

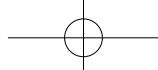


flutter project

CHAPTER 01

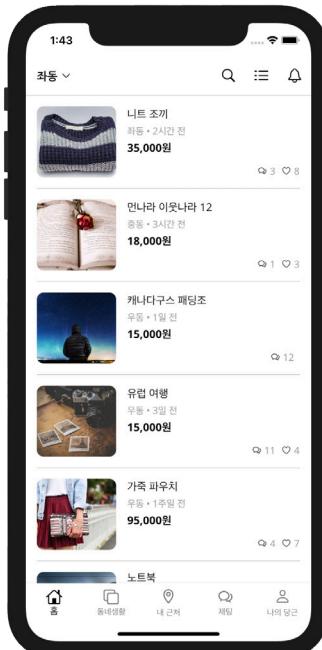
당근마켓 UI 만들어보기

당근마켓은 인기 있는 중고거래 모바일 서비스 앱입니다. 당근마켓 UI 만들어보기는 실제 많은 사람이 사용하고 있는 앱을 직접 개발해보면서 짧은 예제 학습만으로 부족했던 자신감과 실력을 가장 빨리 키울 수 있는 전략입니다. 특히 application 개발을 잘하기 위해서는 먼저 UI를 손쉽게 잘 만드는 것이 필승패턴 중 하나입니다. 하지만 처음부터 너무 어려운 UI를 도전해 보는 것은 부담이 될 수 있습니다. Flutter를 보다 효과적으로 정복하기 위해서 단계적으로 필수적인 개념들을 직접 구현해 보면서 자신감과 실력을 키울 수 있도록 합시다.

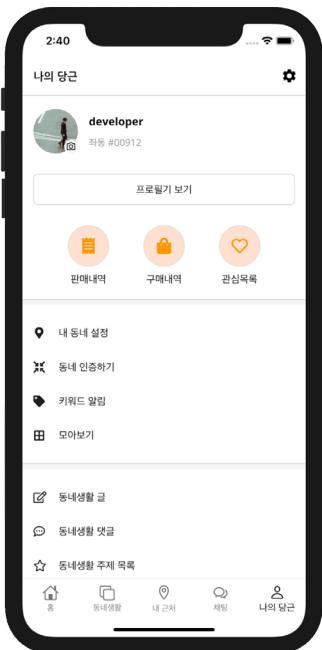


[앱 미리보기] 당근마켓 앱 구조 살펴보기

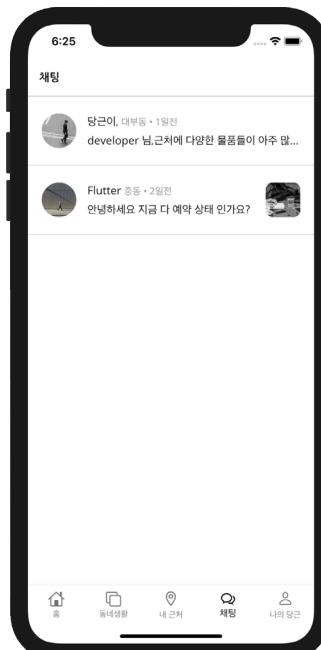
모든 소스 코드는 <https://github.com/flutter-coder/flutter-ui-book2>에 공개되어 있습니다.



