Mobile Application

Pertemuan 15

Flutter Routes

- Sebuah Teknik yang digunakan Flutter untuk melakukan navigasi aplikasi
- Sebelumnya di Android Studio menggunakan Intent untuk berpindah halaman, kini digantikan dengan Routes
- Di Flutter, screen dan page disebut rute
- Hanya dengan menggunakan metode yang sederhana
  - Navigation.pop()
  - Navigation.push()

Lanjutan – Cara Kerja Route

- Navigasi Langsung dengan MaterialPageRoute
- Navigasi Statis dengan Peta Rute
- Navigasi Dinamis dengan onGenerateRoute

Navigasi Langsung dengan MaterialPageRoute

- Navigasi utama yang paling sering digunakan dalam pembuatan prototipe biasanya adalah navigasi langsung dengan MaterialPageRoute.
- Pengembang mengakses instance Navigator dan memasukkan MaterialPageRoute baru ke dalamnya.
- Rute membuat Widget baru yang diletakkan di atas tumpukan navigasi.

```
// Within the `HomeRoute` widget
onPressed: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => SecondRoute()),
  );
}
```

Navigasi Statis dengan Peta Rute

- Opsi kedua yang kami miliki adalah peta rute, dan ini adalah cara yang lebih disukai
- Pengembang memiliki opsi untuk membuat Peta dengan kunci String yang mengidentifikasi Metode WidgetBuilder yang berbeda dan menangani navigasi.

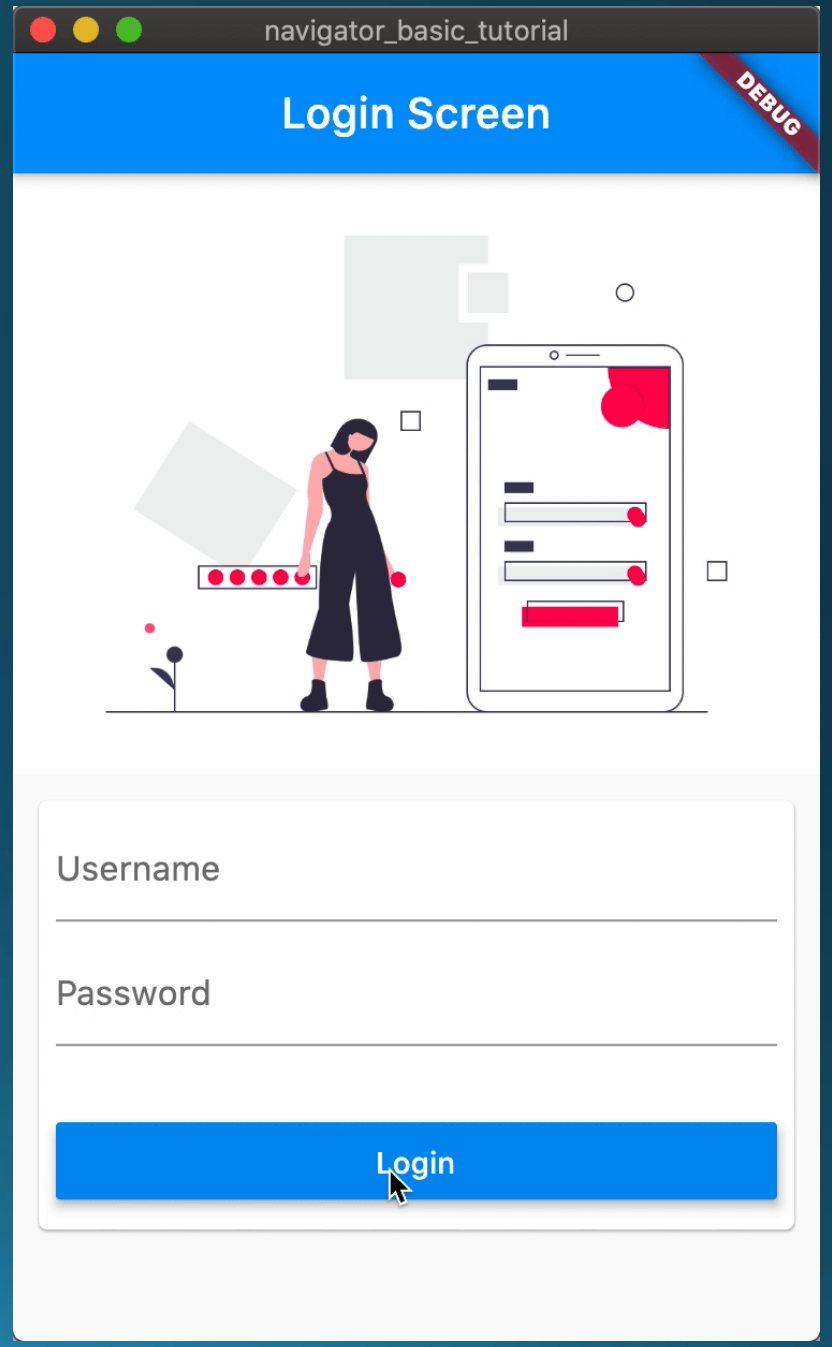
```
routes: {
  LobbyScreen.route: (context) => LobbyScreen(),
  LoginScreen.route: (context) => LoginScreen(),
  GameScreen.route: (context) => GameScreen(),
},
```

