# Lab 07

Instructor: Sidra Khatoon E-mail: skhatoon[@uit.edu](mailto:ad@uit.edu)

# Objective

# The objective of lab is to building a UI with padding and Image widgets, Creating a static layout with

# Rows and Columns.

**Student Information**

|  |  |
| --- | --- |
| **Student Name** |  |
| **Student ID** |  |
| **Date** |  |

**Assessment**

|  |  |
| --- | --- |
| **Marks Obtained** |  |
| **Remarks** |  |
| **Signature** |  |

# Objective

# The objective of lab is to building a UI with padding and Image widgets, Creating a static layout with

# Rows and Columns

# Instructions

You have to perform the following tasks yourselves. Raise your hand if you face any difficulty in understanding and solving these tasks. **Plagiarism** is an abhorrent practice and you should not engage in it.

# How to Submit?

Submit lab work using Teams.

**Image Widgets:**

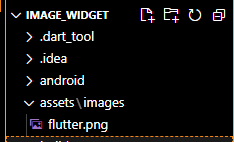
In this section, we are going to see how we can display images in Flutter. When you create an app in Flutter, it includes both code and assets (resources). An asset is a file, which is bundled and deployed with the app and is accessible at runtime. The asset can include static data, configuration files, icons, and images. The Flutter supports many image formats, such as JPEG, WebP, PNG, GIF, animated WebP/GIF, BMP, and WBMP.

How to display the image in Flutter?

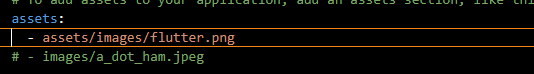
To display an image in Flutter, do the following steps:

Step 1: First, we need to create a new folder inside the root of the Flutter project and named it assets. We can also give it any other name if you want.

Step 2: Next, inside this folder, add one image manually.

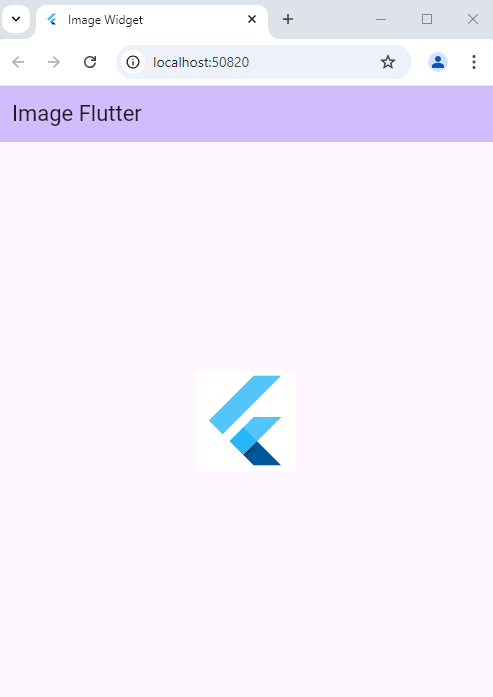


Step 3: Update the pubspec.yaml file. Suppose the image name is tablet.png, then pubspec.yaml file is:



Step 4: Finally, open the main.dart file and insert the following code.





**Flutter Row and Column:**

Row and Column are the two most important and powerful widgets in Flutter. These widgets let you align children horizontally and vertically as per the requirement. As we know that when we design any UI(User Interface) in a flutter, we need to arrange its content in the Row and Column manner so these Row and Column widgets are required when designing UI.

Key Points:

1. Row and Column widgets are the most commonly used layout patterns in the Flutter application.
2. Both may take several child widgets.
3. A child widget can also be a row or column widget.
4. We can stretch or constrain a particular children's widget.
5. Flutter also allows developers to specify how child widgets can use row and column widgets' available space.

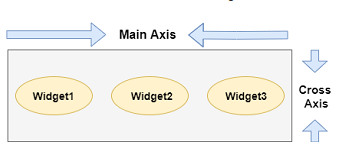
**Row Widgets:**

This widget arranges its children in a horizontal direction on the screen. In other words, it will expect child widgets in a horizontal array. If the child widgets need to fill the available horizontal space, we must wrap the children widgets in an Expanded widget.

A row widget does not appear scrollable because it displays the widgets within the visible view. So it is considered wrong if we have more children in a row which will not fit in the available space. If we want to make a scrollable list of row widgets, we need to use the ListView widget.

