

# Module 02: Component of Flutter

Inspect Flutter Counter App

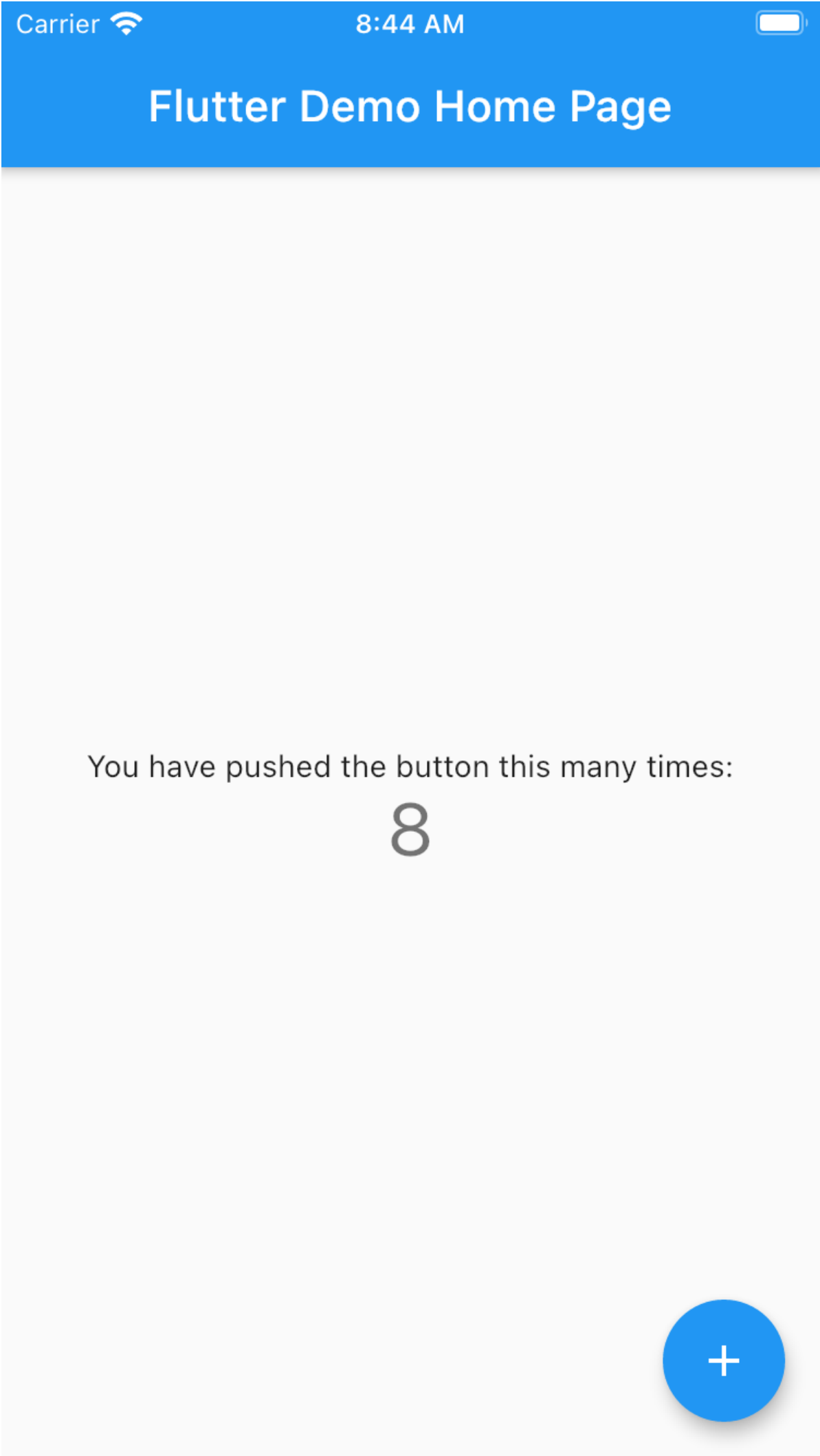
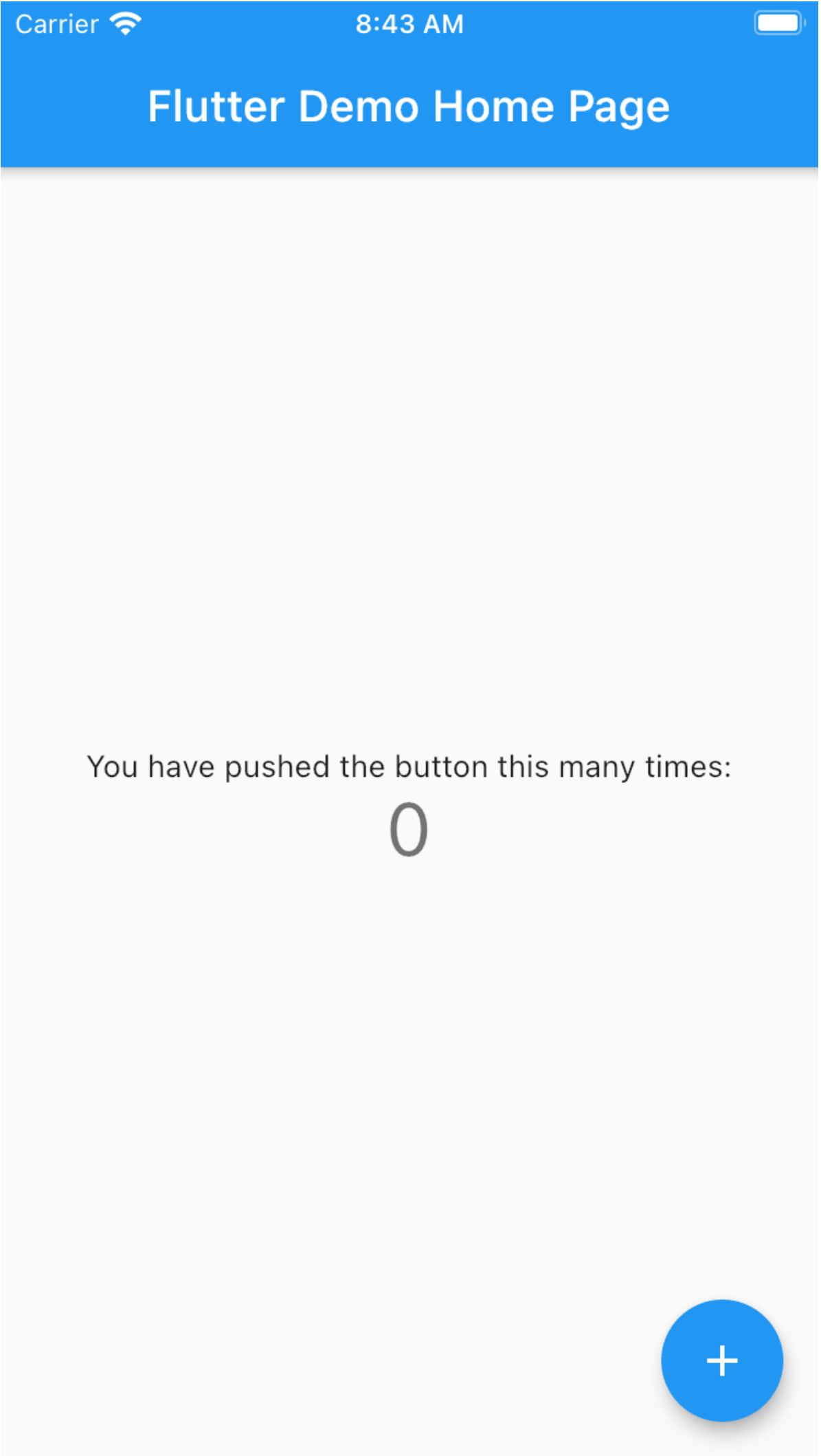
Basic layouts in Flutter

Basic Widgets in Flutter

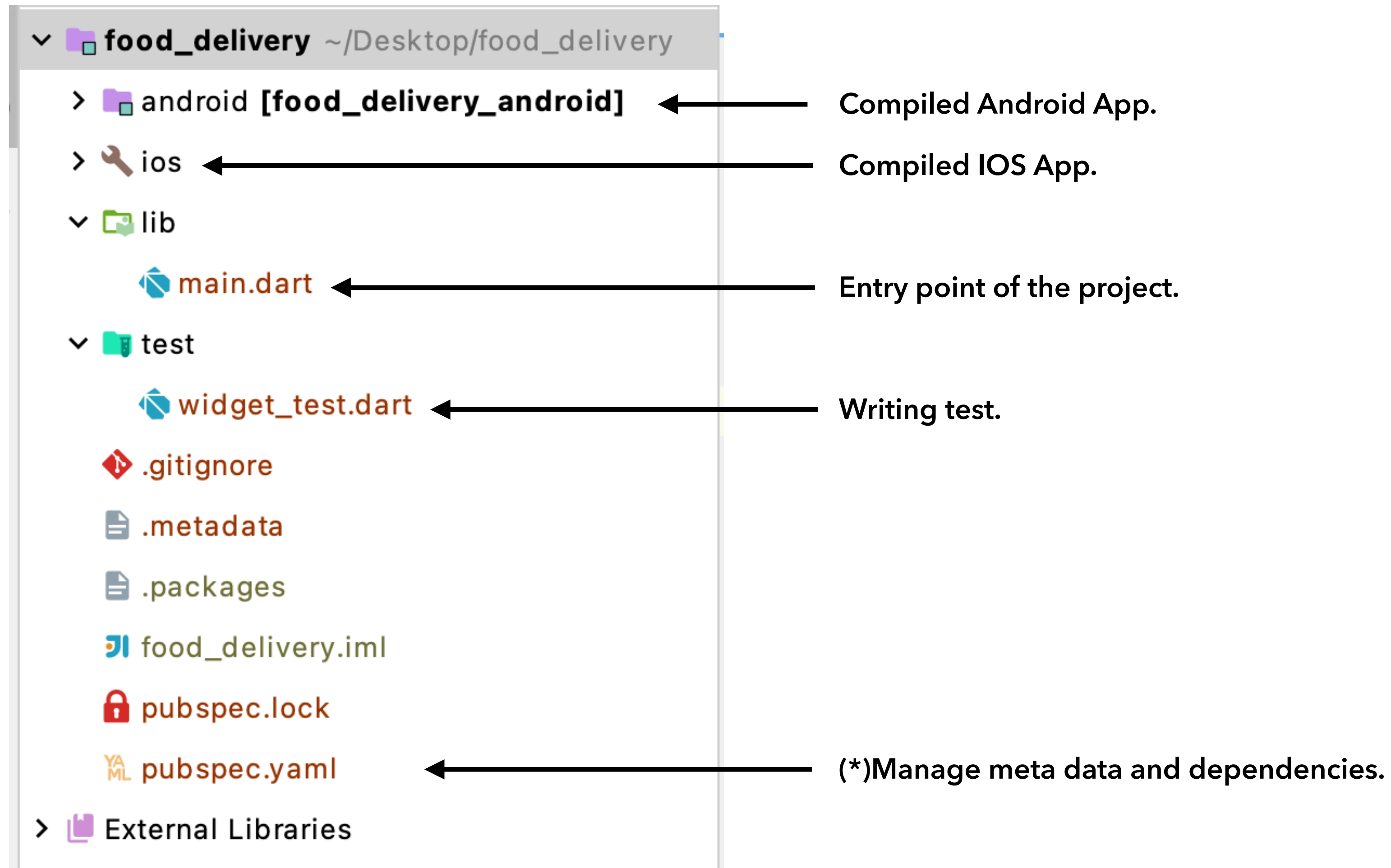
Material and Cupertino

## 2.1 Inspect Flutter Counter App

# Intro to the Counter app



# Flutter project structure



# Application entry point and import library

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

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

# The build method

```
class MyApp extends StatelessWidget {  
  // This widget is the root of your application.  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'Flutter Demo',  
      debugShowCheckedModeBanner: false,  
      theme: ThemeData(  
        primarySwatch: Colors.blue,  
        visualDensity: VisualDensity.adaptivePlatformDensity,  
      ), // ThemeData  
      home: MyHomePage(title: 'Flutter Demo Home Page'),  
    ); // MaterialApp  
  }  
}
```

- MyApp is a widget inherited StatelessWidget.
- Each widget has a method build.
- MaterialApp is a built-in of Flutter.