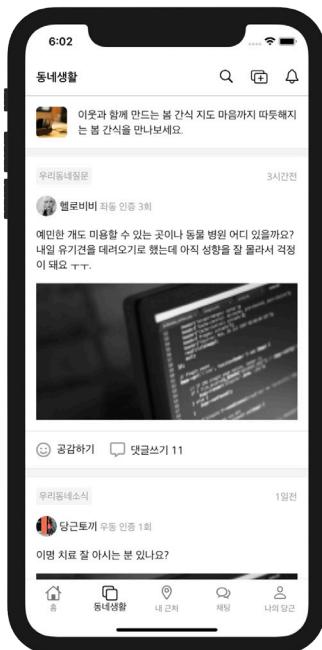
◆ 홈 완성화면



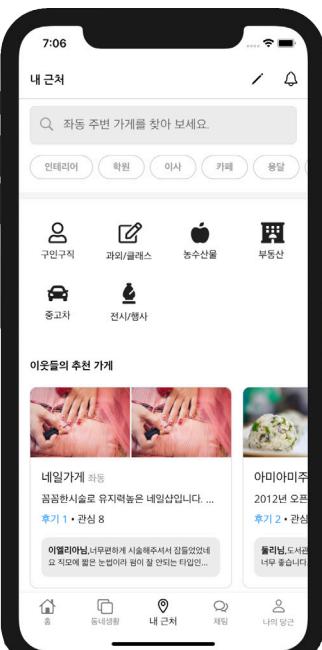
◆ 나의 당근 완성화면



◆ 채팅 완성화면

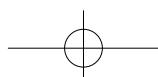


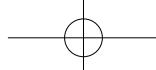
◆ 동네생활 완성화면



◆ 내 근처 완성화면

※ 안내 : 이 책은 플러터로 앱을 직접 만들어볼 수 있는 실전북으로 플러터 기본 문법 등 기초 내용은 설명되어 있지 않습니다. 플러터 기초 내용은 《플러터로 앱 만들기 [입문]》 책을 참조합니다.





01 _ 1 앱 뼈대 만들기

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_01에 공개되어 있습니다.

Flutter로 앱을 처음 만들면 어디서부터 작업을 시작해야 할지 막막할 수 있습니다. 그래서 앱을 만드는 작업 순서를 나열하고 그 작업 순서에 맞게 UI를 만들어보겠습니다. 우리가 해야 할 첫 번째 작업은 앱의 뼈대를 만드는 것입니다.

“ Android Studio와 Flutter를 설치하지 않았다면 아래 주소를 참고해주세요.

Flutter 설치 방법 : <https://blog.naver.com/getinthere/222324876638>

프로젝트를 만들어 주세요. 이름은 carrot_market_ui 로 하겠습니다.

작업 순서

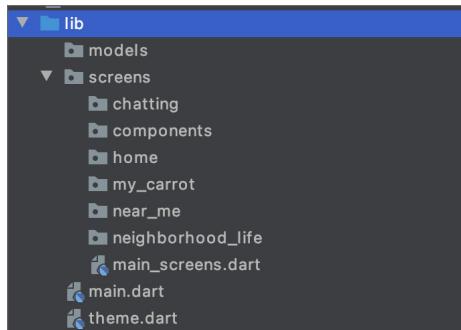
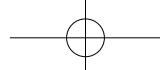
- ① 폴더 및 파일 만들기
- ② pubspec.yaml 파일 설정하기
- ③ main_screens.dart에 기본 코드 입력하기
- ④ 앱 테마 설정하기
- ⑤ main.dart 파일 완성하기

“ 각 장마다 작업 순서를 명시 했습니다. 그 이유는 복잡한 UI를 구성하기 위해서 전체적인 구조를 보고 학습할 수 있게 도움을 주고 싶었습니다. 작업 순서는 여러분이 교재를 학습하기 위한 알고리즘이라고 생각하면 됩니다.

폴더 및 파일 만들기

코드를 한곳에 모두 작성하는 것은 좋은 방법이 아닙니다. 가독성 및 재사용을 위해 위젯이나 코드들을 별도의 폴더와 파일로 나누는 것이 좋습니다. 아래와 같은 구조로 폴더와 파일들을 만들어봅시다.

```
lib
- models // 화면에 필요한 샘플 데이터와 데이터 모델 클래스 관리 폴더
- screens // 5개의 화면 파일이 모여 있는 폴더
  - chatting // 채팅 화면에 사용될 위젯 모음 폴더
  - components // 여러 화면에서 공통으로 사용될 위젯 모음 폴더
  - home // 홈 화면에 사용될 위젯 모음 폴더
  - my_carrot // 나의 당근 화면에 사용될 위젯 모음 폴더
  - near_me // 내 근처 화면에 사용될 위젯 모음 폴더
  - neighborhood_life // 동네생활 화면에 사용될 위젯 모음 폴더
  main_screens.dart // IndexStack, BottomNavigation 위젯을 가지는 파일
main.dart
theme.dart // 앱 테마 관리 파일
```



