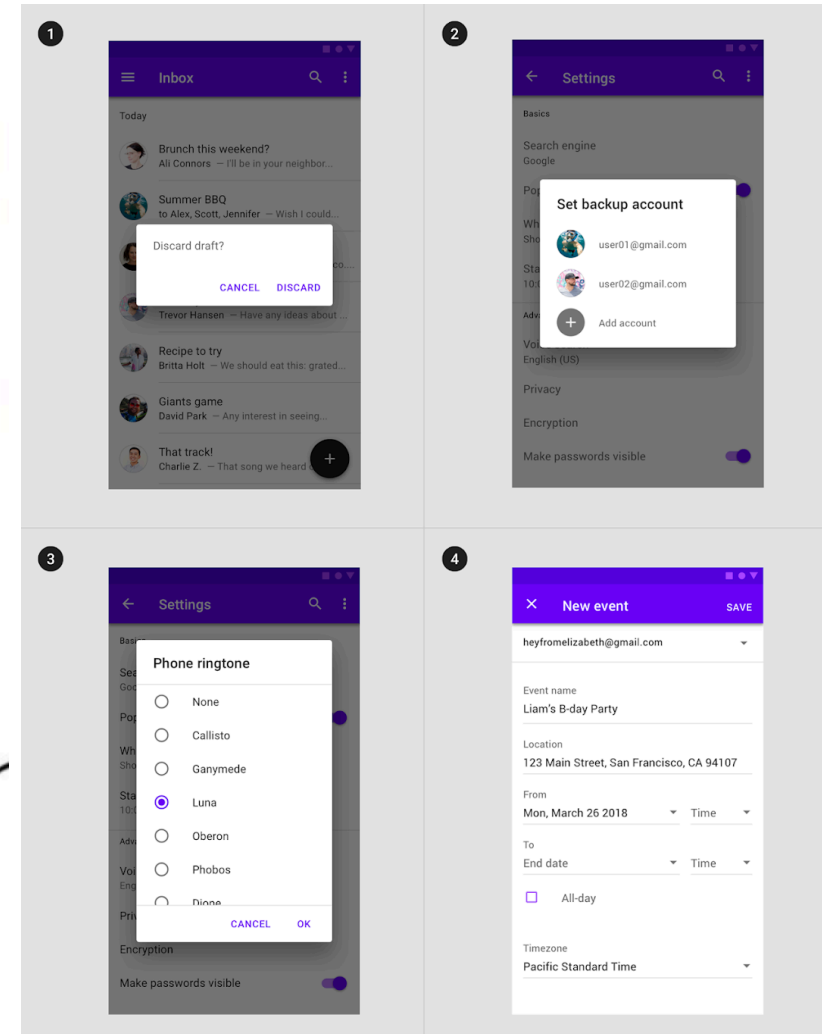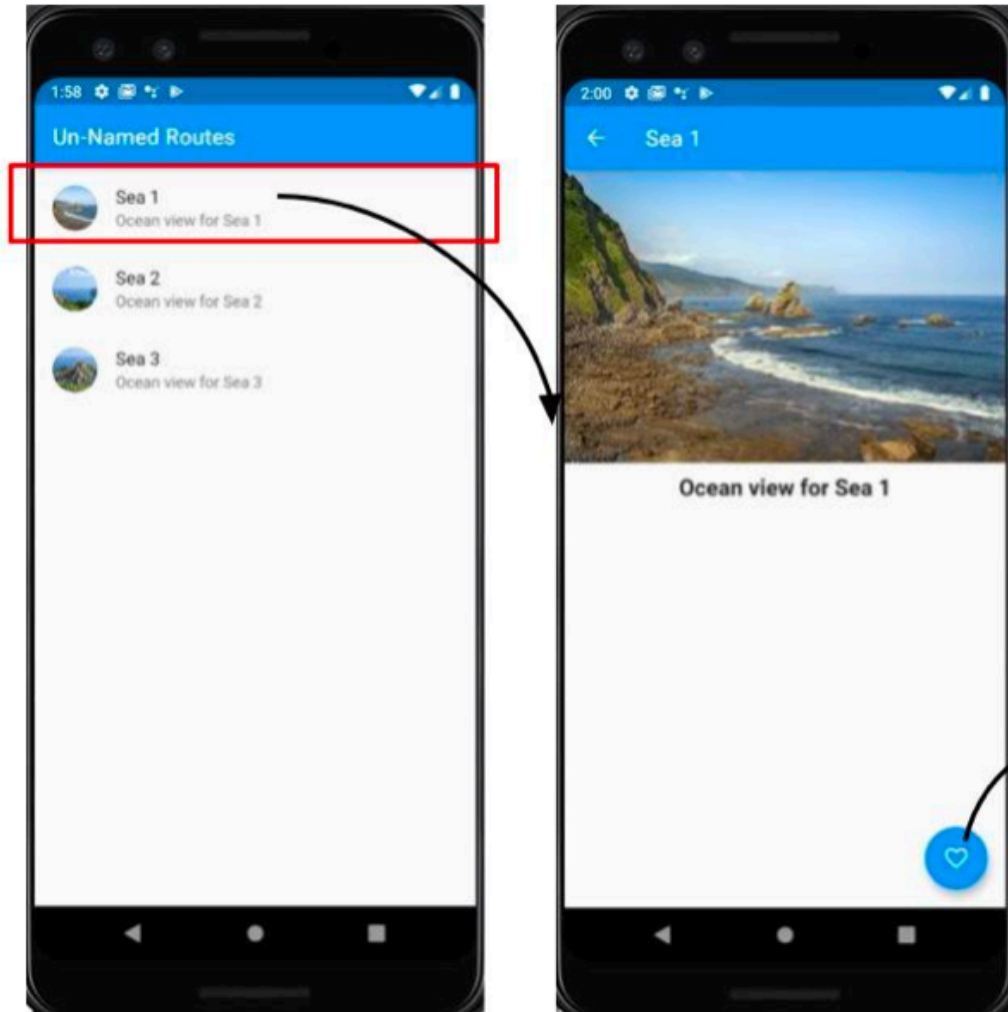# Navigation and Dialogs

Contents

- Navigation

- Dialogs

Navigation (1)

1. Navigator.**push**()

2. Navigator.**pushAndRemoveUntil**()

3. Navigator.**pop**()

4. **RouteSettings**

Navigation (2)

- Let's assume you have three screens: HomeScreen, ProfileScreen, and SettingsScreen
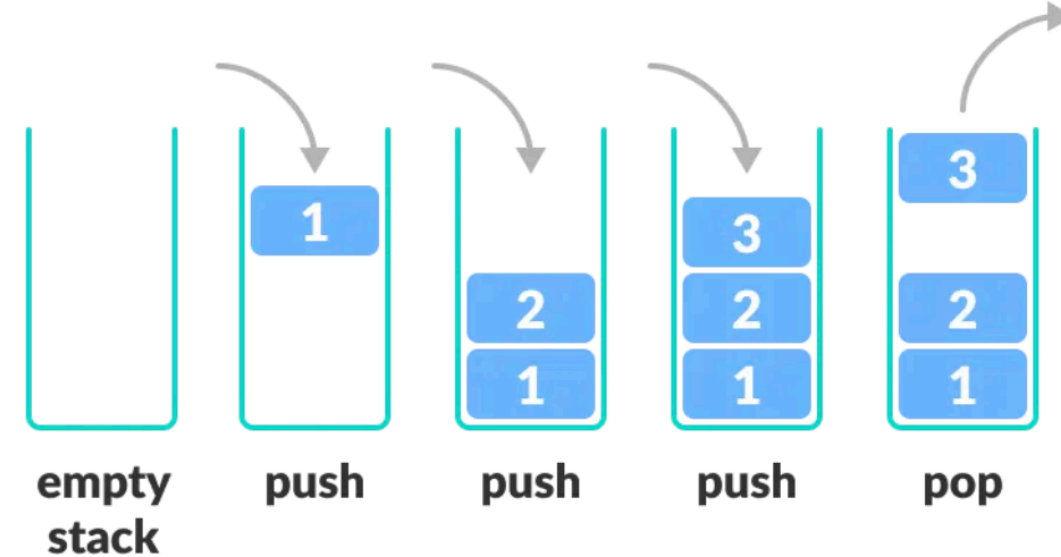- The app starts with the **HomeScreen**

