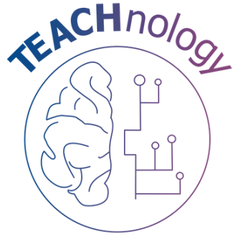
workshop number 4

TEACHnology

powered by AIESEC



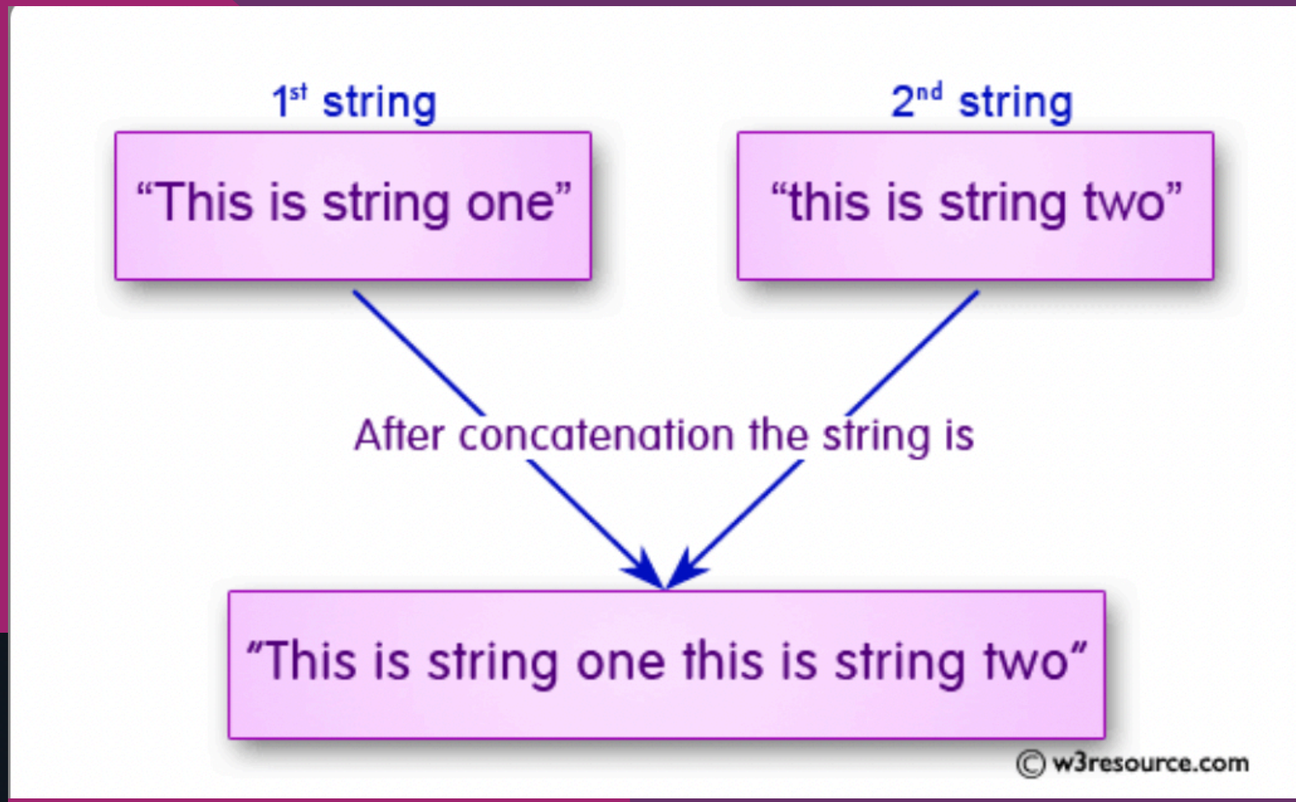
Plan :

- Recap quiz questions
- Classes -> Widgets
- Stateless Widget
- Stateful Widget
- Case Study

syntax for concatenation is

+

```
main(){
  String a="Hello";
  String b="Welcome";
  String c=a+b;
   print(c);
}
```



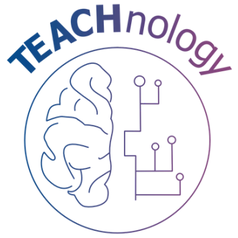

```
HelloWelcome
```

```
1  class MyClass {
2    var MyCheck;
3    var MyNum;
4   MyClass(this.MyCheck,this.MyNum);
5   void MyClassFunction(){
6     print(MyCheck);
7   }
8   }
9   main(){
10    bool Check=true;
11    int a=10;
12    MyClass mine=MyClass(Check,20);
13    mine.MyClassFunction();
14  }
15
```

attributes

```
class Car{
  int HP;
  int Doors;
  String Model;
  Car(this.HP,this.Doors,this.Model);

}

void main(){
  Car bmw = Car(5,2,'E3');
  Car toyota = Car(7,2,'E3');
  print(bmw.HP);
  bmw.HP=3;
  print(bmw.HP);
  print(toyota.HP);
```

Changing the value of the attribute of a class

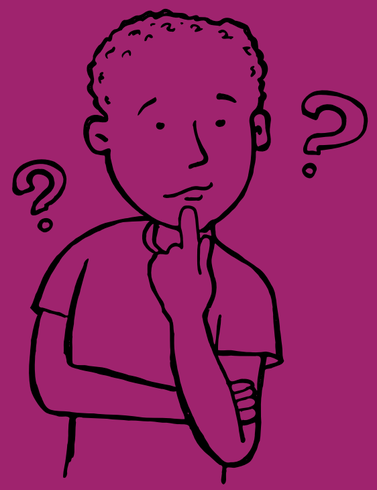


Widgets

- Stateless Widget
- Stateful Widget

Stateless Widgets

- The widgets whose state can not be altered once they are built
- A widget that does not require mutable state.



Now, if you got what I mean, tell me if these widgets are staleless or not?

Icons, Buttons, Text ...

Stateless Widgets

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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
const MyApp({Key? key}) : super(key: key);

@override
Widget build(BuildContext context) {
        return Container();
}
}
```

Stateless Widgets


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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  // MyApp class declaration that extends StatelessWidget
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    // Build method that returns a widget and accepts a BuildContext as a parameter
    return Container();
    // Returns an empty Container widget
    // This is just a placeholder and can be replaced with actual UI components
  }
}
```

Stateless Widgets

BuildContext

- It's a special object that holds important information about the place of a widget in the Flutter app. It's like a map that tells the widget where it is and what's around it.
- Let's say you have a widget called "ProfilePicture" inside a widget called "UserCard." The BuildContext helps the "ProfilePicture" widget find its parent