## 2.2 Basic layouts in Flutter



# The example

Everything in Flutter is Widget. Use Row and Column elements to create layout structure.

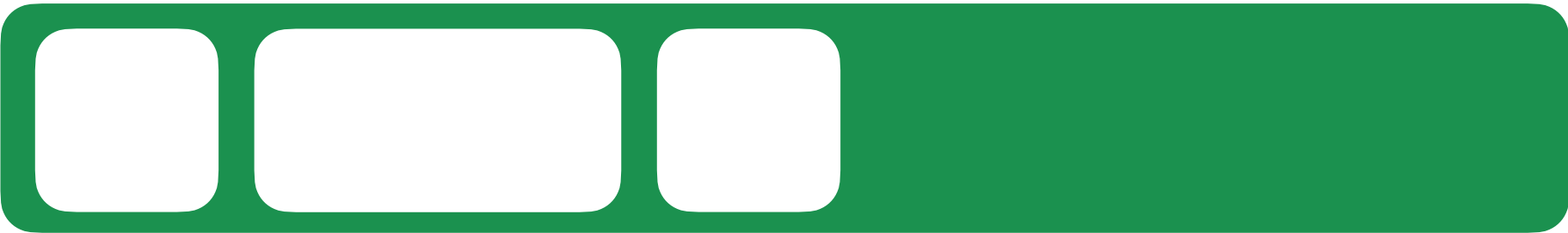


An example from Flutter.Dev



# Row - MainAxisAlignment

start



spaceBetween



center



spaceAround



end

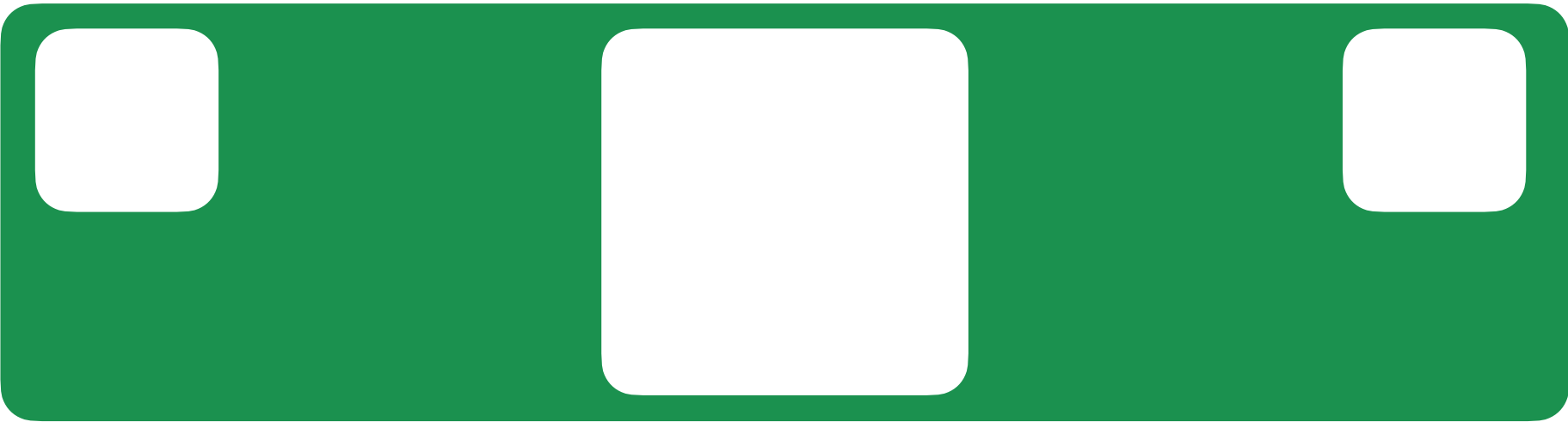


spaceEvenly

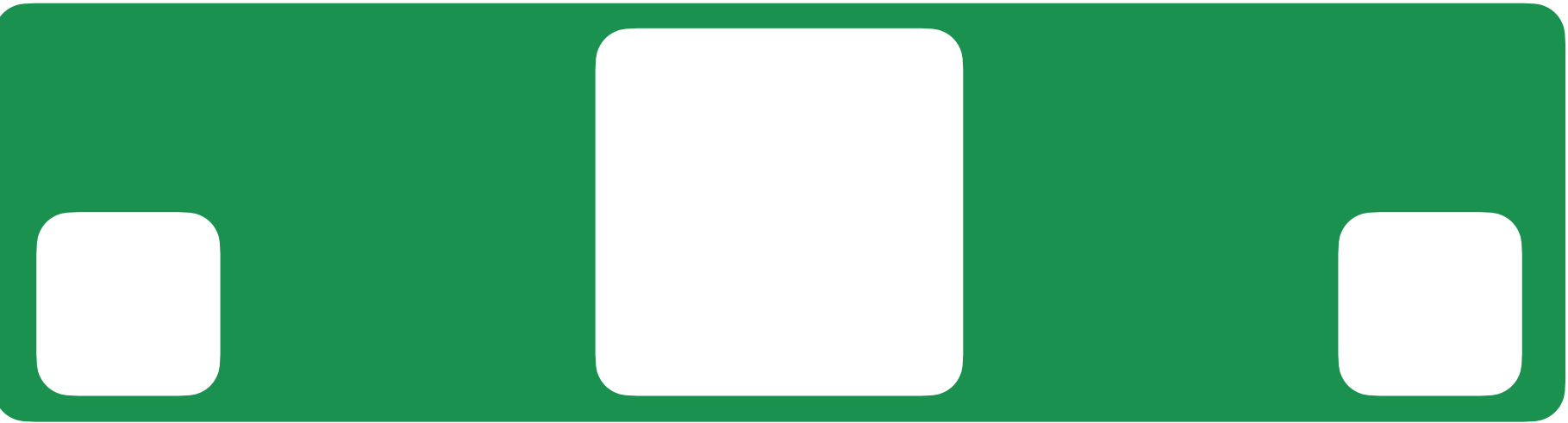


