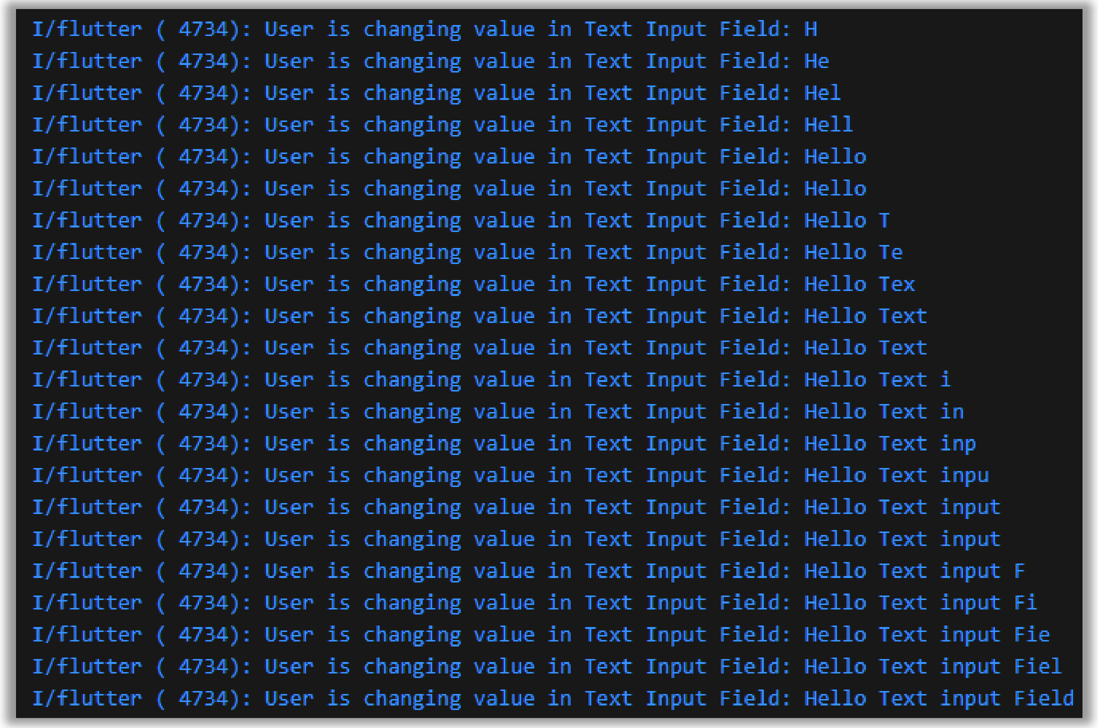
TextField Example



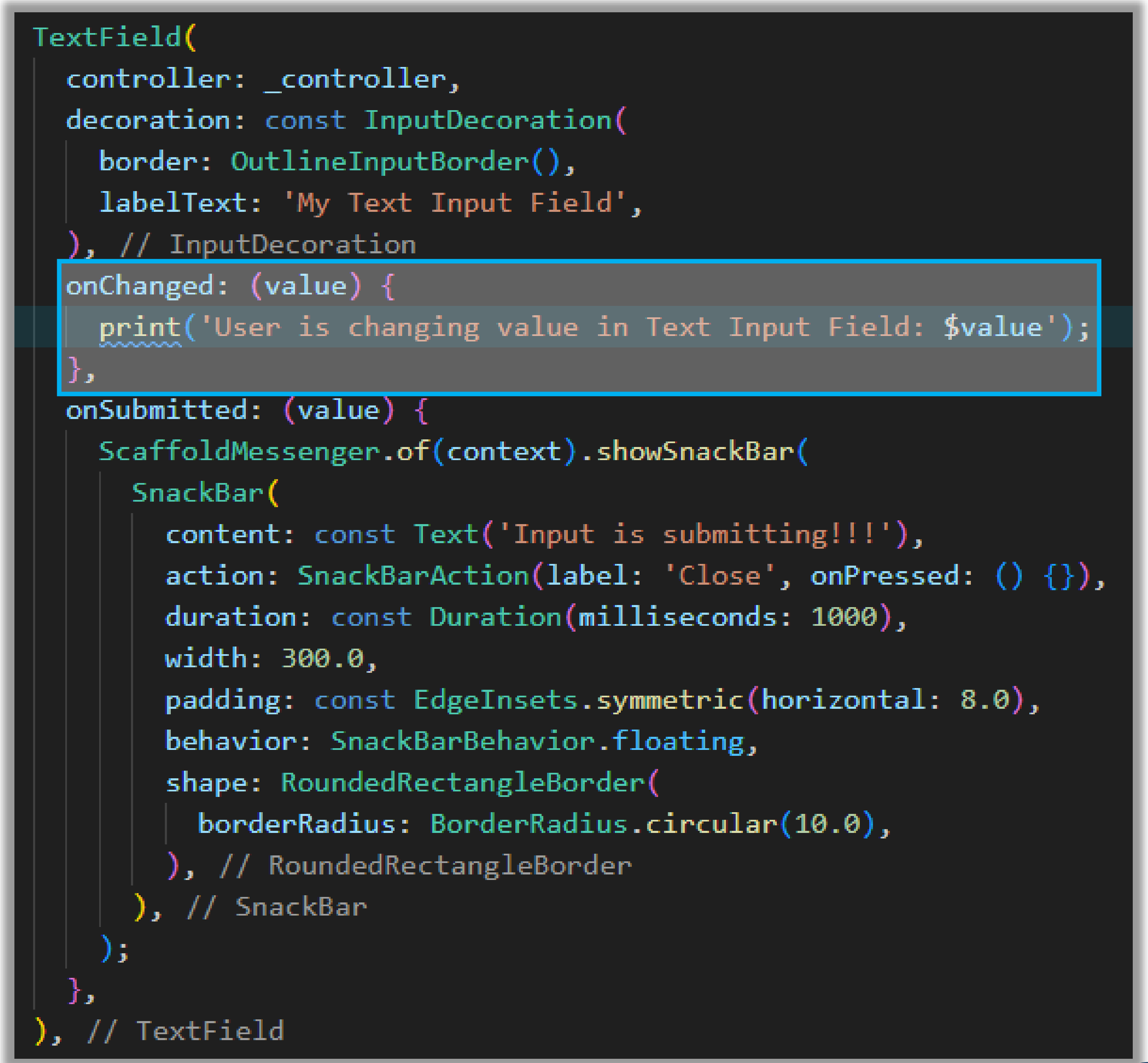


```
TextField(
  controller: _controller,
  decoration: const InputDecoration(
    border: OutlineInputBorder(),
    labelText: 'My Text Input Field',
  ), // InputDecoration
  onChanged: (value) {
    print('User is changing value in Text Input Field: $value');
  },
  onSubmitted: (value) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: const Text('Input is submitting!!!'),
        action: SnackBarAction(label: 'Close', onPressed: () {}),
        duration: const Duration(milliseconds: 1000),
        width: 300.0,
        padding: const EdgeInsets.symmetric(horizontal: 8.0),
        behavior: SnackBarBehavior.floating,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10.0),
        ), // RoundedRectangleBorder
      ), // SnackBar
    );
  },
), // TextField
```

TextField Example



```
I/flutter ( 4734): User is changing value in Text Input Field: H
I/flutter ( 4734): User is changing value in Text Input Field: He
I/flutter ( 4734): User is changing value in Text Input Field: Hel
I/flutter ( 4734): User is changing value in Text Input Field: Hell
I/flutter ( 4734): User is changing value in Text Input Field: Hello
I/flutter ( 4734): User is changing value in Text Input Field: Hello
I/flutter ( 4734): User is changing value in Text Input Field: Hello T
I/flutter ( 4734): User is changing value in Text Input Field: Hello Te
