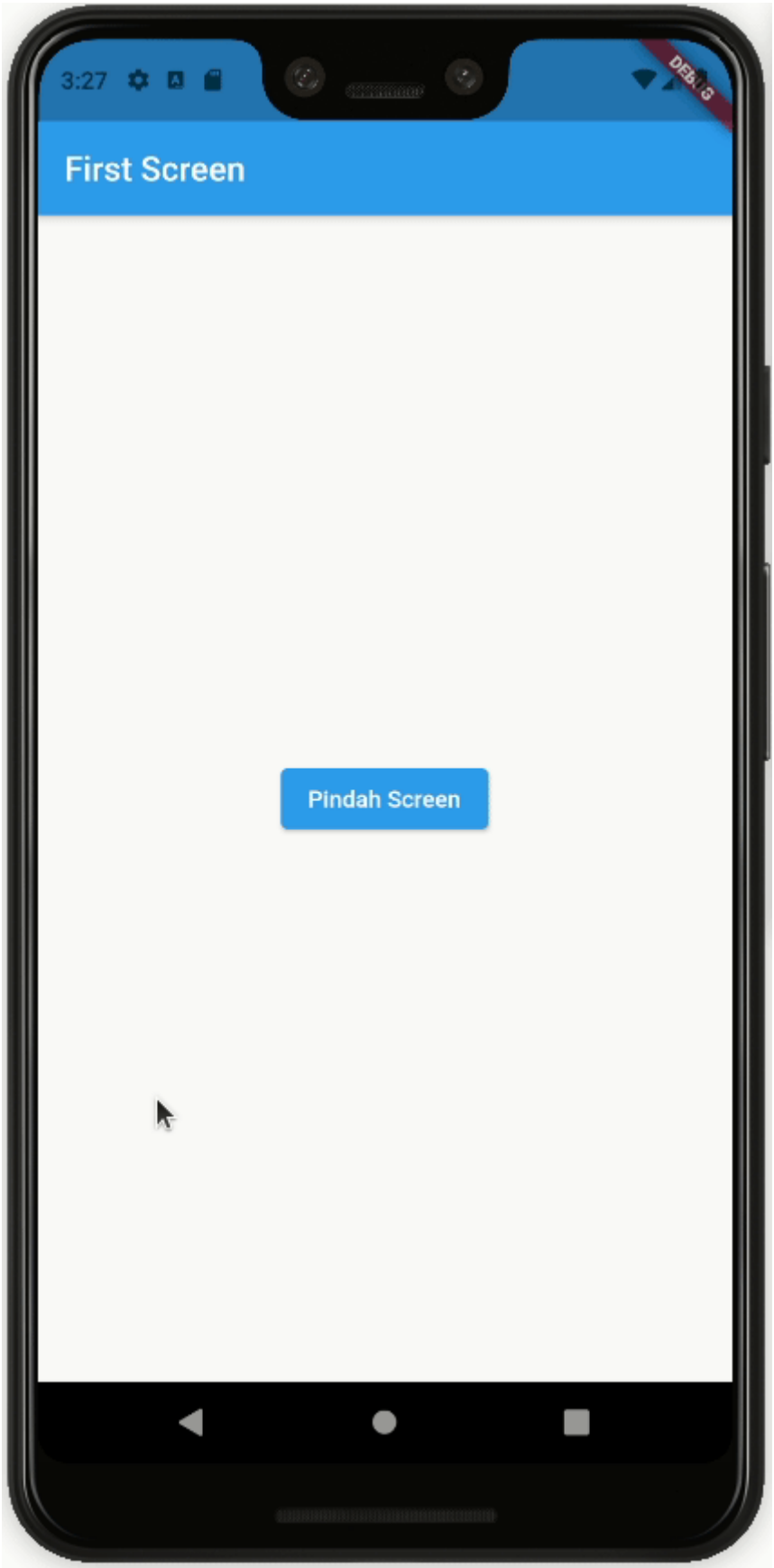
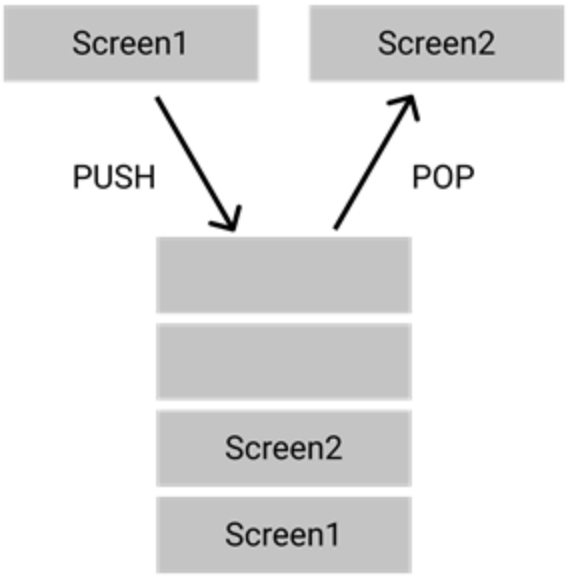
Navigation

Kita telah bisa membuat satu tampilan *screen* (layar/*page*) pada pembelajaran sebelumnya. Namun, pada saat membangun sebuah aplikasi kita akan membuat banyak sekali *screen* dan kita akan berpindah dari satu *screen* ke *screen* lainnya.