# Row - CrossAxisAlignment

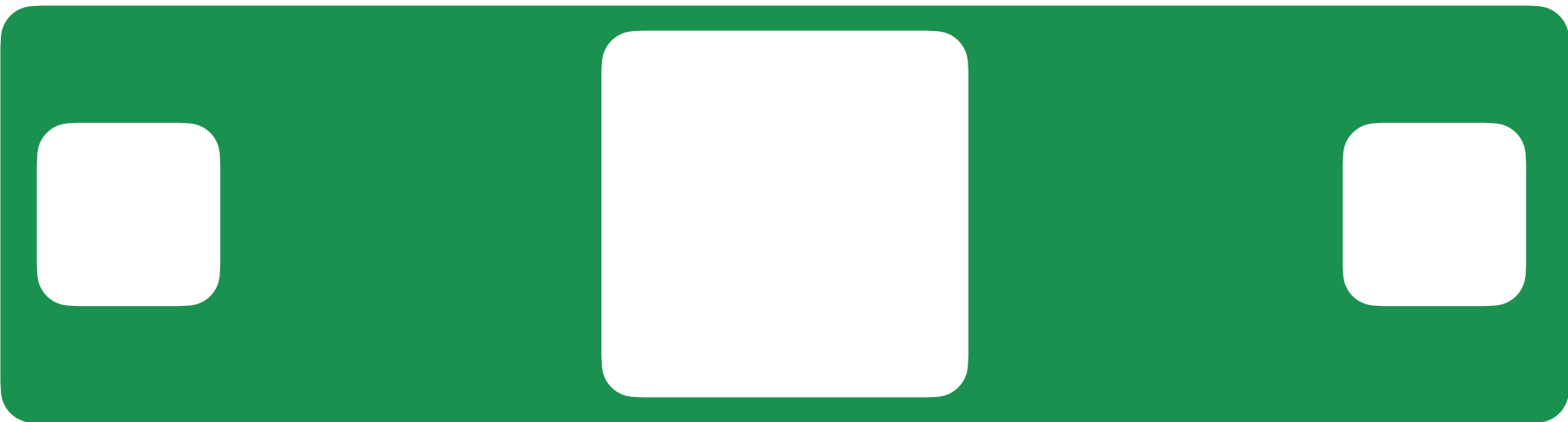
start



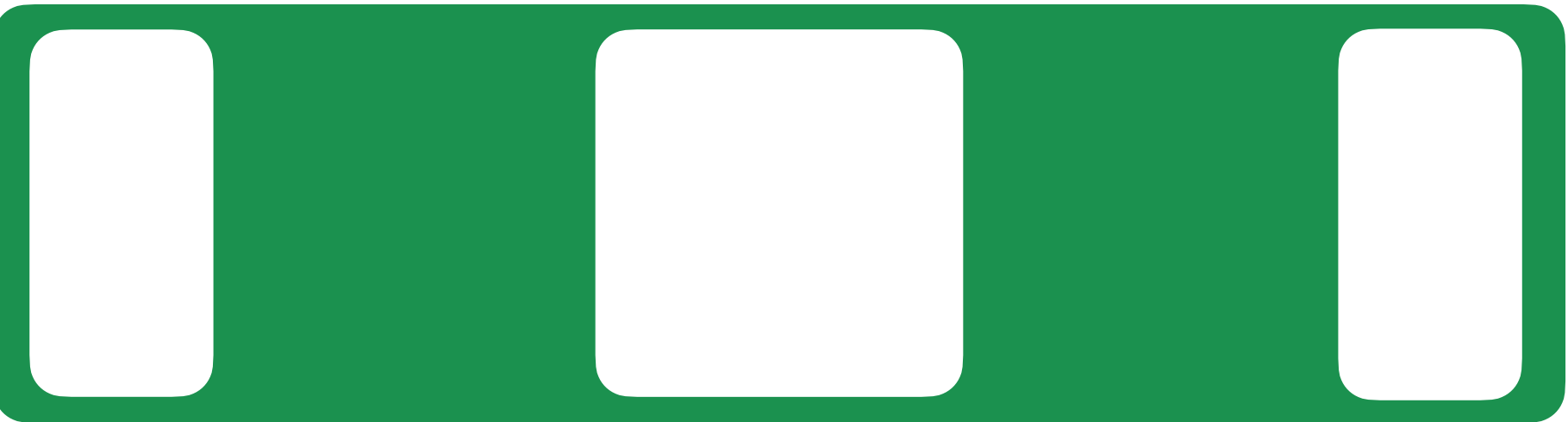
end



center



stretch

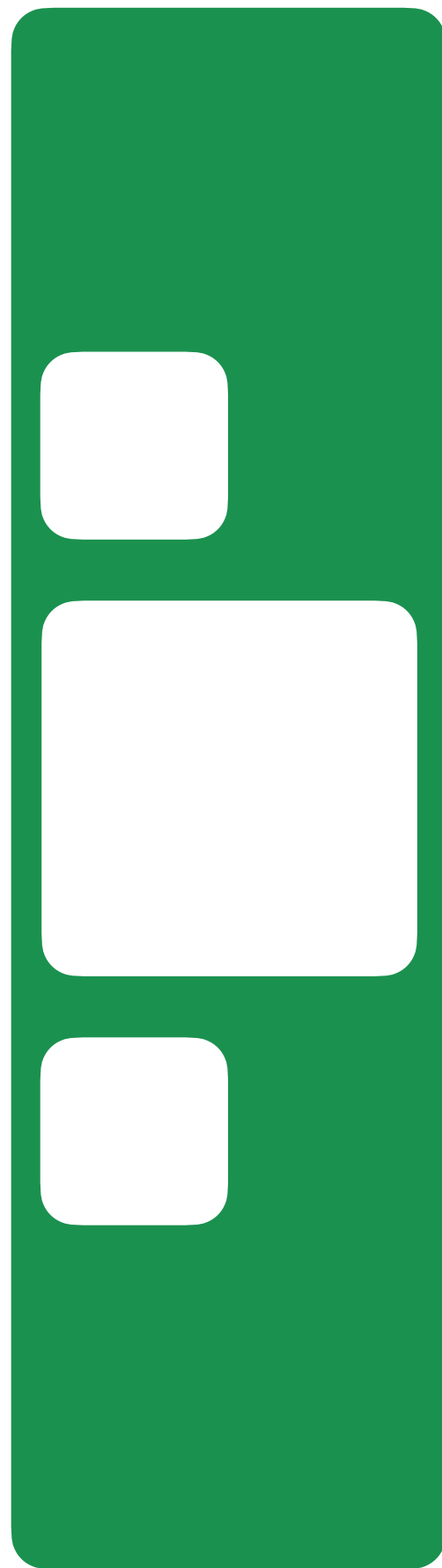


# Column - MainAxisAlignment

start



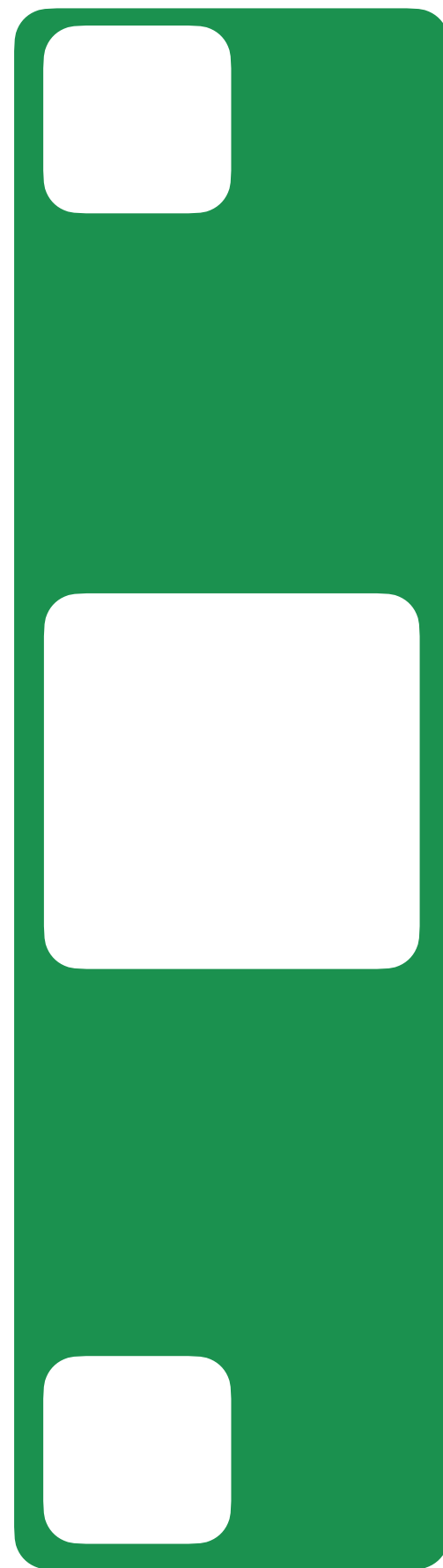
center



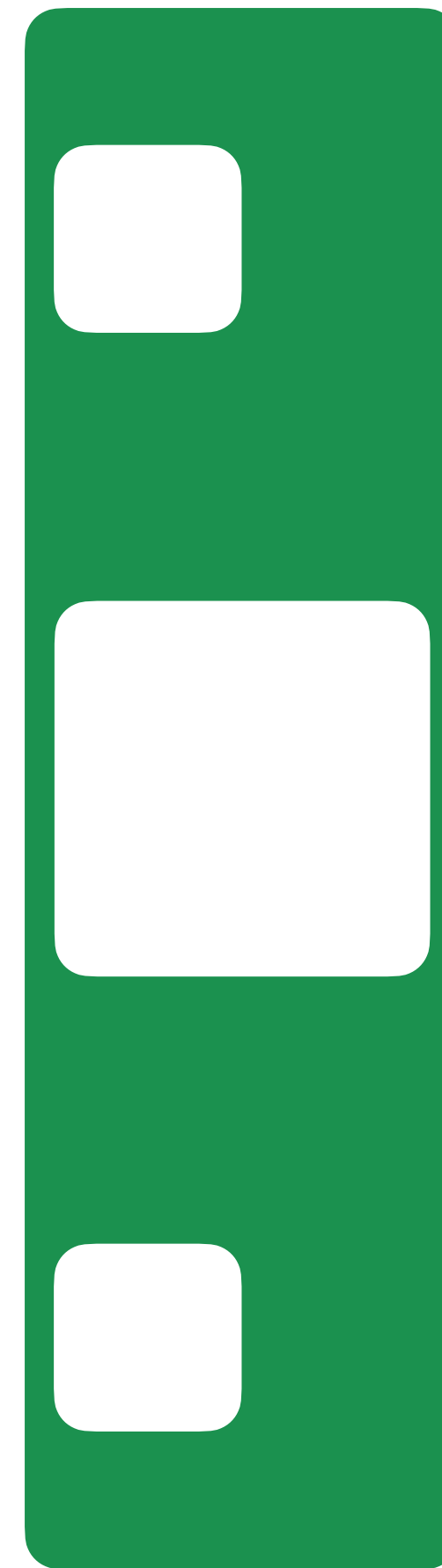
end



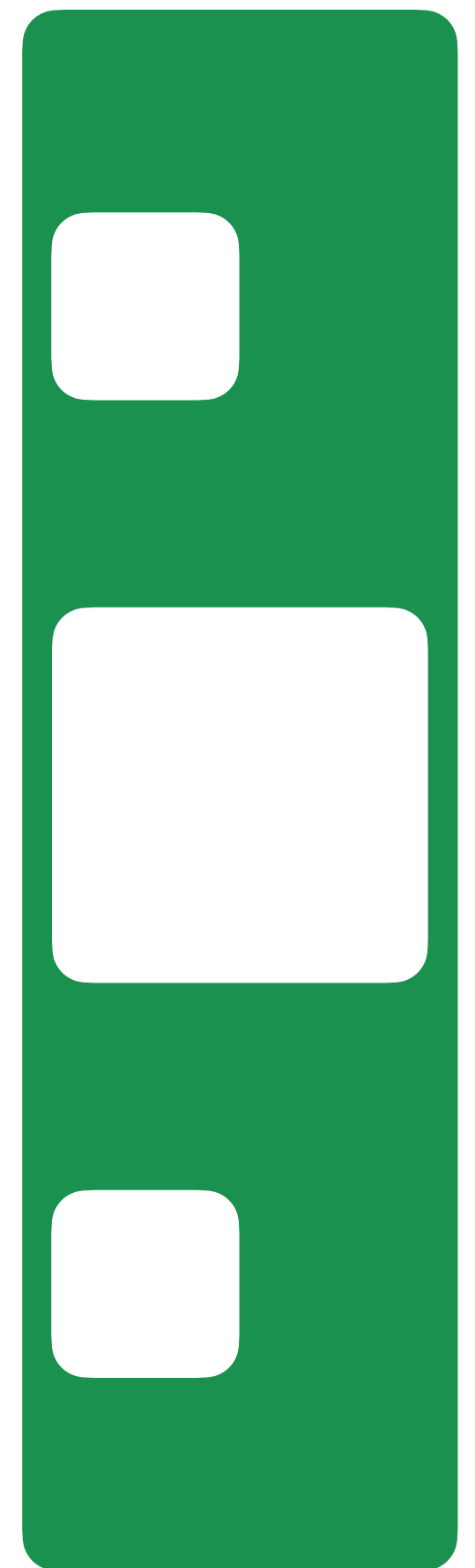
spaceBetween



spaceAround

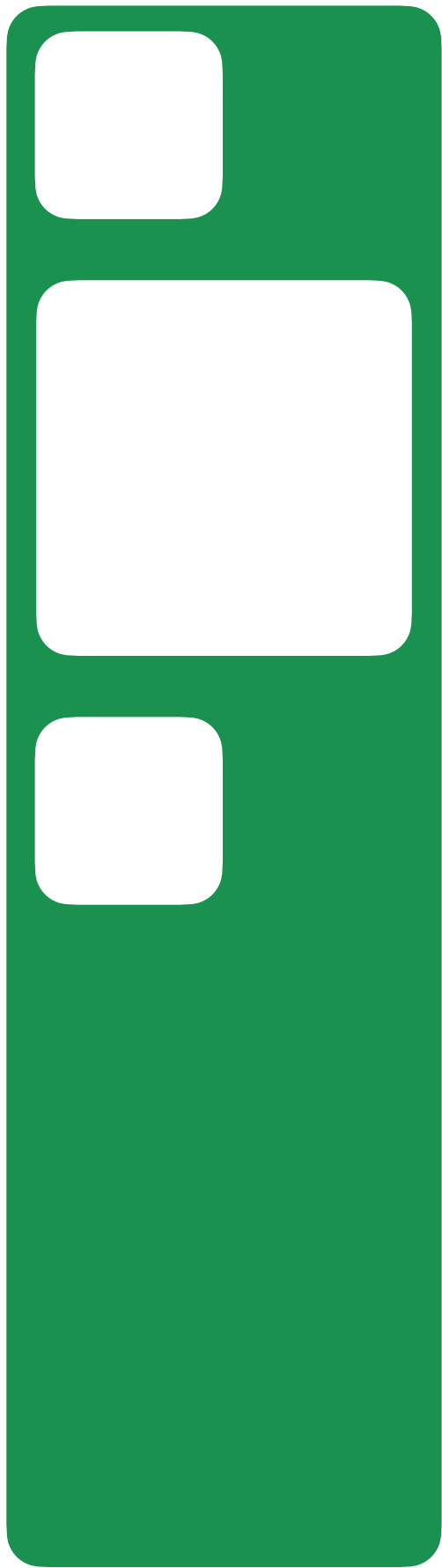


spaceEvenly