I/flutter ( 4734): User is changing value in Text Input Field: Hello Tex
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text i
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text in
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text inp
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text inpu
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input F
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input Fi
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input Fie
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input Fiel
I/flutter ( 4734): User is changing value in Text Input Field: Hello Text input Field
```



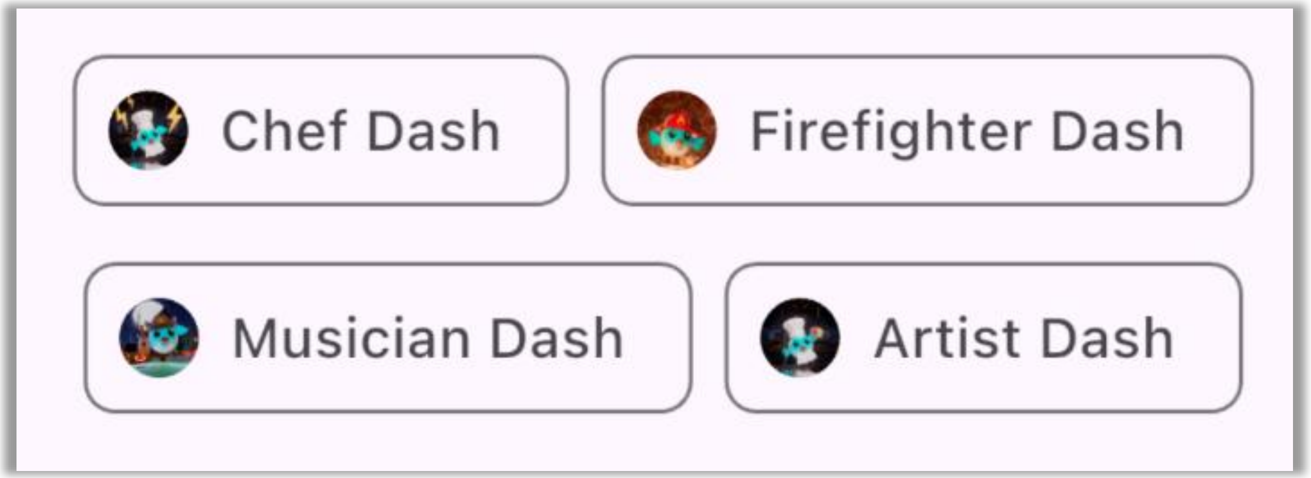
```dart
TextField(
  controller: _controller,
  decoration: const InputDecoration(
    border: OutlineInputBorder(),
    labelText: 'My Text Input Field',
  ), // InputDecoration
  onChanged: (value) {
    print('User is changing value in Text Input Field: $value');
  },
  onSubmitted: (value) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: const Text('Input is submitting!!!'),
        action: SnackBarAction(label: 'Close', onPressed: () {}),
        duration: const Duration(milliseconds: 1000),
        width: 300.0,
        padding: const EdgeInsets.symmetric(horizontal: 8.0),
        behavior: SnackBarBehavior.floating,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10.0),
        ), // RoundedRectangleBorder
      ), // SnackBar
    );
  },
), // TextField
```