Stack: [HomeScreen]

- Navigating from **HomeScreen** to **ProfileScreen**:

Stack: [
ProfileScreen
HomeScreen
]

- Navigating from **ProfileScreen** to **SettingsScreen**

Stack: [
SettingsScreen
ProfileScreen
HomeScreen
]

empty stack   push   push   push   pop

Navigation (3)

- Navigator.push(): Open a new page
- Navigating from **HomeScreen** to **ProfileScreen**

```
Navigator.push(
  context,
  MaterialPageRoute(builder: (context) => ProfileScreen()),
);
```

- Stack now looks like

```
Stack: [
        ProfileScreen
        HomeScreen
        ]
```

Navigation (4)

- Navigator.**pushAndRemoveUntil**(): Open new page and clear all all of the previous screens from the navigation stack

- Example: Suppose you have a login flow, and after successful login, you want to navigate to the home screen and remove all previous routes (so the user can't navigate back to the login page):

```
Navigator.pushAndRemoveUntil(
  context,
  MaterialPageRoute(builder: (context) => HomeScreen()),
);
```

- Stack now looks like

```
Stack: [
        HomeScreen
        ]
```
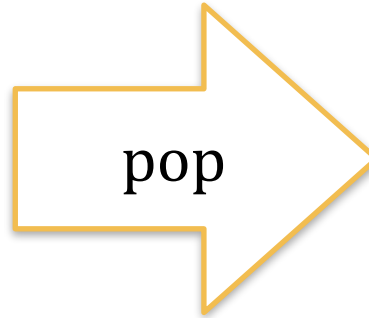
Navigation (5)

- Navigator.**pop**(): Return to the previous
- Pressing the back button (or calling **Navigator.pop()**):

Navigator.pop(context);

- Stack now looks like

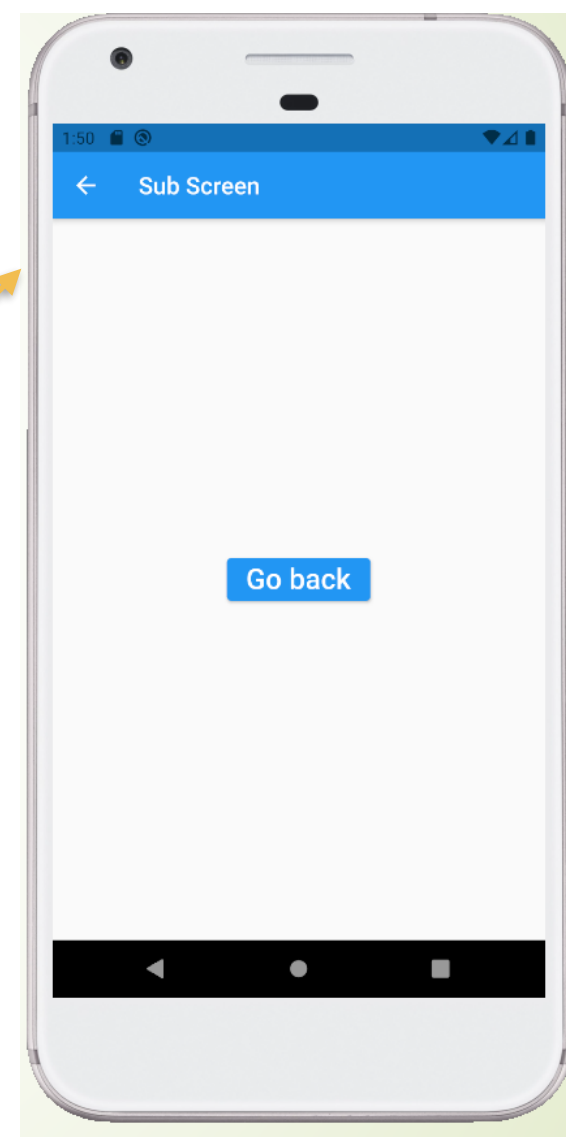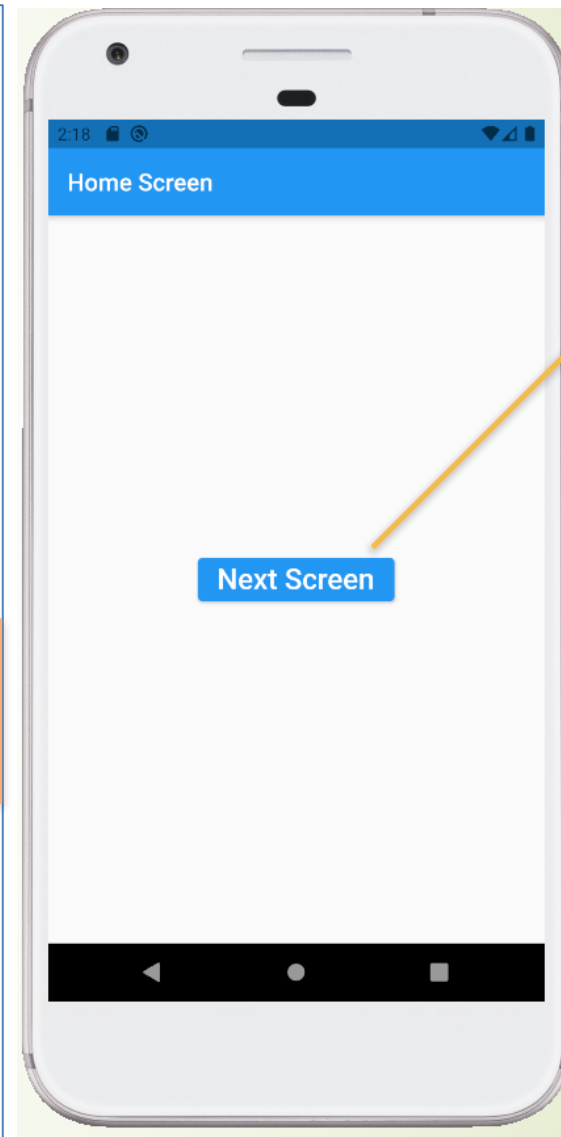Stack: [
    ProfileScreen
    HomeScreen
    ]

pop

Stack: [
    HomeScreen
    ]