# Column - CrossAxisAlignment

start



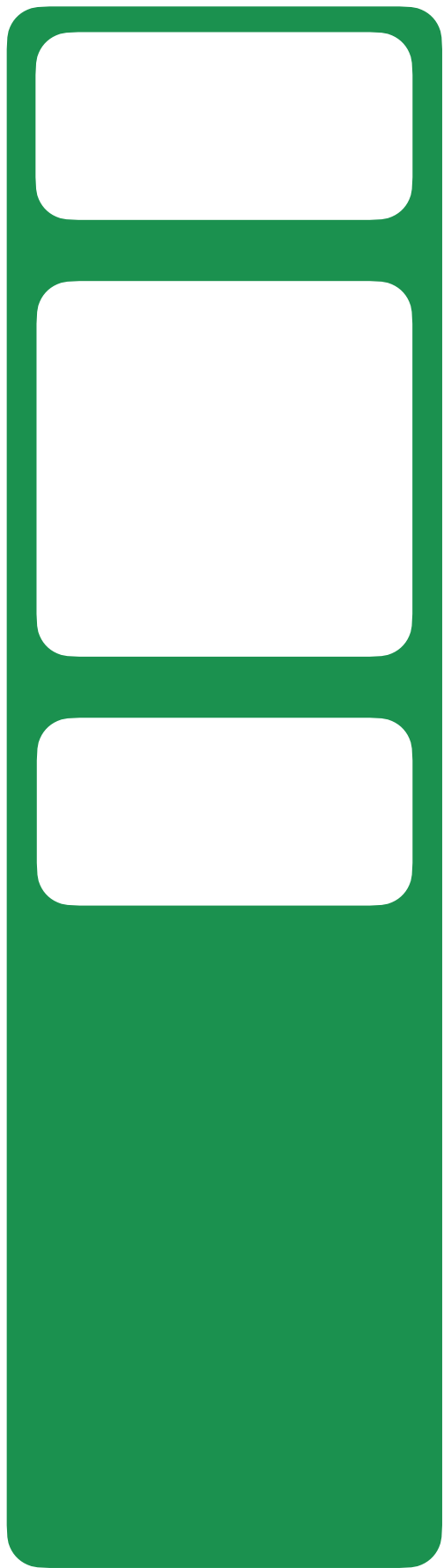
center



end



stretch



# Row/Column - MainAxisSize

Row

min

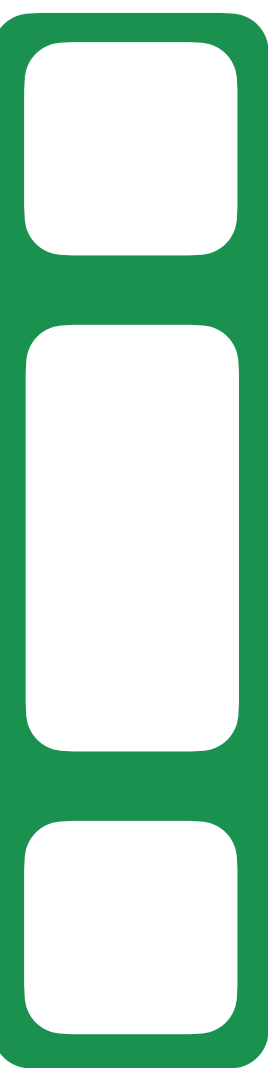


max



Column

min



max



# Expanded



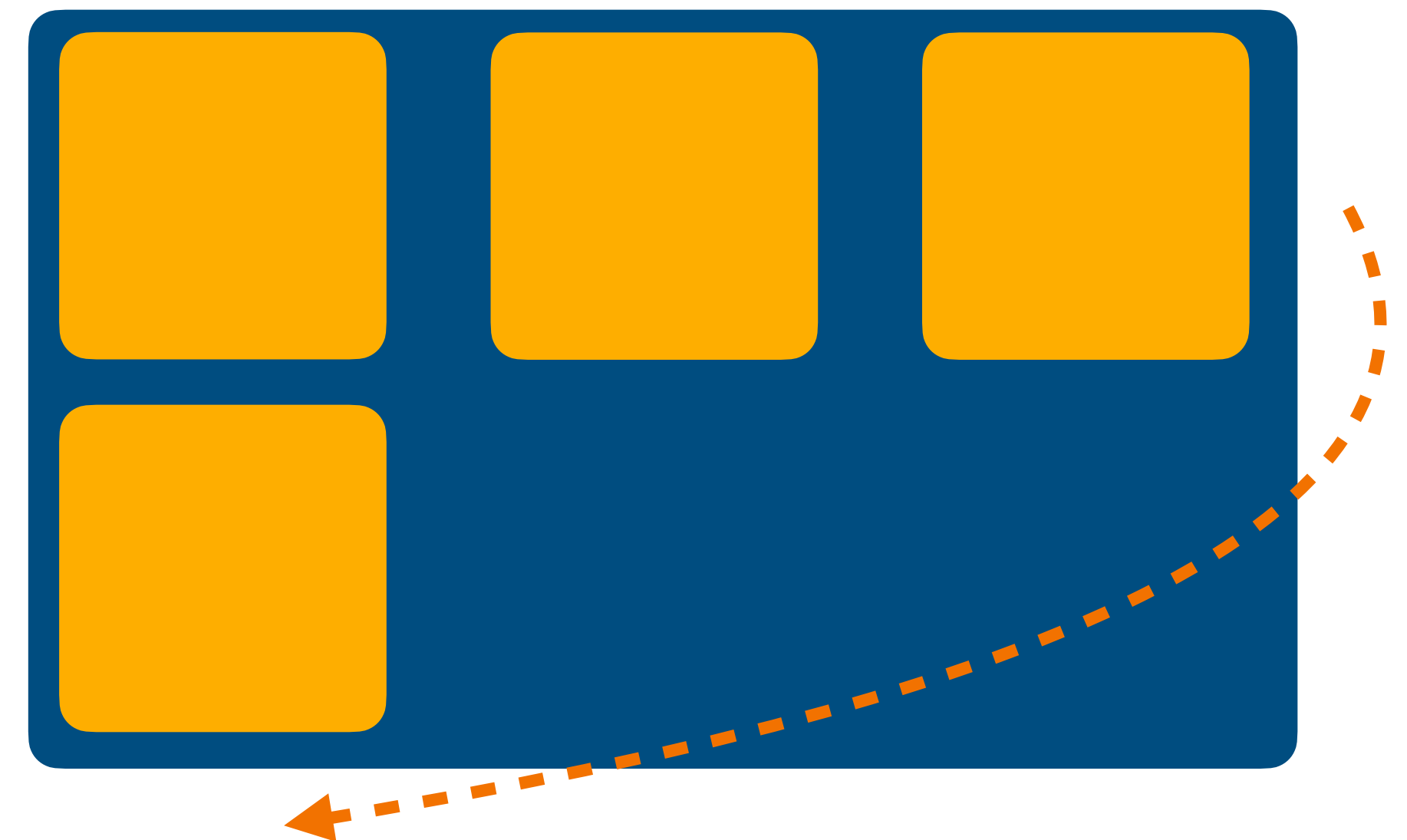
# Wrap

Without Wrap



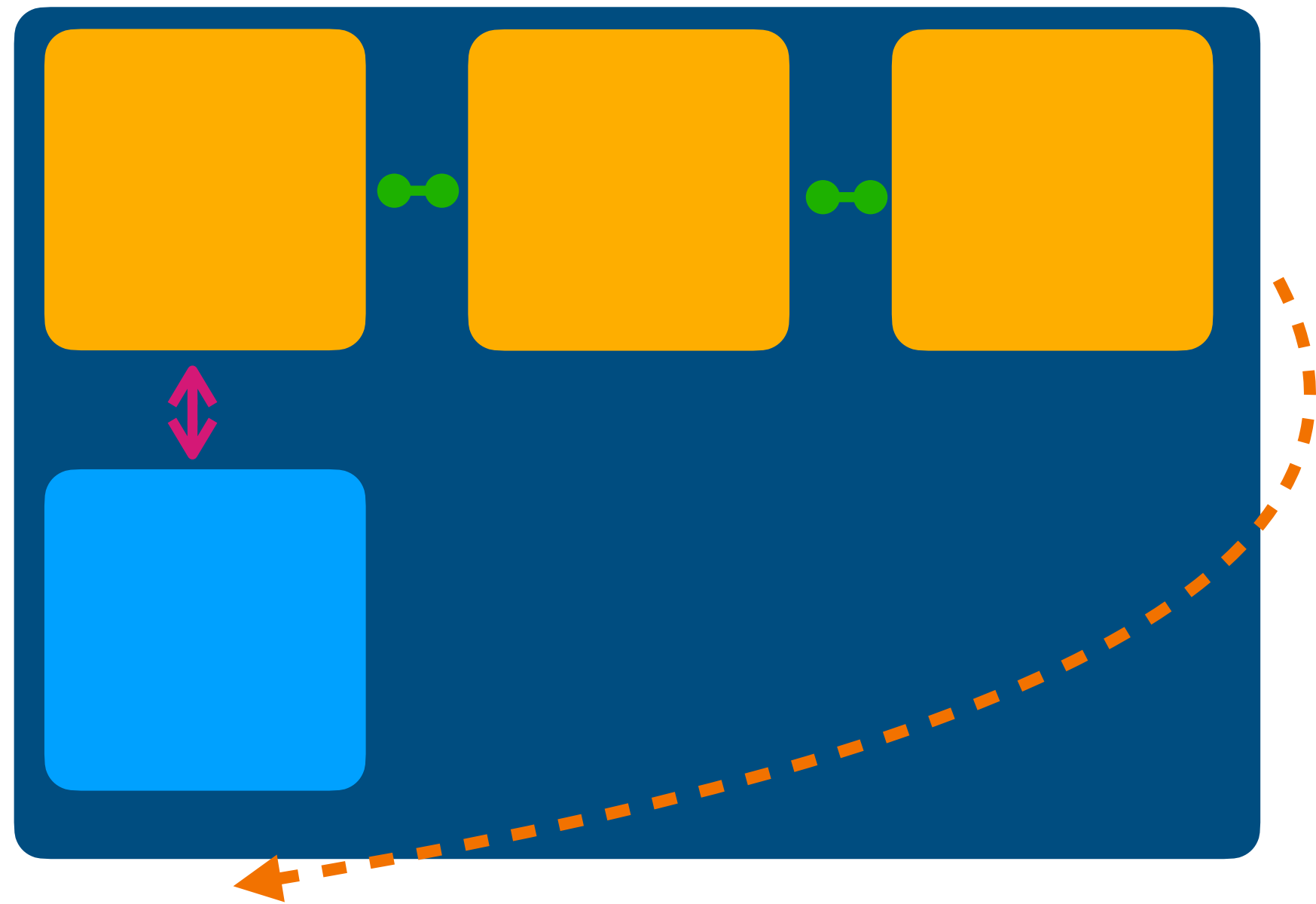
Error layout

With Wrap





# Wrap - Horizontal

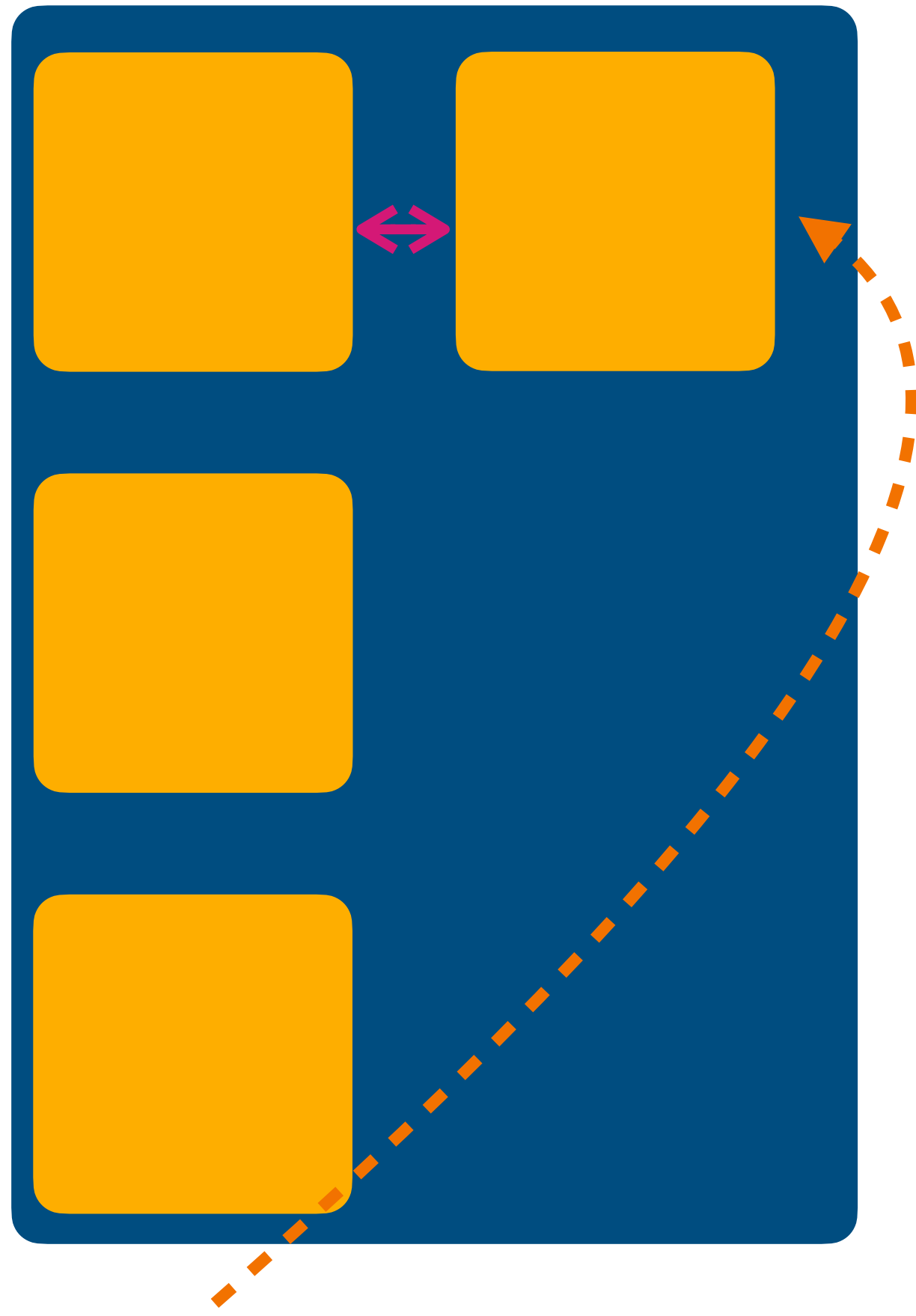