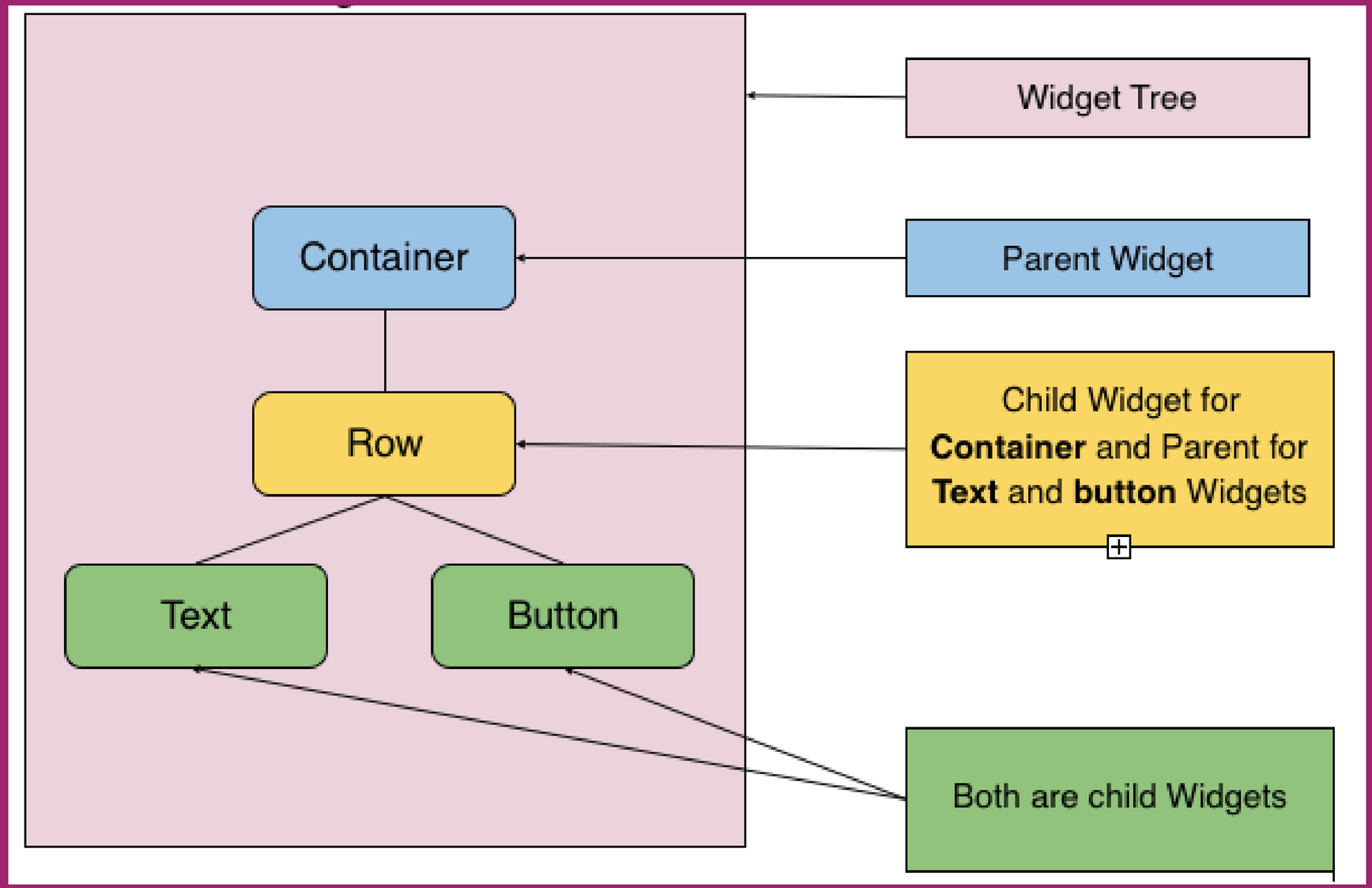
Stateless Widgets

BuildContext

summary

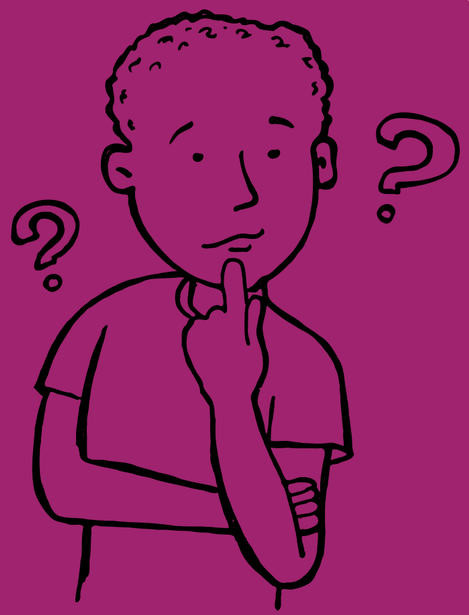
BuildContext is like a helpful guide for widgets in Flutter. It tells them where they are, helps them look good, find other widgets, and perform actions within the app.

Stateless Widgest



Stateful Widgets

- The widgets whose state can be altered once they are built
- A widget that has mutable state



Which one of these is not a stateful widget?

Form

TextField

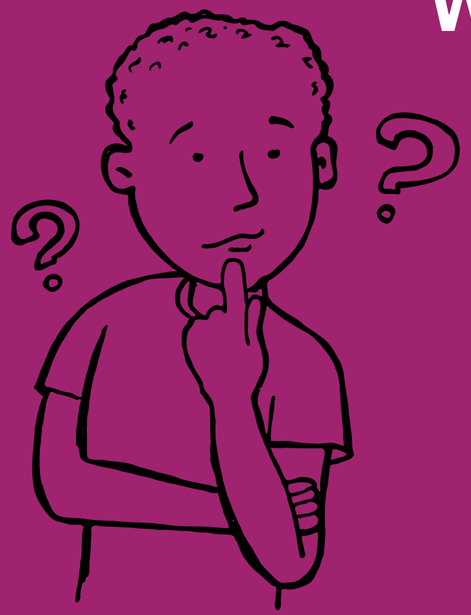
DropdownButton

Container

CheckboxListTile

Stateful Widgets

- The widgets whose state can be altered once they are built
- A widget that has mutable state



Which one of these is not a stateful widget?

Form

TextField

DropdownButton

Container

CheckboxListTile

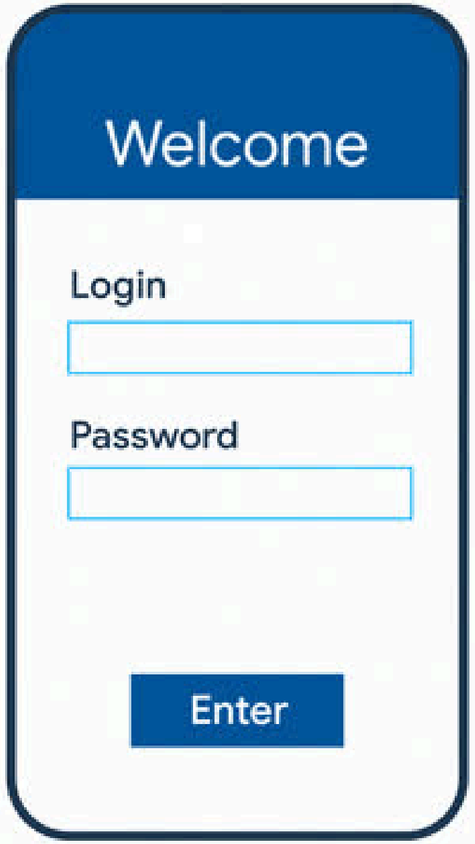
Stateful Widget

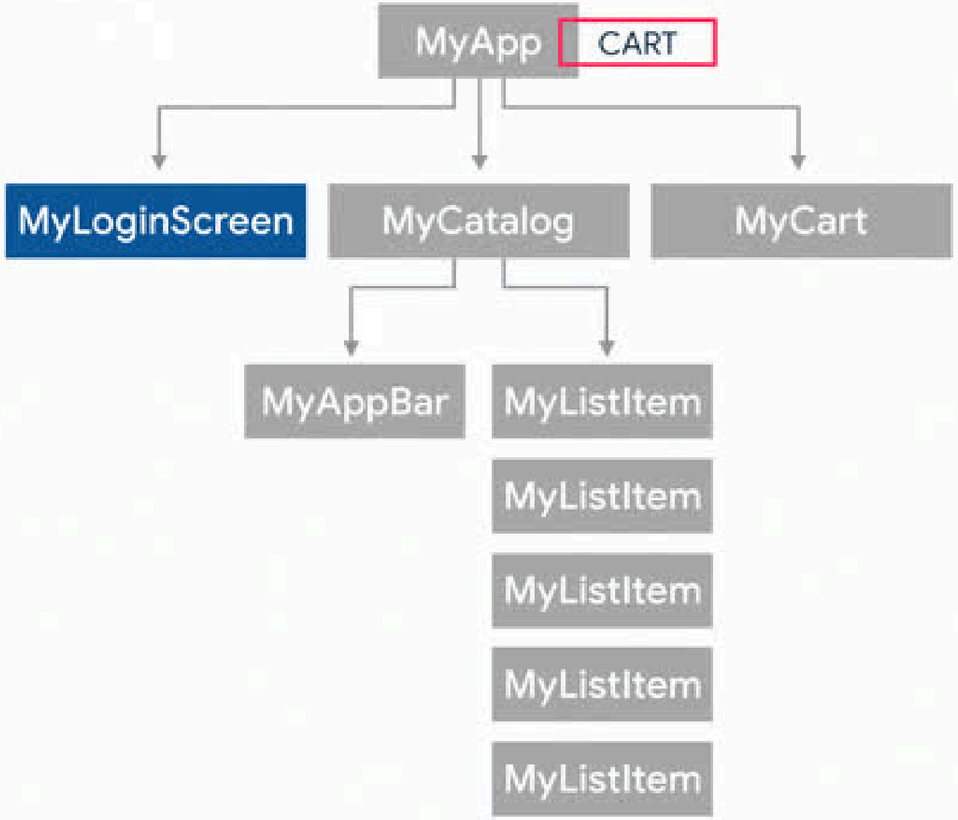
```
class myHomePage extends StatefulWidget {
  const myHomePage({super.key});

  @override
  State<myHomePage> createState() => _myHomePageState();
}

class _myHomePageState extends State<myHomePage> {
  @override
  Widget build(BuildContext context) {
    return const Placeholder();
  }
}
```