◆ 기본 폴더 구조

pubspec.yaml 파일 설정하기

pubspec.yaml 파일은 간단하게 프로젝트를 정의하는 파일입니다. 프로젝트의 이름, 버전, 개발 환경 등을 정의하고 앱 개발에 필요한 폰트, 아이콘 및 편리한 기능들을 가져와서 사용할 수 있게 도와주는 파일입니다. pubspec.yaml 파일을 열고 다음과 같이 작성해 봅시다.

```
carrot_market_ui/pubspec.yaml

name: carrot_market_ui    // 프로젝트의 이름을 명시합니다.
description: A new Flutter application.

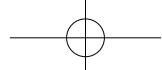
...
version: 1.0.0+1

environment:
  sdk: ">=2.12.0 <3.0.0"  // ❶ Dart 버전을 나타내고 Null safety를 사용할 수 있는 버전입니다.

dependencies:
  flutter:
    sdk: flutter

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  google_fonts: ^2.0.0    // ❷ 여러 가지 글꼴 스타일을 사용할 수 있게 합니다.
  font_awesome_flutter: ^9.0.0  // ❸ 앱에서 사용할 수 있는 아이콘을 제공합니다.
  intl: ^0.17.0   // ❹ DateFormat, NumberFormat 등 다양한 기능을 제공합니다.

dev_dependencies:
  flutter_test:
    sdk: flutter
```



- ① sdk 버전을 확인해 주세요. 2.12.0 보다 버전이 아래이면 위와 같이 수정해주세요.
- ② google_font 패키지를 추가합니다.
- ③ font_awesome_flutter 패키지를 추가합니다.
- ④ intl 패키지를 추가합니다.

main_screens.dart 기본 코드 작성

이 앱의 메인 화면이 될 파일입니다. 앞서 만들었던 lib / screens 폴더 아래 main_screens.dart 파일을 열고 기본 코드를 입력해 봅시다. Android Studio에서 "stf"를 입력하고 자동완성 기능을 이용하면 더욱 편리합니다.

```
lib / screens / main_screens.dart

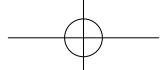
import 'package:flutter/material.dart';

class MainScreens extends StatefulWidget {
    @override
    _MainScreensState createState() => _MainScreensState();
}

class _MainScreensState extends State<MainScreens> {
    @override
    Widget build(BuildContext context) {
        return Container(
            child: Center(
                child: const Text('MainScreens'),
            ),
        );
    }
}
```

앱 테마 설정하기

반복적인 글꼴과 색상 등을 매번 지정하는 것은 번거로운 일이 될 수 있습니다. 우리는 theme.dart 파일에서 자주 사용되는 TextTheme과 appBarTheme를 정의하고 사용하는 방법을 배워 봅시다. 다음과 같이 코드를 입력해 주세요. 주석으로 표시된 부분은 입력하지 않아도 됩니다.



lib / theme.dart

```
import 'package:flutter/material.dart';
// ❶
import 'package:google_fonts/google_fonts.dart';
// ❷
TextTheme textTheme() {
    return TextTheme(
        headline1: GoogleFonts.openSans(fontSize: 18.0, color: Colors.black),
        headline2: GoogleFonts.openSans(
            fontSize: 16.0, color: Colors.black, fontWeight: FontWeight.bold),
        bodyText1: GoogleFonts.openSans(fontSize: 16.0, color: Colors.black),
        bodyText2: GoogleFonts.openSans(fontSize: 14.0, color: Colors.grey),
        subtitle1: GoogleFonts.openSans(fontSize: 15.0, color: Colors.black),
    );
}
// ❸
AppBarTheme appBarTheme() {
    return AppBarTheme(
        centerTitle: false,
        color: Colors.white,
        elevation: 0.0,
        textTheme: TextTheme(
            headline6: GoogleFonts.nanumGothic(
                fontSize: 16,
                fontWeight: FontWeight.bold,
                color: Colors.black,
            ),
        ),
    );
}
// ❹
ThemeData theme() {
    return ThemeData(
        scaffoldBackgroundColor: Colors.white,
        textTheme: textTheme(),
        appBarTheme: appBarTheme(),
    );
}
```

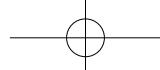
❶ pubspec.yaml 파일에 등록했던 폰트 관련 패키지를 import 합니다.

