Activities With Ansh

Flutter All Widgets

.1. Basic Widgets

These are fundamental widgets used in most Flutter apps.

1.1 Text

Used to display text.

```
Text(
  'Hello, Flutter!',
  style: TextStyle(fontSize: 20, color: Colors.blue),
)
```

1.2 Container

A rectangular box with styling options.

```
Container(
  width: 100,
  height: 100,
  color: Colors.blue,
  child: Text('Hello'),
)
```

1.3 Row

Displays widgets in a horizontal direction.

```
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    Icon(Icons.star),
    Text('Row Widget'),
  ],
)
```

1.4 Column

Displays widgets in a vertical direction.

```
Column(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    Text('Item 1'),
    Text('Item 2'),
  ],
)
```

1.5 Image

Displays an image.

```
Image.network(
  'https://example.com/image.jpg',
  width: 200,
  height: 200,
  fit: BoxFit.cover,
)
```

---

2. Layout Widgets

Used to organize other widgets.

2.1 Center

Aligns its child widget in the center

```
Center(
  child: Text('Centered Text'),
)
```

2.2 Padding

Adds padding around a child widget.

```
Padding(
  padding: EdgeInsets.all(10),
  child: Text('Padded Text'),
)
```

2.3 Align

Aligns the child widget within itself.

```
Align(
  alignment: Alignment.bottomRight,
  child: Text('Aligned Text'),
)
```

2.4 Expanded

Expands a child of Row or Column to fill space.

```
Expanded(
  child: Container(color: Colors.blue),
)
```

2.5 SizedBox

Creates a box with a specific size.

```
SizedBox(
  width: 50,
  height: 50,
)
```

---

3. Input Widgets

Used to gather input from users.

3.1 TextField

Single-line or multi-line text input.

```
TextField(
  decoration: InputDecoration(
    labelText: 'Enter your name',
    border: OutlineInputBorder(),
  ),
)
```

3.2 Button Widgets

ElevatedButton

```
ElevatedButton(
  onPressed: () {
    print('Button pressed');
  },
  child: Text('Elevated Button'),
)
```

TextButton

```
TextButton(
  onPressed: () {},
  child: Text('Text Button'),
)
```

IconButton

```
IconButton(
  icon: Icon(Icons.add),
  onPressed: () {
    print('Icon pressed');
  },
)
```

---

4. Scrolling Widgets

Widgets to display scrollable content.

4.1 ListView

Creates a scrollable list.

```
ListView(
  children: [
    Text('Item 1'),
    Text('Item 2'),
  ],
)
```

4.2 GridView

Displays items in a grid.

```
GridView.count(
  crossAxisCount: 2,
  children: [
    Container(color: Colors.red),
    Container(color: Colors.blue),
  ],
)
```

4.3 SingleChildScrollView

Makes its child scrollable.

```
SingleChildScrollView(
  child: Column(
    children: List.generate(20, (index) => Text('Item $index')),
  ),
)
```

---

5. App Structure Widgets

Essential widgets for app structure.

5.1 Scaffold

The basic structure of a Material app screen.

```
Scaffold(
  appBar: AppBar(title: Text('My App')),
  body: Center(child: Text('Hello, Flutter!')),
)
```

5.2 AppBar

A top app bar with optional title and actions.

```
AppBar(
  title: Text('My App'),
  actions: [Icon(Icons.settings)],
)
```

5.3 BottomNavigationBar

Adds navigation at the bottom.

```
BottomNavigationBar(
  items: [
    BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),
    BottomNavigationBarItem(icon: Icon(Icons.person), label: 'Profile'),
  ],
)
```

---

6. Advanced Widgets

For more complex UI designs.

6.1 Stack

Places widgets on top of each other.

```
Stack(
  children: [
    Container(width: 100, height: 100, color: Colors.red),
    Positioned(
      top: 10,
      left: 10,
      child: Text('Positioned Text'),
    ),
  ],
)
```

6.2 Card

A material design card.

```
Card(
  elevation: 5,
  child: Padding(
    padding: EdgeInsets.all(8),
    child: Text('Card Widget'),
  ),
)
```

6.3 FutureBuilder

Displays data from a Future.

```
FutureBuilder(
  future: fetchData(),
  builder: (context, snapshot) {
    if (snapshot.connectionState == ConnectionState.waiting) {
      return CircularProgressIndicator();
    } else if (snapshot.hasError) {
      return Text('Error: ${snapshot.error}');
    } else {
      return Text('Data: ${snapshot.data}');
    }
  },
)
```

6.4 StreamBuilder

Builds UI based on a Stream.

```
StreamBuilder(
  stream: myStream(),
  builder: (context, snapshot) {
    if (snapshot.hasData) {
      return Text('Data: ${snapshot.data}');
    }
    return CircularProgressIndicator();
  },
)
```

---

7. Animation Widgets

For adding animations.

7.1 AnimatedContainer

Automatically animates changes.

```
AnimatedContainer(
  duration: Duration(seconds: 1),
  width: 100,
  height: 100,
  color: Colors.blue,
)
```

7.2 Hero

Hero animations between screens.

```
Hero(
  tag: 'heroTag',
  child: Image.asset('assets/image.png'),
)
```

---

8. Gesture Widgets

Detect gestures like tap, drag, etc.

8.1 GestureDetector

Adds gesture detection.

```
GestureDetector(
  onTap: () {
    print('Widget tapped');
  },
  child: Container(
    color: Colors.blue,
    height: 50,
    width: 50,
  ),
)
```

---

9. State Management Widgets

For managing state.

9.1 StatefulWidget

A widget with mutable state.

```
class MyWidget extends StatefulWidget {
  @override
  State<MyWidget> createState() => _MyWidgetState();
}

class _MyWidgetState extends State<MyWidget> {
  @override
  Widget build(BuildContext context) {
    return Text('Stateful Widget');
  }
}
```

---

10. Other Widgets

10.1 Divider

Adds a horizontal line.

```
Divider(color: Colors.grey, thickness: 1)
```

10.2 Flexible

Resizes its child within a Row or Column.

```
Flexible(
  child: Container(color: Colors.red),
)
```

All Basics Complete Flutter Widgets PDF

Made By: Ansh Chaurasia