

สูตรลัด Flutter

เขียน Flutter ง่ายๆ จากโค้ดตัวอย่าง

อนุชิต ชโลธ

สำนักพิมพ์ก๊อปวาง

สูตรลัด Flutter

เขียน Flutter ง่ายๆ จากโค้ดตัวอย่าง

พิมพ์ครั้งที่ 3

อนุชิต ชโลธร

สำนักพิมพ์ก๊อปวาง

สารบัญ

บทนำ	6
วิธีการอ่าน	7
1. หน้าไม่จำเป็นต้องใช้ Scaffold	8
2. บอกหน้อยว่าหน้าจอะไร ผ่าน AppBar	12
3. ใส่ Action ให้ AppBar	17
4. เพิ่มเมนู Drawer กันหน้อย	19
5. กำหนด Drawer ให้อยู่ทางขวา	23
6. ใส่เมนูข้างล่างหน้าจ (Bottom Navigation Bar)	27
7. AppBar เดิมไม่ถูกใจสร้างใหม่ซะเลย	33
8. ใส่ปุ่ม FAB	36
9. จัดกึ่งกลางด้วย Center	41
10. จัดตำแหน่งด้วย Align	43
11. ใช้ Align จัดตำแหน่งใน Stack	45
12. จัดเรียงแนวตั้งด้วย Column	48
13. กำหนดขนาด Column	50
14. จัดอยู่ข้างบน Column	53
15. จัดอยู่ตรงกลาง Column	56
16. จัดอยู่ตรงข้างล่าง Column	59
17. จัดวางแบบกระจายใน Column	62
18. จัดวางแบบแยกออกจากกัน Column	65
19. จัดวางแนวนอนด้วย Row	68
20. จัดชิดซ้าย ใน Row	73
21. จัดกึ่งกลาง ใน Row	76
22. จัดชิดขวา ใน Row	79
23. จัดแบบกระจาย ใน Row	82
24. จัดวางแบบแยกออกจากกัน ใน Row	85
25. วาง Widget ช้อนทับกันด้วย Stack	88
26. จัดวางโดยกำหนดตำแหน่ง Positioned	91

27. วางข้อความไว้บนภาพ	96
28. ตกแต่ง Bottom Navigation Bar	99
29. แสดง Icon แบบ Shifting ใน Bottom Navigation Bar	103
30. แสดงเฉพาะไอคอนใน Bottom Navigation Bar	107
31. ใช้ปุ่ม FAB ร่วมกับ Bottom Navigation Bar	111
32. ตั้งค่ารูปแบบปุ่ม Navigation Bar	116
33. แสดงข้อมูลใน ListView	120
34. กำหนด Lead, Title, Subtitle และ Tail ใน ListTile	124
35. แสดงข้อมูลใน ListView แบบแนวนอน	127
36. แสดงข้อมูลใน GridView	130
37. ใส่สีฟุ้งๆ โดยใช้ Color Gradient	140
38. ใส่เงาด้วย BoxShadow	152
39. ทำ Crop ด้วย Clip	154
40. จัดการข้อความล้นด้วย Overflow	158
41. ตกแต่งปุ่ม ElevatedButton ในรูปแบบต่างๆ	164
42. ใส่ไอคอน ให้ Elevated Button	170
43. ใส่ปุ่มข้อความด้วย TextButton	172
44. ใส่ Icon ให้ TextButton	174
45. ทำ Icon ให้เป็นปุ่มกด	176
46. อยากรให้อะไรเป็นปุ่ม ครอบด้วย InkWell	178
47. ช่องกรอกข้อความ TextFormField	181
48. ตรวจสอบข้อมูลในฟอร์ม	186
49. ตกแต่ง TextFormField	191
50. ใส่สีส่นให้ App ด้วย Theme	194
51. เพิ่มความมืดด้วย Dark Theme	201
52. เรียกใช้ Material Design 3	206
53. Color System ที่เปลี่ยนไป	211
54. ใส่ Call back ให้ Widget	219
55. ส่งค่าย้อนกลับด้วย ValueChanged	224
56. ไขความลับ setState	229

57. กำหนดขอบเขตในการวาดซ้ำ	232
58. เรื่องวุ่นๆ ของ Routing	236
59. ปาดซ้ายปาดขวาใน List View	250
60. ใครๆ ก็ใช้ Font Awesome	256
61. เรื่องวุ่นๆ กับ Barcode / QRCode	259
62. สแกน Barcode / QRCode	268
63. จับภาพหน้าจอให้เนียนๆ	273
64. ขอ Permission หน่อยสิ	282
65. เก็บข้อมูลไว้ในเครื่อง	291
66. เก็บข้อมูลให้เป็นฐานข้อมูล	304
67. วาดกราฟหน่อย	314
68. เชื่อมต่อกับ REST API	326
69. JWT หมดอายุ จะต่ออายุได้อย่างไร	354
70. สร้างโมเดลแบบไม่ต้องโค้ด	364
71. ไม่อยากรอ ขอแบบ Complete หน่อย	366
72. App หลายภาษาถ่ายนิดเดียว	373
73. ตัดโฆษณาใน App	388
74. ใส่แผนที่ GoogleMap	399
75. ใส่ Marker ลงใน GoogleMap	406
76. กำหนดสไตล์แผนที่ GoogleMap	411
77. หาเส้นทางและวาดเส้นทางใน Google Map	416
78. อ่านค่าจาก GPS แบบ Realtime	423
79. แปลงข้อมูลตำแหน่งเป็นที่อยู่	433
80. สร้างหน้าจอแบบ Responsive	439
81. Deep links และ App links	449
82. สร้างโค้ดเชื่อมต่อกับ API ด้วย Retrofit	463
83. เปลี่ยน Icon ให้แอป	474
84. เพิ่มหน้า Splash	476
85. หาพาร์ทในแต่ละแพลตฟอร์ม	478
86. เลือกภาพและวิดีโอจากเครื่อง	481

