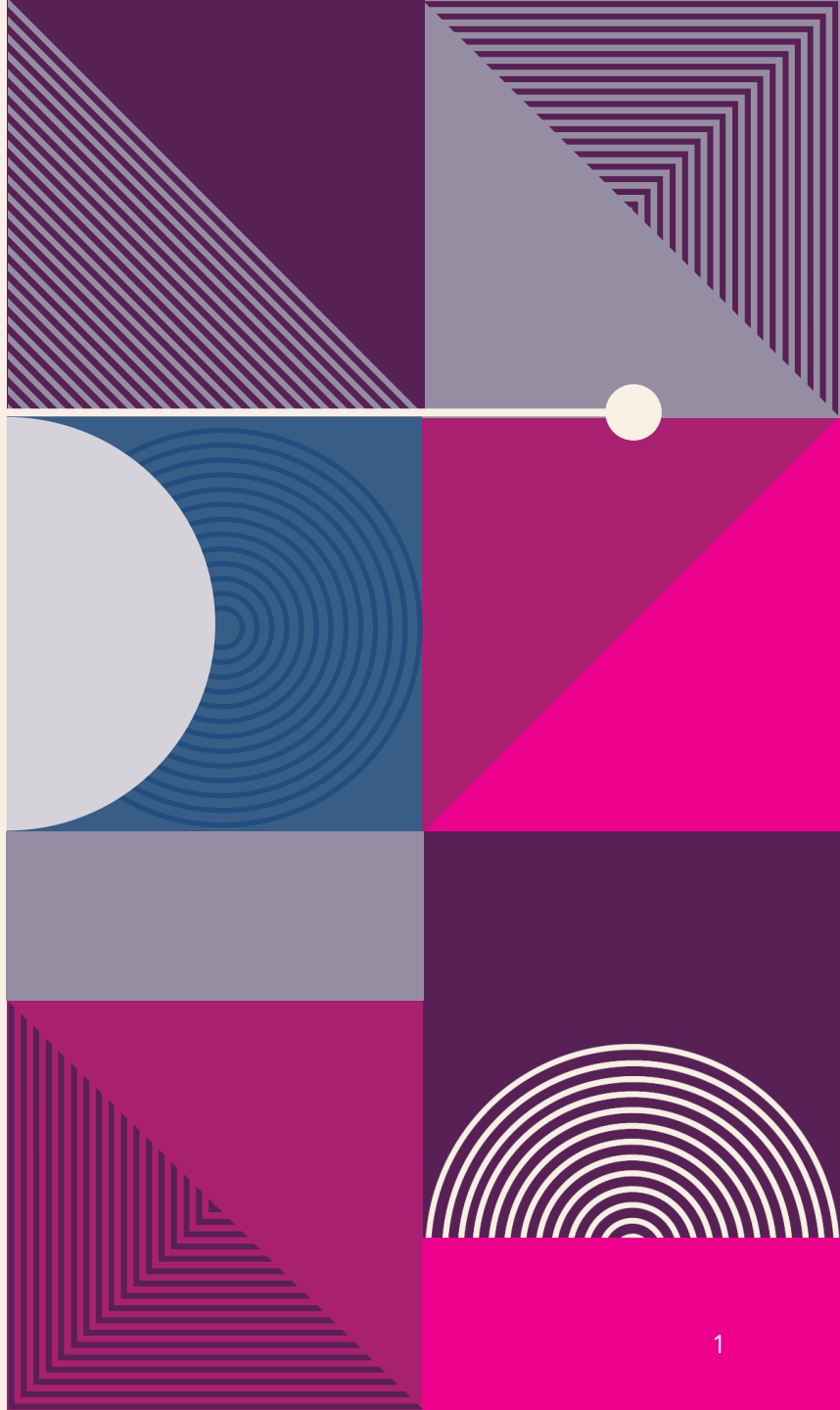