Home Screen

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

class HomeRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Home Screen"),
      ), // AppBar
      body: Center(
        child: ElevatedButton(
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(builder: (context) => SubRoute()),
            );
          },
          child: Text('Next Screen',style: TextStyle(fontSize: 24)),
        ), // ElevatedButton
      ), // Center
    ); // Scaffold
  }
}
```
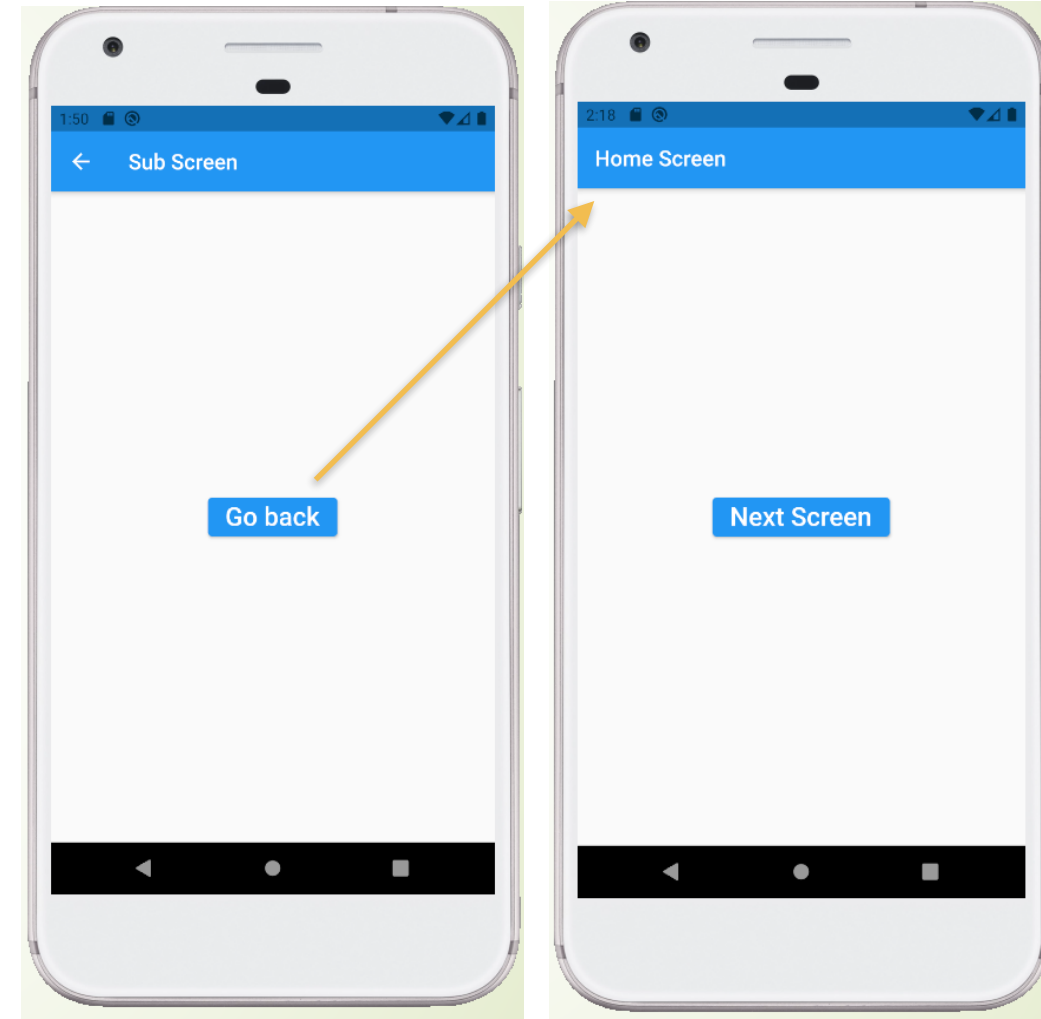
Navigator.push()

Sub Screen

```dart
import 'package:flutter/material.dart';
class SubRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Sub Screen"),
      ), // AppBar
      body: Center(
        child: ElevatedButton(
          onPressed: () {
            Navigator.pop(context);
          },
          child: Text('Go back', style: TextStyle(fontSize: 24)),
        ), // ElevatedButton
      ), // Center
    ); // Scaffold
  }
}
```
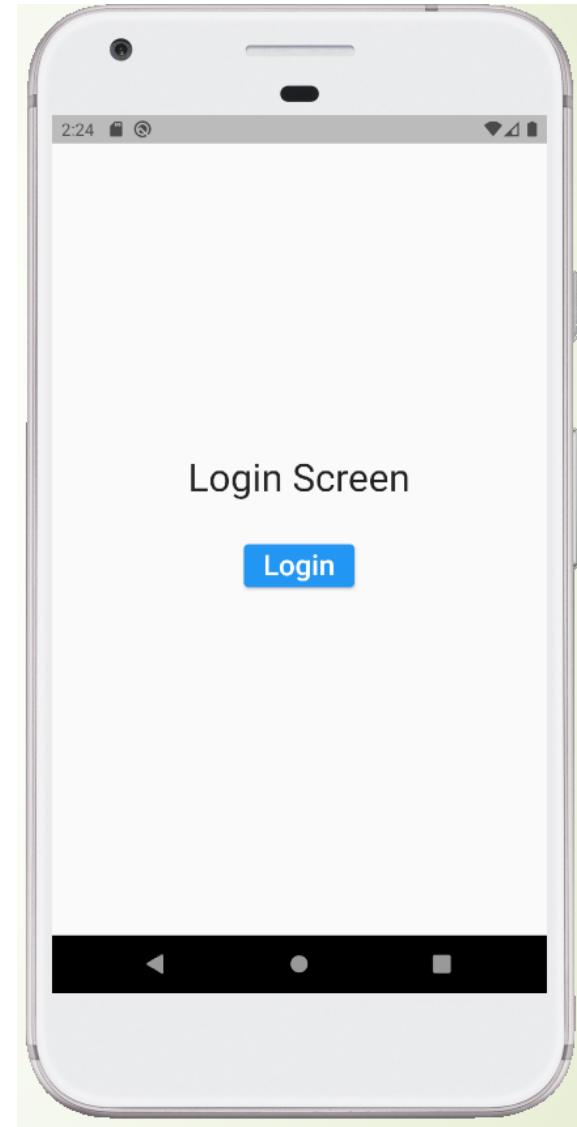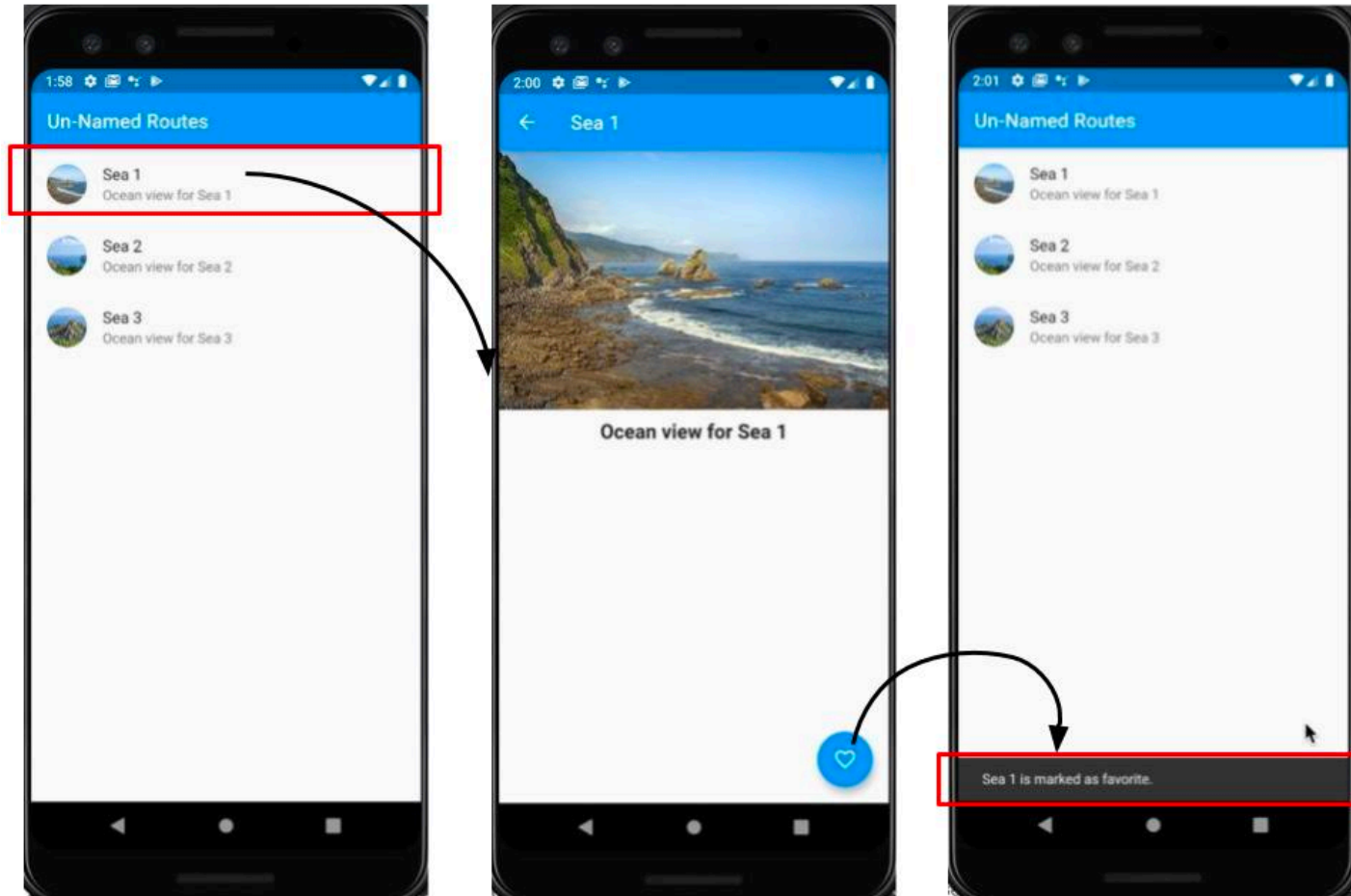
Navigator.pop()

Login Screen

