앱 개발 입문

성균관대학교 컬처앤테크놀로지융합전공 하계 부트캠프

5일차 - 2022. 08. 02 (화)

강의 슬라이드 링크

https://github.com/kunny/skku-bootcamp-2022-summer/blob/main/_slides/day5.pdf

오늘 강의에서 다룰 내용

- 플러터에서 화면을 이동하는 방법을 알아봅니다.
- 노트 목록 화면과 편집 화면을 루트에 추가합니다.
- 노트 보기 화면을 구현합니다.
- 노트 목록, 보기, 편집 화면을 연결합니다.
- 데이터베이스를 알아봅니다.

플러터에서 화면 이동 처리하기

플러터에서 화면 이동 처리하기

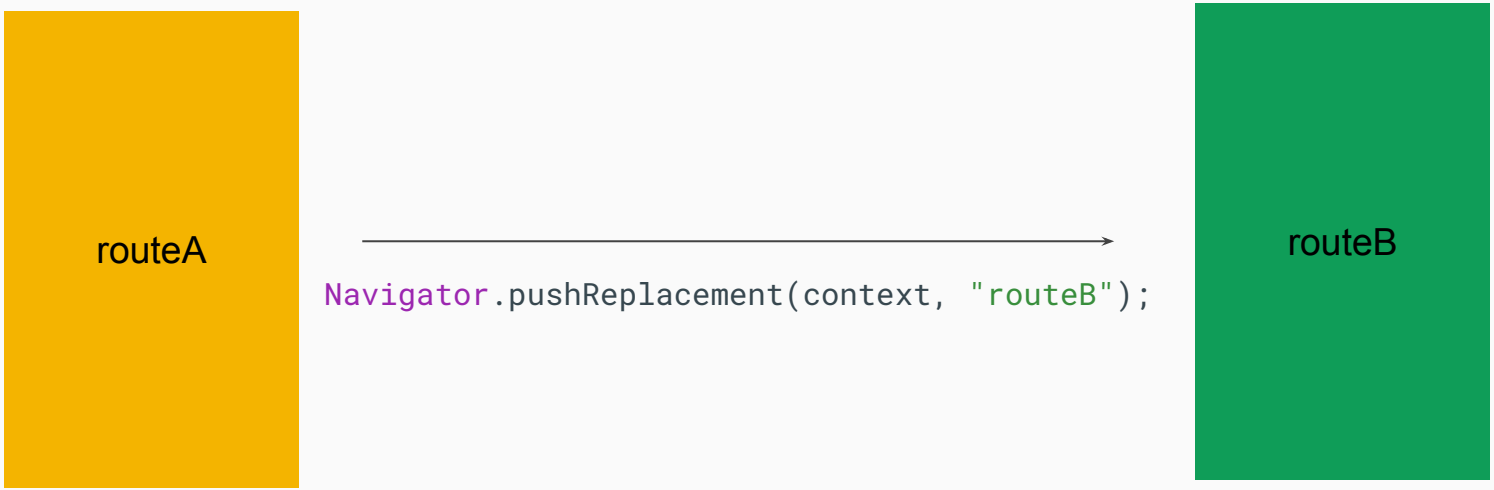
- 각 화면은 스택(Stack) 방식으로 관리되며, **`Navigator`** 클래스 내 다양한 함수를 사용하여 화면을 이동합니다.
- 앱 내의 각 화면은 **루트(Route)**라 부르며, 이름을 지정한 화면은 **Named Route**라 부릅니다.
- **`MaterialApp`**의 **`routes`** 속성을 사용하여 앱에서 이동할 수 있는 화면을 정의합니다.
- Named Route를 호출할 때에는 인자(**`arguments`**)를 추가로 전달할 수 있습니다.
- 호출했던 화면에서 기존 화면으로 돌아왔을 때 이벤트를 받을 수 있습니다. (**`Future`**)