Form - Container

- Optional container
- Group multiple form fields widgets
- Each field should be wrapped in FormField
  - Pre-wrap form field widgets, e.g. TextFormField

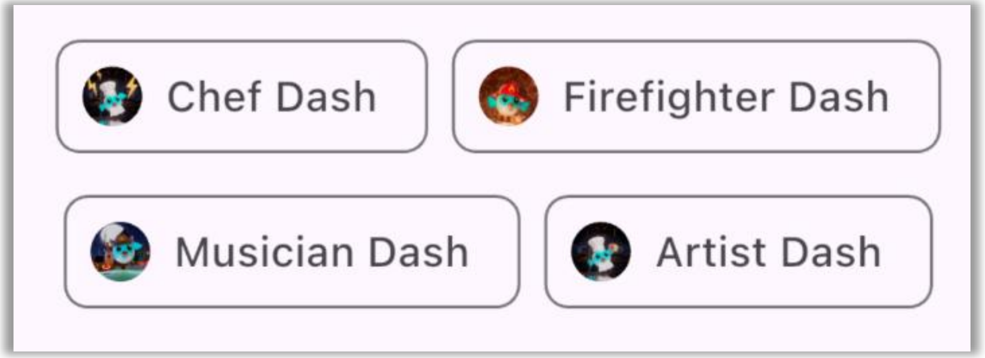
https://github.com/flutter/samples/tree/main/form_app

Chip



- InputChip - complex piece of information
- ChoiceChip - Single choice among multiple options
- FilterChip - uses tags to filter content
- ActionChip - represents action related to primary content

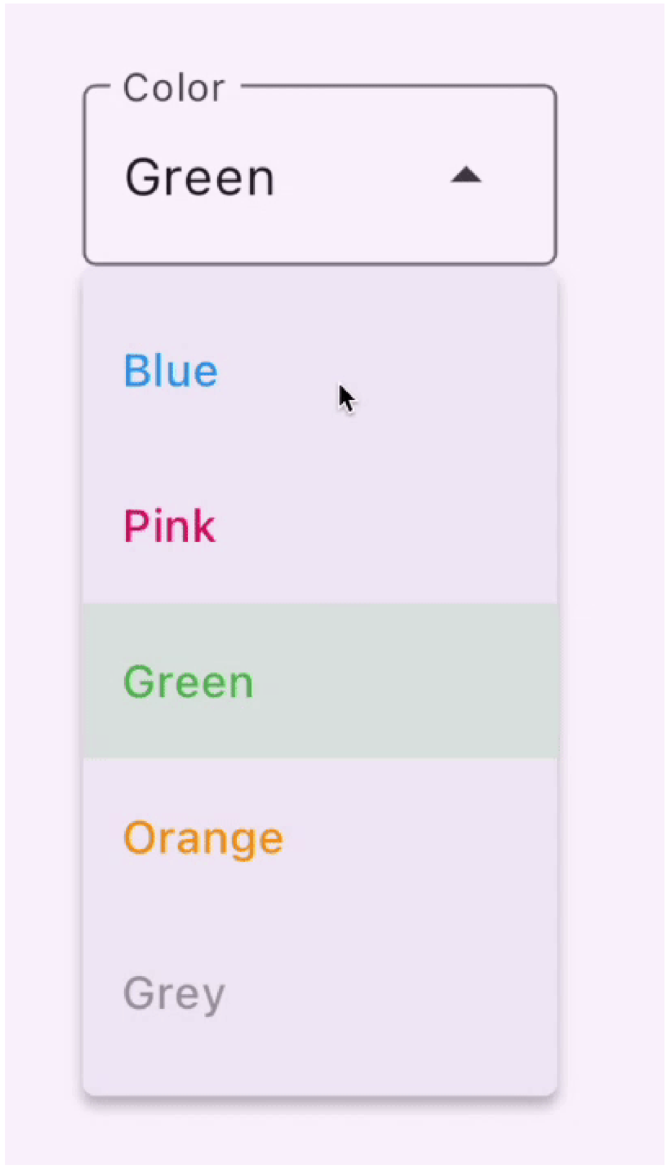
Chip



- label: require property
- avatar: optional property to show icon or a picture
- onDeleted: callback which show a delete icon when triggered delete the chip
- More customization: shape, color, & iconTheme

