**Flutter Widgets**

In this section, we are going to learn the concept of a widget, how to create it, and their different types available in the Flutter framework. We have learned earlier that everything in Flutter is a widget.

If you are familiar with React or Vue.js, then it is easy to understand the Flutter.

Whenever you are going to code for building anything in Flutter, it will be inside a widget. The central purpose is to build the app out of widgets. It describes how your app view should look like with their current configuration and state. When you made any alteration in the code, the widget rebuilds its description by calculating the difference of previous and current widget to determine the minimal changes for rendering in UI of the app.

Widgets are nested with each other to build the app. It means the root of your app is itself a widget, and all the way down is a widget also. For example, a widget can display something, can define design, can handle interaction, etc.

We can create the Flutter widget like this:

 Class ImageWidget extends StatelessWidget {

          // Class Stuff

 }

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



 class MyHomePage extends StatelessWidget {

   MyHomePage({Key key, this.title}) : super(key: key);

   // This widget is the home page of your application.

   final String title;



   @override

   Widget build(BuildContext context) {

     return Scaffold(

       appBar: AppBar(

         title: Text(this.title),

       ),

       body: Center(

         child: Text('Hello World'),

       ),

     );

   }

 }

## Types of Widget

We can split the Flutter widget into two categories:

1. Visible (Output and Input)
2. Invisible (Layout and Control)

### Visible widget

The visible widgets are related to the user input and output data. Some of the important types of this widget are:

**Text**

A Text widget holds some text to display on the screen. We can align the text widget by using **textAlign** property, and style property allow the customization of Text that includes font, font weight, font style, letter spacing, color, and many more. We can use it as like below code snippets.

1. new Text(
2. 'Hello, Javatpoint!',
3. textAlign: TextAlign.center,
4. style: new TextStyle(fontWeight: FontWeight.bold),
5. )

This widget allows you to perform some action on click. Flutter does not allow you to use the Button widget directly; instead, it uses a type of buttons like a **FlatButton** and a **RaisedButton**. We can use it as like below code snippets.

1. //FlatButton Example
2. new FlatButton(
3. child: Text("Click here"),
4. onPressed: () {
5. // Do something here
6. },
7. ),
9. //RaisedButton Example
10. new RaisedButton(
11. child: Text("Click here"),
12. elevation: 5.0,
13. onPressed: () {
14. // Do something here
15. },
16. ),

In the above example, the **onPressed** property allows us to perform an action when you click the button, and **elevation** property is used to change how much it stands out.

This widget allows you to perform some action on click. Flutter does not allow you to use the Button widget directly; instead, it uses a type of buttons like a **FlatButton** and a **RaisedButton**. We can use it as like below code snippets.

1. //FlatButton Example
2. new FlatButton(
3. child: Text("Click here"),
4. onPressed: () {
5. // Do something here
6. },
7. ),
9. //RaisedButton Example
10. new RaisedButton(
11. child: Text("Click here"),
12. elevation: 5.0,
13. onPressed: () {
14. // Do something here
15. },
16. ),

In the above example, the **onPressed** property allows us to perform an action when you click the button, and **elevation** property is used to change how much it stands out.

This widget allows you to perform some action on click. Flutter does not allow you to use the Button widget directly; instead, it uses a type of buttons like a **FlatButton** and a **RaisedButton**. We can use it as like below code snippets.

1. //FlatButton Example
2. new FlatButton(
3. child: Text("Click here"),
4. onPressed: () {
5. // Do something here
6. },
7. ),
9. //RaisedButton Example
10. new RaisedButton(
11. child: Text("Click here"),
12. elevation: 5.0,
13. onPressed: () {
14. // Do something here
15. },
16. ),

In the above example, the **onPressed** property allows us to perform an action when you click the button, and **elevation** property is used to change how much it stands out.

This widget allows you to perform some action on click. Flutter does not allow you to use the Button widget directly; instead, it uses a type of buttons like a **FlatButton** and a **RaisedButton**. We can use it as like below code snippets.

1. //FlatButton Example
2. new FlatButton(
3. child: Text("Click here"),
4. onPressed: () {
5. // Do something here
6. },
7. ),
9. //RaisedButton Example
10. new RaisedButton(
11. child: Text("Click here"),
12. elevation: 5.0,
13. onPressed: () {
14. // Do something here
15. },
16. ),

In the above example, the **onPressed** property allows us to perform an action when you click the button, and **elevation** property is used to change how much it stands out.

This widget allows you to perform some action on click. Flutter does not allow you to use the Button widget directly; instead, it uses a type of buttons like a **FlatButton** and a **RaisedButton**. We can use it as like below code snippets.

1. //FlatButton Example
2. new FlatButton(
3. child: Text("Click here"),
4. onPressed: () {
5. // Do something here
6. },
7. ),
9. //RaisedButton Example
10. new RaisedButton(
11. child: Text("Click here"),
12. elevation: 5.0,
13. onPressed: () {
14. // Do something here
15. },
16. ),

In the above example, the **onPressed** property allows us to perform an action when you click the button, and **elevation** property is used to change how much it stands out.