Stateful Widget

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

class myHomePage extends StatefulWidget {
  const myHomePage({Key? key}) : super(key: key); // Constructor for myHomePage widget

  @override
  State<myHomePage> createState() => _myHomePageState(); // Create state object for myHomePage
}

class _myHomePageState extends State<myHomePage> {
  @override
  Widget build(BuildContext context) {
    return const Placeholder(); // Return a Placeholder widget as the content of the home page
  }
}
```

Widget Tree


Welcome
Login
Password
Enter


MyApp
CART
MyLoginScreen
MyCatalog
MyCart
MyAppBar
MyListItem
MyListItem
MyListItem
MyListItem
MyListItem

Types of Widgets

1) Container

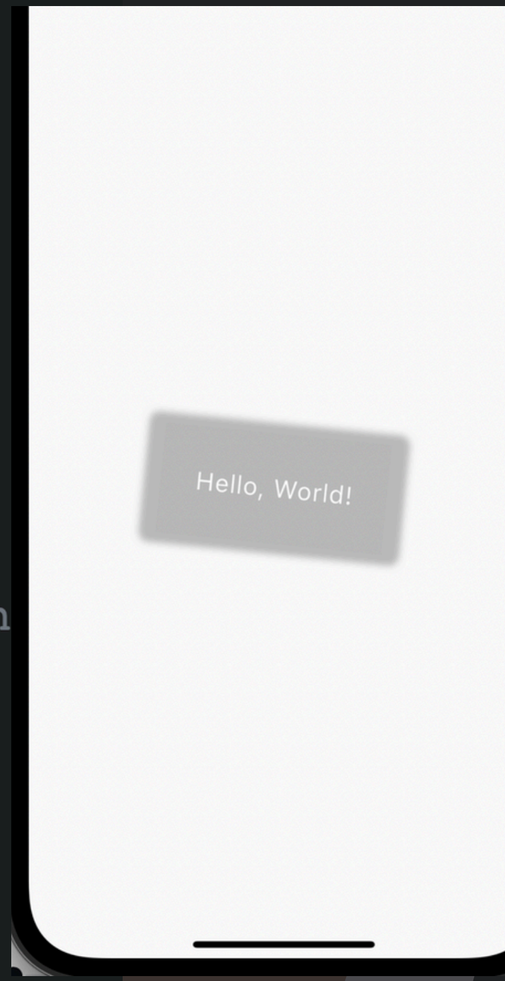
- Margin: Spacing around the container's outer edges.
- Color: Background color of the container.
- Child: The widget contained within the container.
- Height and width: Dimensions of the container.
- Constraints: Restrictions or limitations applied to the container's size.

Types of Widgets

1) Container

- Padding: Spacing between the container's content and its edges.
- Alignment: Positioning of the container's child widget within the container.
- Decoration: Visual appearance or styling applied to the container.
- Transform: Geometric transformation applied to the container or its child widget.

```
child: Container(
  margin: EdgeInsets.all(16.0), // Sets margin around the container
  //color: Colors.blue, // Sets the background color to blue
  width: 200.0, // Sets the width of the container
  height: 100.0, // Sets the height of the container
  constraints:
      BoxConstraints(maxWidth: 300.0), // Sets maximum width constraint
  padding: EdgeInsets.symmetric(
      horizontal: 24.0,
      vertical: 12.0), // Sets padding inside the container
  alignment: Alignment.center, // Aligns the child widget to the center
  decoration: BoxDecoration(
    borderRadius:
        BorderRadius.circular(8.0), // Applies rounded corner decoration
    boxShadow: [
      BoxShadow(
          color: Colors.black26,
          blurRadius: 4.0), // Applies a shadow effect
    ],
  ),
  transform:
      Matrix4.rotationZ(0.1), // Applies a rotation transformation
```



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

void main() {
  runApp(MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        title: Text('Container Example'),
      ),
      body: Center(
        child: Container(
          margin: EdgeInsets.all(16.0), // Sets margin around the container
          //color: Colors.blue, // Sets the background color to blue
          width: 200.0, // Sets the width of the container
          height: 100.0, // Sets the height of the container
          constraints:
              BoxConstraints(maxWidth: 300.0), // Sets maximum width constraint
          padding: EdgeInsets.symmetric(
              horizontal: 24.0,
              vertical: 12.0), // Sets padding inside the container
          alignment: Alignment.center, // Aligns the child widget to the center
          decoration: BoxDecoration(
            borderRadius:
                BorderRadius.circular(8.0), // Applies rounded corner decoration
            boxShadow: [
              BoxShadow(
                  color: Colors.black26,
                  blurRadius: 4.0), // Applies a shadow effect
            ],
          ),
          transform:
              Matrix4.rotationZ(0.1), // Applies a rotation transformation
          child: Text(
            'Hello, World!', // Child widget
            style: TextStyle(color: Colors.white, fontSize: 20.0),
          ),
        ),
      ),
    ),
  ));
}
```

Template for you to work on at home

for container widget

Types of Widgets

2) Row or Column

- VerticalDirection: Determines the vertical direction of widget layout.
- TextDirection: Specifies the text layout direction.
- TextBaseline: Sets the baseline alignment for text within a widget.
- Children: Contains a list of child widgets within a parent widget.
- ClipBehavior: Defines the behavior for clipping overflow within a widget.

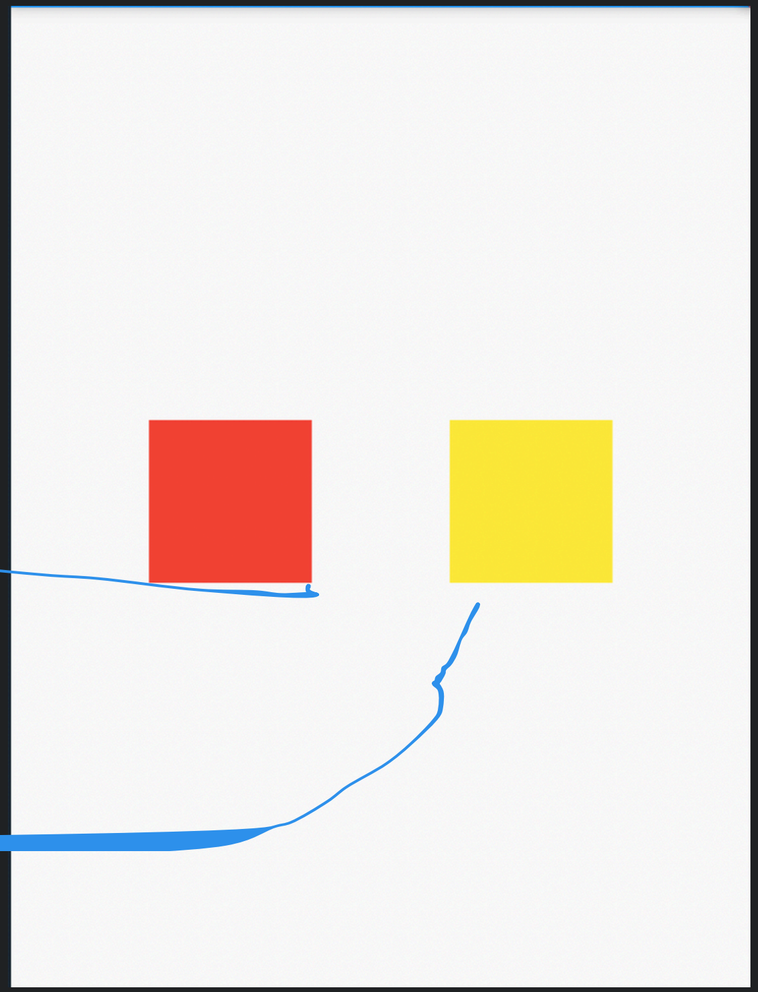
Types of Widgets

2) Row or Column

- Direction: Determines the direction of widget layout.
- MainAxisSize: Controls the size of the main axis of a widget.
- CrossAxisAlignment: Defines how children are aligned along the cross axis.
- MainAxisAlignment: Specifies how children are aligned along the main axis.