DropdownMenu

- Allow user to select a choice from a menu of options
- Places selected choice in TextField
- Can help filter contents



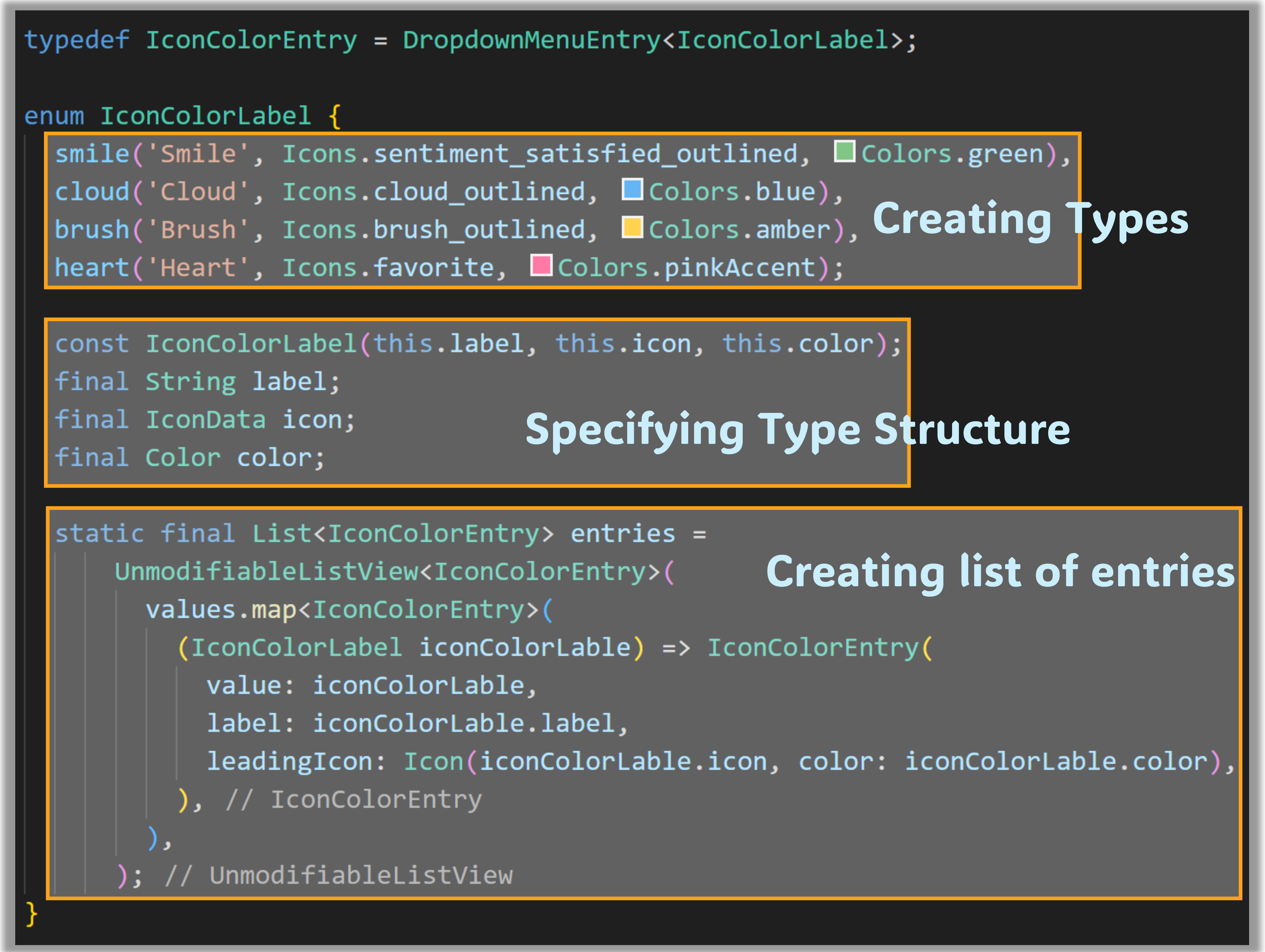
DropdownMenu

- dropdownMenuEntries provides a list of DropdownMenuEntrys that describes each menu item
- controler: TextEditingController
- onSelected: callback triggers when the user selects an option
- initialSelection: Configure the default value.