●● (spacing)

↕ (runSpacing)

.direction: Axis.horizontal

# Wrap - Vertical



●● (spacing)

↕ (runSpacing)

.direction: Axis.vertical

# Stack and Positioned

`.fit: StackFit.expand`

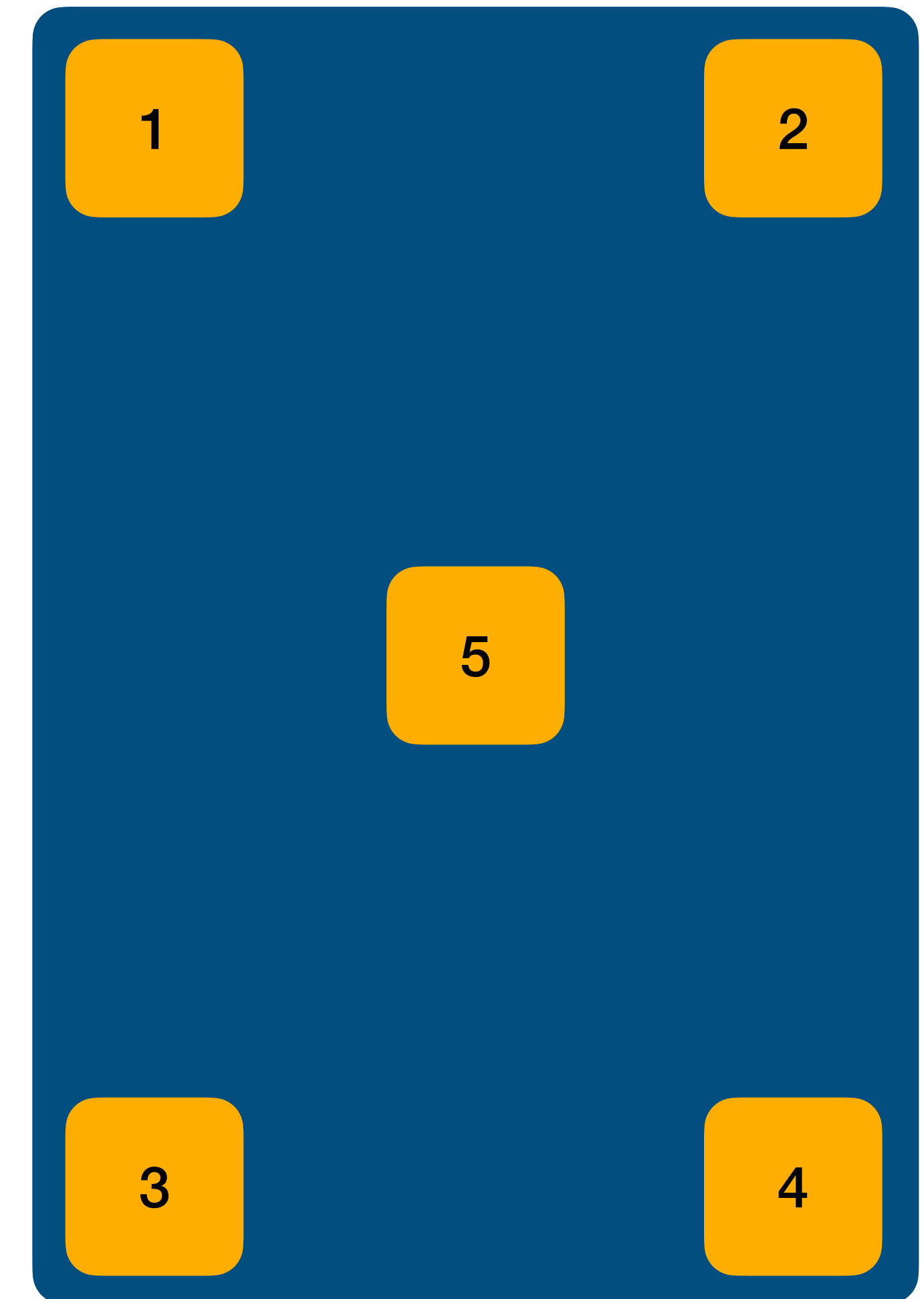
(1): Positioned (top: 0, left: 0).

(2): Positioned (top: 0, right: 0).

(3): Positioned (bottom: 0, left: 0).

(4): Positioned (bottom: 0, right: 0).

(5): Positioned (bottom: 0, right: 0, top: 0, left: 0).