87. เลือกไฟล์จากเครื่อง	488
88. ลากวางวิดเจ็ต	493
89. ย่อขยายวิดเจ็ต	500
90. ปรับขนาดตัวอักษรอัตโนมัติใน Widget Text	508
91. หมุนวิดเจ็ตด้วย Transform	513
92. แสดงภาพจากกล้อง	518
93. พิมพ์สลิปด้วยเครื่องพิมพ์ลู่ทูล	522
94. แสดงไฟล์ PDF	532
95. สร้างไฟล์ PDF	539
96. สร้างไฟล์ PDF จากข้อมูล	602
97. ลายเซ็นต์	608
98. ใส่ตารางข้อมูล	612
99. ปฏิทินและตารางกิจกรรม	618
100. ลิงค์ฟรีวิว	623
101. ป้าย Carousel	626
102. เวลาที่ผ่านมา	629
103. เปลี่ยนรูปแบบตัวเลข	632
104. ใช้งาน Prisma ร่วมกับ Dart Frog	635
105. เชื่อมต่อกับ Gemini	646
ดาวน์โหลดไฟล์ตัวอย่าง	660
แนะนำหนังสือชุด “ก๊อปปวาง”	662
สูตรลัด Flutter	662
สูตรลัด GetX	663
สูตรลัด FlutterFire	664
สูตรลัด Hasura	665
คู่มือประกอบการเรียน Flutter Crash Course	666
Material Design 3	667
สูตรลัด Dart 3	668
สูตรลัด Dart Frog	670
สูตรลัด Appwrite	672

สูตรลัด Serverpod
Prisma Client Dart

674
676

บทนำ

หนังสือเล่มนี้เน้นการเรียนรู้ Flutter และภาษา Dart จากตัวอย่างโค้ด ซึ่งเป็นโค้ดตัวอย่างที่ได้รวบรวมมาจากโครงการต่างๆ ที่ผู้เขียนได้พัฒนาแอปให้กับลูกค้าทั้งในประเทศและต่างประเทศ ผู้อ่านสามารถเรียนรู้แนวคิด การออกแบบหน้าจอ การจัดวางเลย์เอาต์ ตลอดจนเทคนิคการตกแต่งวิดเจ็ตต่างๆ ได้จากตัวอย่างโค้ด และนำไปประยุกต์ใช้งานได้

อนุชิต ชโลธร

วิธีการอ่าน

หนังสือเล่มนี้เน้นความเข้าใจ ผู้อ่านสามารถเรียนรู้แนวคิดการออกแบบหน้าจอ การจัดวางเลย์เอาท์ ตลอดจนเทคนิคการตกแต่งวิดเจ็ตต่างๆ จากตัวอย่างโค้ด ดังนั้น การอ่านหนังสือเล่มนี้ให้เกิดประโยชน์ ผู้อ่านควรลงมือทดลองทำทันที

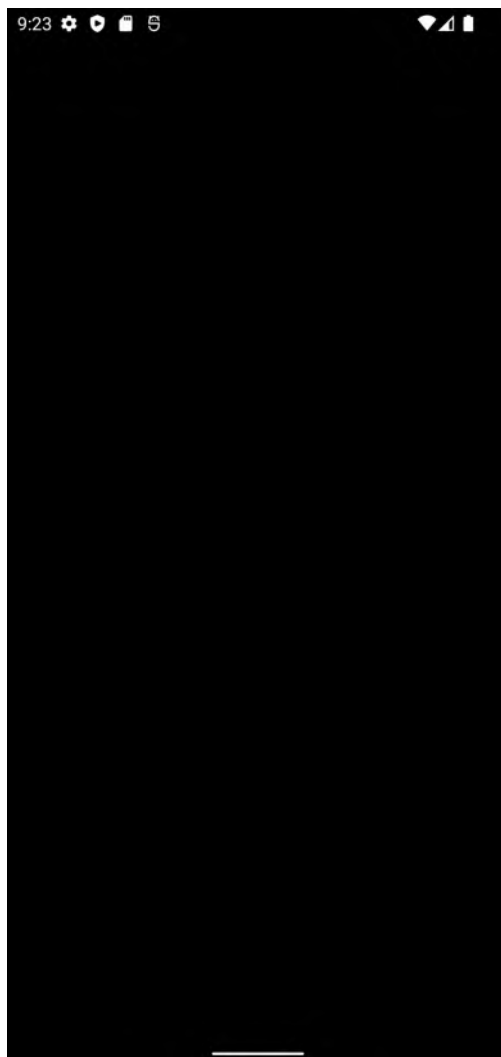
เนื้อหาในแต่ละบทจะแบ่งเนื้อหาออกเป็น 2 ส่วน