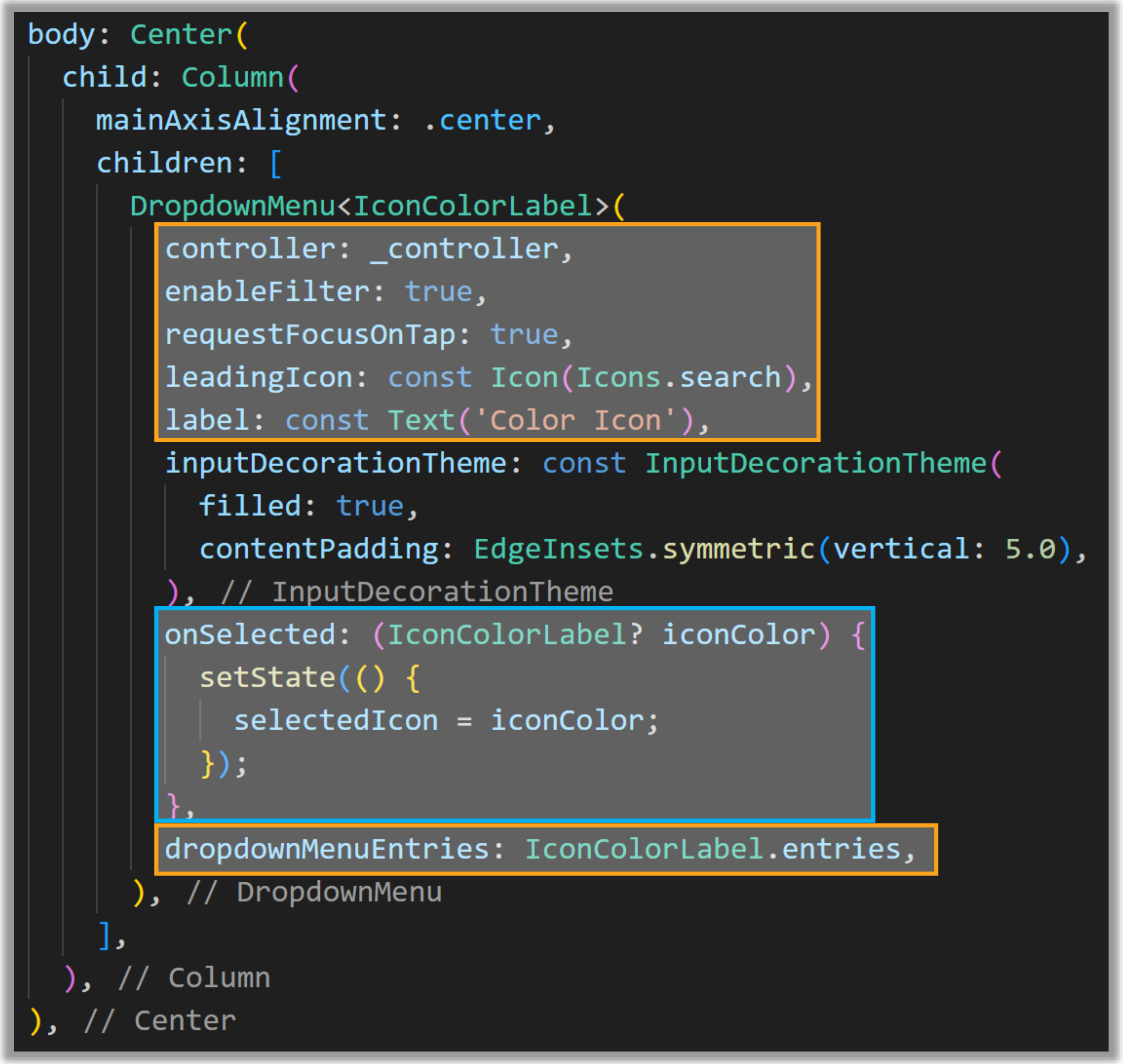
DropdownMenu

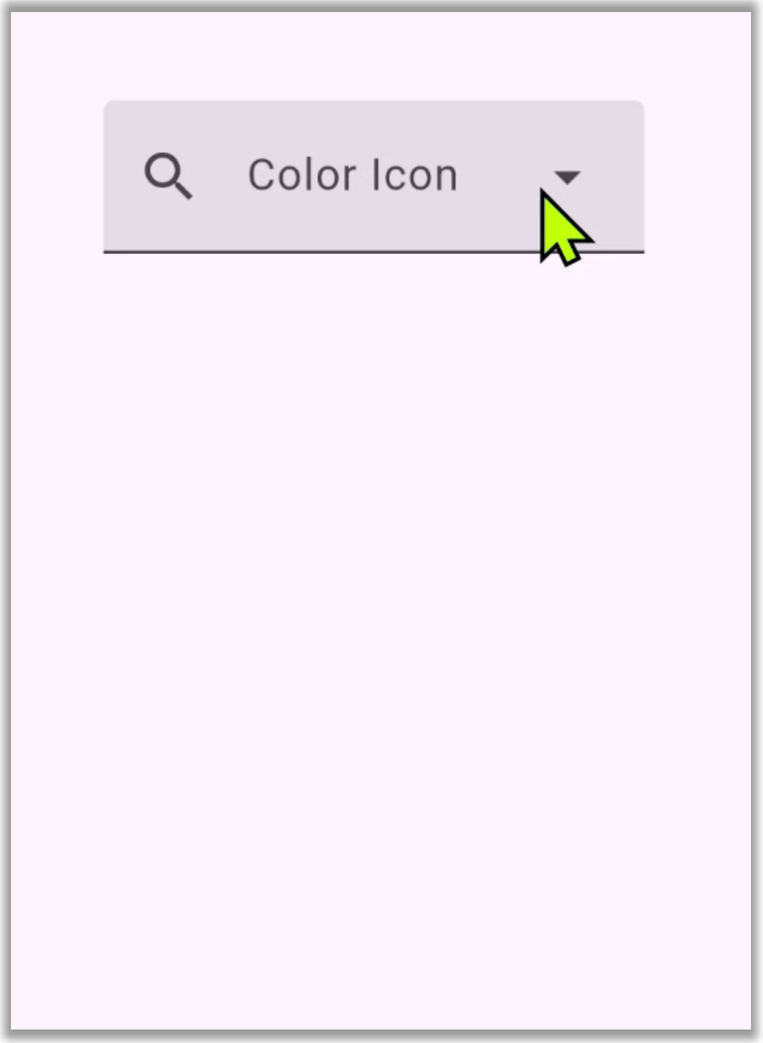
- DropdownMenuEntry

```
typedef IconColorEntry = DropdownMenuEntry<IconColorLabel>;
```



```dart
typedef IconColorEntry = DropdownMenuEntry<IconColorLabel>;

enum IconColorLabel {
  smile('Smile', Icons.sentiment_satisfied_outlined, Colors.green),
  cloud('Cloud', Icons.cloud_outlined, Colors.blue),
  brush('Brush', Icons.brush_outlined, Colors.amber),
  heart('Heart', Icons.favorite, Colors.pinkAccent);

  const IconColorLabel(this.label, this.icon, this.color);
  final String label;
  final IconData icon;
  final Color color;

  static final List<IconColorEntry> entries =
      UnmodifiableListView<IconColorEntry>(
        values.map<IconColorEntry>(
          (IconColorLabel iconColorLable) => IconColorEntry(
            value: iconColorLable,
            label: iconColorLable.label,
            leadingIcon: Icon(iconColorLable.icon, color: iconColorLable.color),
          ), // IconColorEntry
        ),
      ); // UnmodifiableListView