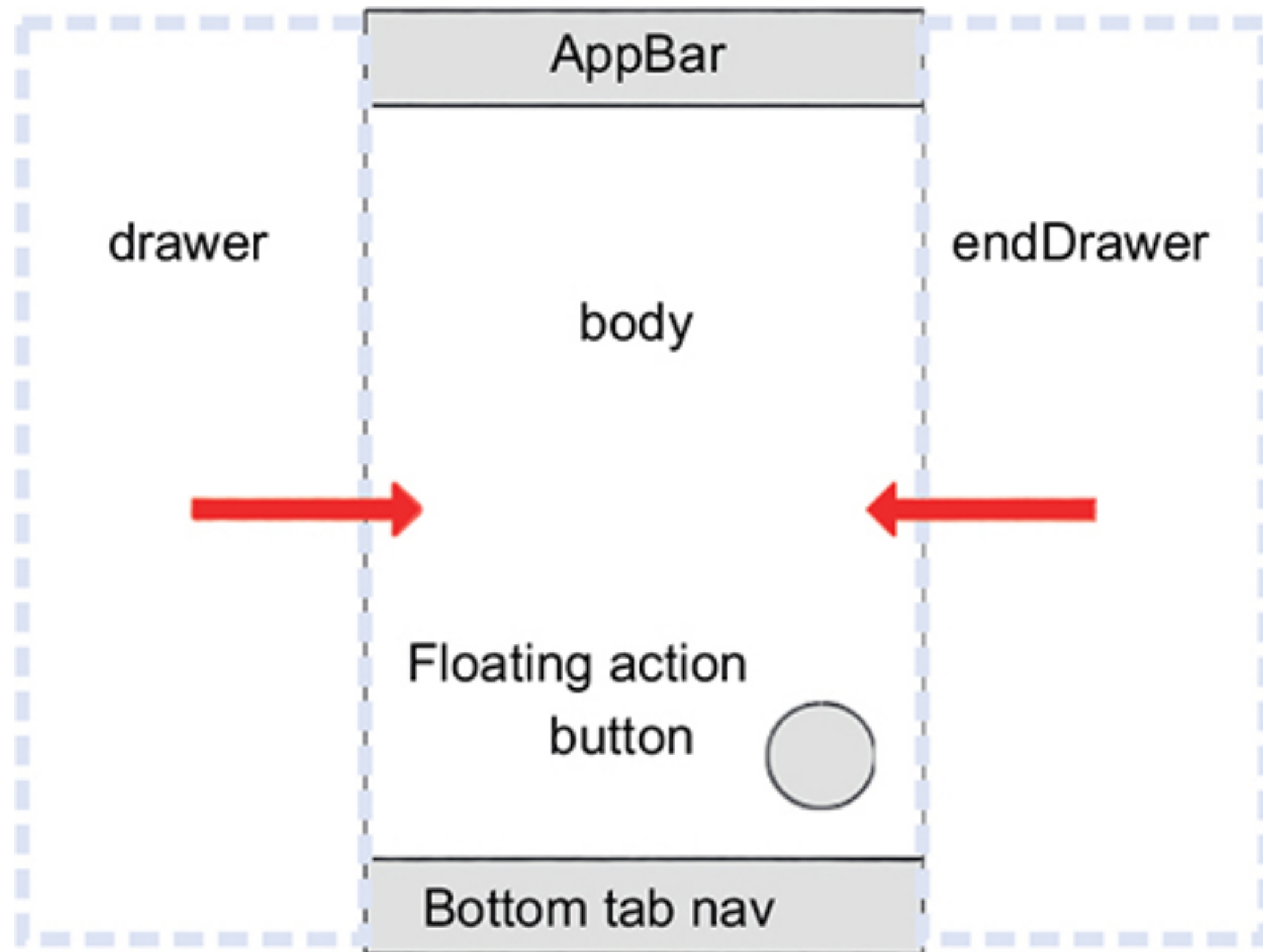


## 2.3 Basic Widgets in Flutter

# MaterialApp Widget

- It leans on Material style guidelines.
- Provides a ton of benefits that effect its entire widget sub-tree: Theme, Router, Localization, Navigator, ...
- There is no drawback to using MaterialApp, even if you don't want to use Material Design guidelines.
- Normally, it is a root of your app.

# Scaffold Widget



- Scaffold is designed to make applications (that follow Material guidelines) as easy as possible to build.
- Per the Flutter docs, it defines the “basic Material Design visual layout,” which means it can make your app look like this pretty easily.
- Constructor method: it has over 10 named arguments.

# Text

- A run of text with a single style.
- The `style widget` is optional to decoration.



```
Text(  
  'Hello, Thea! How are you feel today?',  
  style: TextStyle(  
    fontWeight: FontWeight.bold,  
  )  
)
```





# Rich Text

- A run of text with a multiply style.

```
const Text.rich(  
  TextSpan(  
    text: 'Hello',  
    children: <TextSpan>[  
      TextSpan(text: ' Thea! ', style: TextStyle(  
        fontWeight: FontWeight.bold)),  
      TextSpan(text: 'How are you feel today?'),  
    ],  
  ),  
)
```



# Image

- Several constructors are provided:
  - `Image.asset('assets/image/dart_lang.png')`
  - `Image.network('https://edu.200lab.io/dart_lang.png')`
  - `Image.file(File file)`



# ElevatedButton

- A material Design “elevated button”

```
ElevatedButton(  
  child: Text("200Lab Education"),  
  onPressed: () {  
    //do something  
  }  
);
```



# ElevatedButton.icon

- A material Design “elevated button”

```
ElevatedButton.icon(  
  icon: Icon( Icons.mobile_friendly_outlined ),  
  label: Text('200Lab Education'),  
  onPressed:(){  
    //do something  
  }  
);
```



# TextButton

```
TextButton(  
  child: Text('200Lab Education'),  
  onPressed: () {  
    //do something  
  }  
);
```