1. ส่วนเนื้อหา อธิบายหลักการ และแนวคิดต่างๆ
2. ส่วนซอร์สโค้ดตัวอย่าง

ผู้อ่านสามารถ “ก๊อปปวาง” ซอร์สโค้ดตัวอย่าง เพื่อทดลองได้ด้วยตนเอง

1. หน้าไม่จำเป็นต้องใช้ Scaffold

Scaffold เป็น Widget สำหรับขึ้นโครงหลักใน Material Design ซึ่ง Widget นี้ทำหน้าที่เป็นกำหนดโครงสร้างและธีม ดังนั้นแต่หน้าจอจะควรขึ้นต้นด้วย Scaffold หากไม่ได้ใช้ Scaffold เราจะพบว่าหน้าจอจะไม่มีการเรียกใช้ธีม และหน้าจอจะแสดงผลเป็นสีดำ ดังรูป ดังนั้นควรใช้ Scaffold เป็น Widget หลักของทุกๆ หน้า



หน้าจอแบบไม่ใช้ Scaffold

ตัวอย่างโค้ด

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

class HomePage extends StatelessWidget {
  const HomePage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold();
  }
}
```

ตัวอย่างหน้าจอที่ใช้ Scaffold เพื่อเรียกใช้ริมนจะแสดงผลได้ถูกต้อง



หน้าจอแบบใช้ Scaffold

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return const Scaffold();
  }
}
```

2. บอกหน่อยว่าหน้าจอะไร ผ่าน AppBar

โครงสร้างหน้าจอตาม Material Design จะแบ่งออกเป็น 2 ส่วนหลักๆ คือ AppBar และ Body การเรียกใช้ AppBar จะอยู่ภายใต้ Widget Scaffold เราจะใช้ Widget AppBar ในการแสดงชื่อของหน้าจอ และแสดงปุ่ม Action Button ต่างๆ ได้



หน้าจอแสดง AppBar

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      body: Container(),
    );
  }
}
```

ใน Scaffold นี้ นอกจาก AppBar แล้วยังมีส่วนที่สามารถตั้งค่าเพิ่มเติมได้อีกหลายส่วน เช่น AppBar, Drawer, Navigation Bottom และ Floating Action Button เป็นต้น

ตัวอย่างหน้าจอ เพิ่ม AppBar, Bottom Navigation และ Drawer



เพิ่ม AppBar, Bottom Navigation และ Drawer

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      drawer: Drawer(
        // put drawer items here
        child: Container(),
      ),
      // bottom navigation bar
      bottomNavigationBar: BottomNavigationBar(
        items: const [
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: 'Home',
          ),

```

```
    BottomNavigationBarItem(  
      icon: Icon(Icons.notifications),  
      label: 'Notifications',  
    ),  
    BottomNavigationBarItem(  
      icon: Icon(Icons.account_circle),  
      label: 'Profile',  
    ),  
  ],  
),  
);  
}  
}
```

3. ใส่ Action ให้ AppBar

ในส่วน AppBar สามารถแสดงชื่อหน้าจอ (Title) และใส่ปุ่ม action ต่างๆ ได้ ผ่านการกำหนดค่า action ซึ่งใน action นี้จะเป็น List ของ Widget เราสามารถใส่ปุ่มที่เป็น Elevated Button, Text Button หรือแม้กระทั่ง Icon Button ได้



ใส่ Action ใน AppBar

ตัวอย่างโค้ด

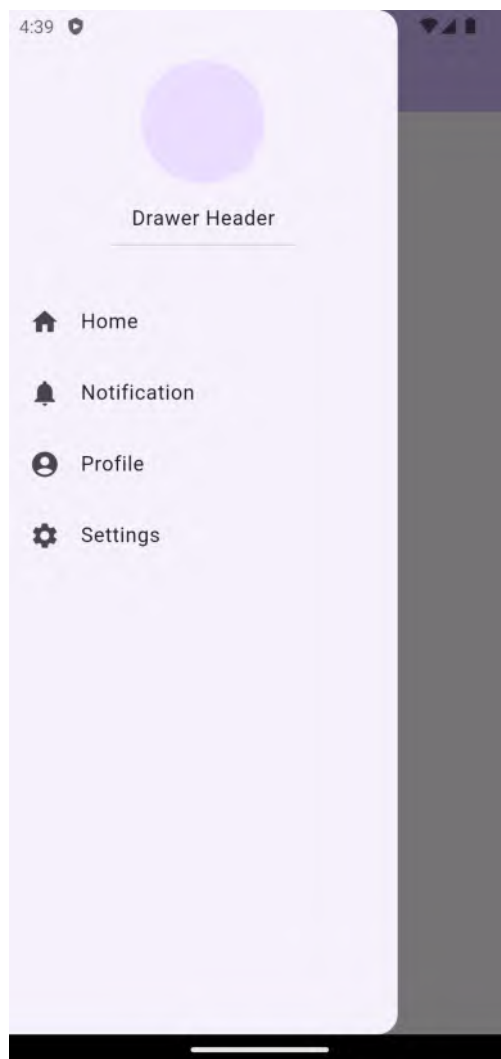
```
import 'package:flutter/material.dart';

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
        actions: [
          IconButton(
            onPressed: () {},
            icon: const Icon(
              Icons.settings,
            ),
          ),
        ],
      ),
      // body
      body: Container(),
    );
  }
}
```