❷ Fluter에서 기본적으로 정의되어 있는textTheme의 속성들을 우리가 사용할 스타일로 재정의합니다. google_font 패키지에 폰트의 종류는 약 970개 중 openSans 폰트를 사용해 크기와 색상을 지정해 놓겠습니다.

❸ AppBar에 사용될 스타일을 미리 정의합니다.

❹ ThemeData 위젯은 우리가 방금 만든 TextTheme과 AppBarTheme를 정의할 수 있는 속성들을 가지고 있습니다. Scaffold 퀄더의 배경 색상을 지정하고 textTheme 속성과 appBarTheme 속성에 우리가 만든 전역 함수들을 지정해 줍시다.

이 프로젝트 앱의 기본적인 디자인 시스템은 Material Design type을 사용합니다. 다음에 나오는 표를 확인해 봅시다.



Example type scale

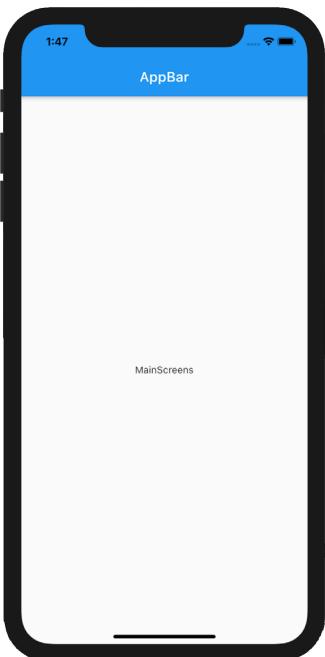
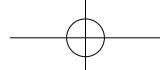
This example type scale uses the Roboto typeface for all headlines, subtitles, body, and captions, creating a cohesive typography experience. Hierarchy is communicated through differences in font weight (Light, Medium, Regular), size, letter spacing, and case.

Scale Category	Typeface	Weight	Size	Case	Letter spacing
H1	Roboto	Light	96	Sentence	-1.5
H2	Roboto	Light	60	Sentence	-0.5
H3	Roboto	Regular	48	Sentence	0
H4	Roboto	Regular	34	Sentence	0.25
H5	Roboto	Regular	24	Sentence	0
H6	Roboto	Medium	20	Sentence	0.15
Subtitle 1	Roboto	Regular	16	Sentence	0.15
Subtitle 2	Roboto	Medium	14	Sentence	0.1
Body 1	Roboto	Regular	16	Sentence	0.5
Body 2	Roboto	Regular	14	Sentence	0.25
BUTTON	Roboto	Medium	14	All caps	1.25
Caption	Roboto	Regular	12	Sentence	0.4
OVERLINE	Roboto	Regular	10	All caps	1.5

The Material Design type scale. (Letter spacing values are compatible with Sketch.)

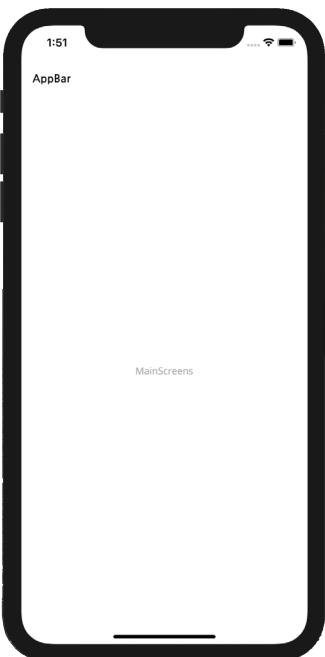
◆ Material 기본 글자 크기

Flutter에서 MaterialApp을 사용한다면 헤드라인, 자막, 본문 및 캡션에 Roboto 서체를 사용하게 됩니다. 예를 들어 Text 위젯의 style을 선언하지 않는다면 표에 나와 있는 기본 스타일이 적용됩니다.

