Flutter Layouts — Row, Column, Container

Arrange widgets on screen: Column stacks down, Row lines up, Container gives margin, padding and shape.

Introduction

A screen is not one widget — it is widgets *arranged*: a photo beside a name, buttons in a line, a card with breathing space around it. Flutter's arrangement tools are three widgets you will use in every single app: **Column** stacks children top to bottom, **Row** lines them side by side, and **Container** wraps a child with margin, padding, colour and shape. Master these three and you can build almost any screen you can sketch. *(Roman Urdu: Column neeche ki taraf, Row side by side, Container dabba — yehi teen har screen banate hain.)*

Key points

- **Column** takes a `children` list (a Dart list — D.7) and stacks them vertically, first child on top. **Row** does the same horizontally, first child on the left. Ninety percent of layout is choosing, for each group: down or across?
- **Alignment has two directions.** `mainAxisAlignment` arranges children *along* the stack (for a Column: up-down; for a Row: left-right) — `start`, `center`, `end`, `spaceBetween`, `spaceAround`. `crossAxisAlignment` arranges them *across* (for a Column: left-right).

  (Roman Urdu: Main axis = stack ki simt, cross axis = us ke aar paar. Column ka main upar-neeche, Row ka main left-right.)
- Between children, space is a widget too: `SizedBox(width: 8)` in a Row, `SizedBox(height: 12)` in a Column. Gaps are never typed as spaces in text — they are boxes.
- **Container** is the all-in-one wrapper. Remember its four layers, outside-in: **margin** (space outside the box) → **border / decoration** (the box's skin: colour, rounded corners, shadow) → **padding** (space inside, between skin and child) → **child**. The words `EdgeInsets.all(16)` mean "16 units on all four sides".
- `decoration: BoxDecoration(...)` is how a Container gets its look: `color`, `borderRadius` for rounded corners, `boxShadow` for a lifted card feel. Note: colour goes *inside* `BoxDecoration` when a decoration exists — setting both `color` and `decoration` on one Container is an error.
- Layouts **nest**: a card is often a Container holding a Row holding a Column holding two Texts. When a screen looks complex, draw its tree of Rows and Columns first — the code is just that drawing, typed.

Code examples

Example 1 — Column: three lines, stacked and centred

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

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: const Text('Column Demo')),
        body: Center(
          child: Column(
            mainAxisSize: MainAxisSize.min,
            children: const [
              Text('Pehli line', style: TextStyle(fontSize: 22)),
              SizedBox(height: 10),
              Text('Doosri line', style: TextStyle(fontSize: 22)),
              SizedBox(height: 10),
              Text('Teesri line', style: TextStyle(fontSize: 22)),
            ],
          ),
        ),
      ),
    );
  }
}
```

What it does: The Column stacks its five children — text, gap, text, gap, text — from top to bottom. `mainAxisSize: MainAxisSize.min` makes the Column only as tall as its children, so the surrounding `Center` can place the whole stack in the middle of the screen.

What you will see: three lines in the exact centre of the screen, one under another, with even gaps. Remove the `Center` and the stack jumps to the top-left — layout widgets obey their parents, and seeing that once teaches more than ten definitions.

Example 2 — Row: a rating line, side by side

```
Row(
  mainAxisSize: MainAxisSize.min,
  children: [
    const Icon(Icons.star, color: Colors.amber),
    const SizedBox(width: 6),
    const Text(
      '4.5',
      style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
    ),
    const SizedBox(width: 16),
    ElevatedButton(
      onPressed: () {},
      child: const Text('Buy'),
    ),
  ],
)
```

What it does: This Row is one line of a shopping app: a gold star, a small gap, the bold rating "4.5", a wider gap, and a Buy button. Put this Row inside `Center` in the body of the Example 1 app (replacing the Column) to run it. A Row's children never wrap to the next line — if they run out of width, Flutter reports an overflow, which the mistakes section below addresses.

What you will see: star, 4.5 and the Buy button on one horizontal line, in that order. This exact pattern — icon + number + action — appears in thousands of real product cards; you now own it.

Example 3 — The profile card: Container + Row + Column together