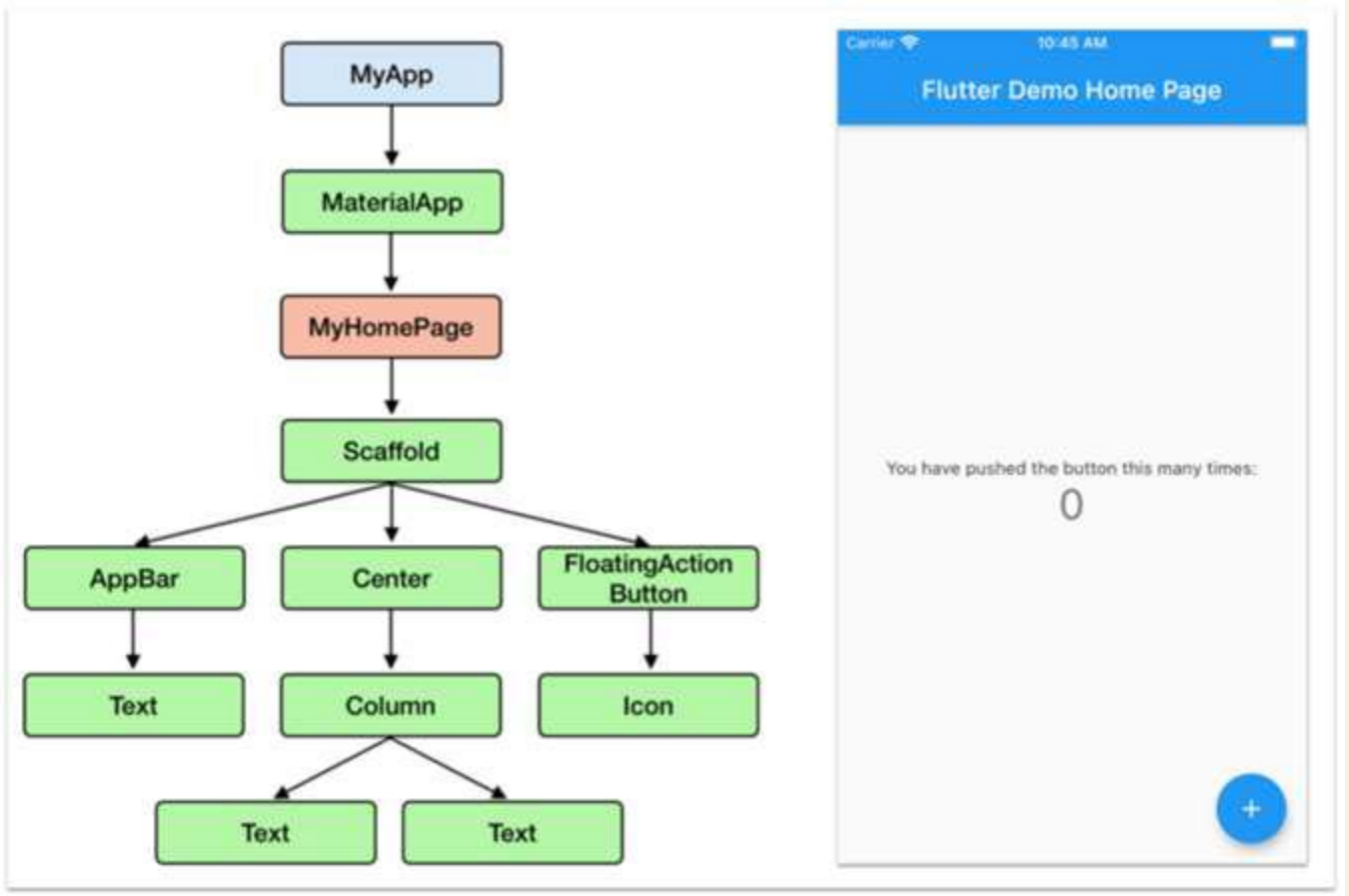
Understanding the Widget Tree



