ListView

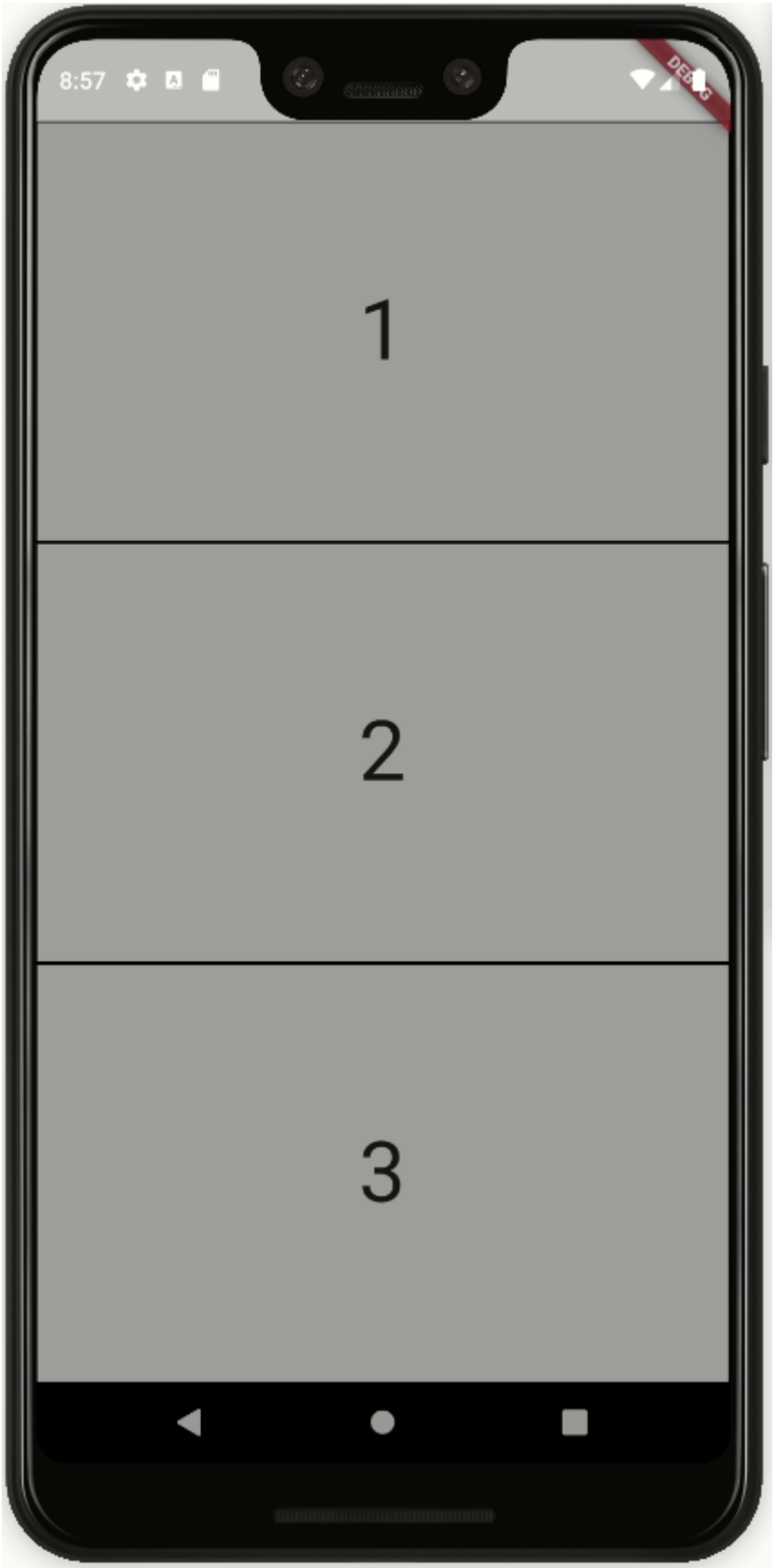
Pada Codelab kedua kita telah menggunakan dan menyinggung sedikit tentang widget **ListView**. Widget ini digunakan untuk menampilkan beberapa item dalam bentuk baris atau kolom dan bisa di-*scroll*.