플러터에서 화면 이동 처리하기

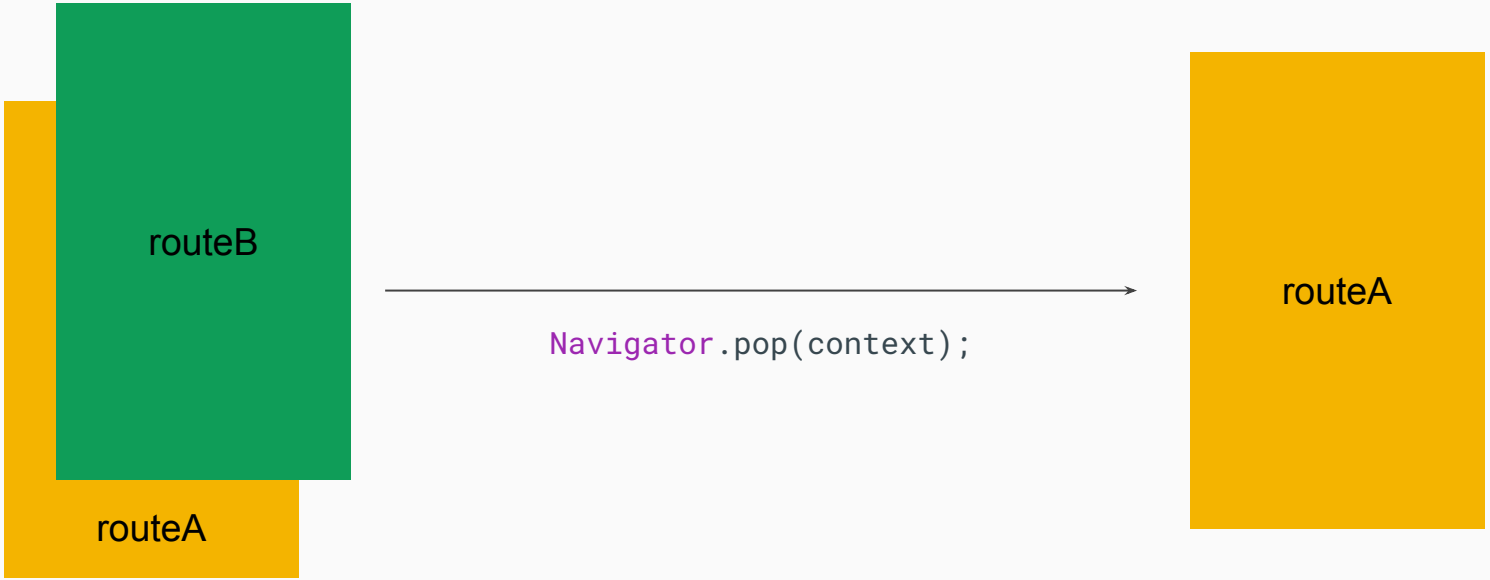
- 새 화면 열기 (현재 화면 유지)
  - Navigator.push() / Navigator.pushNamed()
- 현재 화면을 새 화면으로 대체
  - Navigator.pushReplacement() / Navigator.pushReplacementNamed()
- 현재 화면을 닫고 기존 화면으로 복귀
  - Navigator.pop()
- 현재 화면을 닫고 특정 조건을 만족하는 화면으로 복귀
  - Navigator.popUntil()

플러터에서 화면 이동 처리하기

플러터에서 화면 이동 처리하기



플러터에서 화면 이동 처리하기



플러터에서 화면 이동 처리하기

노트 목록 화면과 편집 화면을 루트에 추가하기

노트 목록 화면과 편집 화면을 루트에 추가하기

1. **NoteListPage**와 **NoteEditPage** 화면에 이름을 지어줍니다.
2. **MyApp** 클래스 내 **MaterialApp**의 **initialRoute**와 **routes** 속성을 추가하여 앱의 첫 페이지와 앱에 포함되는 화면을 구성합니다.