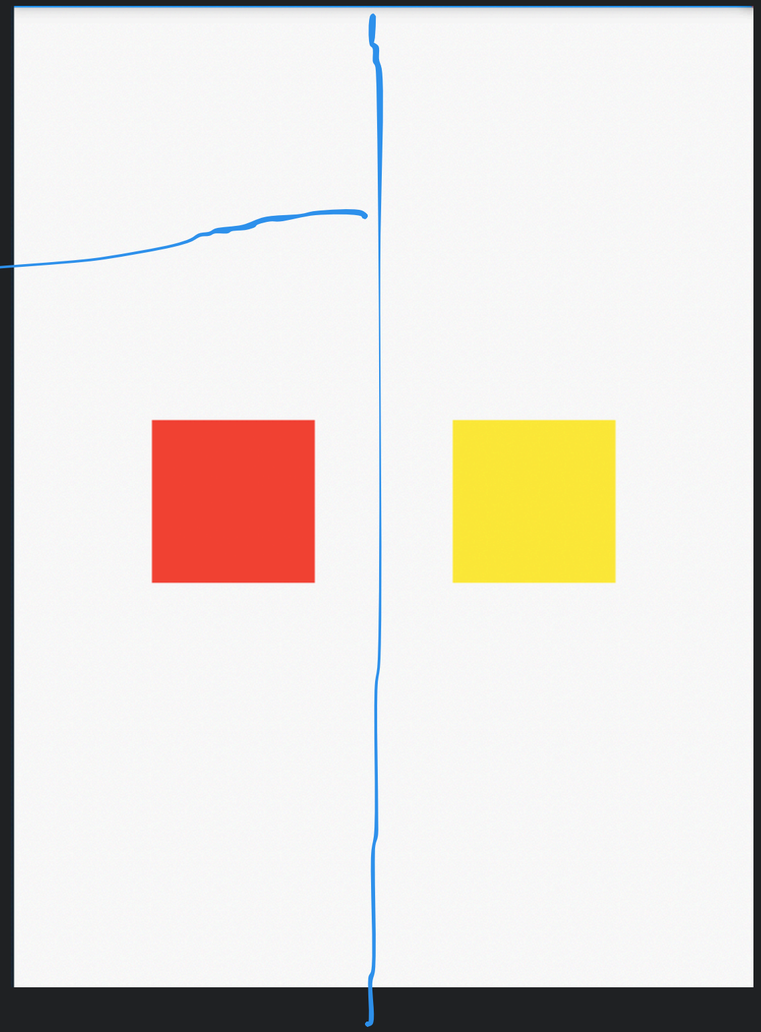
concept of Child/Children

```
16 ▼ Widget _buildWideContainers() {
17     return Center(
18       child: Row(
19         mainAxisAlignment: MainAxisAlignment.spaceEvenly,
20 ▼       children: <Widget>[
21           Container(
22             height: 100.0,
23             width: 100.0,
24             color: Colors.red,
25           ),
26           Container(
27             height: 100.0,
28             width: 100.0,
29             color: Colors.yellow,
30           ),
31         ],
32       ),
33     );
34   }
```



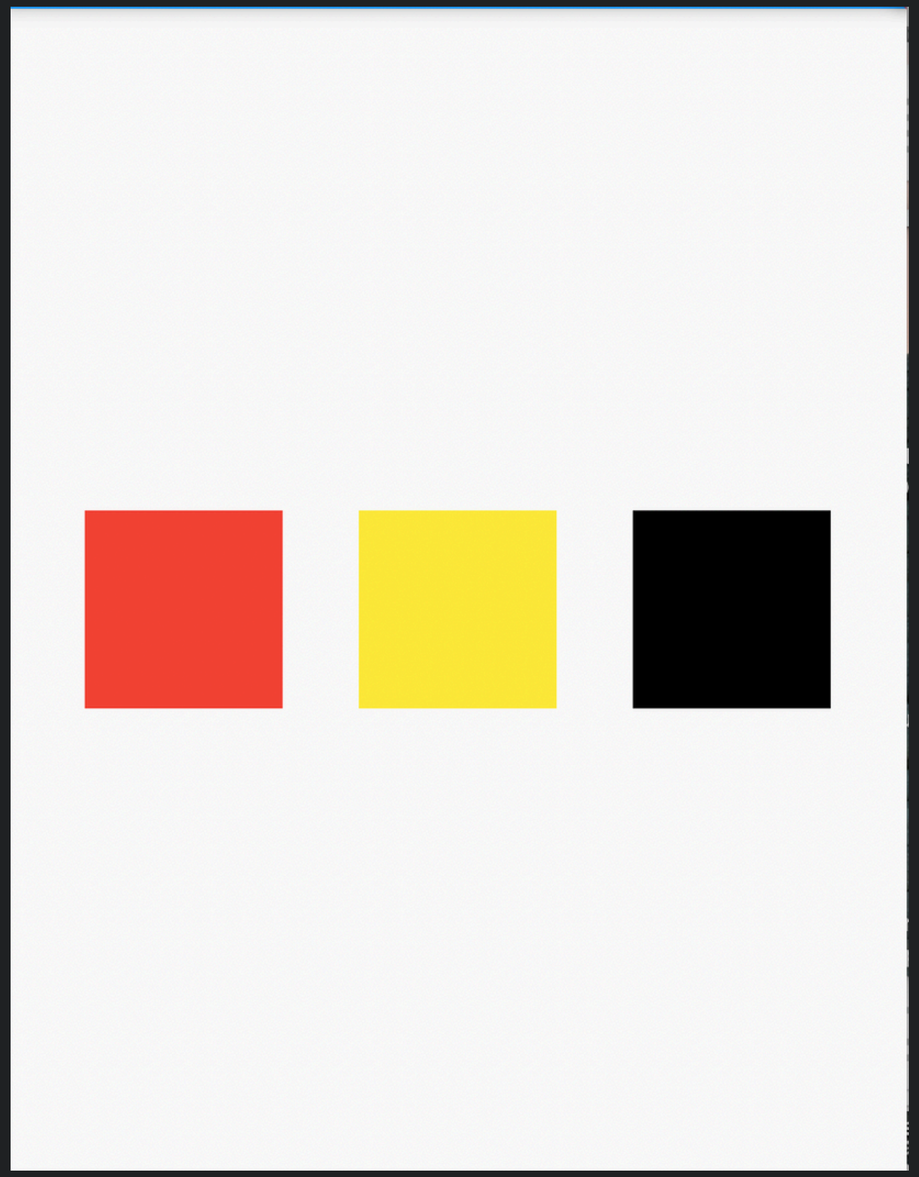
concept of Row

```dart
Widget _buildWideContainers() {
  return Center(
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.red,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.yellow,
        ),
      ],
    ),
  );
}
```