```dart
import 'package:flutter/material.dart';
import 'home.dart';
class LoginRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text('Login Screen', style: TextStyle(fontSize: 32)),
            SizedBox(height: 30),
            ElevatedButton(
              onPressed: () {
                Navigator.pushAndRemoveUntil(context,
                  MaterialPageRoute(builder: (context) => HomeRoute()),
                  (route) => false);
              },
              child: Text('Login', style: TextStyle(fontSize: 24)),
            ), // ElevatedButton
          ],
        )), // Column // Center
    ); // Scaffold
  }
}
```

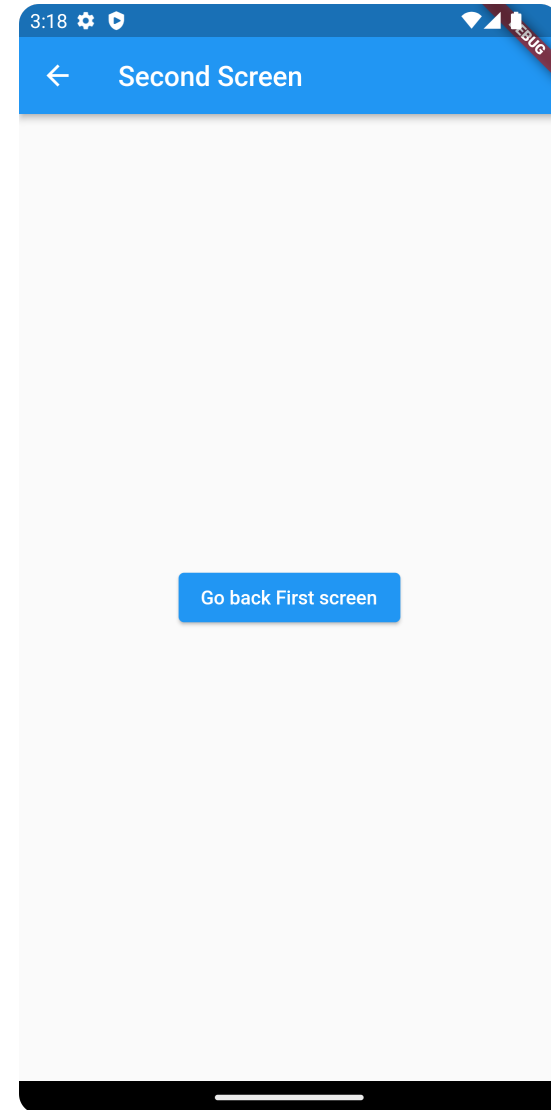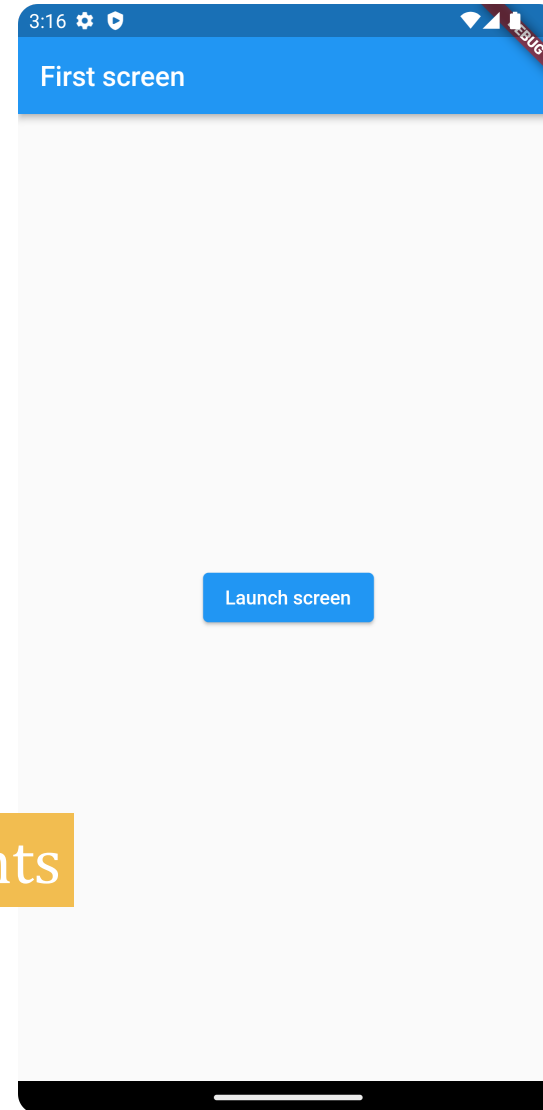**Navigator.pushAndRemoveUntil()**

RouteSettings (1)

- The **information needs to be sent from one screen to another.**

- Pass the arguments using **RouteSettings**

RouteSettings (2)