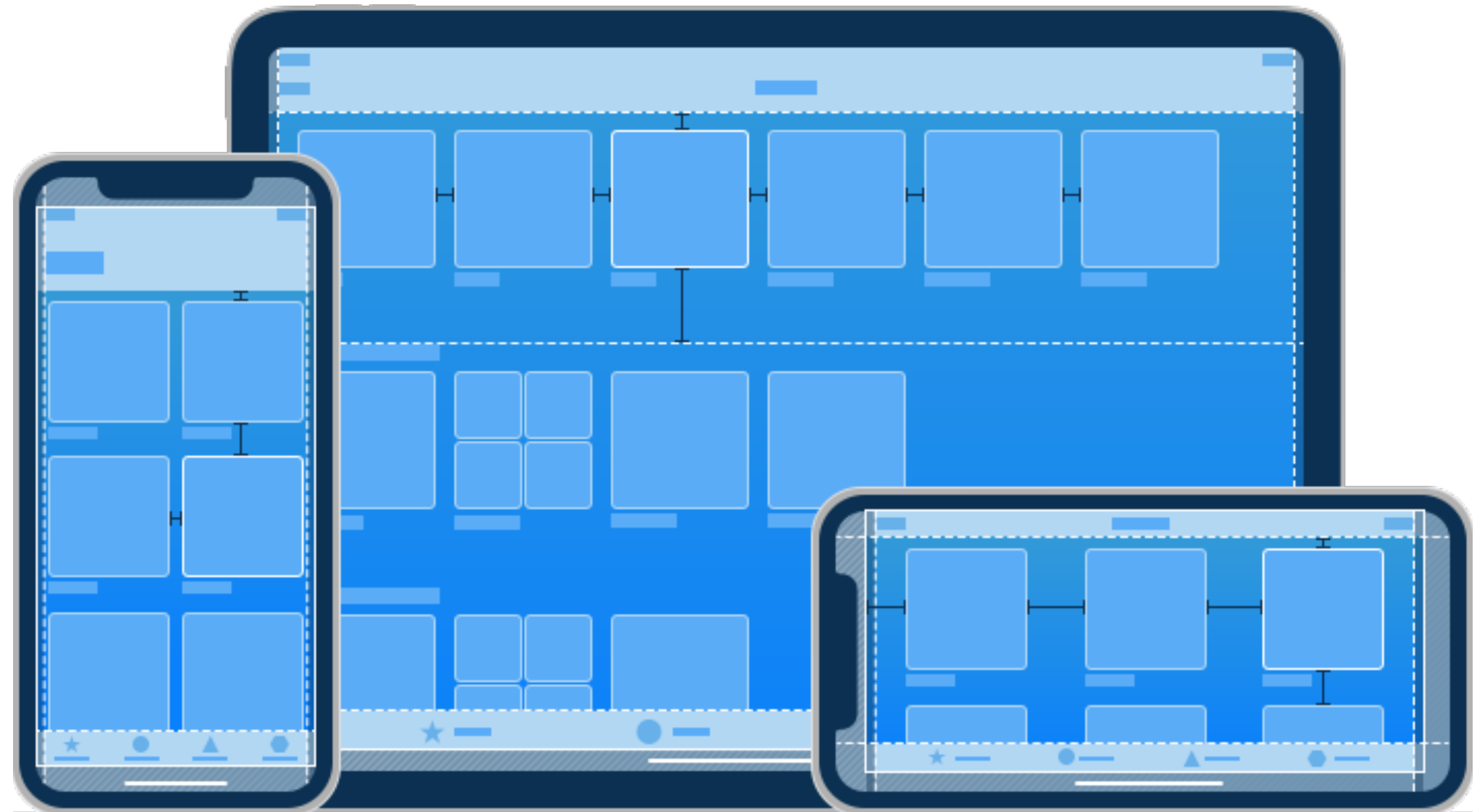
## 2.4 Material and Cupertino

# What is Material design?

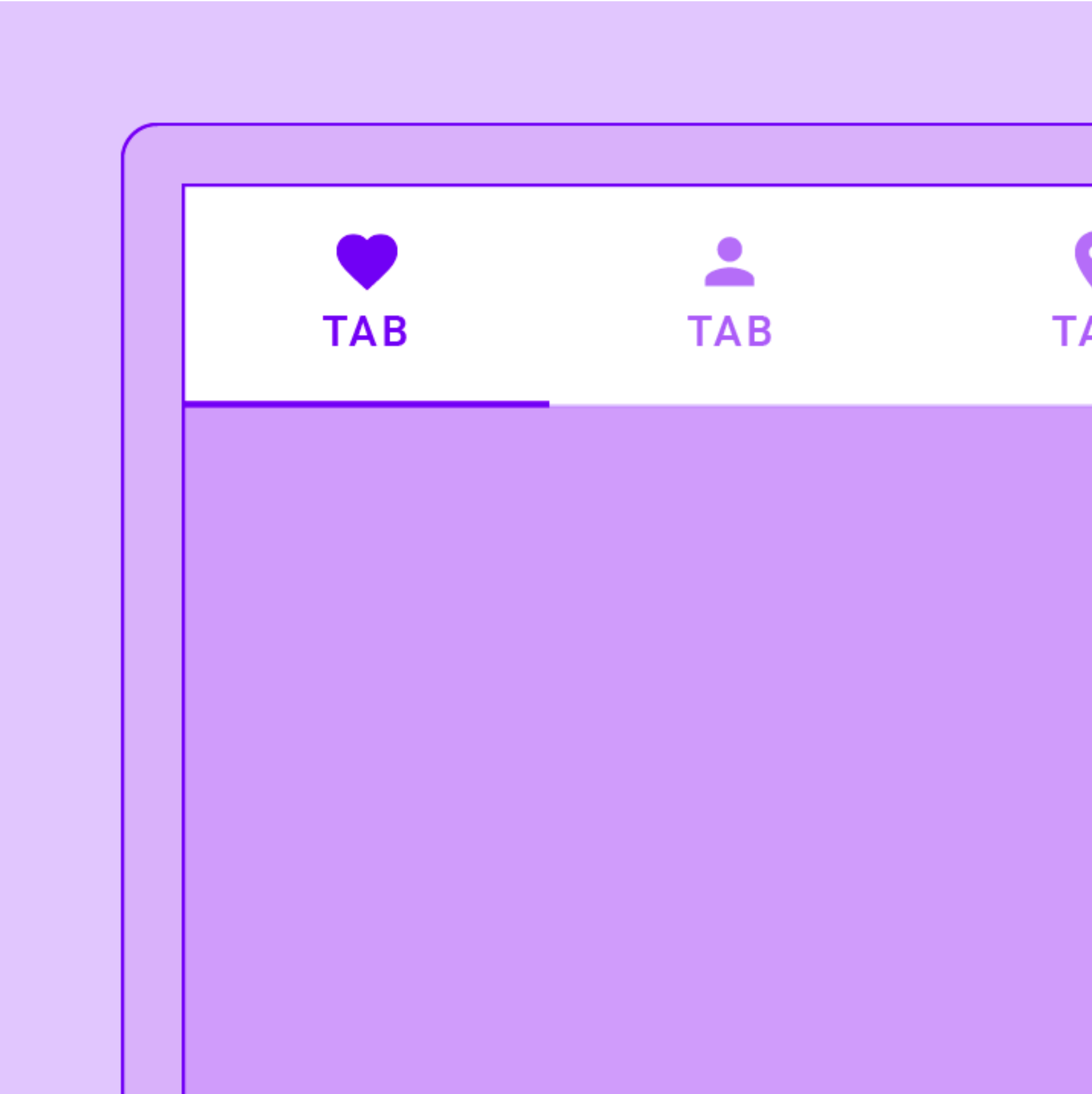


# What is Cupertino design?

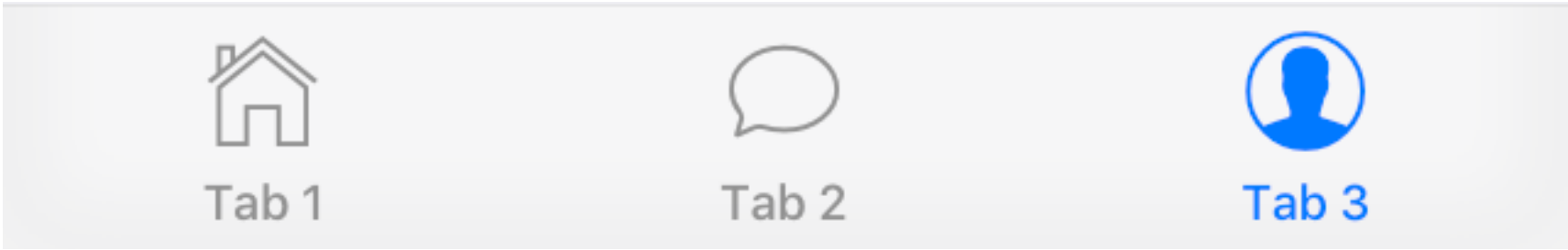
- Clarity
- Depth
- Deference



# Material vs Cupertino

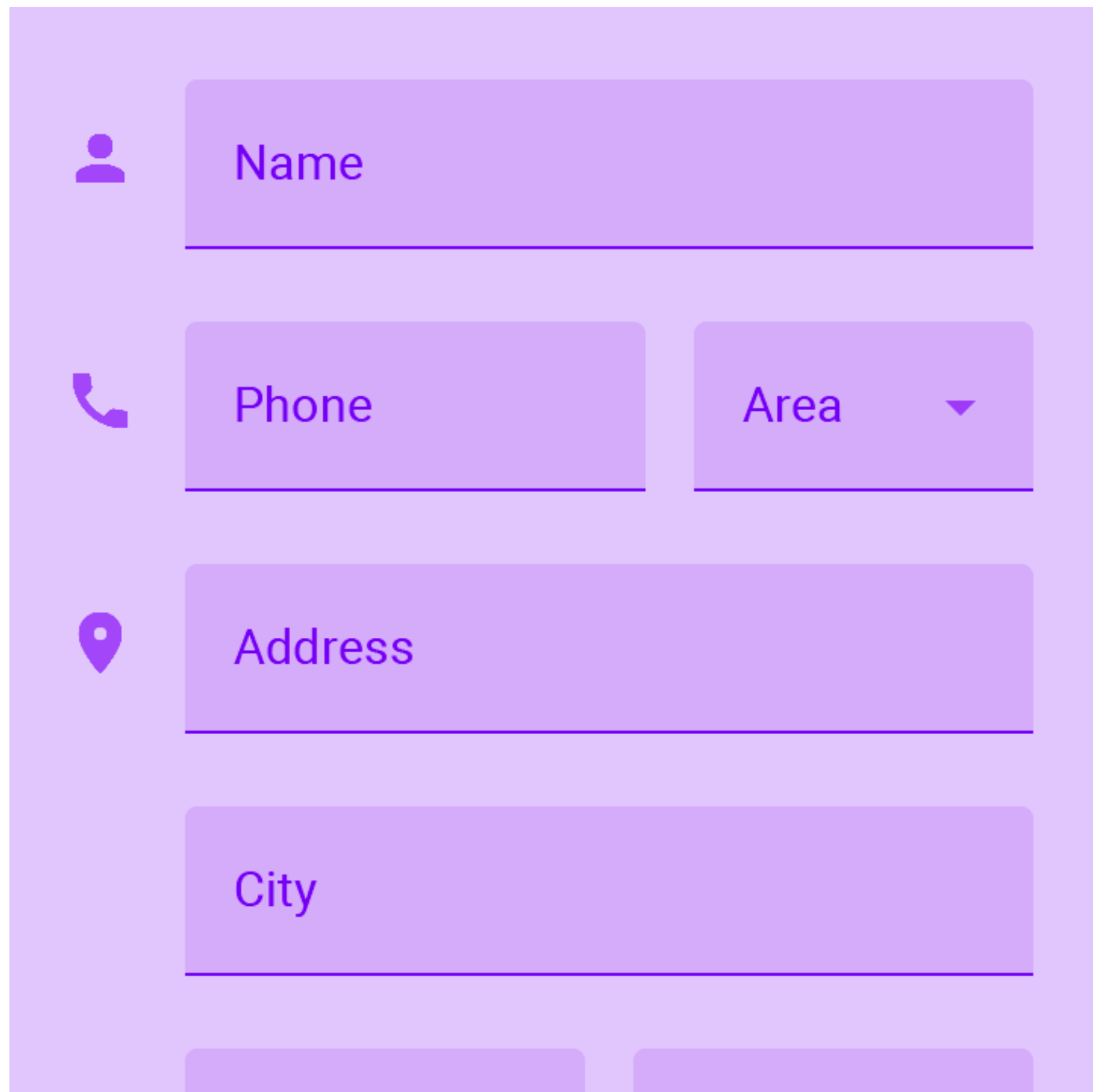


TabBar



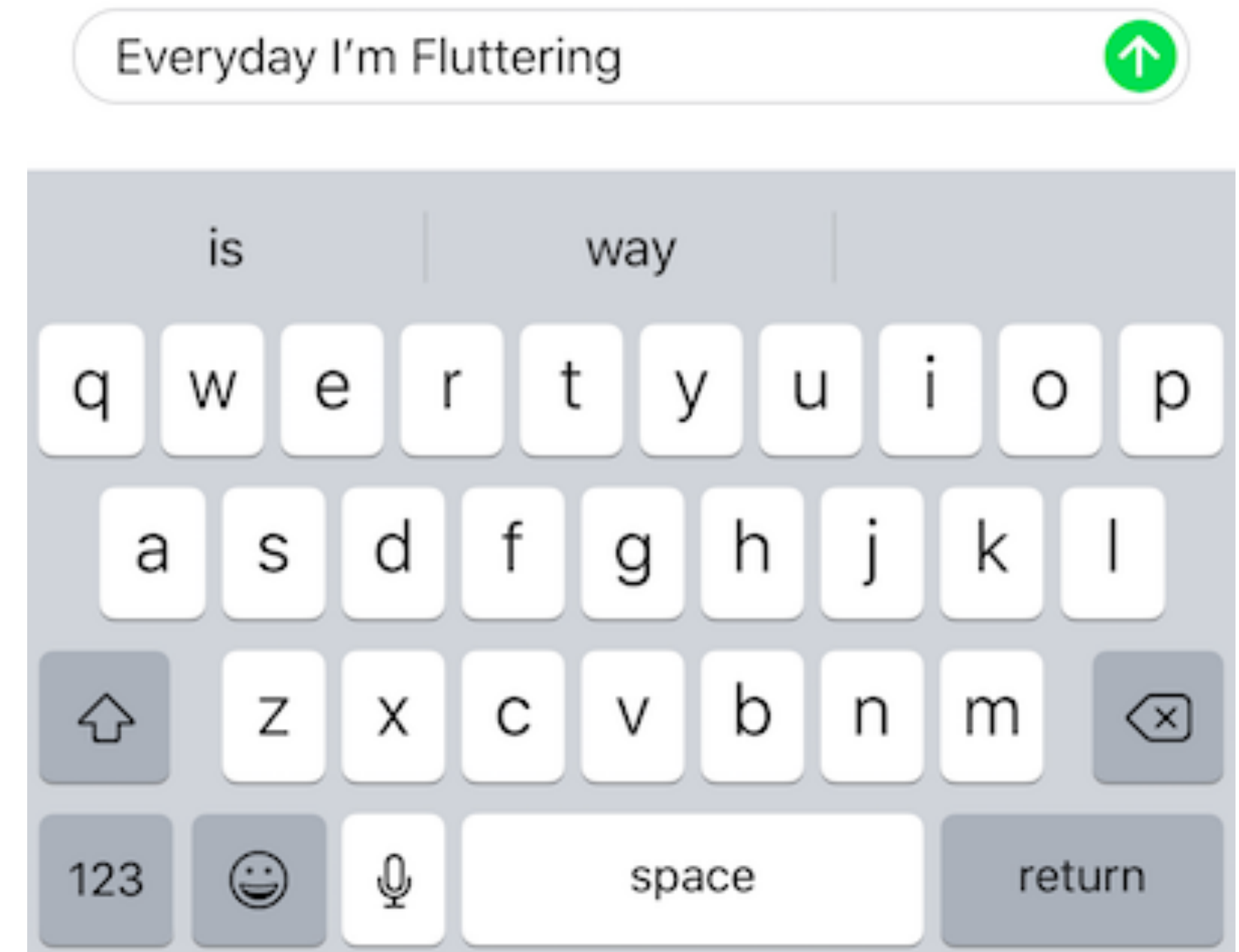
CupertinoTapBar

# Material vs Cupertino



A screenshot of a form using Material Design's TextField widget. The form is set against a light purple background. It contains four input fields: 'Name' (with a person icon), 'Phone' (with a phone icon), 'Address' (with a location pin icon), and 'City'. The 'Phone' field is split into two parts: 'Phone' and 'Area' (with a dropdown arrow). The fields are outlined with a thin purple border and have a light purple background.