Widget tree

- The widget tree is how you create your UI; you position widgets within each other to build simple and complex layouts. Since just about everything in the Flutter framework is a widget, and as you start nesting them, the code can become harder to follow.



INTRODUCTION TO WIDGETS

The following are the widgets (usable only with **Material Design**) that you'll use to create the **full** widget tree projects:

- **Scaffold**: Implements the Material Design **visual layout**.
- **AppBar**: Implements the **toolbar** at the top of the screen.
- **CircleAvatar**: Usually used to show a **rounded** user profile **photo**.
- **Divider**: Draws a **horizontal line** with **padding** above and below.

If the app you are creating is using **Cupertino**, you can use the following widgets instead.

- **CupertinoPageScaffold**: Implements the **iOS** visual **layout for a page**. It works with **CupertinoNavigationBar**.
- **CupertinoTabScaffold**: Implements the **iOS visual layout**. This is used to navigate **multiple pages**, with the tabs at the **bottom** of the screen.
- **CupertinoNavigationBar**: Implements the **iOS** visual layout **toolbar** at the **top** of the screen.

MATERIAL DESIGN	CUPERTINO
Scaffold	CupertinoPageScaffold CupertinoTabScaffold
AppBar	CupertinoNavigationBar
CircleAvatar	n/a
Divider	n/a

The following widgets can be used with both **Material Design** and **Cupertino**:

- **SingleChildScrollview**: This adds **vertical** or **horizontal** scrolling ability to a single child widget.
- **Padding**: This adds **left**, **top**, **right**, and **bottom** padding.
- **Column**: This displays a **vertical list** of child widgets.
- **Row**: This displays a **horizontal list** of child widgets.
- **Container**: This widget can be used as an **empty placeholder** (invisible) or can specify **height**, **width**, **color**, **transform** (rotate, move, skew), and many.
- **Expanded**: This **expands** and **fills the available space for the child** widget that belongs to a **Column** or **Row** widget.
- **Text**: The Text widget is a great way to **display labels** on the screen. It can be configured to be a **single** line or **multiple** lines.
- **Stack**: lets you **stack widgets on top of each other** and use a Positioned (optional) widget to align each child of the Stack for the layout needed.
- **Positioned**: The Positioned widget works with the **Stack widget** to control child **positioning** and **size**.

main.dart | home.dart

lib > home.dart > _HomeState > build

```dart
class Home extends StatefulWidget {
  @override
  _HomeState createState() => _HomeState();
}

class _HomeState extends State<Home> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Widget Tree'),
      ), // AppBar
      body: SafeArea(
        //.....STEP 2
        child: SingleChildScrollView(
          child: Padding(
            padding: EdgeInsets.all(16.0),
            child: Column(
              //.....STEP 3
              children: <Widget>[
                Row(
                  //.....STEP 4
                  children: <Widget>[
                    Container(
                      color: Colors.yellow,
```

Widget Inspector

Widget Tree

- [root]
  - MyApp
    - MaterialApp
      - Home
        - Scaffold
          - SafeArea
            - SingleChildScrollView
              - Padding
                - Column
                  - Row
                    - Container
                    - Padding
                    - Expanded
                      - Container
                    - Padding
                    - Container
                  - Padding
                  - Row
                    - Column

flutter.dev/devtools/inspector · x64 (64 bit) android ·

Creating the Full Widget Tree

Create a new Flutter project called **ch5_widget_tree**. You can follow the instructions from **LECTRUE 4**.

For this project, you need to **create** the **pages folder** only.

1. Open the **home.dart** file.
2. Add to the **Scaffold body** property a **SafeArea** widget with the **child** property set to a **SingleChildScrollview**. Add a **Padding** widget as a **child** of the **SingleChildScrollView**. Set the **padding** property to **EdgeInsets.all(16.0).**

```
body: SafeArea(
    child: SingleChildScrollView(
        child: Padding(
            padding: EdgeInsets.all(16.0),
        ),
    ),
),
```

3. Add to the **Padding child** property a **Column** widget with the **children** property set to a **Row**.

```
body: SafeArea(
    child: SingleChildScrollView(
        child: Padding(
            padding: EdgeInsets.all(16.0),
            child: Column(
               children: <Widget>[
                  Row(
                        children: <Widget>[
                        ],
                   ),
               ],
            ),
        ),
    ),
),
```

4. Add to the **Row children** widgets in this order: **Container**, **Padding**, **Expanded**, **Padding**, **Container**, and **Padding**. You are not done adding widgets; in the next step, you'll add a Row widget with multiple nested widgets.

```
Row(
      children: <Widget>[
            Container(
                  color: Colors.yellow,
                  height: 40.0,
                  width: 40.0,
            ),
            Padding(padding: EdgeInsets.all(16.0),),
            Expanded(
                child: Container(
                    color: Colors.amber,
                    height: 40.0,
                   width: 40.0,
                ),
            ),
            Padding(padding: EdgeInsets.all(16.0),),
            Container(
                 color: Colors.brown,
                 height: 40.0,
                 width: 40.0,
             ),
      ],
)
```

5. Add a **Padding** widget to create a space before the **next Row** widget.

```
Padding(padding: EdgeInsets.all(16.0),),
```