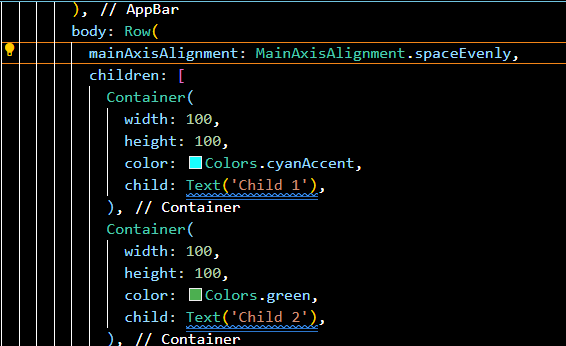
****

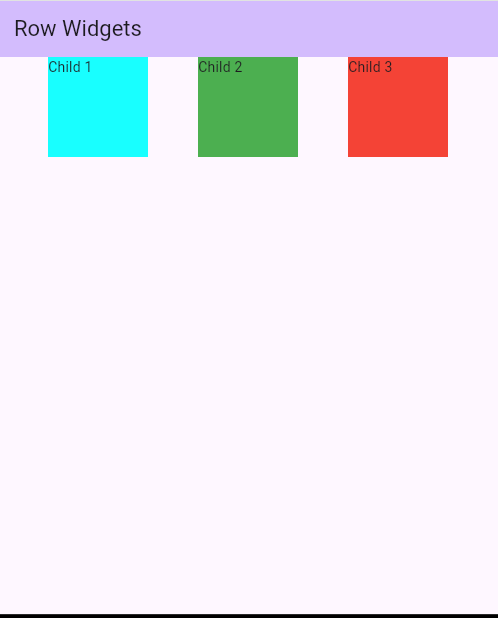
We can control how a row widget aligns its children based on our choice using the property crossAxisAlignment and mainAxisAlignment. The row's cross-axis will run vertically, and the main axis will run horizontally. See the below visual representation to understand it more clearly.



We can align the row's children widget with the help of the following properties:

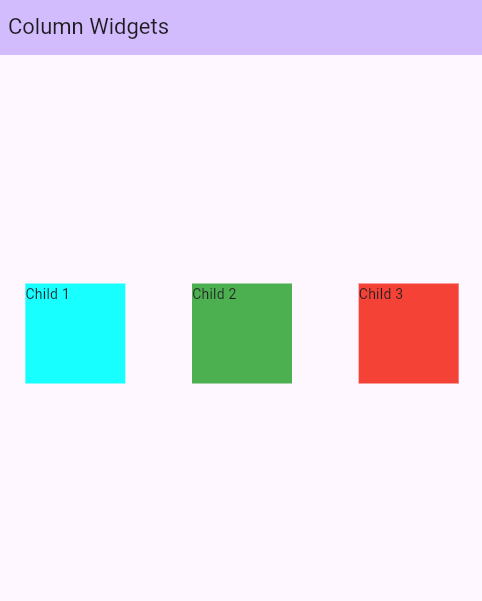
* start: It will place the children from the starting of the main axis.
* end: It will place the children at the end of the main axis.
* center: It will place the children in the middle of the main axis.
* spaceBetween: It will place the free space between the children evenly.
* spaceAround: It will place the free space between the children evenly and half of that space before and after the first and last children widget.
* spaceEvenly: It will place the free space between the children evenly and before and after the first and last children widget.