Dalam pemrograman Android kita mengenal Intent lalu pada pemrograman website terdapat *tag* untuk berpindah dari satu *page* ke *page* lain. Pada Flutter kita akan menggunakan sebuah *class* bernama **Navigator**. Dengan Navigator ini kita akan berpindah dari satu screen ke screen lainnya. Berikut ini contohnya:


3:27
First Screen
Pindah Screen

Perlu kita ketahui bahwa konsep navigasi pada Flutter mirip sekali dengan pemrograman Android, yakni bahwa ketika berpindah *screen*/*activity* akan menjadi tumpukan (*stack*). Jadi ketika berpindah dari satu *screen* ke *screen* lain (*push*), maka *screen* pertama akan ditumpuk oleh *screen* kedua. Kemudian apabila kembali dari screen kedua ke pertama, maka screen kedua akan dihapus (pop).



Kita akan membuat kode seperti contoh di atas. Kita membutuhkan halaman kedua yang kodenya seperti berikut:

```
1. class SecondScreen extends StatelessWidget {
2.   @override
3.   Widget build(BuildContext context) {
4.     return Scaffold(
5.       appBar: AppBar(
6.         title: Text('Second Screen'),
7.       ),
8.       body: Center(
9.         child: OutlinedButton(
10.            child: Text('Kembali'),
11.            onPressed: () {},
12.          ),
13.        ),
14.      );
15.    }
16. }
```

Lalu, kode untuk halaman pertama akan seperti berikut:

```
class FirstScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('First Screen'),
      ),
      body: Center(
        child: ElevatedButton(
            child: Text('Pindah Screen'),
            onPressed: () {},
          ),
        ),
      );
  }
}
```

Navigator.push

Untuk berpindah ke *screen* kedua kita akan menggunakan sebuah method Navigator.push, method tersebut ditulis seperti berikut:

```
Navigator.push(context, MaterialPageRoute(builder: (context) {
    return WidgetScreen();
}));
```

Pada kode di atas Navigator.push memiliki dua parameter. Pertama ialah context dan yang kedua Route. Parameter *context* ini merupakan variabel *BuildContext* yang ada pada *method build*.
Parameter *route* berguna untuk menentukan tujuan ke mana kita akan berpindah *screen*. Route tersebut kita isikan dengan MaterialPageRoute yang di dalamnya terdapat *builder* yang nantinya akan diisi dengan tujuan *screen*-nya. Maka untuk melakukan perpindahan *screen* kita akan membuat event onPressed pada tombol RaisedButton yang ada pada *screen* pertama:

```
class FirstScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('First Screen'),
      ),
```

```
8.        body: Center(
9.          child: ElevatedButton(
10.             child: Text('Pindah Screen'),
11.             onPressed: () {
12.               Navigator.push(context, MaterialPageRoute(builder: (con
    text) {
13.                 return SecondScreen();
14.               }));
15.             },
16.           ),
17.         ),
18.       );
19.     }
20.   }
```

Navigator.pop

Setelah dapat berpindah ke *screen* lain dan kembali ke *screen* sebelumnya, maka kita akan belajar menggunakan Navigator.pop. Penulisan Navigator.pop seperti berikut:

```
1. Navigator.pop(context)
```

Pada Navigator.pop kita hanya cukup menambahkan parameter *context* yang merupakan variabel dari *method build*.

Untuk kembali dari screen kedua kita dapat menambahkan *event* onPressed pada RaisedButton yang ada pada *screen* kedua dan kita masukkan Navigator.pop pada *event*, seperti berikut:

```
1. class SecondScreen extends StatelessWidget {
2.   @override
3.   Widget build(BuildContext context) {
4.     return Scaffold(
5.       appBar: AppBar(
6.         title: Text('Second Screen'),
7.       ),
8.       body: Center(
9.         child: OutlinedButton(
10.            child: Text('Kembali'),
11.            onPressed: () {
12.              Navigator.pop(context);
13.            },
14.          ),
15.        ),
```

```
    );
  }
}
```

Mengirimkan Data Antar Halaman

Seringkali beberapa halaman pada aplikasi perlu saling berinteraksi dengan berbagi dan saling mengirimkan data. Pada Flutter kita memanfaatkan *constructor* dari sebuah class untuk mengirimkan data antar halaman.