}
```


```
body: Center(
  child: Column(
    mainAxisAlignment: .center,
    children: [
      DropdownMenu<IconColorLabel>(
        controller: _controller,
        enableFilter: true,
        requestFocusOnTap: true,
        leadingIcon: const Icon(Icons.search),
        label: const Text('Color Icon'),
        inputDecorationTheme: const InputDecorationTheme(
          filled: true,
          contentPadding: EdgeInsets.symmetric(vertical: 5.0),
        ), // InputDecorationTheme
        onSelected: (IconColorLabel? iconColor) {
          setState(() {
            selectedIcon = iconColor;
          });
        },
        dropdownMenuEntries: IconColorLabel.entries,
      ), // DropdownMenu
    ],
  ), // Column
), // Center
```

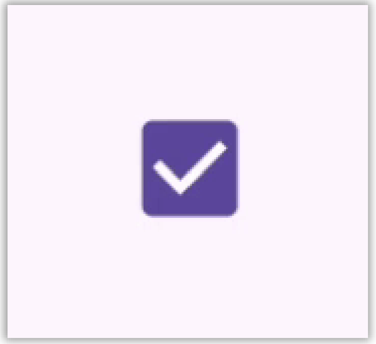


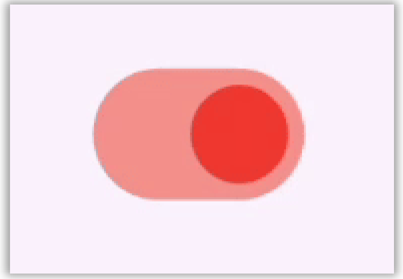
Slider

- Adjust a value by moving an indicator
    - Volume, Brightness
- value: current value
- onChanged: callback triggered when changed
- min: & max: minimum and maximum allowable values
- divisions: a discrete interval