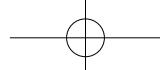


◆ Flutter 기본 스타일

- AppBar의 title 속성의 Text 위젯 style : Flutter 기본 스타일이 적용
- body 영역에서 Text 위젯 style : Flutter 기본 스타일이 적용



◆ 스타일 적용



- Appbar의 title 속성의 Text 위젯 style : theme.dart 파일에 정의한 스타일이 적용
- body 영역에서 Text 위젯 style : theme.dart 파일에 정의한 스타일이 적용

TIP

Text 위젯에 style 속성을 사용하지 않으면 위 사진의 Body 2의 스타일로 정의됩니다. Flutter에서는 Body 2를 bodyText2로 표기합니다.

Appbar의 Text 위젯은 기본적으로 H6 스타일로 표시됩니다. Flutter에서는 headline6 속성으로 표기됩니다.

우리가 만든 lib / theme.dart 파일을 Text 위젯에 개별적으로 style을 사용해야 된다면 다음과 같이 사용할 수 있습니다.

Text 위젯 style 개별 사용 예시

```
Text ('MainScreens', style :textTheme ().bodyText2 )  
// bodyText2: GoogleFonts.openSans(fontSize: 14.0, color: Colors.grey) 스타일 적용
```

```
Text ('MainScreens', style :textTheme ().headline1 )  
// headline1: GoogleFonts.openSans(fontSize: 18.0, color: Colors.black) 스타일 적용
```

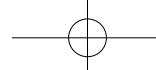
theme 개념이 낯설다면 그냥 넘어가세요. 교재를 계속 따라가다 보면 자연스럽게 개념을 잡을 수 있습니다.

main.dart 파일 완성하기

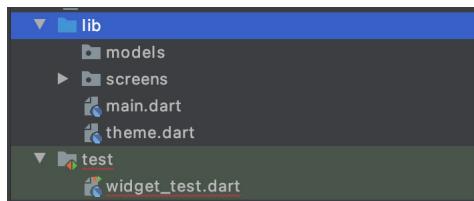
Flutter에서 새로운 프로젝트를 만들면 main.dart 파일에서 샘플 코드를 확인할 수 있습니다. 불필요한 코드를 다 제거하고 다음과 같이 코드를 작성해 봅시다.

lib / main.dart

```
import 'package:carrot_market_ui/screens/main_screens.dart';  
import 'package:carrot_market_ui/theme.dart';  
import 'package:flutter/material.dart';  
  
void main() {  
    // ❶  
    runApp(CarrotMarketUI());  
}  
  
class CarrotMarketUI extends StatelessWidget {  
    @override  
    Widget build(BuildContext context) {  
        // ❷  
        return MaterialApp(  
            title: 'carrot_market_ui',  
            debugShowCheckedModeBanner: false,  
            // ❸  
            home: MainScreens(),  
            // ❹  
            theme: theme(),  
        );  
    }  
}
```



- ❶ main() 함수는 앱이 시작될 때 코드의 진입점입니다. runApp() Flutter에게 앱의 최상위 위젯이 무엇인지 알려 줍니다.
- ❷ 일반적으로 앱을 만들 때 필요한 Material Design type의 여러 편의 위젯들을 제공합니다.
- ❸ MaterialApp의 home 속성은 애플리케이션이 정상적으로 시작될 때 처음 표시되는 경로(화면)를 Flutter에게 알립니다. lib / screens / main_screens.dart 파일에 정의한 MainScreens 위젯으로 지정하였습니다.
- ❹ theme 속성에 theme.dart 파일에서 작업한 전역 함수 theme() 함수를 연결합니다. MaterialApp에서 기본적으로 정의되어 있는 스타일에서 우리가 새롭게 정의한 스타일을 사용하게 됩니다.