노트 목록 화면과 편집 화면을 루트에 추가하기

```dart
class NoteListPage extends StatefulWidget {
  static const routeName = '/';

  const NoteListPage({Key? key}) : super(key: key);

  @override
  State createState() => _NoteListPageState();
}
```

note_list_page.dart

노트 목록 화면과 편집 화면을 루트에 추가하기

```dart
class NoteEditPage extends StatefulWidget {
  static const routeName = '/edit';

  const NoteEditPage({Key? key}): super(key: key);

  @override
  State createState() => _NoteEditPageState();
}
```

note_edit_page.dart

노트 목록 화면과 편집 화면을 루트에 추가하기

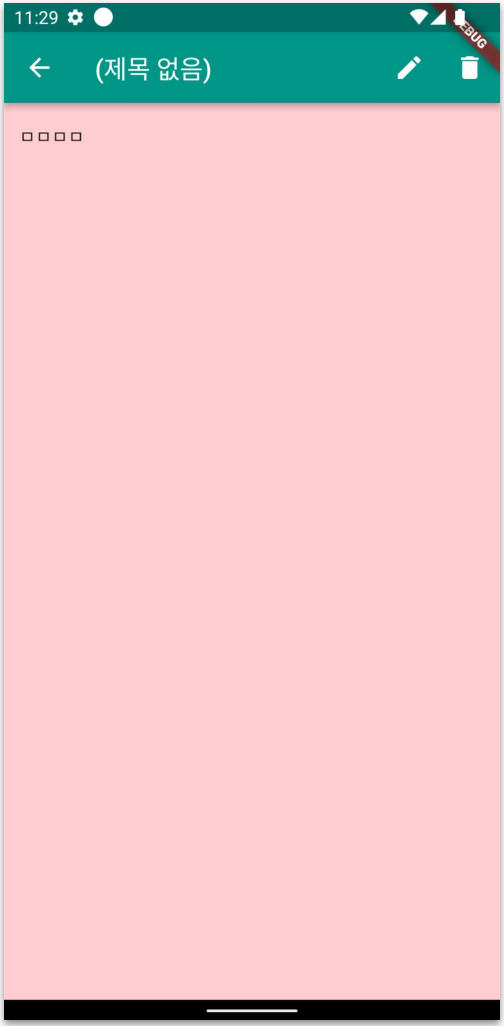
```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      initialRoute: NoteListPage.routeName,
      routes: {
        NoteListPage.routeName: (context) => const NoteListPage(),
        NoteEditPage.routeName: (context) => const NoteEditPage(),
      },
    );
  }
}
```

main.dart

노트 보기 화면 구현하기

노트 보기 화면 구현하기

1. **page** 폴더에 **note_view_page.dart** 파일을 추가합니다.
2. **note_view_page.dart** 파일에 **NoteViewPage** 클래스를 구현합니다.
3. **MyApp** 클래스 내 **MaterialApp**의 **routes** 속성에 **NoteViewPage**를 추가합니다.


11:29
(제목 없음)
□□□□

노트 본문 화면 작성하기

```dart
class NoteViewPage extends StatefulWidget {
  static const routeName = '/view';

  final int index;

  const NoteViewPage(this.index, {Key? key}) : super(key: key);

  @override
  State createState() => _NoteViewPageState();
}
```

note_view_page.dart

노트 보기 화면 작성하기

```dart
class _NoteViewPageState extends State<NoteViewPage> {
  @override
  Widget build(BuildContext context) {
    final note = noteService().getNote(widget.index);
    return Scaffold(
      appBar: AppBar(
        title: Text(note.title.isEmpty ? '(제목 없음)' : note.title),
        actions: [
          IconButton(
            icon: const Icon(Icons.edit),
            tooltip: '편집',
            onPressed: null,
          ),
          IconButton(
            icon: const Icon(Icons.delete),
            tooltip: '삭제',
            onPressed: null,
          ),
        ],
      ),
      body: Container(
        width: double.infinity,
        height: double.infinity,
        color: note.color,
        child: SingleChildScrollView(
          padding: const EdgeInsets.symmetric(
              horizontal: 12.0, vertical: 16.0),
          child: Text(note.body),
        ),
      ),
    );
  }
}
```