CheckBox & Switch

- value: either true or false
- onChanged: callback triggered when toggles



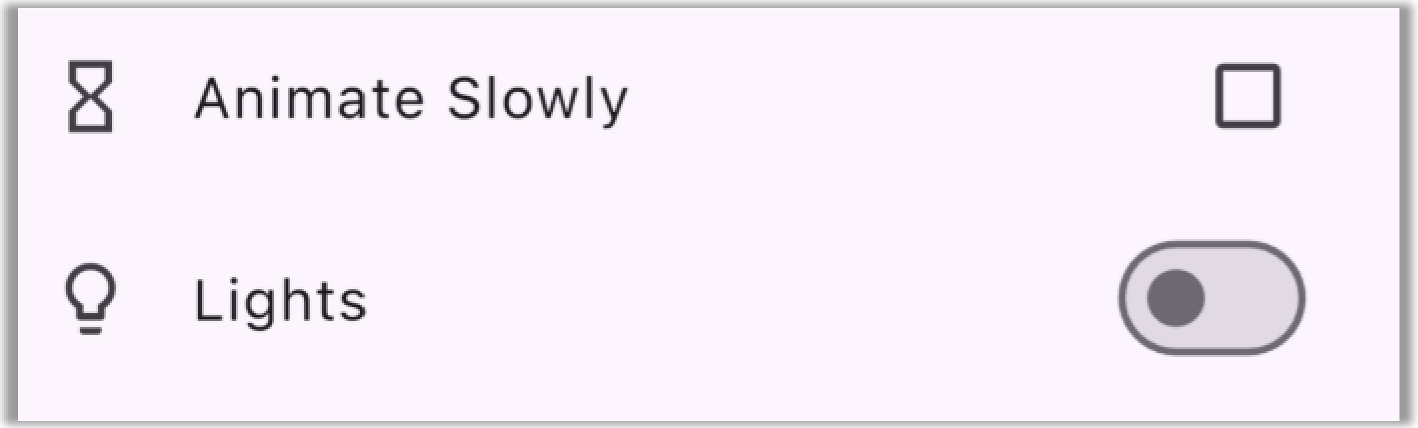


Radio

- RadioGroup contains Radio buttons
- Select between mutually exclusive options
- value: represents button's value
- groupValue: selected value for the group
- onChnaged: callback triggered when user clicks a radio button in group