```dart
class FirstScreen extends StatelessWidget {
  const FirstScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('First screen'),
      ), // AppBar
      body: Center(
        child: ElevatedButton(
          onPressed: () => {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => const SecondScreen(),
                settings: const RouteSettings(arguments: 'First screen')))
          },
          child: const Text('Launch screen'),
        ), // ElevatedButton
      ), // Center
    ); // Scaffold
  }
}
```
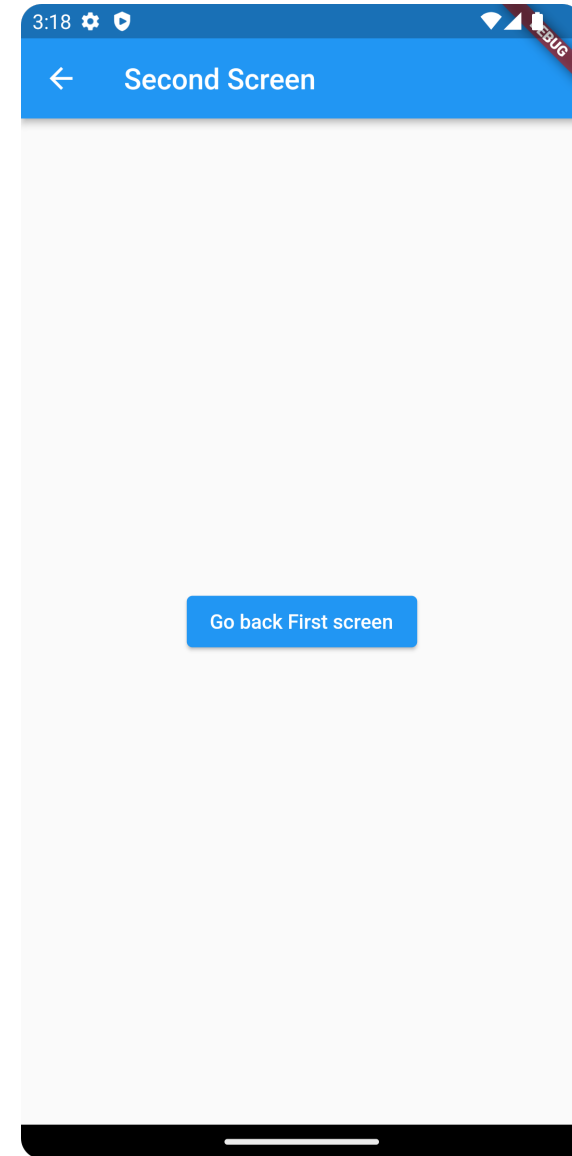
Pass the arguments

First screen

Launch screen

← Second Screen

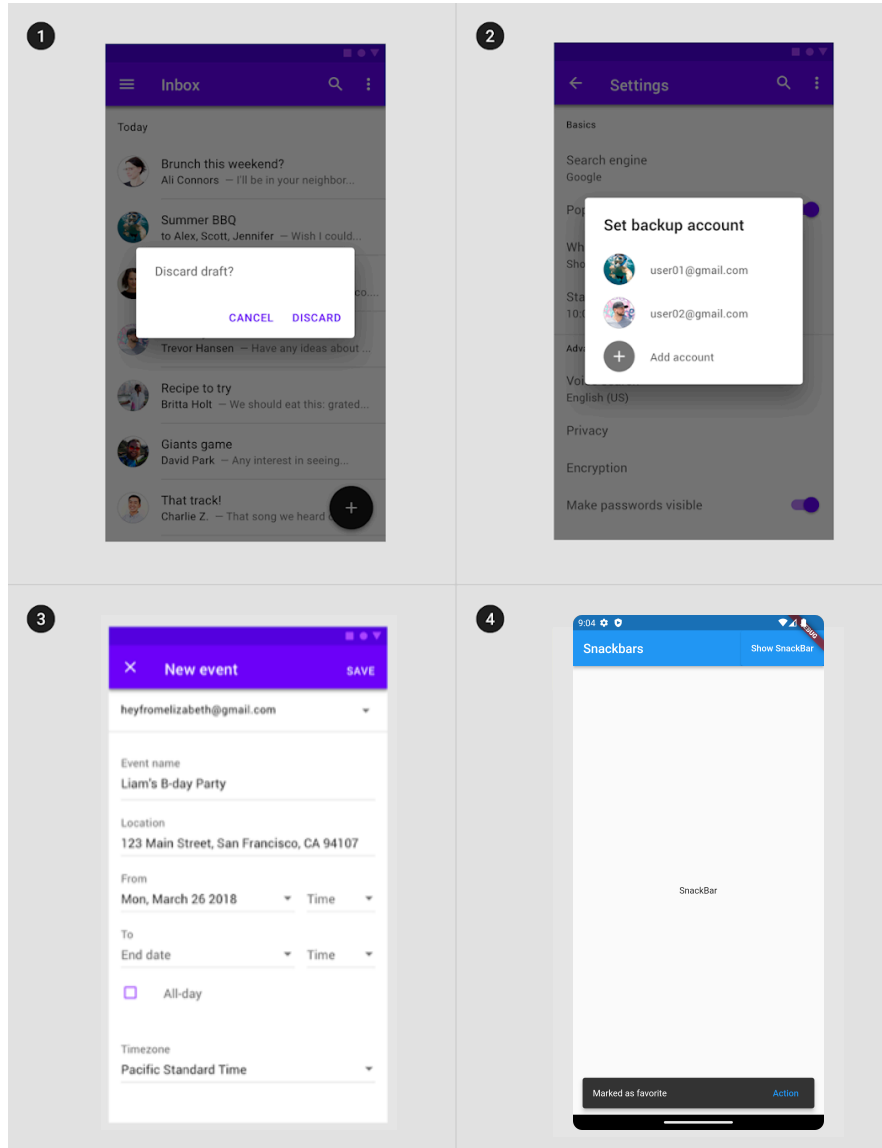Go back First screen

RouteSettings (3)

-

```dart
class SecondScreen extends StatelessWidget {
  const SecondScreen({super.key});

  @override
  Widget build(BuildContext context) {
    final data = ModalRoute.of(context)?.settings.arguments as String;
```

Receive the arguments