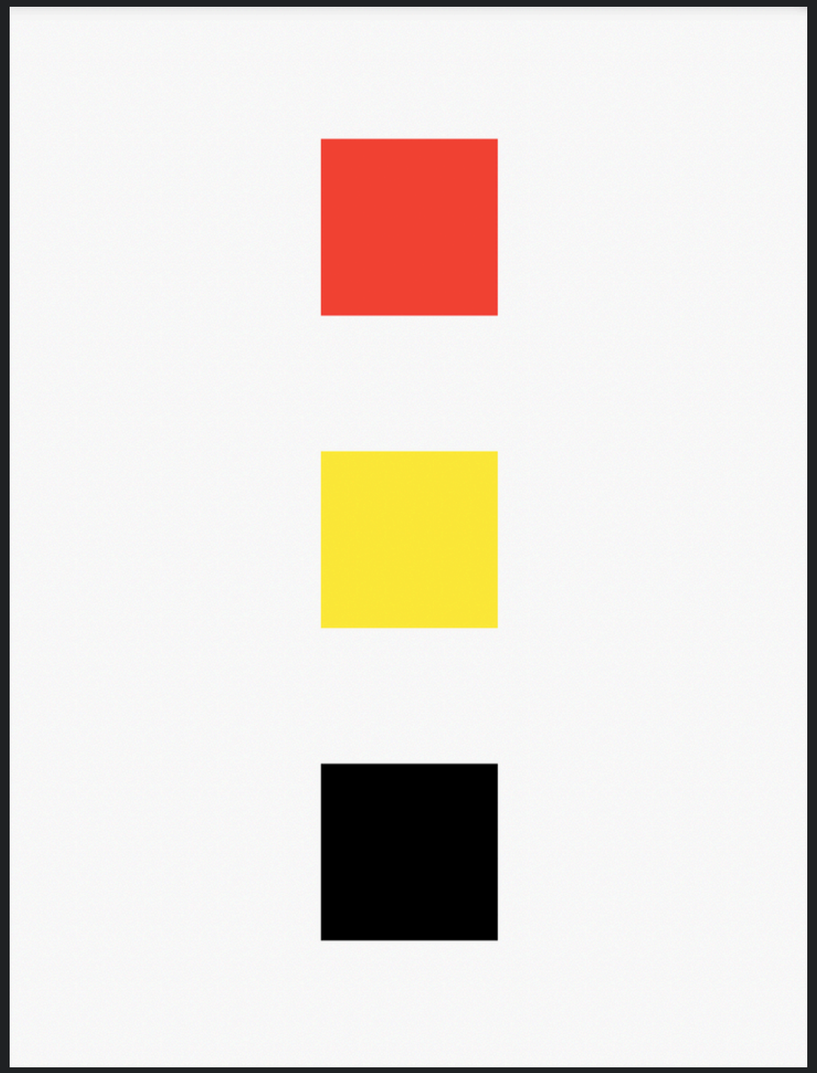
Adding a widget

```
Widget _buildWideContainers() {
  return Center(
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.red,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.yellow,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.black,
        ),
      ],
    ),
  );
```



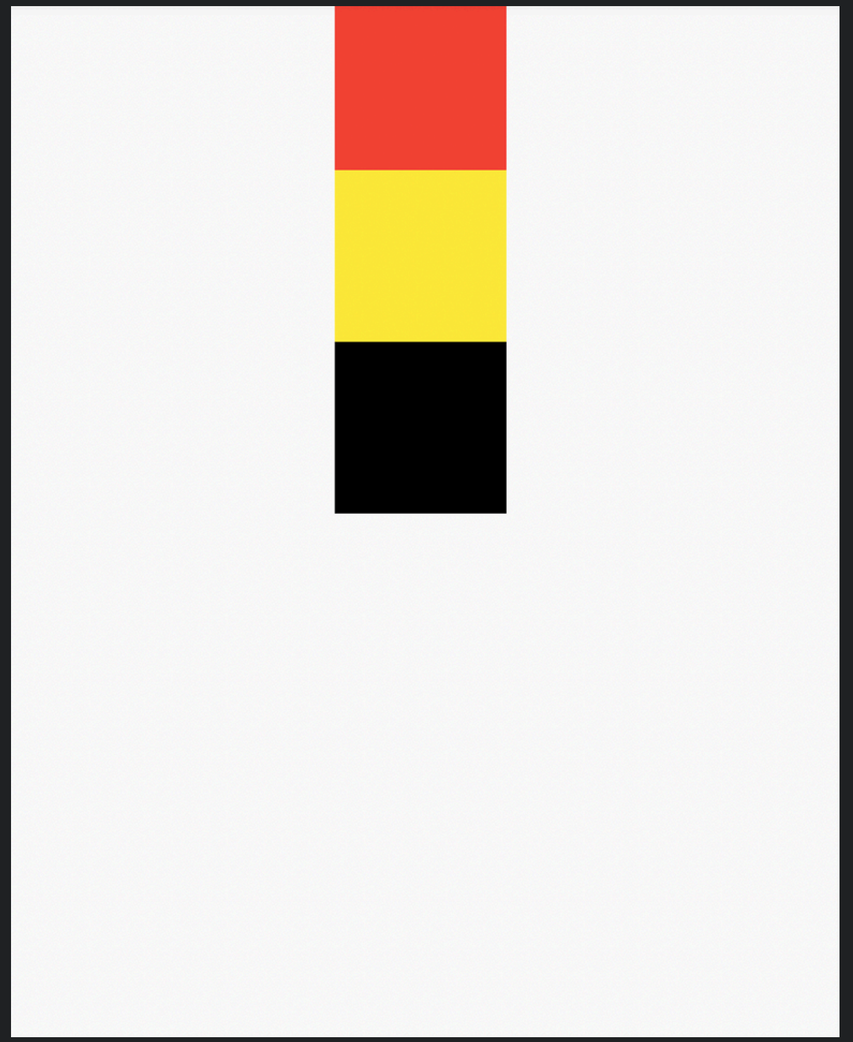
concept of Column

```
Widget _buildWideContainers() {
  return Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.red,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.yellow,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.black,
        ),
      ],
    ),
  );
}
```



Alignment

```
Widget _buildWideContainers() {
  return Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.start,
      children: <Widget>[
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.red,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.yellow,
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.black,
        ),
      ],
    ),
  );
}
```



Types of Widgets

3) Text

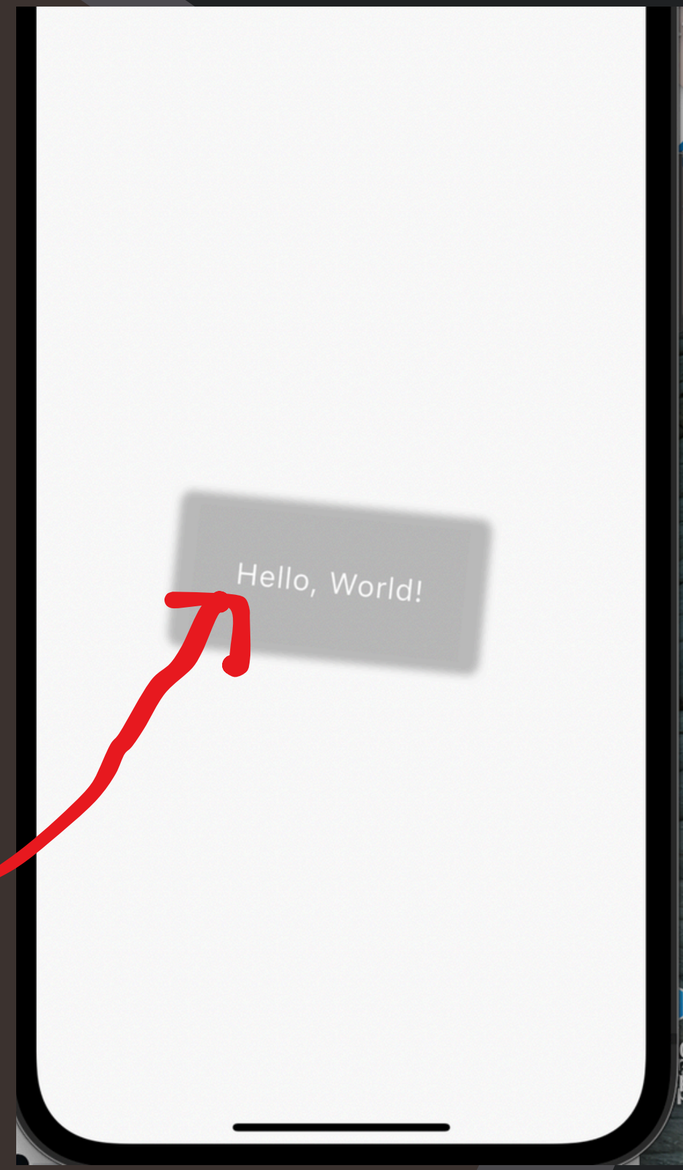
- Key: Identifier for tracking and managing widgets efficiently.
- Locale: Represents a specific region for localization and formatting.
- MaxLines: Limits the number of displayed lines for text.
- Data: Raw information used in a program.
- Overflow: Behavior when content exceeds available space.
- SoftWrap: Enables text wrapping within a container.