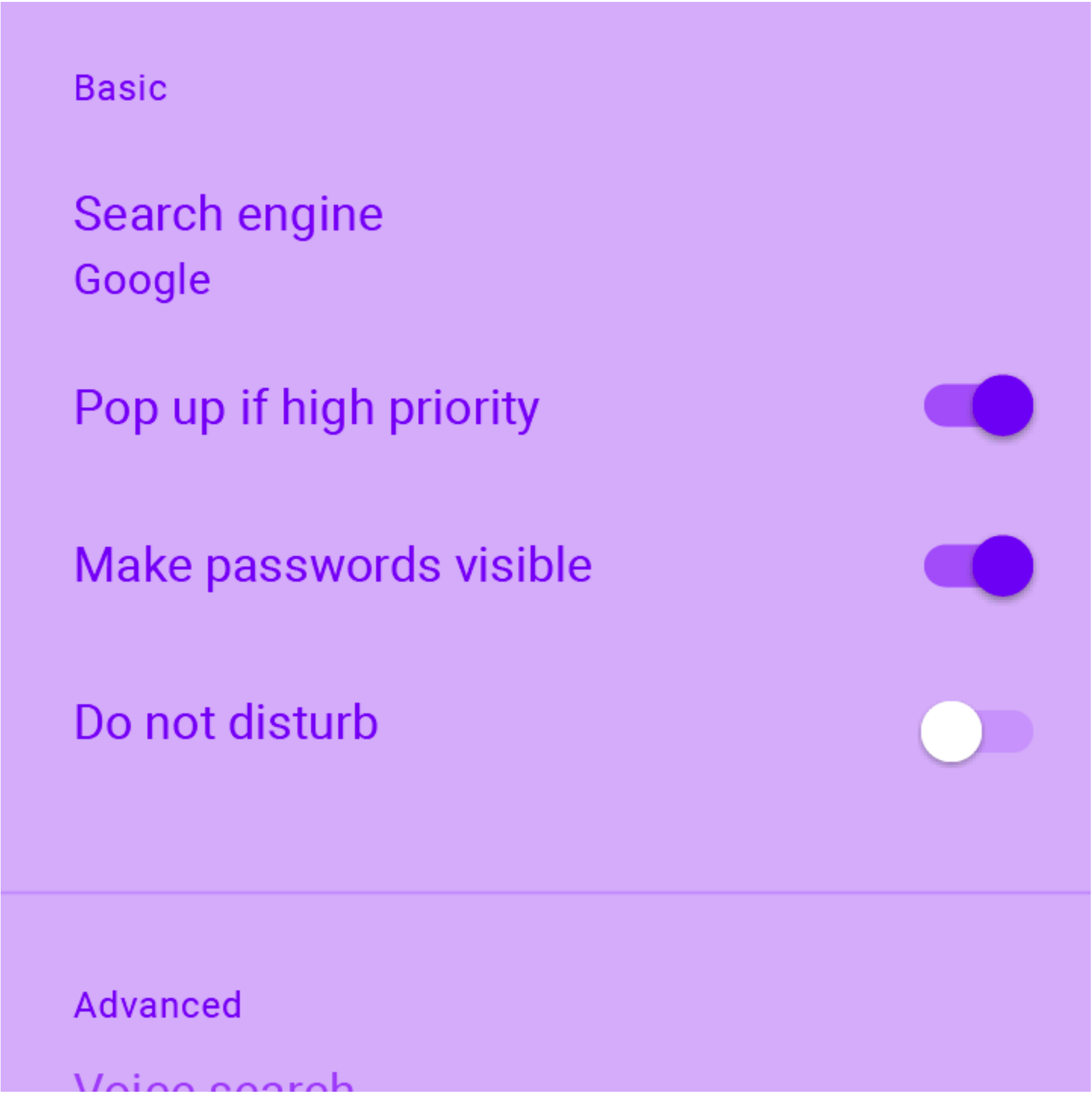
TextField



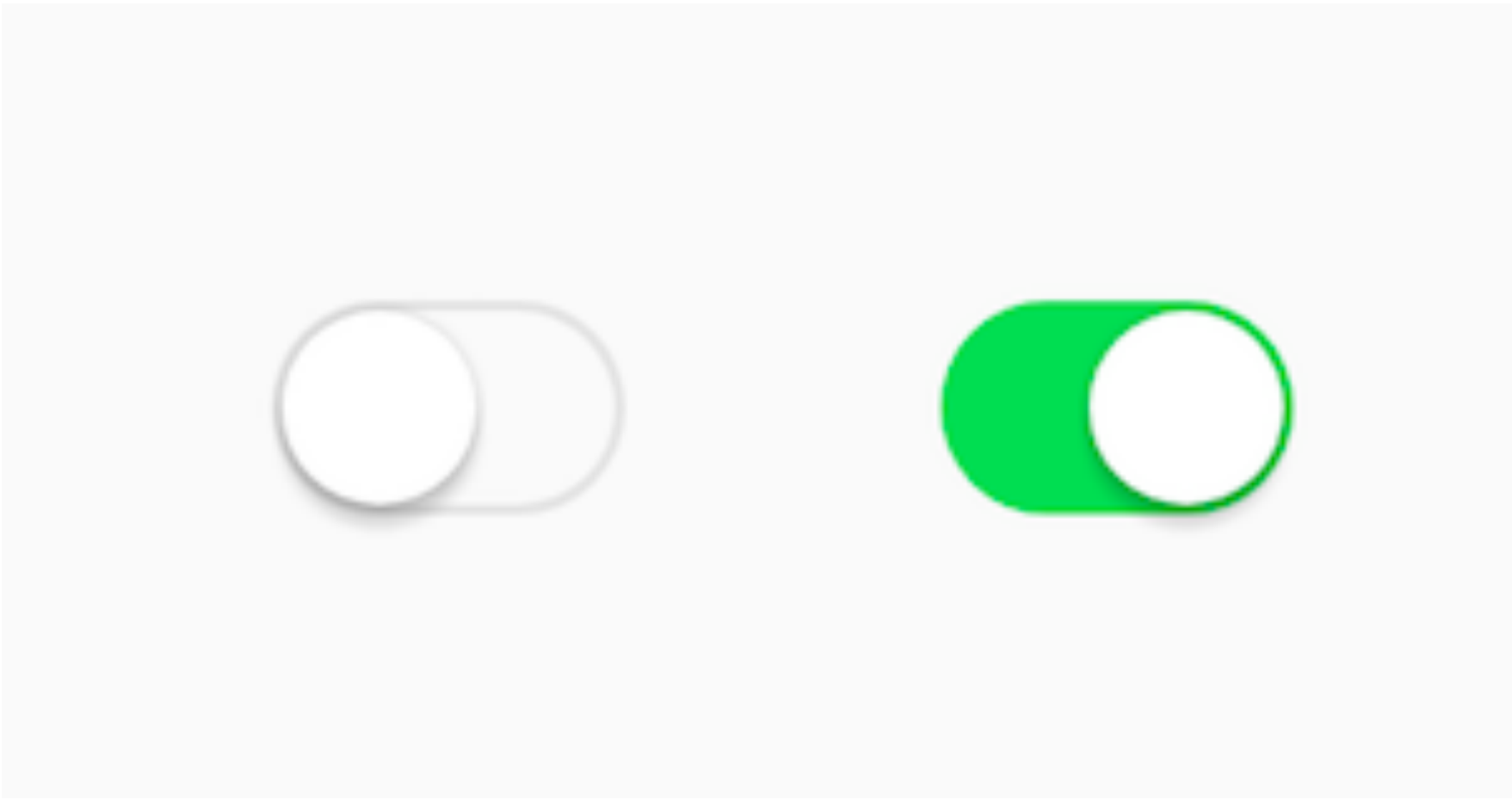
A screenshot of a text input field using Cupertino Design's CupertinoTextField widget. The field is a rounded rectangle with a light gray background and a thin gray border. It contains the text 'Everyday I'm Fluttering' and a green circular button with a white upward arrow on the right. Below the field is a virtual keyboard with a light gray background. The keyboard has four rows of keys: the first row has 'is' and 'way' as suggestions; the second row has letters 'q' through 'p'; the third row has a shift key, letters 'z' through 'm', and a delete key; the fourth row has a '123' key, an emoji key, a microphone key, a 'space' key, and a 'return' key.

CupertinoTextField

# Material vs Cupertino



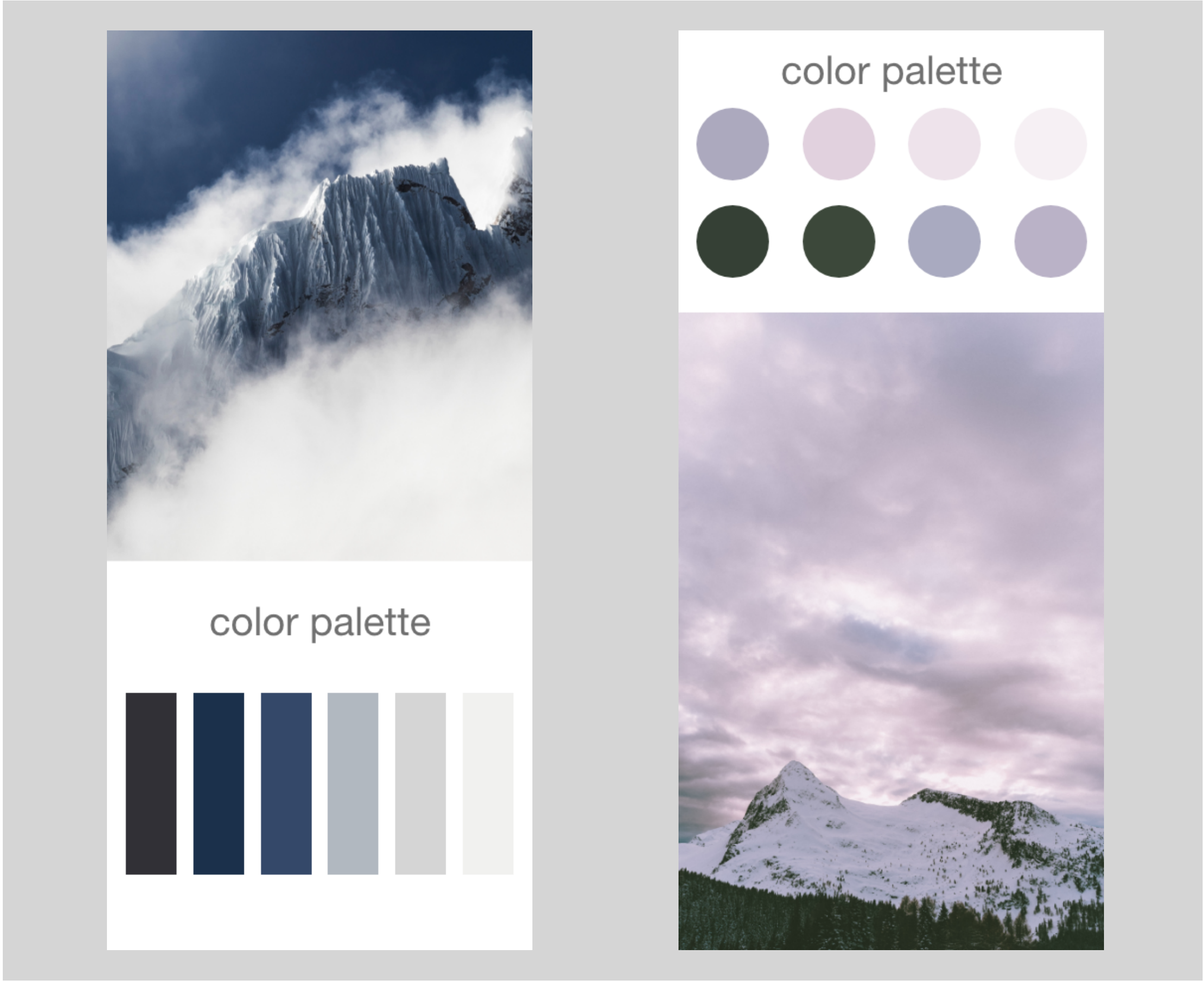
Switch



CupertinoSwitch

# Code time

# Practice 1: Choice one in two design



# Challenge