```dart
    return Scaffold(
      appBar: AppBar(
        title: const Text('Second Screen'),
      ), // AppBar
      body: Center(
        child: ElevatedButton(
          onPressed: () => {Navigator.pop(context)},
          child: Text('Go back $data'),
        ), // ElevatedButton
      ), // Center
    ); // Scaffold
  }
}
```
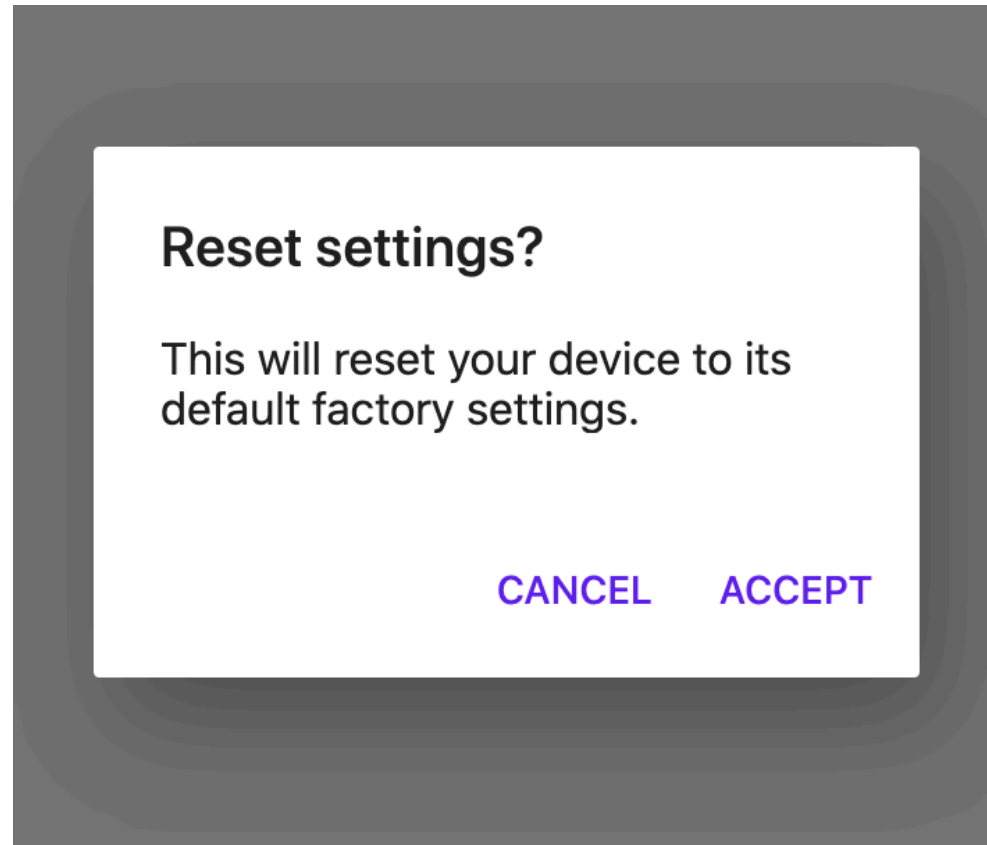
Dialogs

1. Alert

2. Simple

3. Full-screen

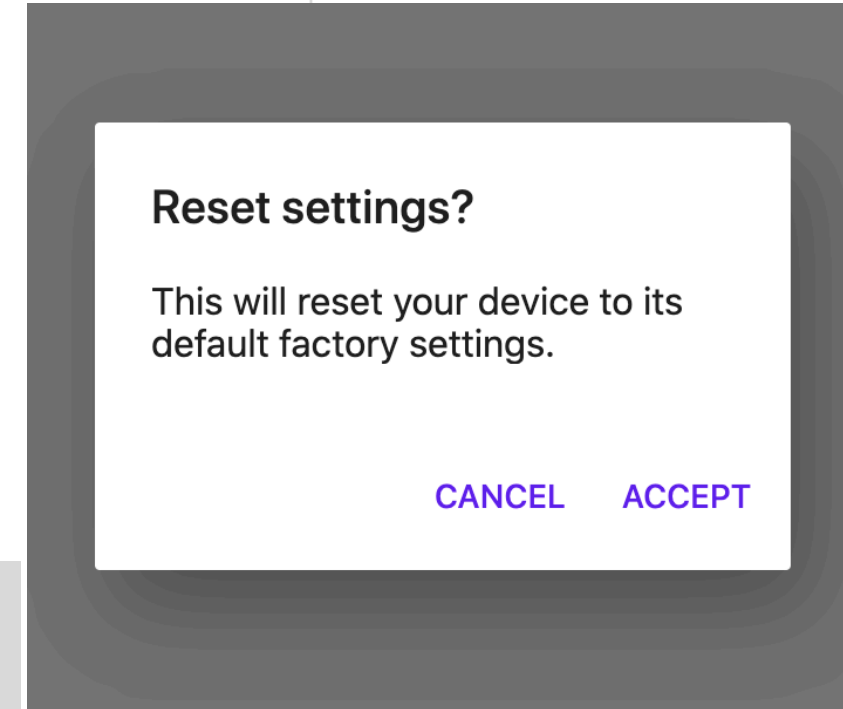4. Snackbars

Alert Dialog (1)

- Alert dialogs **interrupt users** with urgent information, details, or actions.

- The following example shows an alert dialog.

Alert Dialog (2)

```
final AlertDialog dialog = AlertDialog(
├── title: const Text('Reset settings?'),
│   content:
├── const Text('This will reset your device to its default factory settings.'),
│   actions: [
│   ├── ElevatedButton(
│   │       onPressed: () => Navigator.pop(context),
│   │   └── child: const Text('CANCEL'),
│   │   ), // ElevatedButton
│   ├── ElevatedButton(
│   │       onPressed: () => Navigator.pop(context),
│   │   └── child: const Text('ACCEPT'),
│   │   ), // ElevatedButton
│   ],
); // AlertDialog
```

Reset settings?

This will reset your device to its default factory settings.
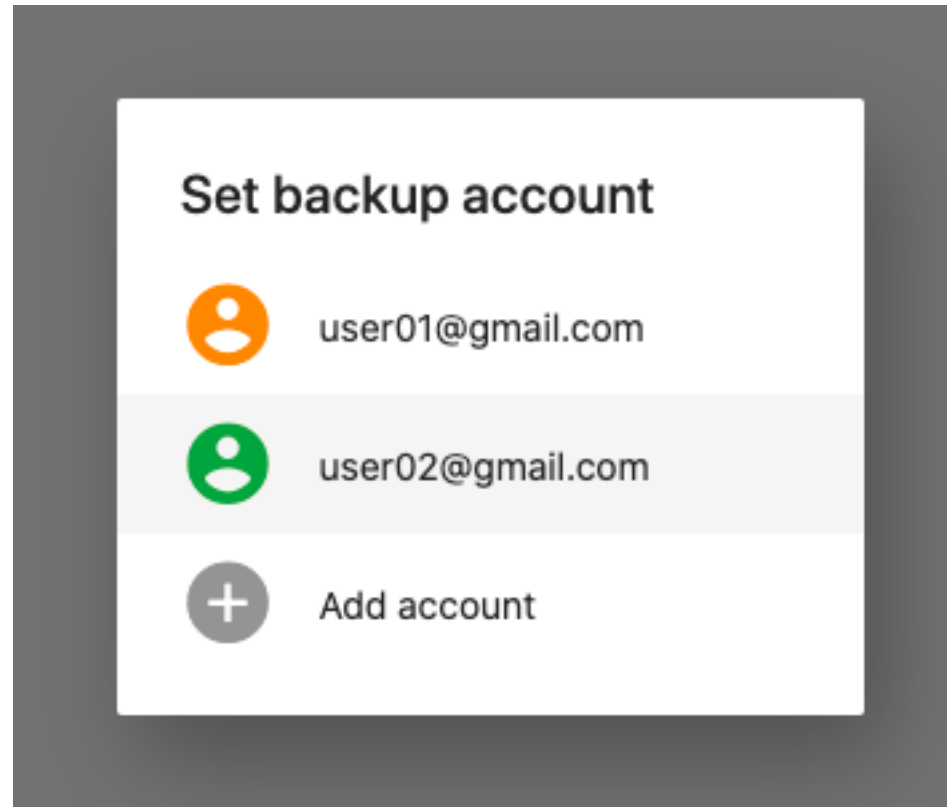
CANCEL   ACCEPT

```
showDialog<void>(
        context: context,
        builder: (context) => dialog);
```

Simple Dialog (1)

- Simple dialogs can **display items** that are **immediately actionable** when **selected**. They **don't have text buttons**.

- The following example shows a simple dialog.