Types of Widgets

3) Text

- TextAlign: Horizontal alignment of text within a widget.
- Style: Visual appearance and formatting properties.
- TextScaleFactor: Scaling factor for text size.

```dart
        child: Text(
          'Hello, World!', // Child widget
          style: TextStyle(color: Colors.white, fontSize: 20.0),
        ),
```



Types of Widgets

4) Image

- Key: Identifier used for tracking and managing objects efficiently.
- Size: Represents the dimensions or extent of an object, such as width and height.
- Style: Visual appearance and formatting properties of an object.
- Duration: Time span or length of a specific action or event.

Types of Widgets

4) Image

- Curve: Mathematical function defining the interpolation or easing behavior of an animation.
- HashCode: Unique numeric value generated for an object based on its contents.
- TextColor: Color applied to the text of an object, such as a widget or a text element.
- Runtime: Period during which a program is executing or running.

Location of the image you want to add

Attributes

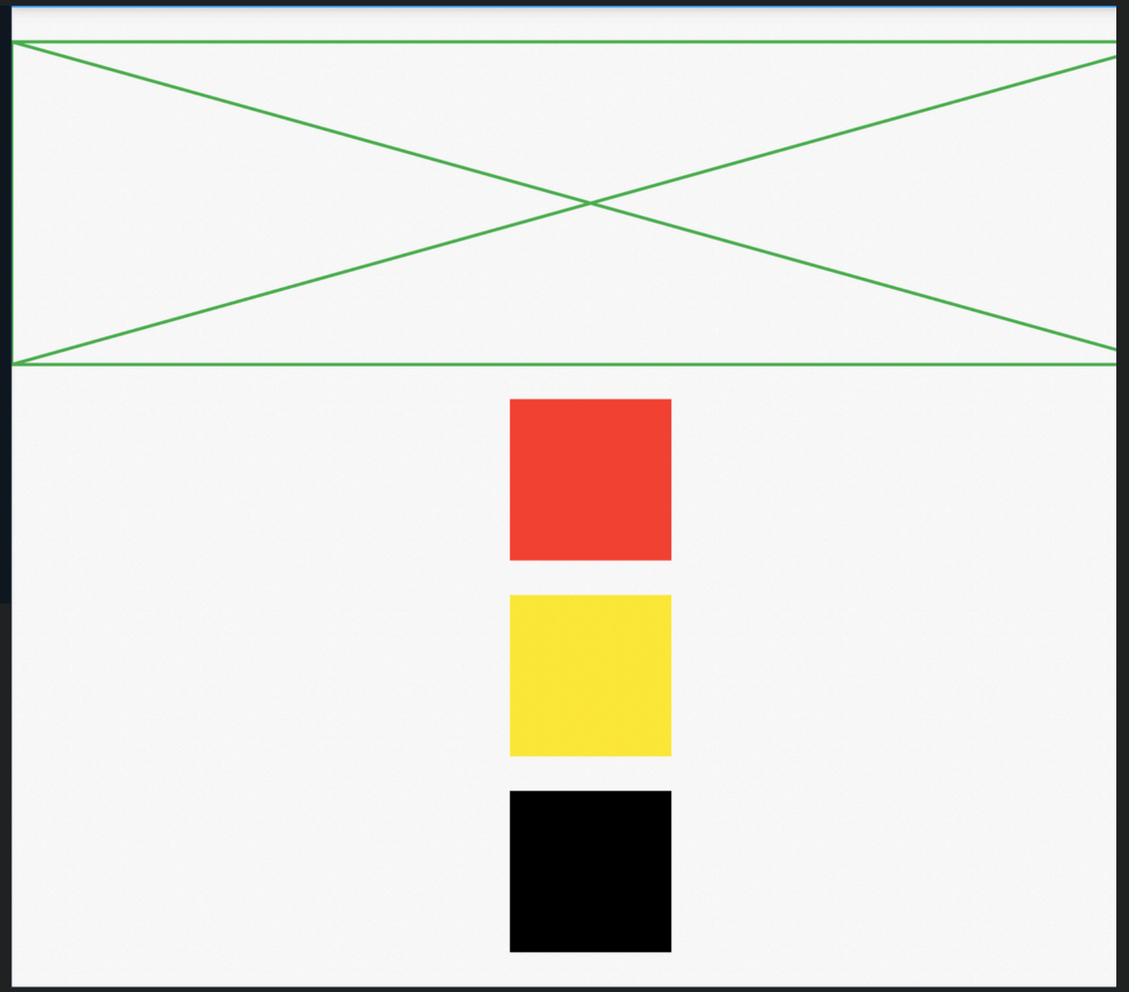
```
Image(
    image: AssetImage('assets/images/my_image.png'),
    width: 200.0,
    height: 200.0,
    fit: BoxFit.cover,),
```

Types of Widgets

5) Placeholder

Note that the Placeholder widget is mainly used for development purposes and is not intended for production usage. Once you have the actual image ready, you should replace the Placeholder widget with the Image widget as shown in the previous response to display the actual image.

```
Placeholder(
  fallbackHeight: 200.0,
  fallbackWidth: 200.0,
  color: Colors.grey,
),
```



Types of Widgets

6)AppBar

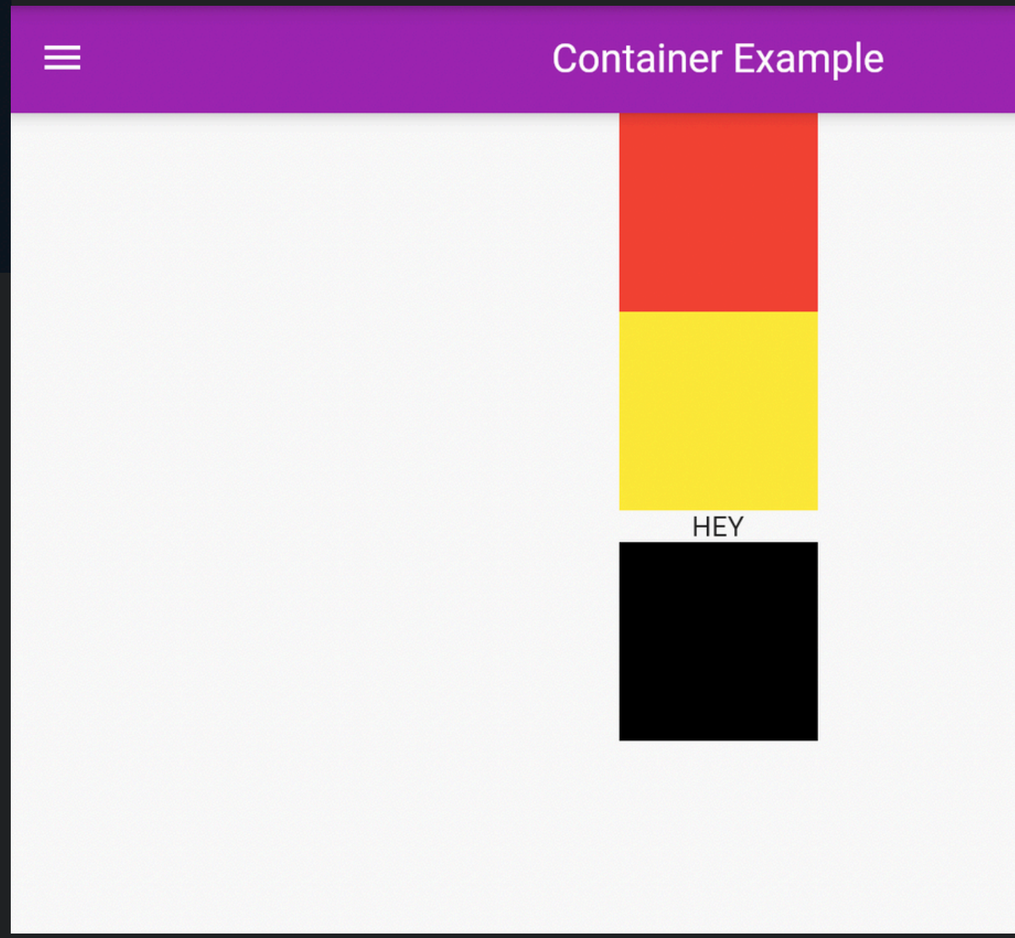
- Brightness: The brightness level (light or dark) of the AppBar.
- Bottom: The widget displayed at the bottom of the AppBar.
- BackgroundColor: The color applied to the background of the AppBar.
- ActionsIconTheme: The icon theme applied to the action icons in the AppBar.
- BottomOpacity: The opacity of the bottom widget in the AppBar.
- Elevation: The elevation or shadow depth of the AppBar.
- CenterTitle: Whether the title is centered within the AppBar.