4. เพิ่มเมนู Drawer ก็น้อย

จากที่ทราบกันในเบื้องต้นแล้วว่า Scaffold มีส่วนตั้งค่า Drawer หรือแถบเมนูเพิ่มได้ด้วย ซึ่งในส่วนของ Drawer เราสามารถสร้างเมนูผ่าน Drawer Widget ได้ ทั้งนี้ขึ้นอยู่กับวิธีการออกแบบหน้าจอของแอปนั้นๆ ซึ่ง Widget Scaffold จะเพิ่มเมนูเพื่อเรียก Drawer ให้อัตโนมัติ โดยที่เราไม่ต้องทำปุ่มเพื่อเปิด/ปิด Drawer เพิ่มเติมแต่อย่างใด



แสดง Navigation Drawer

ตัวอย่างโค้ด

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

class HomePage extends StatelessWidget {
  const HomePage({super.key});
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // drawer
      drawer: Drawer(
        child: Column(
          children: [
            // drawer header
            const DrawerHeader(
              child: Column(
                mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                children: [
                  CircleAvatar(
                    radius: 48,
```

```

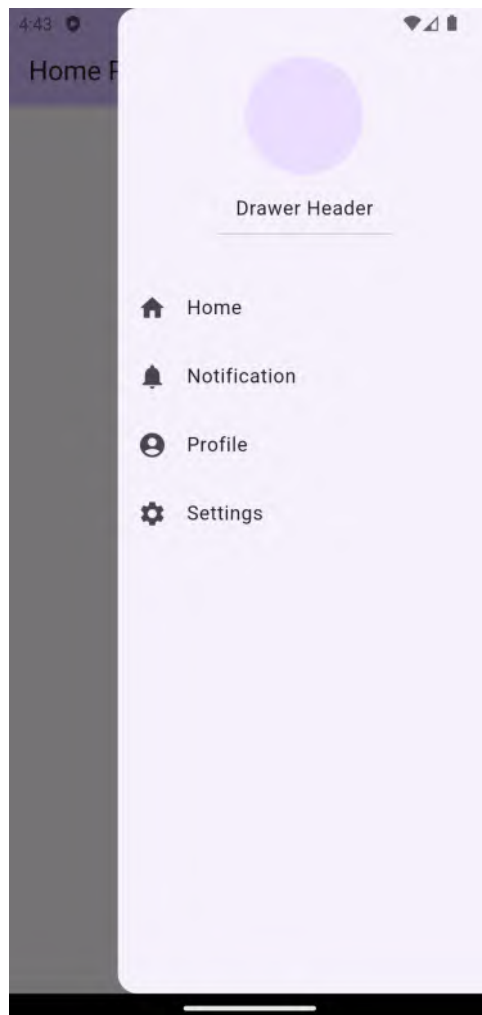
        ),
        Text("Drawer Header"),
    ],
),
),
// list menu
Expanded(
    child: ListView(
        children: const [
            ListTile(
                leading: Icon(Icons.home),
                title: Text("Home"),
            ),
            ListTile(
                leading: Icon(Icons.notifications),
                title: Text("Notifification"),
            ),
            ListTile(
                leading: Icon(Icons.account_circle),
                title: Text("Profile"),
            ),
            ListTile(
                leading: Icon(Icons.settings),
                title: Text("Settings"),
            ),
        ],
    ),
)

```

```
    ],  
    ),  
  ),  
);  
}  
}
```

5. กำหนด Drawer ให้อยู่ทางขวา

Navigation Drawer ยังสามารถกำหนดการแสดงผลทางซ้ายหรือทางขวาของหน้าจอได้ เราไม่จำเป็นต้องสร้าง Widget ขึ้นมาใหม่ เพียงแค่กำหนดค่าใน drawer เพื่อแสดง Navigation Drawer เท่านั้น หากต้องการให้ Drawer แสดงผลทางซ้าย ให้ใส่ค่าลงใน drawer หากต้องการแสดง Navigation Drawer ทางขวา ให้ใส่ค่าลงใน endDrawer ซึ่งเป็นการระบุตำแหน่งของ Navigation Drawer ได้อย่างง่ายดาย



แสดง Drawer ด้านขวา

ตัวอย่างโค้ด

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

class HomePage extends StatelessWidget {
  const HomePage({super.key});
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // drawer
      endDrawer: Drawer(
        child: Column(
          children: [
            // drawer header
            const DrawerHeader(
              child: Column(
                mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                children: [
```

```

        CircleAvatar(
            radius: 48,
        ),
        Text("Drawer Header"),
    ],
),
),
// list menu
Expanded(
    child: ListView(
        children: const [
            ListTile(
                leading: Icon(Icons.home),
                title: Text("Home"),
            ),
            ListTile(
                leading: Icon(Icons.notifications),
                title: Text("Notification"),
            ),
            ListTile(
                leading: Icon(Icons.account_circle),
                title: Text("Profile"),
            ),
            ListTile(
                leading: Icon(Icons.settings),
                title: Text("Settings"),
            ),
        ],
    ),
),

```

```
        ),
    )
    ],
),
),
);
}
}
```

6. ใส่เมนูข้างล่างหน้าจอ (Bottom Navigation Bar)