6. Add a **Row** widget with the **children** property set to a **Column**. Add to the **Column children** a **Container**, **Padding**, **Container**, **Padding**, **Container**, **Divider**, **Row**, **Divider** and **Text**.

```
Row(
      children: <Widget>[
            Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  mainAxisSize: MainAxisSize.max,
                 children: <Widget>[
                        Container(
                              color: Colors.yellow,
                              height: 60.0,
                              width: 60.0,
                        ),
                       Padding(padding: EdgeInsets.all(16.0),),
                       Container(
                              color: Colors.amber,
                              height: 40.0,
                              width: 40.0,
                       ),
                       Padding(padding: EdgeInsets.all(16.0),),
                       Container(
                              color: Colors.brown,
                              height: 20.0,
                             width: 20.0,
                       ),
```

```
Divider(),
                  Row(
                        children: <Widget>[
                       // Next step we'll add more widgets
                              ],
                         ),
                        Divider(),
                        Text('End of the Line'),
                     ],
               ),
         ],
),
```

7. Modify the **last Row widget** (from **step 6**) and set the **children** property to a **CircleAvatar** with a **child** as a **Stack**. Add to the **Stack children** property three **Container** widgets.

```
Row(
    children: <Widget>[
        CircleAvatar(
            backgroundColor: Colors.lightGreen,
            radius: 100.0,
            child: Stack(
              children: <Widget>[
                  Container(
                      height: 100.0,
                      width: 100.0,
                      color: Colors.yellow,
                   ),
                  Container(
                      height: 60.0,
                      width: 60.0,
                      color: Colors.amber,
                  ),
                  Container(
                      height: 40.0,
                      width: 40.0,
                      color: Colors.brown,
                  ),
              ],
            ),
        ),
    ],
),
```

8. After the **Stack** widget (from **step 7**), add a **Divider** widget and then a **Text** widget with a string of '**End of the Line'**.

```
Divider(),
Text('End of the Line'),
```

THE FULL CODE

Lib/home.dart

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


class Home extends StatefulWidget {
   @override
    _HomeState createState() => _HomeState();
}
class _HomeState extends State<Home> {
   @override
   Widget build(BuildContext context) {
         return Scaffold(
            appBar: AppBar(
                title: Text('Widget Tree'),
            ),
```

```
body: SafeArea( //……..……STEP 2
        child: SingleChildScrollView(
            child: Padding(
             padding: EdgeInsets.all(16.0),
               child: Column( //……..……STEP 3
                children: <Widget>[
               Row( //…….STEP 4
                  children: <Widget>[
                     Container(
                      color: Colors.yellow,
                      height: 40.0,
                      width: 40.0,
                         ),
```

```
Padding(padding: EdgeInsets.all(16.0),),
                        Expanded(
                                    child: Container(
                                        color: Colors.amber,
                                        height: 40.0,
                                        width: 40.0,
                                    ),
                                ),
                                Padding(padding: EdgeInsets.all(16.0),),
                                Container(
                                    color: Colors.brown,
                                    height: 40.0,
                                    width: 40.0,
                                ),
                            ],
                        ),
Padding(padding: EdgeInsets.all(16.0),), //.....STEP 5
```

```
Row(    //.....STEP 6
                    children: <Widget>[
                       Column(
                          crossAxisAlignment:
CrossAxisAlignment.start,
                                mainAxisSize: MainAxisSize.max,
                             children: <Widget>[
                                Container(
                                      color: Colors.yellow,
                                      height: 60.0,
                                      width: 60.0,
                                ),
```

```
Padding(padding: EdgeInsets.all(16.0),),
                                    Container(
                                            color: Colors.amber,
                                            height: 40.0,
                                            width: 40.0,
                                    ),
 Divider(),
                                Row(  //.....STEP 7
                                            children: <Widget>[
                                              CircleAvatar(
                                    backgroundColor: Colors.lightGreen,
                                            radius: 100.0,
```

```
child: Stack(
                    children: <Widget>[
                        Container(
                           height: 100.0,
                           width: 100.0,
                           color: Colors.yellow,
                                  ),
                         Container(
                            height: 60.0,
                            width: 60.0,
                            color: Colors.amber,
                                   ),
                          Container(
                             height: 40.0,
                             width: 40.0,
                             color: Colors.brown,
                                                ),
                                              ],
                                           ),
```

```
    ),
                                ],
                            ),
                            Divider(),  //..... STEP 8
                            Text('End of the Line'),
                        ],
                    ),
                ],
            ),
        ],
    ),
  ),
),
  );
  }
}
```


12:46
Widget Tree
End of the Line

Flutter Inspector

- The Flutter widget inspector is a powerful tool for visualizing and exploring Flutter widget trees. The Flutter framework uses widgets as the core building block for anything from controls (such as text, buttons, and toggles), to layout (such as centering, padding, rows, and columns).

Flutter Inspector