Cara penggunaan *ListView* ini mirip dengan **Column** atau **Row** di mana Anda memasukkan widget yang ingin disusun sebagai **children** dari **ListView**.

```
class ScrollingScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: ListView(
        children: <Widget>[
          Container(
            height: 250,
            decoration: BoxDecoration(
                color: Colors.grey,
                border: Border.all(color: Colors.black),
            ),
            child: Center(
              child: Text(
                '1',
                style: TextStyle(fontSize: 50),
              ),
            ),
          ),
          Container(
            height: 250,
            decoration: BoxDecoration(
              color: Colors.grey,
              border: Border.all(color: Colors.black),
            ),
            child: Center(
              child: Text(
                '2',
                style: TextStyle(fontSize: 50),
              ),
            ),
          ),
          Container(
            height: 250,
            decoration: BoxDecoration(
              color: Colors.grey,
              border: Border.all(color: Colors.black),
            ),
            child: Center(
              child: Text(
```

```
                  '3',
                  style: TextStyle(fontSize: 50),
                ),
              ),
            ),
            Container(
              height: 250,
              decoration: BoxDecoration(
                color: Colors.grey,
                border: Border.all(color: Colors.black),
              ),
              child: Center(
                child: Text(
                  '4',
                  style: TextStyle(fontSize: 50),
                ),
              ),
            ),
          ],
        ),
      );
    }
  }
```

Ketika dijalankan, aplikasi akan menjadi seperti berikut:


8:57
1
2
3

Menampilkan Item Secara Dinamis

Selain memasukkan widget satu per satu ke dalam **children** dari **ListView**, Anda juga dapat menampilkan *list* secara dinamis. Ini sangat berguna ketika Anda memiliki banyak item dengan jumlah yang tidak menentu.

Misalnya kita ingin menampilkan daftar angka dari 1 sampai 10.

```
1. List<int> numberList = <int>[1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
```

Caranya, masukkan variabel atau *list* Anda sebagai *children* lalu panggil fungsi `map()`. Fungsi map ini berguna untuk memetakan atau mengubah setiap item di dalam list menjadi objek yang kita inginkan. Fungsi map ini membutuhkan satu buah parameter berupa fungsi atau lambda.

```
1. class ScrollingScreen extends StatelessWidget {
2.   final List<int> numberList = <int>[1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
3.
4.   @override
5.   Widget build(BuildContext context) {
6.     return Scaffold(
7.       body: ListView(
8.         children: numberList.map((number) {});
9.       ),
10.      );
11.    }
12.  }
```

Karena parameter children ini membutuhkan nilai berupa *list* widget, maka kita perlu mengembalikan setiap item dari **numberList** menjadi **widget** yang akan ditampilkan. Ubah fungsi **lambda** Anda menjadi seperti berikut:

```
1. class ScrollingScreen extends StatelessWidget {
2.   final List<int> numberList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
3.
4.   @override
5.   Widget build(BuildContext context) {
6.     return Scaffold(
7.       body: ListView(
8.         children: numberList.map((number) {
9.           return Container(
10.              height: 250,
11.              decoration: BoxDecoration(
12.                color: Colors.grey,
```

```
                border: Border.all(color: Colors.black),
              ),
              child: Center(
                child: Text(
                  '$number', // Ditampilkan sesuai item
                  style: TextStyle(fontSize: 50),
                ),
              ),
            );
          }).toList(),
        ),
      );
    }
  }
```

Perhatikan di akhir kita perlu mengembalikan fungsi map menjadi objek *List* lagi dengan fungsi `.toList()`. Lakukan **hot reload** pada aplikasi Anda untuk melihat hasil perubahan.

Menggunakan ListView.builder

Selain mengisi parameter children dari ListView seperti sebelumnya, kita juga bisa memanfaatkan method ListView.builder. ListView.builder lebih cocok digunakan pada ListView dengan jumlah item yang cukup besar. Ini karena Flutter hanya akan merender tampilan item yang terlihat di layar dan tidak me-*render* seluruh item ListView di awal.

ListView.builder memerlukan dua parameter yaitu itemBuilder dan itemCount. Parameter itemBuilder merupakan fungsi yang mengembalikan widget untuk ditampilkan. Sedangkan itemCount kita isi dengan jumlah seluruh item yang ingin ditampilkan.

Berikut ini adalah contoh kode penggunaan ListView.builder:

```
ListView.builder(
  itemBuilder: (BuildContext context, int index) {
    return Container(
      height: 250,
      decoration: BoxDecoration(
        color: Colors.grey,
        border: Border.all(color: Colors.black),
      ),
      child: Center(
        child: Text(
          '${numberList[index]}',
```

```
            style: TextStyle(fontSize: 50),
          ),
        ),
      );
    },
    itemCount: numberList.length,
  ),
```