ใน Material Design เมนูข้างล่างหน้าจอหรือ Bottom Navigation Bar เราสามารถใช้ Widget ที่ชื่อ Bottom Navigation Bar หรือ Navigation Bar ได้ ซึ่งรูปแบบการแสดงผล และการใช้งานจะแตกต่างกัน เล็กน้อย หากเน้นการแสดงผลรูปแบบของ Material Design 3 แนะนำให้ใช้ Navigation Bar ทั้งนี้ขึ้นอยู่กับงานออกแบบว่าจะเลือกใช้ Bottom Navigation Bar แบบใด



Bottom Navigation Bar

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // bottom navigation bar
      bottomNavigationBar: BottomNavigationBar(
        items: const [
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: 'Home',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.notifications),
            label: 'Notifications',
          ),
        ],
      ),
    );
  }
}
```

```
        BottomNavigationBarItem(  
            icon: Icon(Icons.account_circle),  
            label: 'Profile',  
        ),  
    ],  
),  
);  
}  
}
```

ตัวอย่างหน้าจอที่ใช้ Navigation Bar



Navigation Bar

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // navigation bar
      bottomNavigationBar: NavigationBar(
        destinations: const [
          NavigationDestination(
            icon: Icon(Icons.home),
            label: 'Home',
          ),
          NavigationDestination(
            icon: Icon(Icons.notifications),
            label: 'Notifications',
          ),
        ],
      ),
    );
  }
}
```

```
        NavigationDestination(  
            icon: Icon(Icons.account_circle),  
            label: 'Profile',  
        ),  
    ],  
),  
);  
}  
}
```

จากตัวอย่างข้างต้น เป็นการใช้งานหน้าจอหลักที่เริ่มต้นด้วย Scaffold เราสามารถปรับเปลี่ยนการแสดงผลได้หลากหลายขึ้นอยู่กับงานออกแบบหน้าจอ

7. AppBar เดิมไม่ถูกใจสร้างใหม่ซะเลย

เราสามารถออกแบบ AppBar ได้เอง โดยไม่ใช่ Widget AppBar ได้เช่นกัน โดยอ้างอิงจากงานออกแบบของ designer ได้ ซึ่งอย่าลืมว่าใน Flutter ทุกอย่างที่เราแสดงผลบนหน้าจอคือ Widget ดังนั้นเราสามารถสร้าง AppBar ตามที่เราต้องการได้ ทั้งนี้ขึ้นอยู่กับงานออกแบบเป็นหลัก



หน้าจอ AppBar ออกแบบเอง

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Column(
        children: [
          // custom design appbar
          Padding(
            padding: const EdgeInsets.only(
              top: 32.0,
              bottom: 16.0,
            ),
            child: Row(
              children: [
                // profile
                const Padding(
                  padding: EdgeInsets.all(8.0),
                  child: CircleAvatar(),
                ),
                // title
                Text(
                  "John Doe",
                  style:
```

```

Theme.of(context).textTheme.titleLarge,
    ),
    // spacer
    const Spacer(),
    // button
    IconButton(
      onPressed: () {},
      icon: const Icon(Icons.arrow_back),
    ),
  ],
),
),
],
),
);
}
}

```

เราจะเห็นได้ว่า เราสามารถสร้าง Widget ใหม่ๆ ได้ตามต้องการ และมีรูปแบบตามงานออกแบบของ Designer ได้ ทั้งนี้ขึ้นอยู่กับความละเอียดของงานออกแบบ

8. ใส่ปุ่ม FAB

Floating Action Button หรือ FAB เป็นปุ่มที่ลอยอยู่บนหน้าจอ เราสามารถปรับแต่งการตั้งค่าได้หลายอย่าง เช่น การใส่ Icon การใส่ข้อความ การจัดตำแหน่ง เป็นต้น การใช้งาน Fab ควรระมัดระวังเรื่องการแสดงผลของปุ่ม ซึ่งอาจไปทับเนื้อหาที่อยู่ในหน้าจอได้



Floating Action Button

ตัวอย่างโค้ด

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

class HomePage extends StatelessWidget {
  const HomePage({super.key});
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // floating action button
      floatingActionButton: FloatingActionButton(
        onPressed: () {},
        child: const Icon(
          Icons.add,
        ),
      ),
    );
  }
}
```

ตัวอย่าง Floating Action Button แบบกำหนดตำแหน่ง กึ่งกลางหน้าจอ



Floating Action Button แบบกำหนดตำแหน่ง

ตัวอย่างโค้ด

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

class HomePage extends StatelessWidget {
  const HomePage({super.key});
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(),
      // floating action button
      floatingActionButtonLocation:
FloatingActionButtonLocation.centerFloat,
      floatingActionButton: FloatingActionButton(
        onPressed: () {},
        child: const Icon(
          Icons.add,
        ),
      ),
    );
  }
}
```

นอกจากการเรียกใช้ Floating Action button แบบปกติแล้ว ในเว็บไซต์ pub.dev ยังมี plugin อื่นๆ ที่น่าสนใจที่เกี่ยวข้องกับ Floating Action Button อยู่หลายตัว เช่น floating action bubble, animated floating buttons, form floating action button เป็นต้น สามารถนำมาประยุกต์ใช้งานได้

9. จัดกึ่งกลางด้วย Center

Widget ที่เกี่ยวกับการจัดวางมีหลายตัวขึ้นอยู่กับว่าเราเรียกใช้ Widget อะไรอยู่ ตัวอย่างเช่น การจัดกึ่งกลางหน้าจอเราจะใช้ Widget ที่ชื่อว่า Center แต่หากต้องการจัดในรูปแบบอื่น เช่น ซิดซ้าย ซิดขวา เราสามารถใช้ Widget ที่ชื่อว่า Align เข้ามาช่วยใช้ในการจัดรูปแบบได้เช่นกัน และยังมี Widget อื่นๆ อีกหลายตัวที่ช่วยจัดรูปแบบได้ ขึ้นอยู่กับการประยุกต์ใช้งาน



Widget Center

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      body: const Center(
        child: Text("Center Text"),
      ),
    );
  }
}
```