Cross axis alignment of box around the height

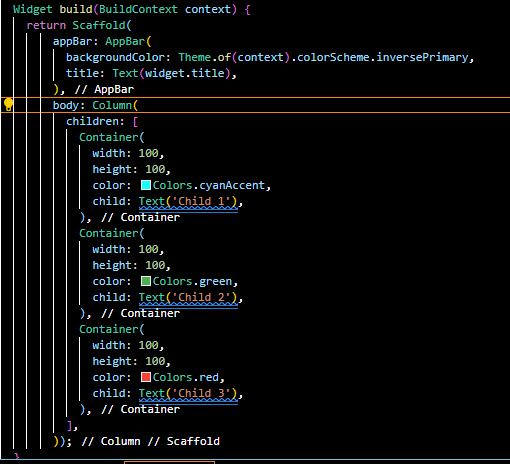




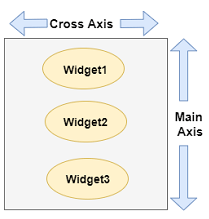
**Column Widgets:**

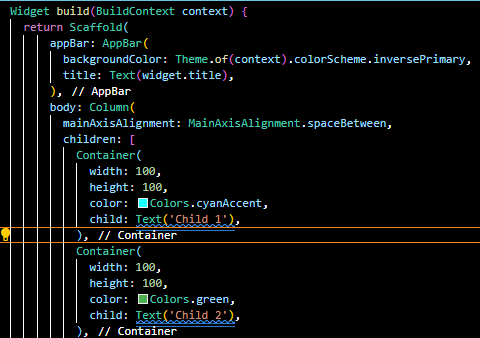
This widget arranges its children in a vertical direction on the screen. In other words, it will expect a vertical array of children widgets. If the child widgets need to fill the available vertical space, we must wrap the children widgets in an Expanded widget.

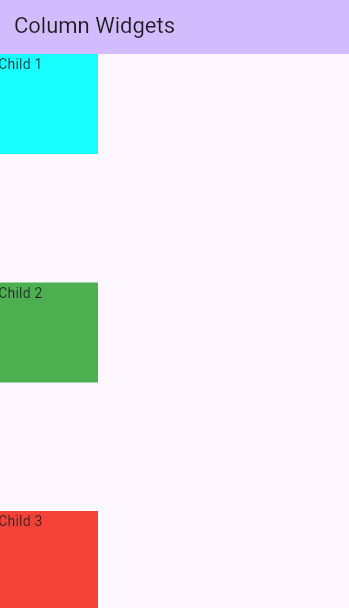
A column widget does not appear scrollable because it displays the widgets within the visible view. So it is considered wrong if we have more children in a column which will not fit in the available space. If we want to make a scrollable list of column widgets, we need to use the ListView Widget.



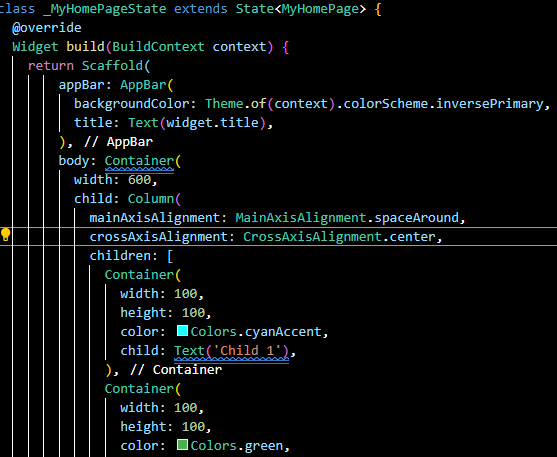
We can also control how a column widget aligns its children using the property mainAxisAlignment and crossAxisAlignment. The column's cross-axis will run horizontally, and the main axis will run vertically. The below visual representation explains it more clearly.

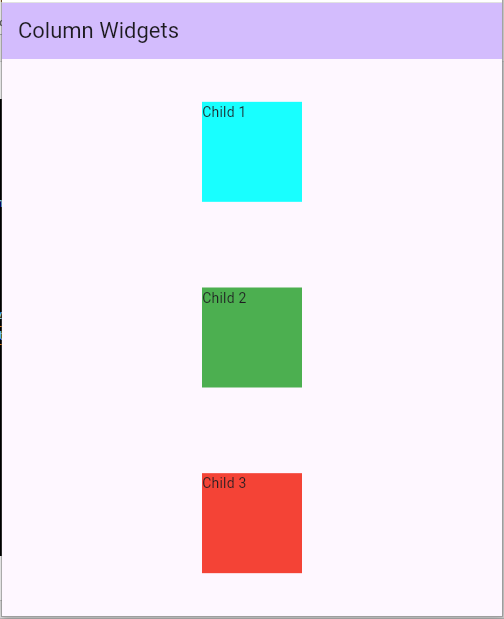






Cross axis alignment of box around the width





**Assessment:**

1. Create a simple app in which use column and row widget to align 3 or 4 images in row and column. Create 2 app one for column and other for column. Also add app bar in application. Submit me code and screenshot of output.