note_view_page.dart

노트 보기 화면 작성하기

```
class _NoteViewPageState extends State<NoteViewPage> {
  ...
  void _edit(int index) {
    Navigator.pushNamed(
      context,
      NoteEditPage.routeName,
      arguments: index,
    ).then((value) {
      setState(() {});
    });
  }
}
```

note_view_page.dart

노트 보기 화면 작성하기

```
class _NoteViewPageState extends State<NoteViewPage> {
  ...
  void _confirmDelete(int index) {
    showDialog(
      context: context,
      builder: (context) {
        return AlertDialog(
          title: const Text('노트 삭제'),
          content: const Text('노트를 삭제할까요?'),
          actions: [
            TextButton(
              child: const Text('아니오'),
              onPressed: () {
                Navigator.pop(context);
              },
            ),
            TextButton(
              child: const Text('예'),
              onPressed: () {
                noteService().deleteNote(index);
                Navigator.popUntil(context, (route) => route.isFirst);
              },
            ),
          ],
        );
      },
    );
  }
}
```

note_view_page.dart

노트 보기 화면 작성하기

```
class _NoteViewPageState extends State<NoteViewPage> {
  @override
  Widget build(BuildContext context) {
    ...
    return Scaffold(
      appBar: AppBar(
        title: Text(note.title.isEmpty ? '(제목 없음)' : note.title),
        actions: [
          IconButton(
            icon: const Icon(Icons.edit),
            tooltip: '편집',
            onPressed: () {
              _edit(widget.index);
            },
          ),
          IconButton(
            icon: const Icon(Icons.delete),
            tooltip: '삭제',
            onPressed: () {
              _confirmDelete(widget.index);
            },
          ),
        ],
      ),
    );
  }
}
```

note_view_page.dart

노트 보기 화면 작성하기

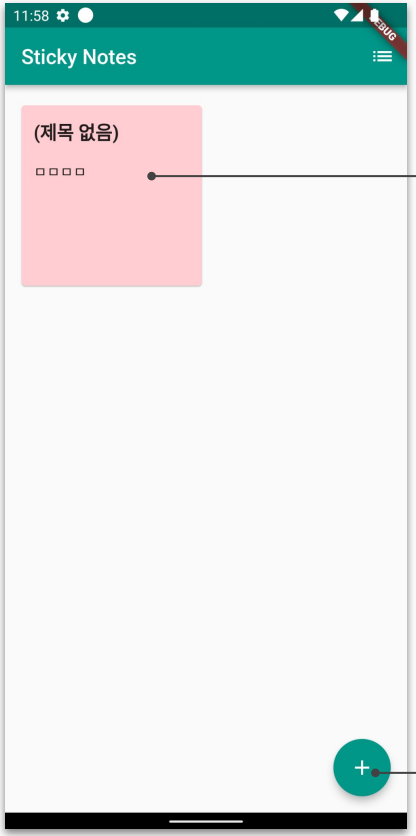
```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      routes: {
        ...
        NoteViewPage.routeName: (context) {
          final index = ModalRoute.of(context)!.settings.arguments as int;
          return NoteViewPage(index);
        },
      },
    );
  }
}
```

main.dart

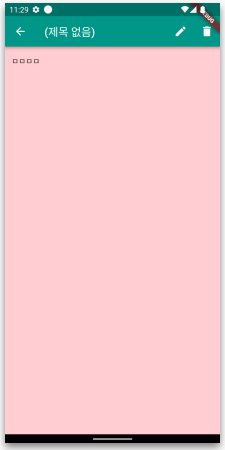
노트 목록, 보기, 편집 화면 연결하기

노트 목록, 보기, 편집 화면 연결하기

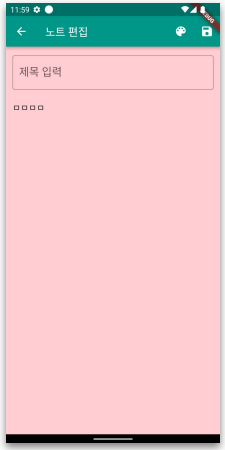
1. 노트 작성 화면이 새 노트를 작성하는 모드와 편집 모드를 지원하게끔 수정합니다.
2. 노트 목록 화면에 새 노트를 추가할 수 있는 버튼을 추가합니다.
3. 노트 목록에 표시된 노트를 눌러 노트 보기 화면으로 이동하게끔 코드를 수정합니다.