10. จัดตำแหน่งด้วย Align

Align เป็น Widget สำหรับกำหนดตำแหน่งของ Widget ที่อยู่ภายใต้ Widget นี้ ในหลายรูปแบบ เช่น จัดกึ่งกลาง, ซ้ายบน, ขวาบน, ซ้ายล่าง, ขวาล่าง, ซ้ายกลาง, ขวากลาง และ กึ่งกลาง เป็นต้น



ใช้ Align

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appbar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: const Align(
        alignment: Alignment.center,
        child: Text("Center Text"),
      ),
    );
  }
}
```

11. ใช้ Align จัดตำแหน่งใน Stack

Stack เป็น Widget ที่ใช้แสดง Widget ซ้อนทับกัน เป็นชั้นๆ (layer) การใช้ Align ร่วมกับ Stack จะทำให้เราสามารถจัด Widget ในแต่ละ layer ให้อยู่ในตำแหน่งต่างๆ ได้อย่างอิสระ



การใช้ Align ร่วมกับ Stack

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Stack(
        children: [
          // top left
          Align(
            alignment: Alignment.topLeft,
            child: Container(
              width: 100,
              height: 100,
              color: Colors.red,
            ),
          ),
          // center
```

```
Align(  
  alignment: Alignment.center,  
  child: Container(  
    width: 100,  
    height: 100,  
    color: Colors.green,  
  ),  
),  
// bottom right  
Align(  
  alignment: Alignment.bottomRight,  
  child: Container(  
    width: 100,  
    height: 100,  
    color: Colors.blue,  
  ),  
),  
],  
),  
);  
}  
}
```

12. จัดเรียงแนวตั้งด้วย Column

การจัดรูปแบบแนวตั้งโดยใช้ Column Widget ที่อยู่ใน Column นั้นจะมีการเรียงตัวแบบแนวตั้ง ดังนั้นการจัดรูปแบบข้อมูลที่อยู่ใน Column สามารถทำได้ 2 แบบ ได้แก่ การจัดวางเรียงตัวแนวตั้งและการจัดวางเรียงตัวแนวนอน

การจัดวางแนวตั้งของ Column เราสามารถกำหนดขนาดของ Column ได้ซึ่งโดยปกติจะอิงขนาดของ Widget ที่อยู่ก่อนหน้าเป็นหลัก หาก Widget ก่อนหน้ามีขนาดเท่ากับหน้าจอหลัก Column ที่วางจะมีขนาดความสูงเท่ากับขนาดของหน้าจอหลักเช่นกัน



สีเทาใน Container แสดงให้เห็นว่า Column มีความสูงเท่ากับหน้าจอ

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        color: Colors.grey,
        child: const Column(
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

13. กำหนดขนาด Column

การกำหนดขนาดของ Column ให้สูงเท่ากับ Widget ที่อยู่ภายใน Column สามารถกำหนดขนาด ด้วย MainAxisSize



สีใน Container แสดงให้เห็นว่า Column มีความสูงเท่ากับ
ขนาดของ Widget ที่อยู่ข้างใน

ตัวอย่างโค้ด

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

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

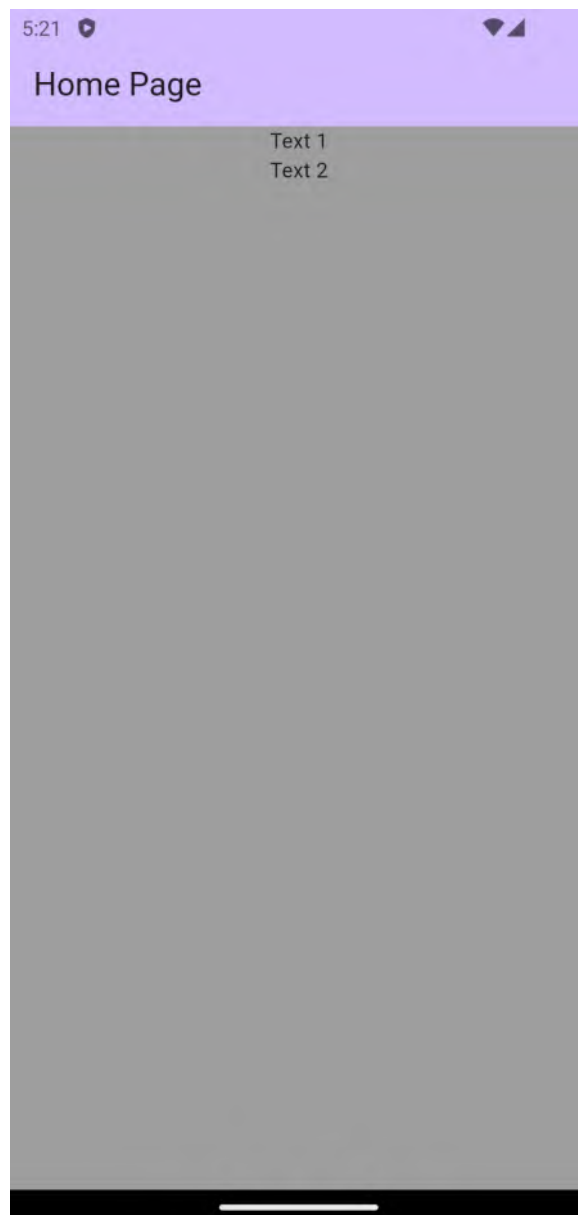
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.min,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

```
}
```