```
Container(
  margin: const EdgeInsets.all(16),
  padding: const EdgeInsets.all(16),
  decoration: BoxDecoration(
    color: Colors.white,
    borderRadius: BorderRadius.circular(16),
    boxShadow: const [
      BoxShadow(color: Colors.black12, blurRadius: 8, offset: Offset(0, 3)),
    ],
  ),
  child: Row(
    children: [
      const CircleAvatar(
        radius: 28,
        child: Text('A', style: TextStyle(fontSize: 22)),
      ),
      const SizedBox(width: 14),
      Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: const [
          Text(
            'Ali Raza',
            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
          ),
          Text('BS Computer Science'),
        ],
      ),
    ],
  ),
)
```

What it does: The full nesting, outside-in: a Container gives the card its margin, padding, white skin, round corners and soft shadow. Inside, a Row places the avatar circle beside a Column, and the Column stacks the name (bold) above the degree, left-aligned with `crossAxisAlignment: CrossAxisAlignment.start`. Place this Container in a Scaffold body to see it.

What you will see: a floating white card: a circle with "A", and beside it "Ali Raza" in bold with "BS Computer Science" under it. This is the skeleton of every contact, product and student card in real apps — and it is the direct ancestor of the Student Cards project in Lesson F.5.

Quick recap

- Column = down, Row = across; `children` is a list; gaps are `SizedBox` widgets.

- `mainAxisAlignment` = along the stack; `crossAxisAlignment` = across it.
- Container layers, outside-in: margin → decoration → padding → child.
- Colour lives inside `BoxDecoration` when a decoration is used.
- Complex screens are just Rows and Columns nested — draw the tree first.

Common mistakes

1. **The yellow-black overflow stripes** — a Row (or Column) whose children need more room than the screen gives shows an overflow warning. Usual cures: smaller children, `SizedBox` gaps trimmed, or wrapping a child in `Expanded` so it takes only the space that remains.
2. **Setting color twice on a Container** — `Container(color: ..., decoration: ...)` is an error. When decoration exists, the colour goes inside `BoxDecoration`. One home for the colour, always.
3. **Guessing alignment names on the wrong axis** — centring a Column's children *horizontally* is `crossAxisAlignment`, not `mainAxisAlignment`. Say the stack's direction first ("this Column runs down"), then main = down, cross = sideways. The names stop being confusing the moment the direction is spoken aloud.

Practice

1. Build a "subject row" screen: a Column of three Rows — each Row an icon, a subject name and its marks at the end. Use SizedBox gaps and one Container card around the whole Column.
2. Redesign Example 3 into a *product* card: avatar circle becomes a square Container with an icon, name becomes a product name, degree line becomes a price. Keep the same tree — only the leaves change.

Practice quiz

1. Which way do Column and Row stack their children?

- Column across, Row down
- Column down (vertical), Row across (horizontal)
- Both down
- Both across

Answer: Column down (vertical), Row across (horizontal)

2. What is the difference between mainAxisAlignment and crossAxisAlignment?

- They are the same
- mainAxisAlignment arranges along the stack, crossAxisAlignment arranges across it
- mainAxisAlignment is for Rows only
- crossAxisAlignment sets the size

Answer: mainAxisAlignment arranges along the stack, crossAxisAlignment arranges across it

3. What are the layers of a Container, outside-in?

- child, padding, decoration, margin
- margin, then border/decoration, then padding, then child
- padding, margin, child, decoration
- decoration, child, margin, padding

Answer: margin, then border/decoration, then padding, then child

---

Watch the video lesson: https://youtu.be/7Flr88LFIc8

Read online: https://scopetreasure.com/notes/flutter-layouts-row-column-container · More free tools and notes at https://scopetreasure.com