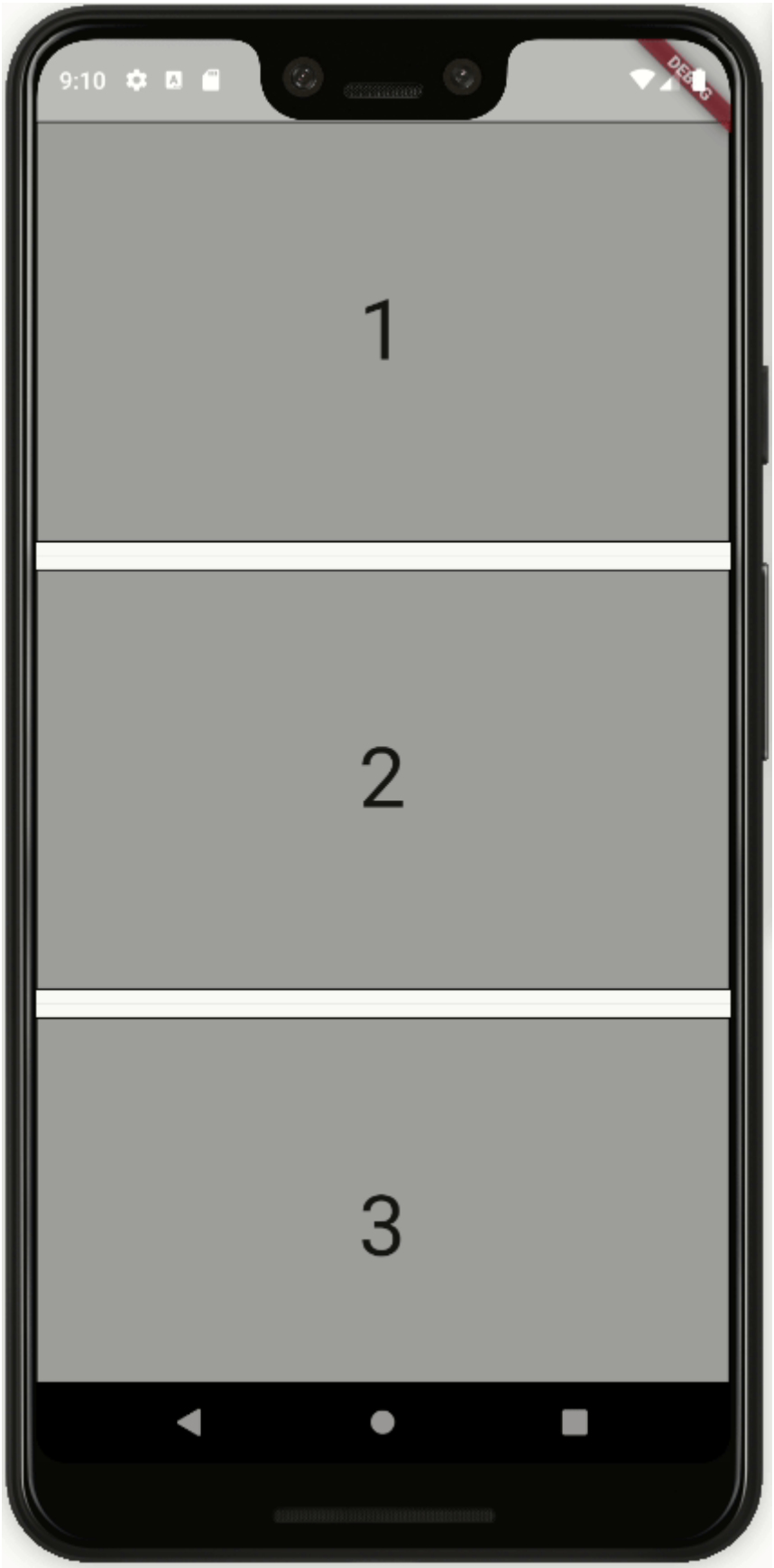
ListView.separated

Cara lain untuk membuat ListView adalah dengan metode ListView.separated. ListView jenis ini akan menampilkan daftar item yang dipisahkan dengan separator. Penggunaan ListView.separated mirip dengan builder, yang membedakan adalah terdapat satu parameter tambahan wajib yaitu separatorBuilder yang mengembalikan Widget yang akan berperan sebagai separator.

Berikut ini adalah contoh kode dari ListView.separated:

```
ListView.separated(
   itemBuilder: (BuildContext context, int index) {
     return Container(
       height: 250,
       decoration: BoxDecoration(
         color: Colors.grey,
         border: Border.all(color: Colors.black),
       ),
       child: Center(
           child: Text(
             '${numberList[index]}',
             style: TextStyle(fontSize: 50),
           ),
         ),
       );
    },
    separatorBuilder: (BuildContext context, int index) {
      return Divider();
    },
    itemCount: numberList.length,
  ),
```

Jika kode di atas dijalankan, maka tampilan aplikasi adalah seperti ini:


9:10
1
2
3

Untuk mempelajari fitur ListView selengkapnya dapat Anda baca pada tautan dokumentasi ListView Class.