การเรียงตัวของ Widget ที่อยู่ภายใน Column จะถูกบังคับให้เรียงตัวตามแนวตั้ง โดยอัตโนมัติ เราสามารถจัดวาง Widget ในรูปแบบแนวตั้งได้หลายแบบ เช่น กำหนดให้อยู่ข้างบน กำหนดกึ่งกลาง กำหนดให้อยู่ด้านล่าง หรือกระจาย ทั้งนี้ขึ้นอยู่กับงานออกแบบของนักออกแบบ

14. จัดอยู่ข้างบน Column

การจัดเรียง Widget ใน Column ยังสามารถทำได้หลายแบบ เช่น การจัดวางแนวตั้ง จัด Widget ให้อยู่ด้านบนของ Column



จัด Widget ให้อยู่ด้านบนของ Column

ตัวอย่างโค้ด

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

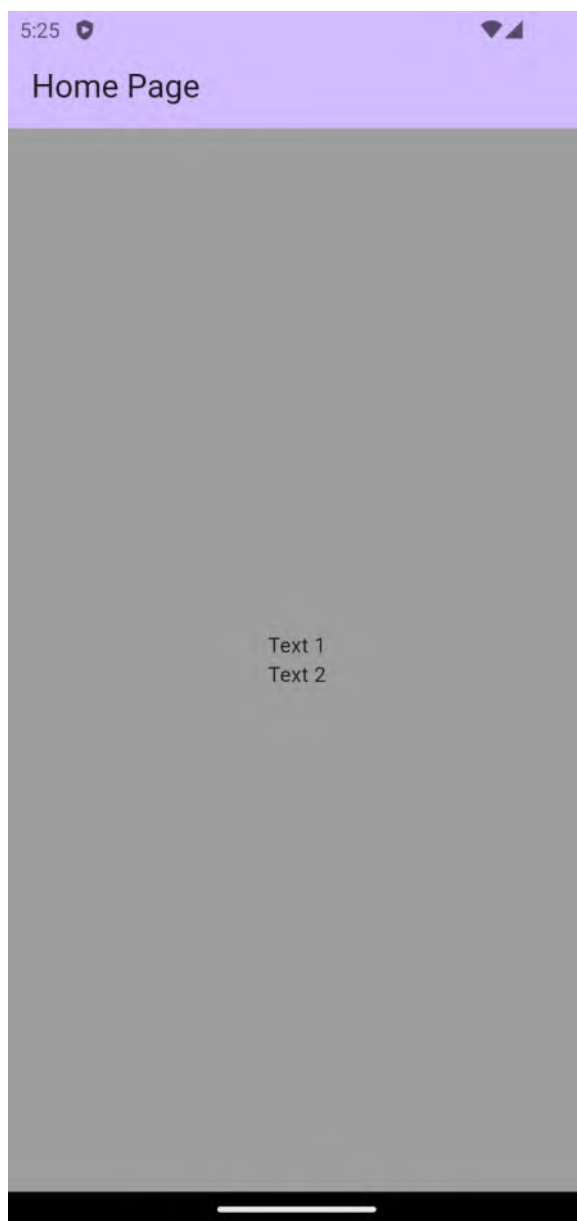
class HomePage extends StatelessWidget {
  const HomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        width: MediaQuery.of(context).size.width,
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.start,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

```
    ));  
}  
}
```