Navigasi Dinamis dengan onGenerateRoute

- Opsi ketiga yang ditambahkan adalah opsi untuk membuat onGenerateRoute. OnGenerateRoute memiliki keuntungan untuk dapat membuat rute dan mengakses argumen yang diteruskan.

```
onGenerateRoute: (settings) {
  switch (settings.name) {
    case LoginScreen.route:
      return MaterialPageRoute(builder: (_) => LoginScreen());
      break;
    case LobbyScreen.route:
      return MaterialPageRoute(builder: (_) => LobbyScreen());
      break;
    case GameScreen.route:
      return MaterialPageRoute(builder: (_) => GameScreen());
      break;
    default:
      return MaterialPageRoute(builder: (_) => LoginScreen());
  }
}
```

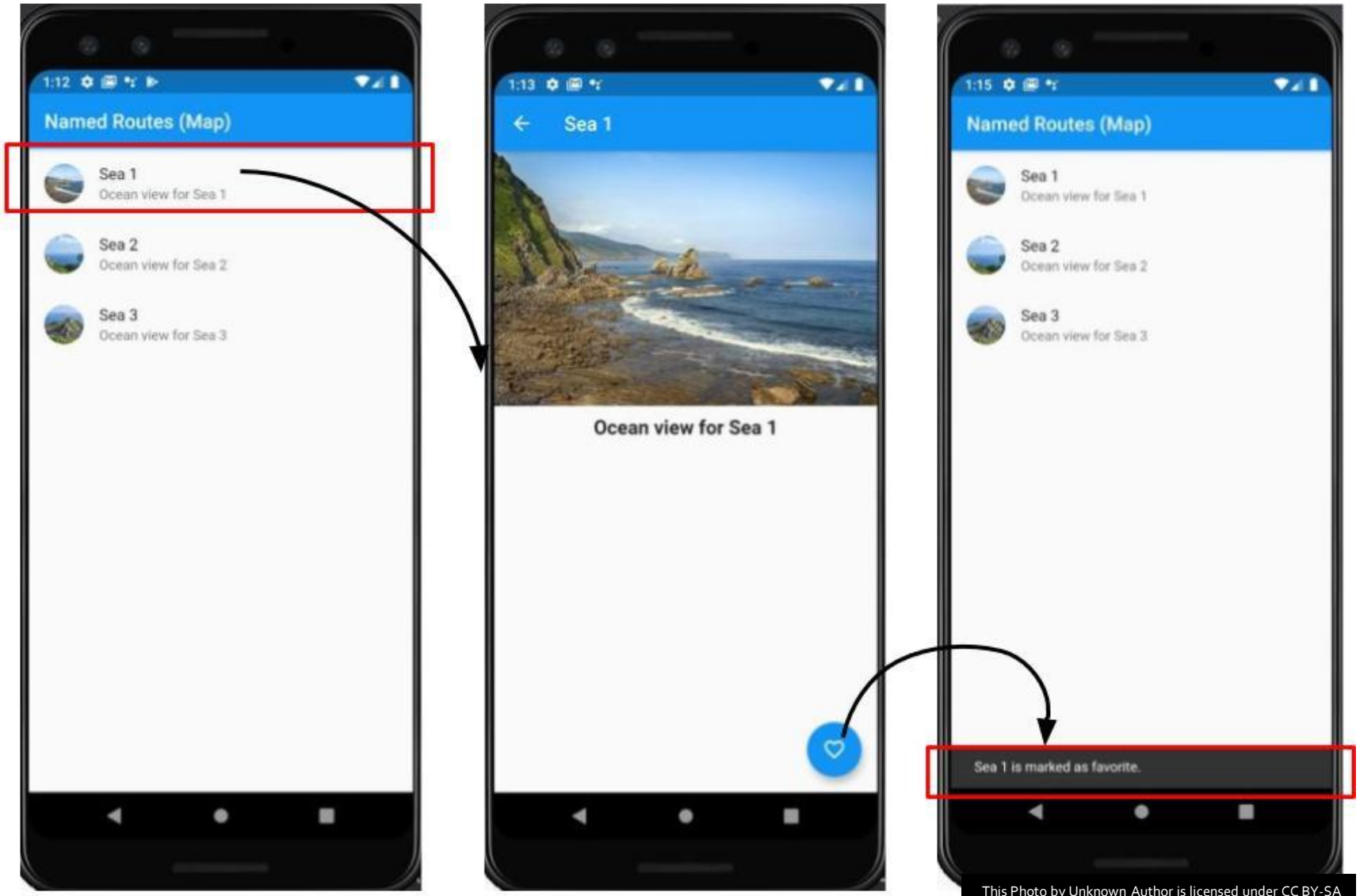
- Navigasi dengan menggunakan Flutter
- Semua operasinya dilakukan dengan Route



