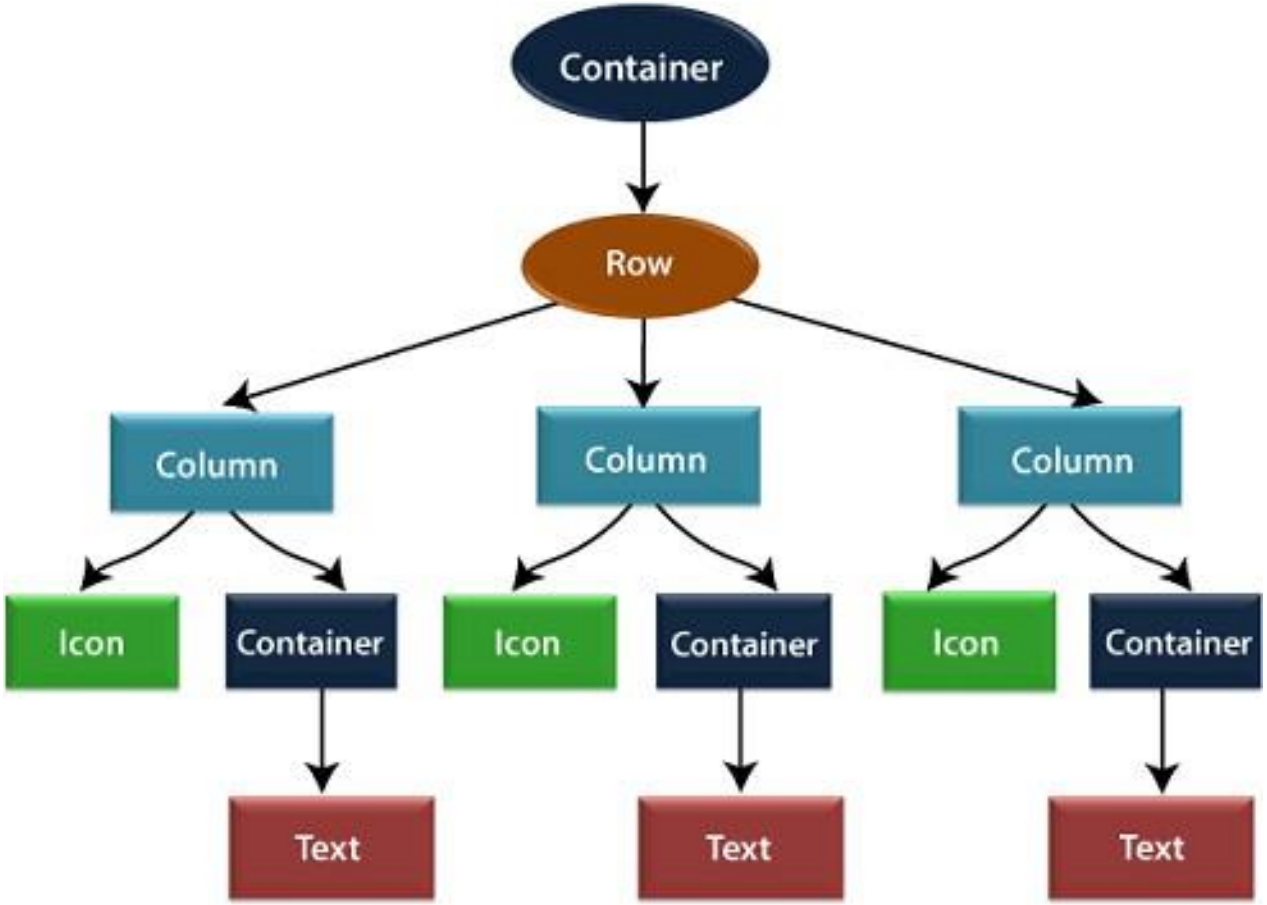
Flutter Layouts

Girma B.

Layouts

- The main concept of the layout mechanism is the widget.
- We know that flutter assume everything as a widget.
- So the image, icon, text, and even the layout of your app are all widgets.
- Here, some of the things you do not see on your app UI, such as rows, columns, and grids that arrange, constrain, and align the visible widgets are also the widgets.
- Flutter allows us to create a layout by composing multiple widgets to build more complex widgets.