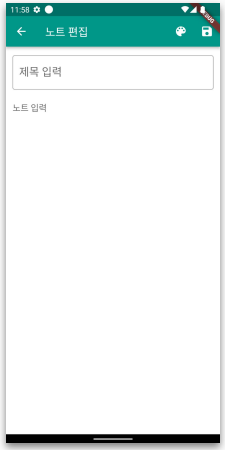
NoteListPage



NoteViewPage



NoteEditPage



노트 목록, 보기, 편집 화면 연결하기

```dart
class NoteEditPage extends StatefulWidget {
  static const routeName = '/edit';

  final int? index;

  const NoteEditPage(this.index, {Key? key}): super(key: key);

  @override
  State createState() => _NoteEditPageState();
}
```

note_edit_page.dart

노트 목록, 보기, 편집 화면 연결하기

```dart
class _NoteEditPageState extends State<NoteEditPage> {
  final titleController = TextEditingController();

  final bodyController = TextEditingController();

  Color color = Note.colorDefault;

  @override
  void initState() {
    super.initState();
    final noteIndex = widget.index;
    if (noteIndex != null) {
      final note = noteService().getNote(noteIndex);
      titleController.text = note.title;
      bodyController.text = note.body;
      color = note.color;
    }
  }
}
```

note_edit_page.dart

노트 목록, 보기, 편집 화면 연결하기

```
class _NoteEditPageState extends State<NoteEditPage> {

  void _saveNote() {
    if (bodyController.text.isNotEmpty) {
      final note = Note(
        bodyController.text,
        title: titleController.text,
        color: color,
      );

      final noteIndex = widget.index;
      if (noteIndex != null) {
        noteService().updateNote(noteIndex, note);
      } else {
        noteService().addNote(note);
      }

      Navigator.pop(context);
    } else { ... }
  }
}
```

note_edit_page.dart

노트 목록, 보기, 편집 화면 연결하기

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      routes: {
        ...
        NoteEditPage.routeName: (context) {
          final args = ModalRoute.of(context)!.settings.arguments;
          final index = args != null ? args as int : null;
          return NoteEditPage(index);
        },
      },
    );
  }
}
```

main.dart

노트 목록, 보기, 편집 화면 연결하기

```
class _NoteListPageState extends State<NoteListPage> {

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      ...
      floatingActionButton: FloatingActionButton(
        tooltip: '새 노트',
        onPressed: () {
          Navigator.pushNamed(context, NoteEditPage.routeName).then((value) {
            setState(() {});
          });
        },
        child: const Icon(Icons.add),
      ),
    );
  }
}
```

note_list_page.dart

노트 목록, 보기, 편집 화면 연결하기

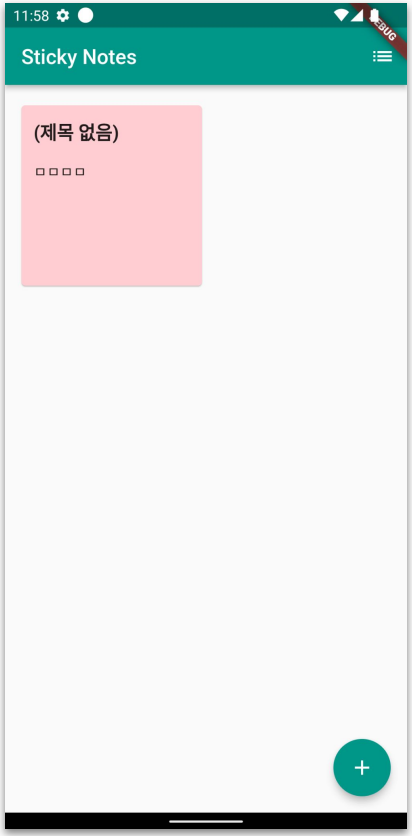
```
class _NoteListPageState extends State<NoteListPage> {
  ...