Sebagai contoh kita memiliki pesan yang akan dikirimkan dari *First Screen* menuju *Second Screen*.

```
String message = 'Hello from First Screen!';
```

Untuk mengirimkan variabel message tersebut ke Second Screen, maka kita akan mengirimkannya sebagai parameter dari constructor kelas SecondScreen seperti berikut:

```
class FirstScreen extends StatelessWidget {
  final String message = 'Hello from First Screen!';

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('First Screen'),
      ),
       body: Center(
         child: ElevatedButton(
           child: Text('Pindah Screen'),
           onPressed: () {
             Navigator.push(context,
                 MaterialPageRoute(builder: (context) => SecondScree
n(message)));
           },
         ),
       ),
    );
  }
}
```

Agar *Second Screen* bisa menerima data tersebut, maka kita perlu mengubah *default constructor*-nya dan menambahkan variabel untuk menampung datanya.

```
class SecondScreen extends StatelessWidget {
  final String message;

  SecondScreen(this.message);
}
```

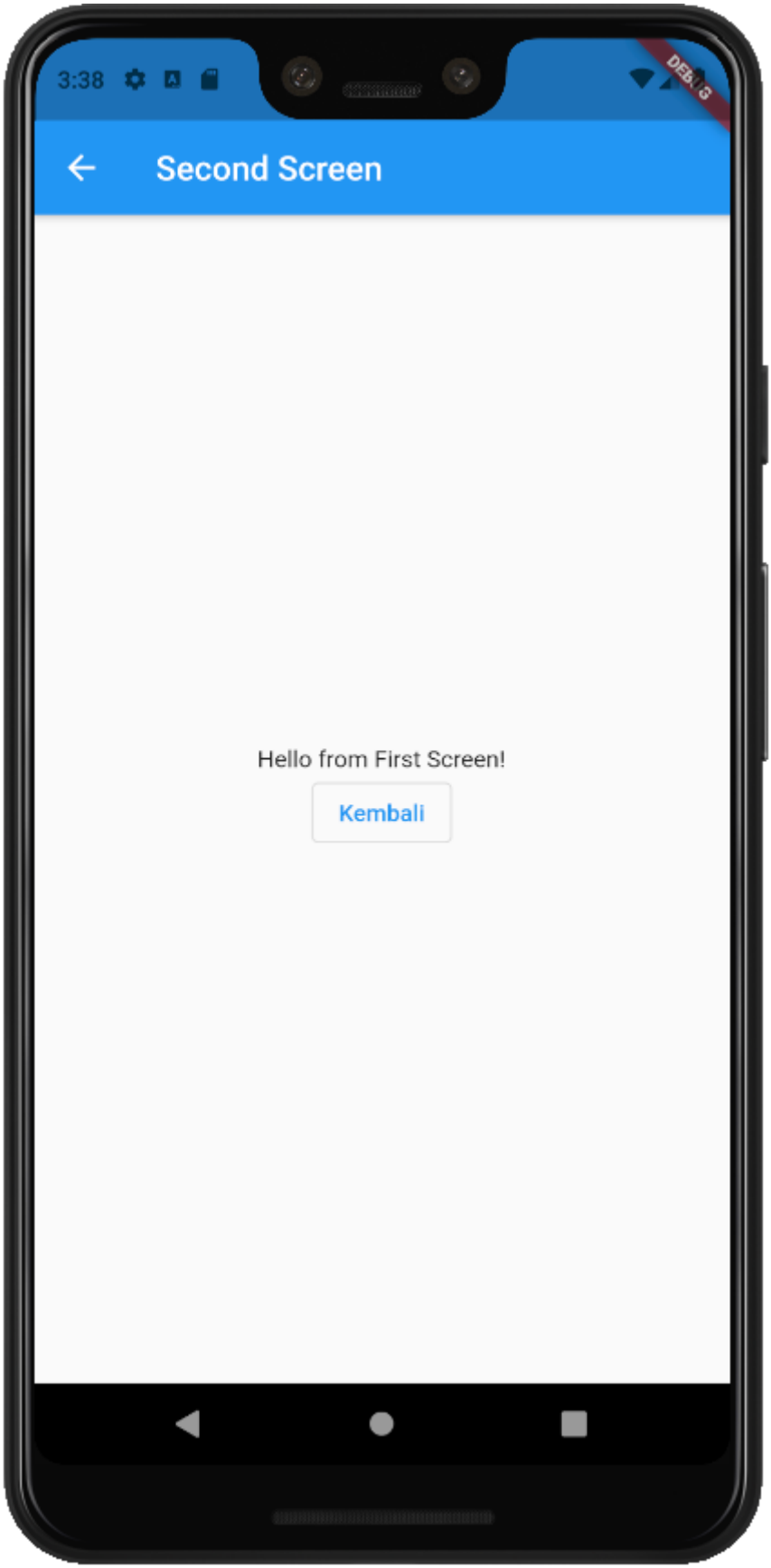
Kemudian kita dapat menampilkan data yang diterima melalui variabel yang kita buat.

```
class SecondScreen extends StatelessWidget {
  final String message;

  SecondScreen(this.message);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Second Screen'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(message),
            OutlinedButton(
              child: Text('Kembali'),
              onPressed: () {
                Navigator.pop(context);
              },
            ),
          ],
        ),
      ),
    );
  }
}
```

Sehingga tampilan *Second Screen* akan menampilkan pesan dari *First Screen* seperti berikut:


3:38
DEBUG
Second Screen
Hello from First Screen!
Kembali

Anda dapat memahami *Navigation* secara mendalam dengan membaca dokumentasi Navigation Cookbook.