Layouts


Container
Row
Column
Column
Column
Icon
Container
Icon
Container
Icon
Container
Text
Text
Text

Container

- In the above image, the container is a widget class that allows us to customize the child widget.
- It is mainly used to add borders, padding, margins, background color, and many more. Here, the text widget comes under the container for adding margins.
- The entire row is also placed in a container for adding margin and padding around the row. Also, the rest of the UI is controlled by properties such as color, text.style, etc.

Layout a widget

- Let us learn how we can create and display a simple widget. The following steps show how to layout a widget:
- Step 1: First, you need to select a Layout widget.
- Step 2: Next, create a visible widget.
- Step 3: Then, add the visible widget to the layout widget.
- Step 4: Finally, add the layout widget to the page where you want to display.

Widget Types

- We can categories the layout widget into two types:
  - Single Child Widget
  - Multiple Child Widget

tusharkute.com

Single Child Widget

- The single child layout widget is a type of widget, which can have only one child widget inside the parent layout widget.
- These widgets can also contain special layout functionality. Flutter provides us many single child widgets to make the app UI attractive.
- If we use these widgets appropriately, it can save our time and makes the app code more readable. The list of different types of single child widgets are:
- Container: It is the most popular layout widget that provides customizable options for painting, positioning, and sizing of widgets.

tusharkute.com

Container

- Container: It is the most popular layout widget that provides customizable options for painting, positioning, and sizing of widgets.

```
Center(
    child: Container(
      margin: const EdgeInsets.all(15.0),
      color: Colors.blue,
      width: 42.0,
      height: 42.0,
    ),
  )
```

tusharkute.com

Padding

- Padding: It is a widget that is used to arrange its child widget by the given padding. It contains EdgeInsets and EdgeInsets.fromLTRB for the desired side where you want to provide padding.

```
const Greetings(
      child: Padding(
        padding: EdgeInsets.all(14.0),
        child: Text('Hello JavaTpoint!'),
      ),
  )
```

Align

- Align: It is a widget, which aligns its child widget within itself and sizes it based on the child's size. It provides more control to place the child widget in the exact position where you need it.

```
Center(
 child: Container(
  height: 110.0,
  width: 110.0,
  color: Colors.blue,
  child: Align(
   alignment: Alignment.topLeft,
   child: FlutterLogo(
    size: 50,
   ),
  ),
 ),
)
```

Align

- SizedBox: This widget allows you to give the specified size to the child widget through all screens.

```
SizedBox(
  width: 300.0,
  height: 450.0,
  child: const Card(child: Text('Hello MITU!')),
)
```

Aspect Ratio

- AspectRatio: This widget allows you to keep the size of the child widget to a specified aspect ratio.

```
AspectRatio(
    aspectRatio: 5/3,
    child: Container(
    color: Colors.bluel,
    ),
),
```

tusharkute.com

ConstrainedBox

- ConstrainedBox: It is a widget that allows you to force the additional constraints on its child widget. It means you can force the child widget to have a specific constraint without changing the properties of the child widget.

```
ConstrainedBox(
  constraints: new BoxConstraints(
    minHeight: 150.0,
    minWidth: 150.0,
    maxHeight: 300.0,
    maxWidth: 300.0,
  ),
  child: new DecoratedBox(
    decoration: new BoxDecoration(color: Colors.red),
  ),
),
```

FittedBox

- FittedBox: It scales and positions the child widget according to the specified fit.

- Example

tusharkute.com

Layouts

- FractionallySizedBox: It is a widget that allows to sizes of its child widget according to the fraction of the available space.
- IntrinsicHeight and IntrinsicWidth: They are a widget that allows us to sizes its child widget to the child's intrinsic height and width.
- LimitedBox: This widget allows us to limits its size only when it is unconstrained.
- Offstage: It is used to measure the dimensions of a widget without bringing it on to the screen.
- OverflowBox: It is a widget, which allows for imposing different constraints on its child widget than it gets from a parent. In other words, it allows the child to overflow the parent widget.

Layouts

- ListView: It is the most popular scrolling widget that allows us to arrange its child widgets one after another in scroll direction.
- GridView: It allows us to arrange its child widgets as a scrollable, 2D array of widgets. It consists of a repeated pattern of cells arrayed in a horizontal and vertical layout.
- Expanded: It allows to make the children of a Row and Column widget to occupy the maximum possible area.

Layouts

- Table: It is a widget that allows us to arrange its children in a table based widget.
- Flow: It allows us to implements the flow-based widget.
- Stack: It is an essential widget, which is mainly used for overlapping several children widgets. It allows you to put up the multiple layers onto the screen. The following example helps to understand it.

Example

- Stack and multi child

Thank you