Types of Widgets

6) AppBar

- HashCode: Unique numeric value generated for an AppBar based on its contents.
- Key: Identifier used for tracking and managing AppBar efficiently.
- Leading: The widget displayed at the start of the AppBar.
- Shape: The shape of the AppBar, such as flat, rounded, or with custom borders.
- TextTheme: The text theme applied to the title and other text elements in the AppBar.

```
appBar: AppBar(
  title: Text('Container Example'),
    backgroundColor: Colors.purple,
  leading: IconButton(
      icon: Icon(Icons.menu),
      onPressed: () {}
    ),
),
```



BRAINSTORM

Groups

Come up With the idea for the app

```dart
void main() {
  runApp(MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        title: Text('Container Example'),
          backgroundColor: Colors.purple,
        leading: IconButton(
            icon: Icon(Icons.menu),
            onPressed: () {}
          ),
      ),
         body: Container(
          height: 100.0,
          width: 200.0,
          color: Colors.black,
        ),
    ),
  ));
}
```

Code a container that is red

should write your apps name on top.

5 mins

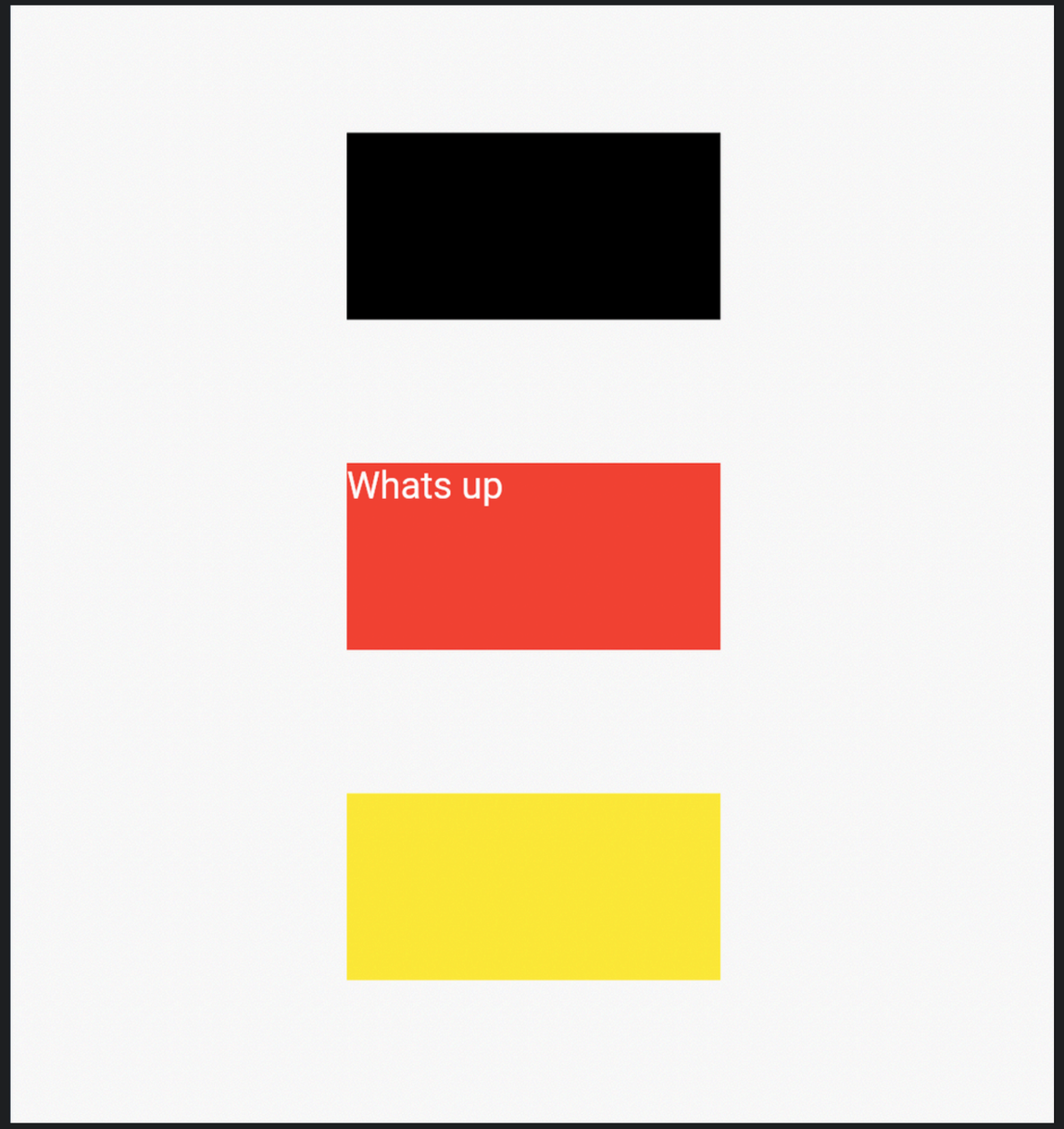
```
void main() {
  runApp(MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        title: Text('Container Example'),
          backgroundColor: Colors.purple,
        leading: IconButton(
            icon: Icon(Icons.menu),
            onPressed: () {}
          ),
      ),
         body:_2widgets(),
    ),
    ),);
  }

Widget _2widgets(){
  return Center(
  child:Column(
  mainAxisAlignment:MainAxisAlignment.start,
  children:<Widget>[
    Container(
    height:100.0,
    width:200.0,
    color:Colors.black,
    ),
    Placeholder(
    color:Colors.grey),
  ],),);
}
```

2widgets in body?? How to

Add a placeholder beneath container.