```dart
class SimpleDialogItem extends StatelessWidget {
  const SimpleDialogItem(
      {super.key, this.icon, this.color, this.text, this.onPressed});

  final IconData? icon;
  final Color? color;
  final String? text;
  final VoidCallback? onPressed;

  @override
  Widget build(BuildContext context) {
    return SimpleDialogOption(
      onPressed: onPressed,
      child: Row(
        mainAxisAlignment: MainAxisAlignment.start,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Icon(icon, size: 36.0, color: color),
          Padding(
            padding: const EdgeInsetsDirectional.only(start: 16.0),
            child: Text(text!),
          ), // Padding
        ],
      ), // Row
    ); // SimpleDialogOption
  }
}
```
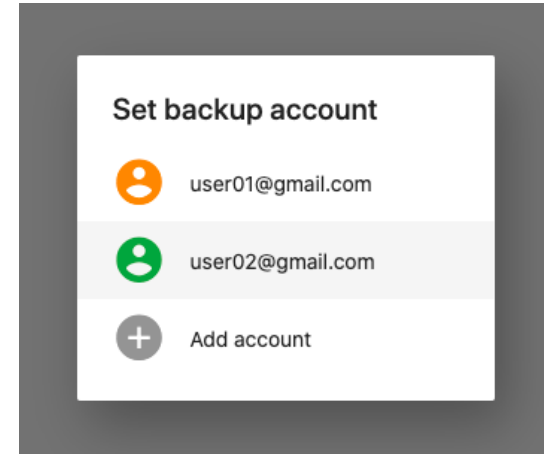
1

```dart
SimpleDialog createSimpleDialog(BuildContext context) {
  return SimpleDialog(
    title: const Text('Set backup account'),
    children: [
      SimpleDialogItem(
        icon: Icons.account_circle,
        color: Colors.orange,
        text: 'user01@gmail.com',
        onPressed: () {
          Navigator.pop(context, 'user01@gmail.com');
        },
      ), // SimpleDialogItem
      SimpleDialogItem(
        icon: Icons.account_circle,
        color: Colors.green,
        text: 'user02@gmail.com',
        onPressed: () {
          Navigator.pop(context, 'user02@gmail.com');
        },
      ), // SimpleDialogItem
      SimpleDialogItem(
        icon: Icons.add_circle,
        color: Colors.grey,
        text: 'Add account',
        onPressed: () {
          Navigator.pop(context, 'Add account');
        },
      ), // SimpleDialogItem
    ],
  ); // SimpleDialog
}
```

2

Set backup account

👤 user01@gmail.com

👤 user02@gmail.com

➕ Add account

```dart
final SimpleDialog dialog = createSimpleDialog(context);

return Scaffold(
  body: Center(
    child: ElevatedButton(
      onPressed: () {
        showDialog<void>(context: context, builder: (context) => dialog);
      },
      child: const Text("SHOW DIALOG"),
    ), // ElevatedButton
  ), // Center
); // Scaffold
```
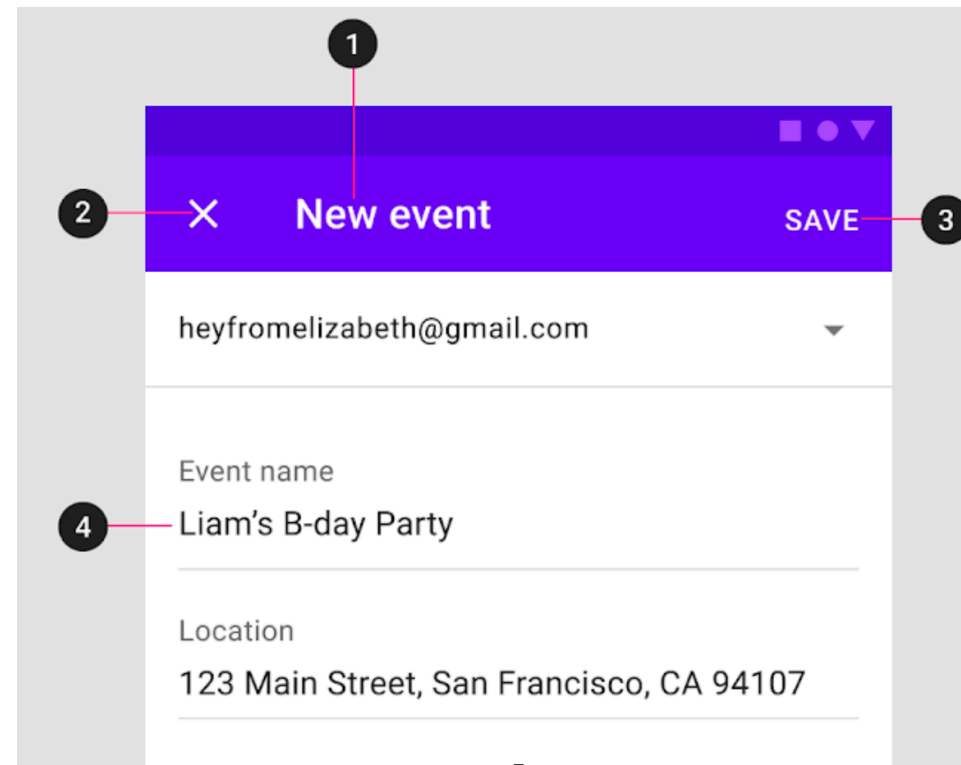
3

Full-screen Dialog (1)

- Full-screen dialogs group a series of tasks, such as creating a calendar entry with the event title, date, location, and time.

- To use a full-screen dialog, simply set the **fullscreenDialog** to true when pushing a new **MaterialPageRoute**.
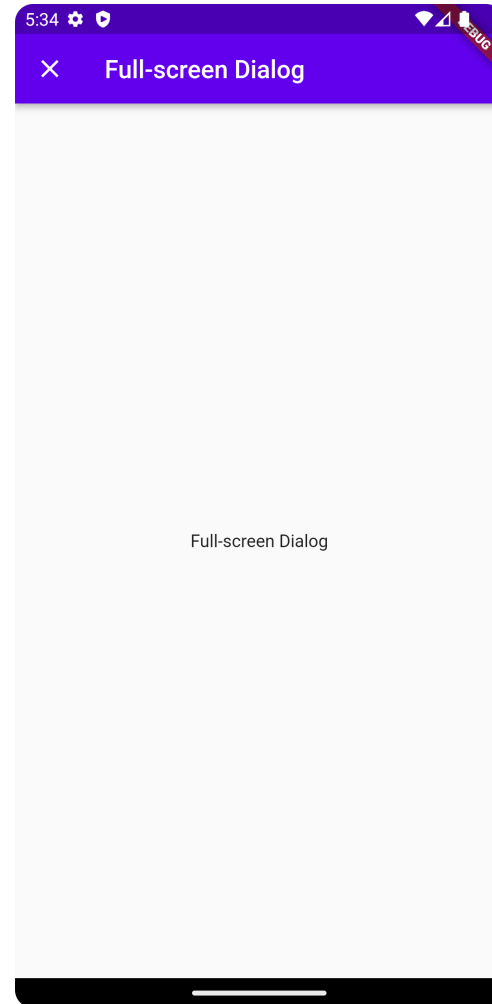
```dart
class FullScreenDialog extends StatelessWidget {
  const FullScreenDialog({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: const Color(0xFF6200EE),
        title: const Text('Full-screen Dialog'),
      ), // AppBar
      body: const Center(
        child: Text('Full-screen Dialog'),
      ), // Center
    ); // Scaffold
  }
}
```
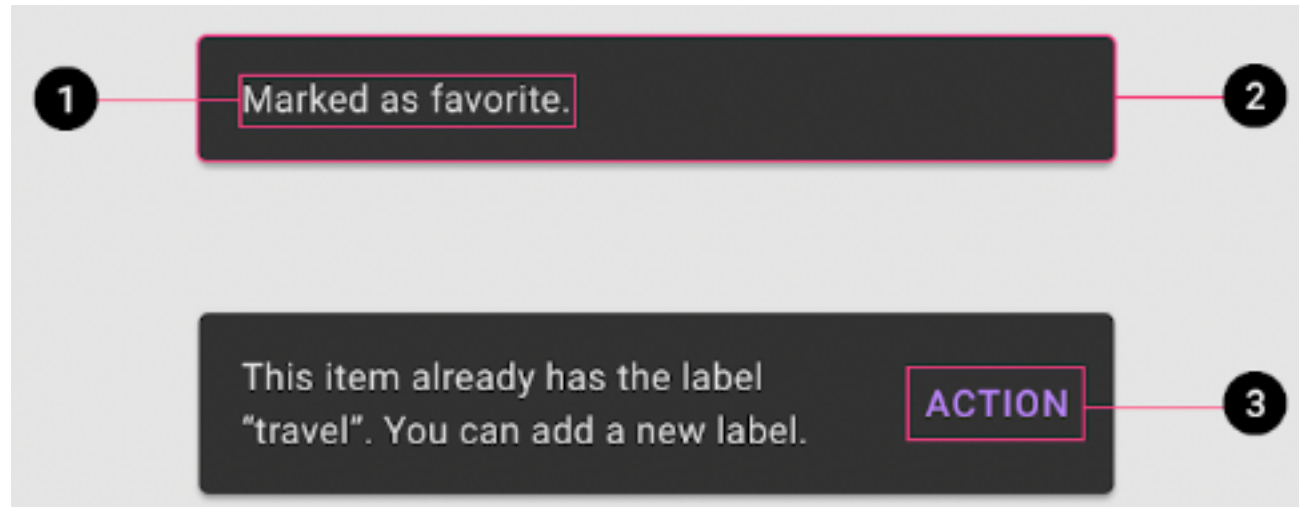
1