  Widget _buildCard(int index, Note note) {
    return InkWell(
      onTap: () {
        Navigator.pushNamed(
          context,
          NoteViewPage.routeName,
          arguments: index,
        ).then((value) {
          setState(() {});
        });
      },
      child: Card( ... ),
    );
  }
}
```

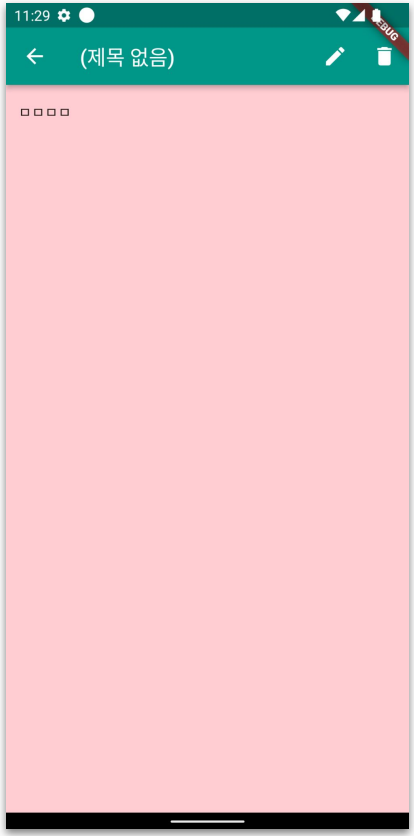
note_list_page.dart

노트 목록, 보기, 편집 화면 연결하기

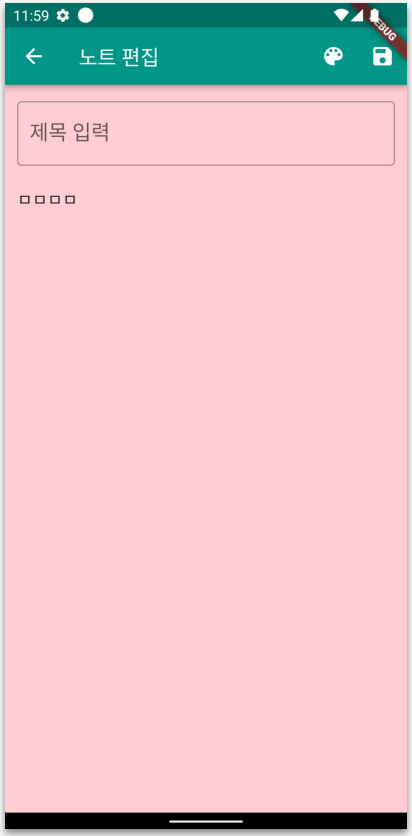
```
Widget _buildCards(List<Note> notes) {
  return _showAsGrid
      ? GridView.builder(
          padding:
              const EdgeInsets.symmetric(horizontal: 12.0, vertical: 16.0),
          itemBuilder: (context, index) => _buildCard(index, notes[index]),
          itemCount: notes.length,
          gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
            crossAxisCount: 2,
            childAspectRatio: 1,
          ),
        )
      : ListView.builder(
          padding:
              const EdgeInsets.symmetric(horizontal: 12.0, vertical: 16.0),
          itemCount: notes.length,
          itemBuilder: (context, index) {
            return SizedBox(
              height: 160,
              child: _buildCard(index, notes[index]),
            );
          },
        );
}
```

note_list_page.dart


11:58
Sticky Notes
(제목 없음)
□□□□


11:29
(제목 없음)
□□□□


1:04
(제목 없음)
□□□□
노트 삭제
노트를 삭제할까요?
아니오
예


11:59
노트 편집
제목 입력
□□□□

완성된 코드

https://github.com/kunny/skku-bootcamp-2022-summer/tree/main/sticky_notes/step4