Extra Hard Exercise

Create This



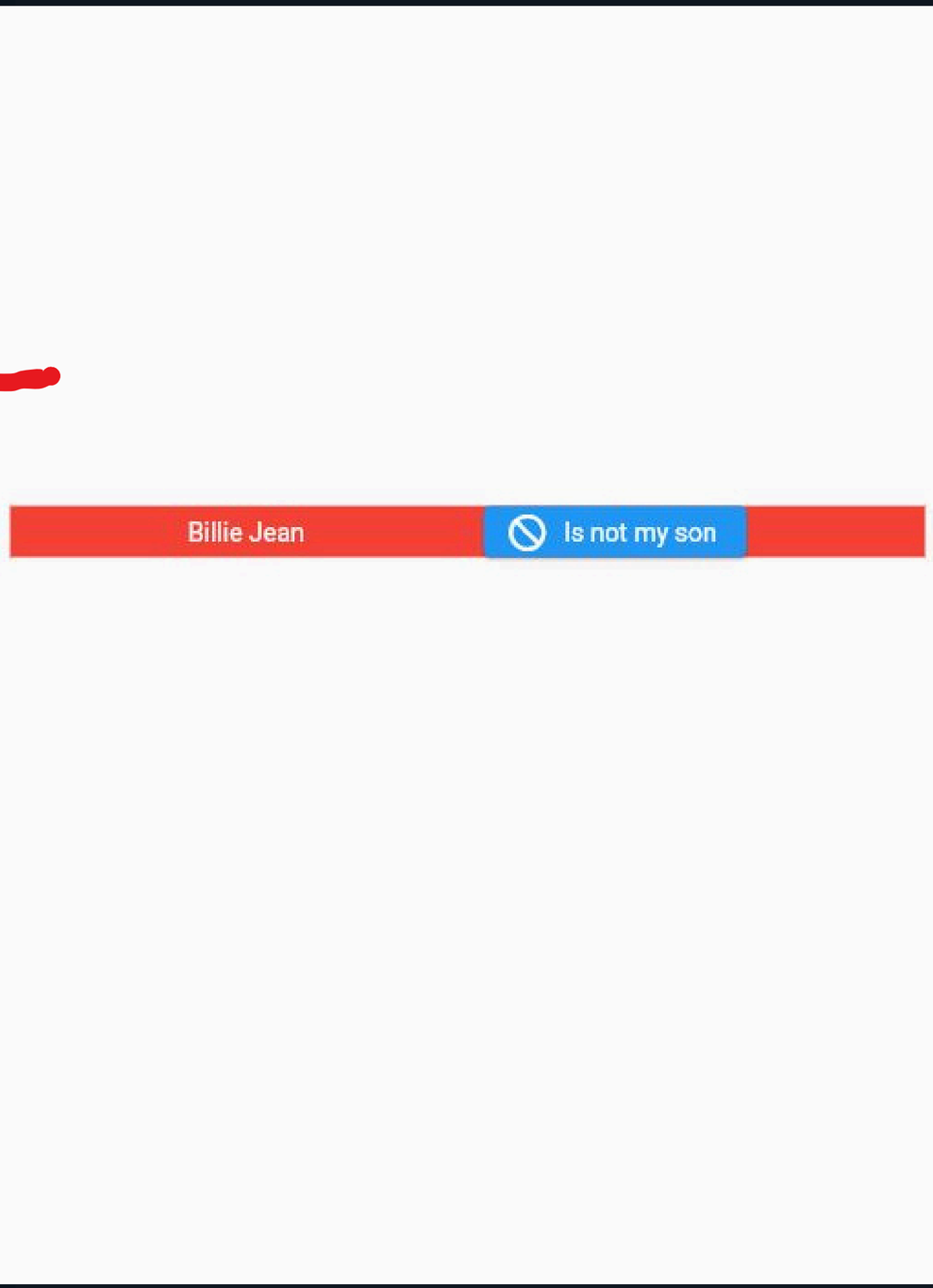
QUIZ

go to https://quizizz.com/join

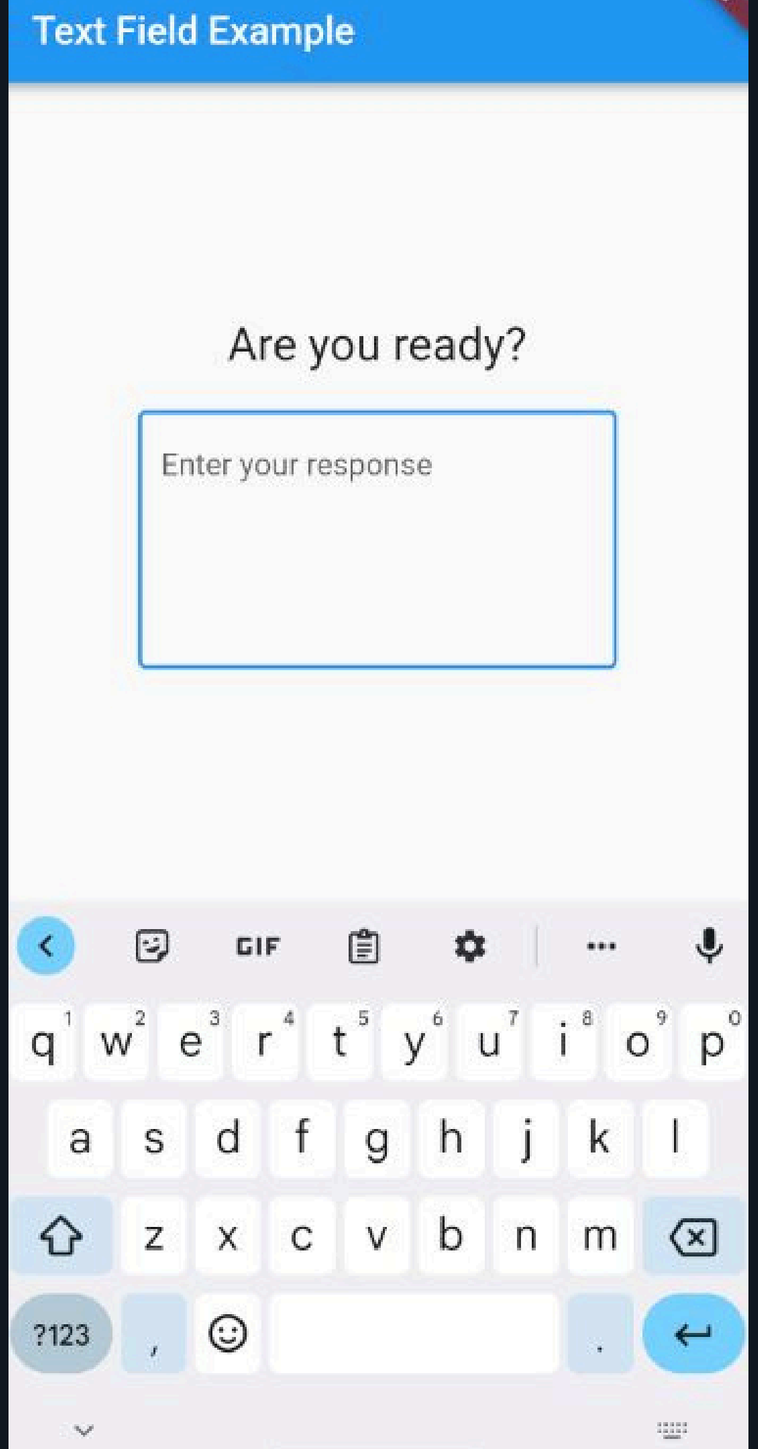
enter code:

699623

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

void main() {
  runApp(MaterialApp(
    home: Scaffold(

      body:  Center(
      child: Container(
          constraints: const BoxConstraints(maxHeight: 500,maxWidth: 500),
          color: Colors.red,
          child:Row(
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,
            children: [
              const Text(
                'Billie Jean',
                style: TextStyle(
                  color: Colors.white,
                ),
              ),
              ElevatedButton.icon(
                  onPressed: (){},
                  icon: Icon(Icons.not_interested),
                  label: Text('Is not my son'),
              ),
            ],
          ),

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



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

void main() {
  runApp(MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        title: const Text('Text Field Example'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            const Text(
              'Are you ready?',
              style: TextStyle(fontSize: 24),
            ),
            SizedBox(height: 20),
            Container(
              width: 250,
              child: TextField(
                decoration: InputDecoration(
                  border: OutlineInputBorder(),
                  hintText: 'Enter your response',
                ),
                maxLines: 5,
              ),
            ),
          ],
        ),
      ),
    ),
  ));
}
```



```dart
Widget _buildWideContainers() {
  return Center(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Placeholder(
          fallbackHeight: 200.0,
          fallbackWidth: 200.0,
          color:Colors.green,
        ),

        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.red,
        ),
        Text(
        "HEY MOM",
        style: TextStyle(color:Colors.blue,fontSize:20.0)
        ),
        Container(
          height: 100.0,
          width: 100.0,
          color: Colors.black,
        ),
      ],
    ),
  );
}
```