```dart
class MyFullScreenPage extends StatelessWidget {
  const MyFullScreenPage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Using Dialogs'),
      ), // AppBar
      body: Center(
        child: ElevatedButton(
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => const FullScreenDialog(),
                fullscreenDialog: true,
              )); // MaterialPageRoute
          },
          child: const Text('Show Full-screen Dialog'),
        ), // ElevatedButton
      ), // Center
    ); // Scaffold
  }
}
```

2

5:34

×   Full-screen Dialog

Full-screen Dialog

Snackbars (1)

- Snackbars inform users of a **process** that an app has **performed** or **will perform**. They **appear temporarily**, towards the **bottom** of the screen. They shouldn't interrupt the user experience, and they don't require user input to disappear.

- The following is an anatomy diagram of a snackbar

  1. Text label
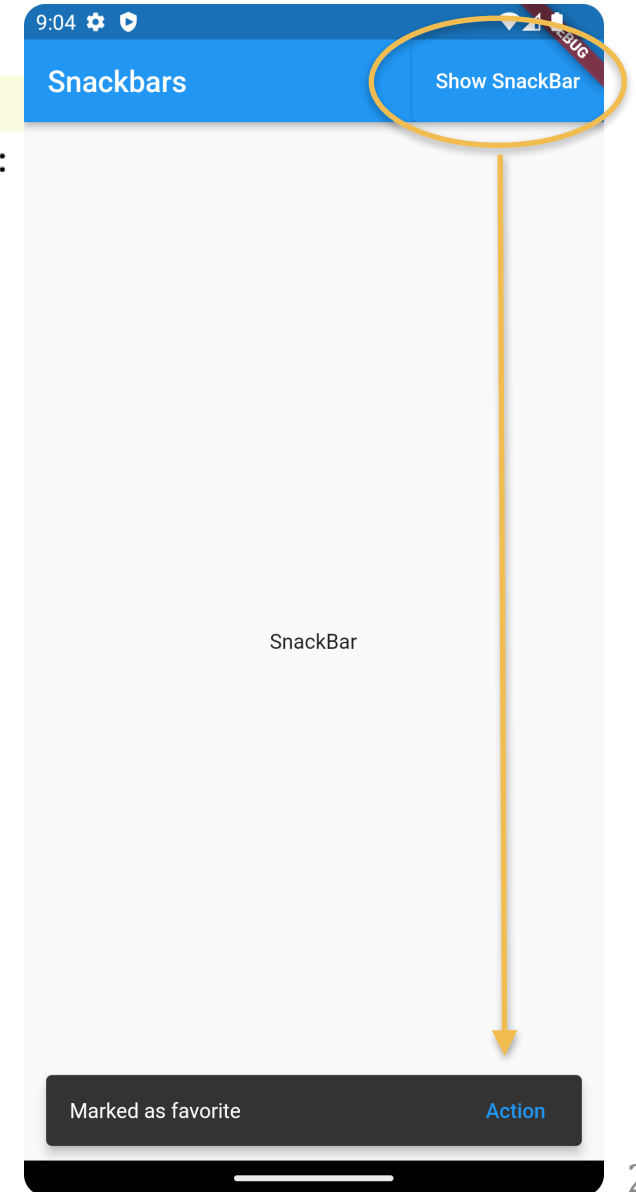 2. Container
  3. Action (optional)

Snackbars (2)

```dart
class SnackBarButton extends StatelessWidget {
  const SnackBarButton({super.key});

  @override
  Widget build(BuildContext context) {
    return ElevatedButton(
        onPressed: () {
          final snackBar = SnackBar(
            content: const Text('Marked as favorite'),
            behavior: SnackBarBehavior.floating,
            action: SnackBarAction(
                label: 'Action',
                onPressed: () {}),  // SnackBarAction
          );  // SnackBar

          ScaffoldMessenger.of(context).showSnackBar(snackBar);
        },
        child: const Text('Show SnackBar'));  // ElevatedButton
  }
}
```

1

2

```dart
class SnackBarsDemo extends StatelessWidget {
  const SnackBarsDemo({Key? key}) : super(key:

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Snackbars'),
        actions: const [SnackBarButton()],
      ),  // AppBar
      body: const Center(
        child: Text('SnackBar'),
      ),  // Center
    );  // Scaffold
  }
}
```
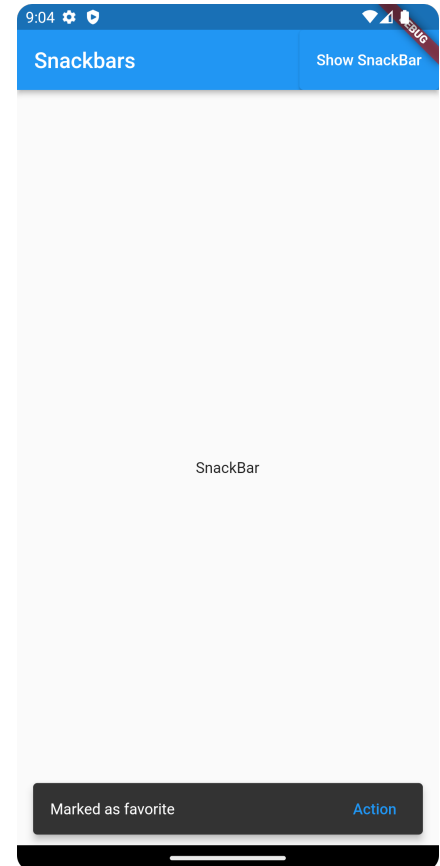
- The default duration of a SnackBar in Flutter is 4 seconds.

- If you want to customize the duration, you can modify the **duration** parameter as shown below:

```
final snackBar = SnackBar(
  content: Text(msg),
  behavior: SnackBarBehavior.floating,
  action: SnackBarAction(label: label, onPressed: () {}),
  duration: Duration(seconds: 2), // Custom duration
);
```

*Keeping up those **inspiration** and the **enthusiasm** in the **learning path**.*
*Let confidence to bring it into **your career path** for getting gain the **success***
as your expectation.

Thank you

Contact

- **Name: R2S Academy**
- **Email: daotao@r2s.edu.vn**
- **Phone/Zalo: 0919 365 363**
- **FB: https://www.facebook.com/r2s.tuyendung**
- **Website: www.r2s.edu.vn**

Questions and Answers