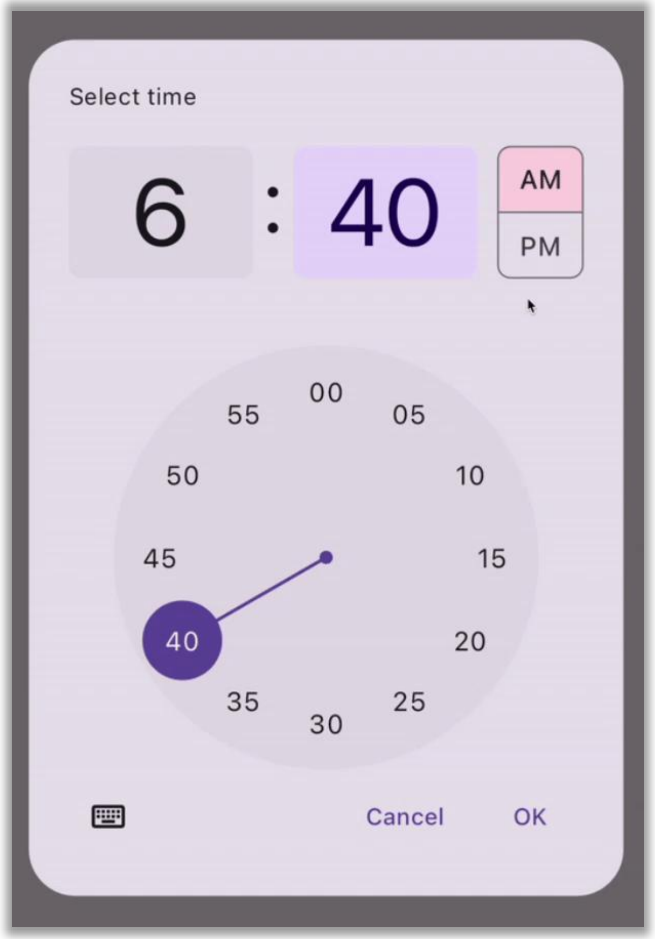
More Toggle Values Widgets

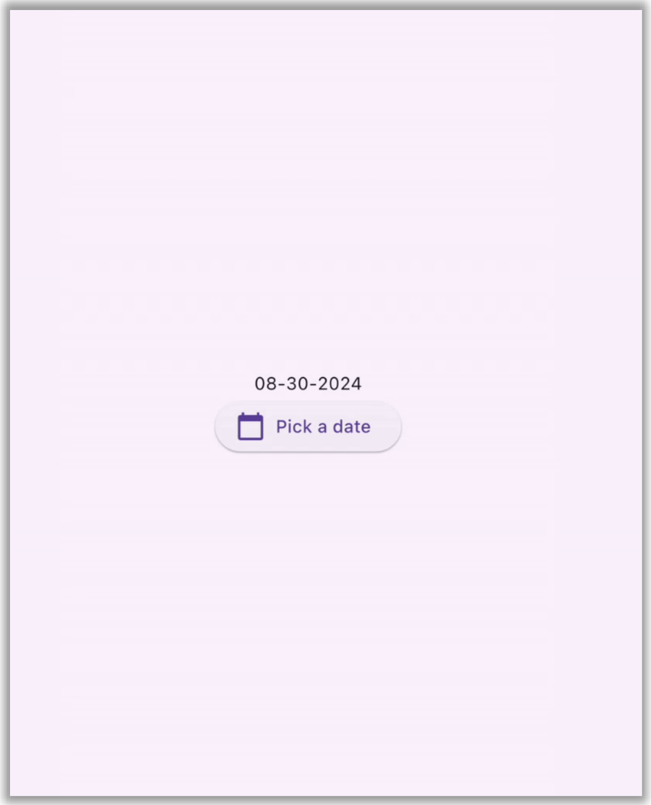
- CheckboxListTile
- SwitchListTile



Date or Time Input

- DatePickerDialog
- TimePickerDialog





Splash Screen

- Screen that appears when App first launched
- Typically displaying the app's logo or brand
- Enhance user experience

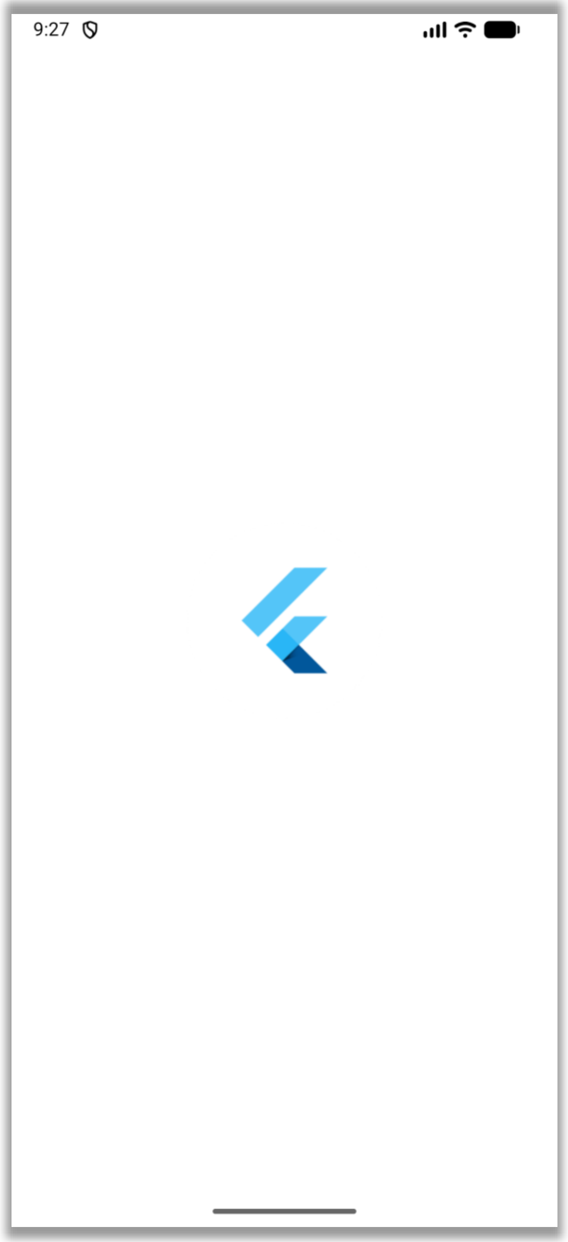
Brand Reinforcement

Loading Feedback

Background Tasks

User Engagement

Smooth Transition



Custom Splash Screen

1. Add package flutter_native_splash
2. Add Image assets that you want to use for Splash Screen
3. Add flutter_native_splash section in pubspec.yaml file
4. Run the package

```
dart run flutter_native_splash:create
```

Time to Practical: Teenage Mutant Ninja Turtle

- Custom Splash Screen
- Show character's Image of TMNT
- Change image when different character radio button is selected

References

- https://docs.flutter.dev/
- https://dart.dev/language
- https://docs.flutter.dev/ui/widgets
- https://www.youtube.com/watch?v=0Xn1QhNtPkQ&list=PLjxrf2q8roU1nbstACpBBSwHa-BuOILlM
- https://www.youtube.com/watch?v=zcJlHVVM84I
- https://docs.flutter.dev/data-and-backend/state-mgmt/ephemeral-vs-app
- https://api.flutter.dev/flutter/widgets/InheritedWidget-class.html
- https://docs.flutter.dev/get-started/fundamentals/user-input