By using the Navigator.pop() method. The pop() method removes the current Route from the stack of routes managed by the Navigator.

Navigasi Pop() dan Push()

- Untuk beralih ke rute baru, gunakan metode Navigator.push(). Metode push() menambahkan Route ke tumpukan rute yang dikelola oleh Navigator
- Dengan menggunakan metode Navigator.pop(). Metode pop() menghapus Rute saat ini dari tumpukan rute yang dikelola oleh Navigator.



Navigasi Named Routes

- Jika perlu menavigasi ke screen yang sama di banyak bagian aplikasi, pendekatan sebelumnya dapat mengakibatkan duplikasi kode.
- Solusinya adalah dengan menentukan Named Routes, dan menggunakannya untuk navigasi.
- Untuk bekerja dengan cara ini, gunakan fungsi Navigator.pushNamed().

Lanjutan

- Cara menggunakan rute bernama menggunakan langkah-langkah berikut:
  - Buat dua layar.
  - Tentukan rute.
  - Navigasikan ke layar kedua menggunakan Navigator.pushNamed().
  - Kembali ke layar pertama menggunakan Navigator.pop().

Contoh

```
routes: {
  // When navigating to the "/" route, build the FirstScreen widget.
  '/': (context) => const FirstScreen(),
  // When navigating to the "/second" route, build the SecondScreen widget.
  '/second': (context) => const SecondScreen(),
},
```

```
// Within the `FirstScreen` widget
onPressed: () {
  // Navigate to the second screen using a named route.
  Navigator.pushNamed(context, '/second');
}
```

Mengirim Argumen

- Mirip dengan Android biasanya untuk mengirimkan data antar Activity
- Flutter juga bisa mengirimkan argument ke layer/screen lainnya
- Navigator menyediakan kemampuan untuk menavigasi ke rute bernama dari bagian mana pun dari aplikasi menggunakan pengenal umum.
- Dalam beberapa kasus, pengembang mungkin juga perlu meneruskan argumen ke rute bernama

Lanjutan

- Cara meneruskan argumen ke rute bernama dan membaca argumen menggunakan ModalRoute.of() dan onGenerateRoute() menggunakan langkah-langkah berikut:

- Tentukan argumen yang perlu disampaikan.
- Buat widget yang mengekstrak argumen.
- Daftarkan widget di tabel rute.
- Arahkan ke widget.

Tentukan argumen yang perlu disampaikan.

- Pertama, tentukan argumen yang perlu disampaikan ke rute baru. Dalam contoh ini, berikan dua bagian data: Judul layar dan pesan.
- Untuk meneruskan kedua bagian data, buat kelas yang menyimpan informasi ini.

```
// You can pass any object to the arguments parameter.
// In this example, create a class that contains both
// a customizable title and message.
class ScreenArguments {
  final String title;
  final String message;

  ScreenArguments(this.title, this.message);
}
```

Buat widget yang mengekstrak argumen

- Selanjutnya, buat widget yang mengekstrak dan menampilkan judul dan pesan dari ScreenArguments. Untuk mengakses ScreenArguments, gunakan metode ModalRoute.of().

```
final args = ModalRoute.of(context)!.settings.arguments as ScreenArguments;

return Scaffold(
  appBar: AppBar(
    title: Text(args.title),
  ),
  body: Center(
    child: Text(args.message),
  ),
);
```

Daftarkan widget di tabel rute.

- Selanjutnya, tambahkan entri ke rute yang disediakan ke widget MaterialApp.

```
MaterialApp(
  routes: {
    ExtractArgumentsScreen.routeName: (context) =>
        const ExtractArgumentsScreen(),
  },
)
```

Arahkan ke widget.

- Selanjutnya, tambahkan entri ke rute yang disediakan ke widget MaterialApp.

```
Navigator.pushNamed(
  context,
  ExtractArgumentsScreen.routeName,
  arguments: ScreenArguments(
    'Extract Arguments Screen',
    'This message is extracted in the build method.'
  ),
);
```

The End

- Selamat Menikmati UAS