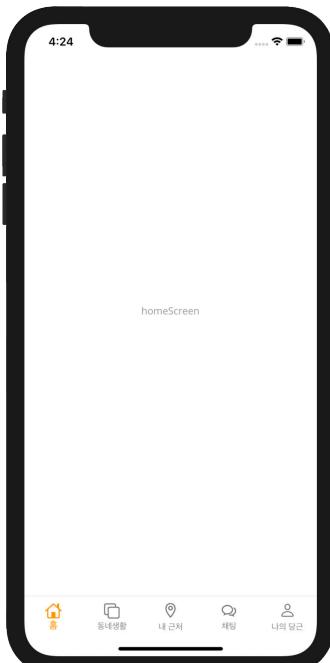


◆ widget_text 파일 삭제

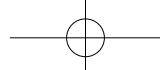
“main.dart 파일을 수정하면 test / widget.dart 파일에 빨간색 표시(오류)가 생깁니다. 해당 파일은 코드를 테스트할 때 사용하는 파일입니다. 우리 교재에서는 테스트 파일이 필요 하지 않기 때문에 파일을 삭제해 주세요.”

메인화면 만들기

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_02에 공개되어 있습니다.



◆ 메인 기본 화면



메인 화면에는 IndexedStack와 bottomNavigationBar를 함께 사용하는 방법과 동작 방식을 알아보겠습니다.

화면을 만들 때 매우 인기 있는 위젯 구성 방식이기 때문에 작업 순서에 맞춰 학습해 보겠습니다.

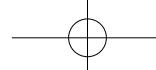
작업 순서

- ① MainScreens 위젯 기본 코드 작성하기
- ② primaryColor 설정하기
- ③ IndexedStack의 하위 위젯 만들기
- ④ MainScreens 위젯 완성하기

MainScreens 위젯 기본 코드 작성하기



먼저 이번 프로젝트 앱의 메인 화면의 동작 방식을 알아 둘 필요가 있습니다. 그중 가장 핵심이 되는 위젯은 IndexedStack 위젯과 BottomNavigationBar 위젯입니다. 이 두 위젯을 활용해서 사용자가 하단 아이콘 버튼을 눌렀을 때 위젯들의 상태가 변경되는 화면을 만들 수 있습니다. 앱 뼈대 만들기에서 작업했던 main_screens.dart 파일을 열고 다음과 같이 코드를 작성해 봅시다.



lib / screens / main_screens.dart

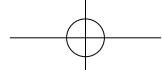
```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';

class MainScreens extends StatefulWidget {
    @override
    _MainScreensState createState() => _MainScreensState();
}

class _MainScreensState extends State<MainScreens> {
    // ❶
    int _selectedIndex = 0;

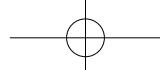
    @override
    Widget build(BuildContext context) {
        // ❷
        return Scaffold(
            // ❸
            body: IndexedStack(
                index: _selectedIndex,
                children: [
                    Container( // index 0
                        color: Colors.orange[100],
                        child: Center(
                            child: Text(
                                'IndexedStack 1',
                                style: TextStyle(fontSize: 20, color: Colors.black),
                            ),
                        ),
                    ),
                    Container( // index 1
                        color: Colors.redAccent[100],
                        child: Center(
                            child: Text(
                                'IndexedStack 2',
                                style: TextStyle(fontSize: 20, color: Colors.black),
                            ),
                        ),
                    ),
                ],
            ),
        );
    }
    // ❹
    bottomNavigationBar: BottomNavigationBar(
        items: [

```



```
BottomNavigationBarItem(
    label: '홈',
    icon: Icon(
        CupertinoIcons.home,
    ),
),
BottomNavigationBarItem(
    label: '채팅',
    icon: Icon(
        CupertinoIcons.chat_bubble,
    ),
),
],
// ❸
onTap: (index) {
    setState(
        () {
            _selectedIndex = index;
        },
    );
},
// ❹
currentIndex: _selectedIndex,
);
);
}
}
```

- ❶ 사용자가 하단 아이콘 버튼을 눌렀을 때 위젯은 index 값을 저장하는 변수입니다.
- ❷ Scaffold 위젯은 기본적인 시각적 레이아웃 구조를 간편하게 만들 수 있게 도와주는 위젯입니다. AppBar, BottomSheet, BottomNavigationBar, Drawer, Body, FloatingActionButton, Snackbar 등을 편리하게 사용할 수 있게 합니다.
- ❸ IndexedStack은 한 번에 하위 항목 하나만을 보여주는 스택 위젯입니다. index 속성을 사용하여 현재 보여줘야 할 위젯을 선택합니다.
- ❹ 일반적으로 세 개에서 다섯 개 사이의 앱의 최상위 화면을 빠르게 탐색할 수 있게 하는 하단에 표시되는 material 위젯입니다.
- ❺ 사용자가 하단 아이콘 버튼을 눌렀을 때 index 값을 반환하는 메서드입니다. 우리는 setState 함수를 사용해서 멤버 변수 _selectedIndex 변수에 값을 변경할 수 있습니다.
- ❻ currentIndex 속성은 현재 선택된 BottomNavigationBarItem 항목에 대한 인덱스입니다. 이 속성을 설정해야 BottomNavigationBarItem의 활성화된 상태를 표시합니다.