15. จัดอยู่ตรงกลาง Column

การจัดวาง Widget ให้อยู่ตรงกลางของ Column



จัด Widget ให้อยู่ตรงกลางของ Column

ตัวอย่างโค้ด

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

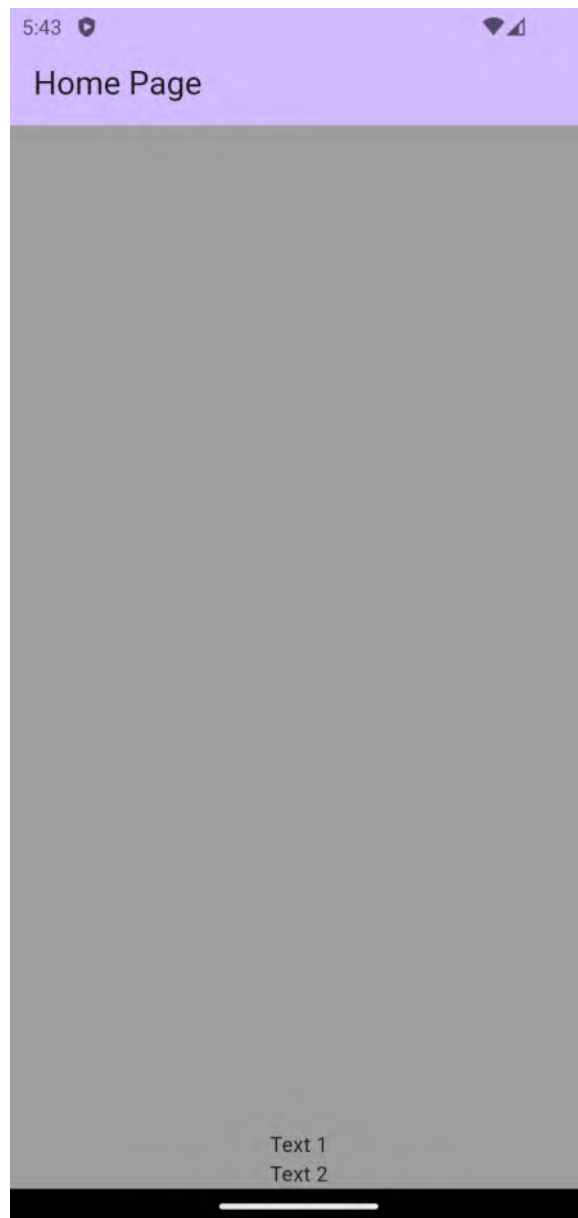
class HomePage extends StatelessWidget {
  const HomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        width: MediaQuery.of(context).size.width,
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

```
    ),  
  );  
}  
}
```

16. จัดอยู่ตรงข้างล่าง Column

การจัดวาง Widget ให้อยู่ตรงด้านล่างของ Column



จัดวาง Widget ให้อยู่ตรงด้านล่างของ Column

ตัวอย่างโค้ด

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

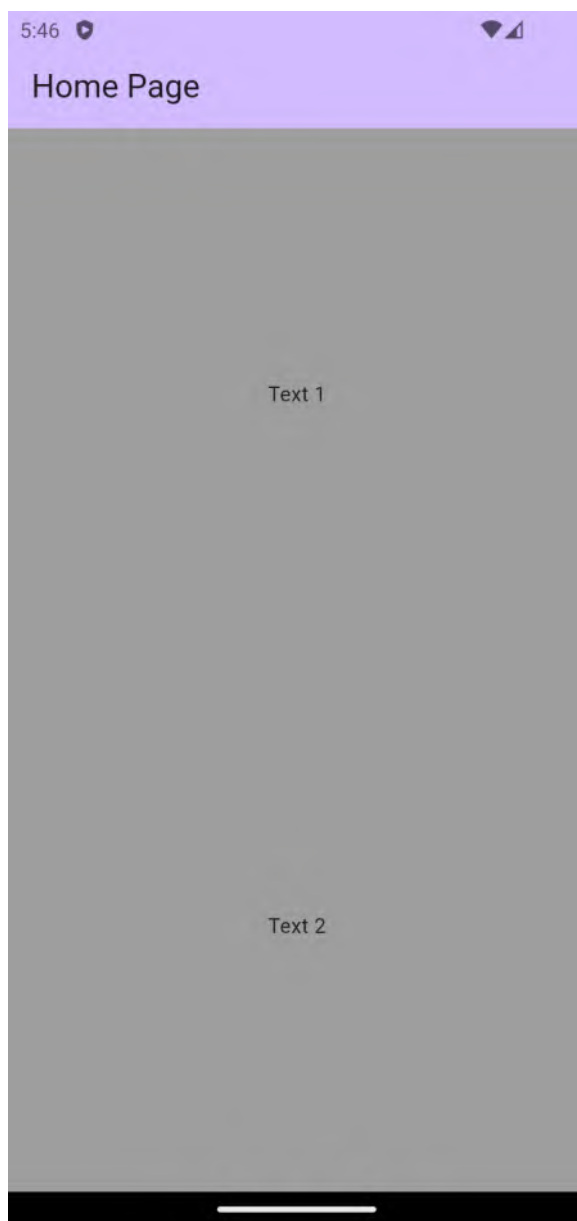
class HomePage extends StatelessWidget {
  const HomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        width: MediaQuery.of(context).size.width,
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.end,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

```
    ));  
}  
}
```

17. จัดวางแบบกระจายใน Column

การจัดวางแบบกระจายโดยมีช่องว่างระหว่าง Widget ขนาดเท่าๆกัน



จัดวางแบบกระจายโดยช่องว่างระหว่าง Widget มีขนาดเท่าๆกัน

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        width: MediaQuery.of(context).size.width,
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.spaceAround,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```

```
    ),  
  );  
}  
}
```

18. จัดวางแบบแยกออกจากกัน Column

การจัดวางแบบกระจายโดยแสดง Widget แยกกัน แสดงอยู่ข้างบนและข้างล่าง



การจัดวางแบบกระจาย แยก Widget แสดงอยู่ข้างบนและข้างล่าง

ตัวอย่างโค้ด

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // appBar
      appBar: AppBar(
        title: const Text("Home Page"),
        backgroundColor:
Theme.of(context).colorScheme.inversePrimary,
      ),
      // body
      body: Container(
        width: MediaQuery.of(context).size.width,
        color: Colors.grey,
        child: const Column(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [
            Text("Text 1"),
            Text("Text 2"),
          ],
        ),
      ),
    );
  }
}
```