primaryColor 설정하기



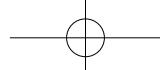
primaryColor는 앱의 브랜드 색상이라 할 수 있습니다. 우리가 만든 프로젝트 앱의 브랜드 색상은 orange로 사용하기 때문에 앱 뼈대 만들기에서 작업했던 theme.dart 파일을 열고 코드를 추가해봅시다.

lib / theme.dart

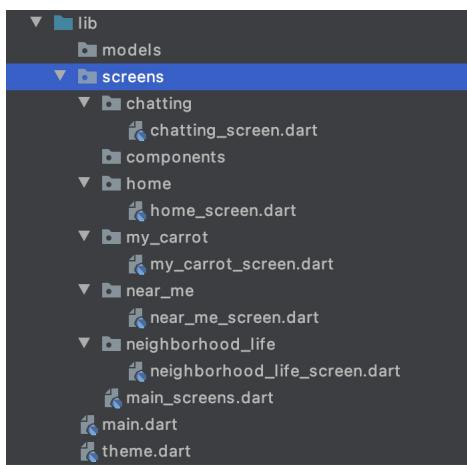
... 생략

```
ThemeData theme() {
    return ThemeData(
        scaffoldBackgroundColor: Colors.white,
        textTheme: textTheme(),
        appBarTheme: appTheme(),
        // ❶
        primaryColor: Colors.orange,
    );
}
```

❶ lib / theme.dart 파일에서 우리가 만든 전역 함수 theme() 함수에 ThemeData 위젯의 primaryColor 색상을 추가합니다.



IndexedStack의 하위 위젯 만들기



◆ 하위 폴더 생성

메인 화면을 구성하기 위해 IndexedStack 위젯의 하위 항목으로 구성될 다섯 개의 화면 위젯이 필요합니다. 앱 뼈대 만들기에서 작업했던 screens 폴더 내부에 파일을 만들고 기본 코드를 작성해 보겠습니다.

① chatting_screen.dart 파일 만들기

하단 BottomNavigationBarItem의 채팅 아이콘 버튼을 눌렀을 때 사용하게 될 위젯입니다.

lib / screens / chatting 폴더에 chatting_screen.dart 파일을 만들어 주세요.

```
lib / screens / chatting / chatting_screen.dart

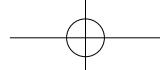
import 'package:flutter/material.dart';

class ChattingScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Center(
            child: Text('chattingScreen'),
        );
    }
}
```

② home_screen.dart 파일 만들기

하단 BottomNavigationBarItem의 홈 아이콘 버튼을 눌렀을 때 사용하게 될 위젯입니다.

lib / screens / home 폴더에 home_screen.dart 파일을 만들어 주세요.



lib / screens / home / home_screen.dart

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

class HomeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Center(
            child: Text('homeScreen'),
        );
    }
}
```

③ my_carrot_screen.dart 파일 만들기

하단 BottomNavigationBarItem의 나의당근 아이콘 버튼을 눌렀을 때 사용하게 될 위젯입니다.

lib / screens / my_carrot 폴더에 my_carrot_screen.dart 파일을 만들어 주세요.

lib / screens / my_carrot / my_carrot_screen.dart

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

class MyCarrotScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Center(
            child: Text('myCarrotScreen'),
        );
    }
}
```

④ near_me_screen.dart 파일 만들기

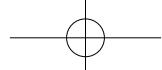
하단 BottomNavigationBarItem의 내 근처 아이콘 버튼을 눌렀을 때 사용하게 될 위젯입니다. lib

/ screens / near_me 폴더에 near_me_screen.dart 파일을 만들어 주세요.

lib / screens / near_me / near_me_screen.dart

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

class NearMeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Center(
            child: Text('nearMeScreen'),
        );
    }
}
```



⑤ neighborhood_life_screen.dart 파일 만들기

하단 BottomNavigationBarItem의 동네생활 아이콘 버튼을 눌렀을 때 사용하게 될 위젯입니다.

lib / screens / neighborhood_life 폴더에 neighborhood_life_screen.dart 파일을 만들어 주세요.

lib / screens / neighborhood_life / neighborhood_life_screen.dart

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

class NeighborhoodLifeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Center(
            child: Text('neighborhoodLifeScreen'),
        );
    }
}
```

MainScreens 위젯 완성하기

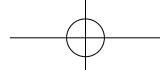
main_screens.dart 파일을 완성해 봅시다. 다음과 같이 코드를 수정해주세요.

lib / screens / main_screens.dart

```
class MainScreens extends StatefulWidget {
    @override
    _MainScreensState createState() => _MainScreensState();
}

class _MainScreensState extends State<MainScreens> {
    int _selectedIndex = 0;

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            body: IndexedStack(
                index: _selectedIndex,
                // ❶
                children: [
                    HomeScreen(),
                    NeighborhoodLifeScreen(),
                    NearMeScreen(),
                    ChattingScreen(),
                    MyCarrotScreen()
                ],
            ),
            bottomNavigationBar: BottomNavigationBar(
                // ❷
            )
        );
    }
}
```



```
backgroundColor: Colors.white,  
// ③  
type: BottomNavigationBarType.fixed,  
// ④  
currentIndex: _selectedIndex,  
onTap: (index) {  
    setState(() {  
        _selectedIndex = index;  
    });  
},  
// ⑤  
items: [  
    const BottomNavigationBarItem(  
        label: '홈', icon: Icon(CupertinoIcons.home)),  
    const BottomNavigationBarItem(  
        label: '동네생활', icon: Icon(CupertinoIcons.square_on_square)),  
    const BottomNavigationBarItem(  
        label: '내 근처', icon: Icon(CupertinoIcons.placemark)),  
    const BottomNavigationBarItem(  
        label: '채팅', icon: Icon(CupertinoIcons.chat_bubble_2)),  
    const BottomNavigationBarItem(  
        label: '나의 당근', icon: Icon(CupertinoIcons.person)),  
],  
),  
);  
}  
}  
}
```

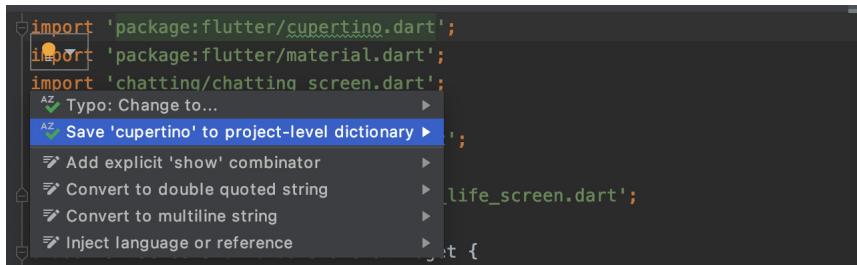
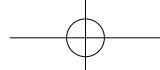
- ❶ IndexedStack 위젯의 하위 항목에 우리가 만든 위젯을 넣어 주세요. import 구문은 자동완성 기능으로 만들어 주면 편리합니다.
- ❷ bottomNavigationBar 배경색을 white로 지정했습니다.
- ❸ BottomNavigationBarType.fixed는 3개 이상의 아이템을 표시할 때 설정을 해줍니다.
- ❹ BottomNavigationBarItem 앞에 const 선언하는 이유는 Flutter에게 변하지 않는 위젯임을 알려 줍니다. 그럼 Flutter는 컴파일 시점에서 미리 위젯을 만들어 두기 때문에 앱이 실행되는 런타임 시점에서 위젯을 만들어 줄 필요가 없어 성능에 있어 도움이 됩니다.

```
import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'chatting/chatting_screen.dart';  
import 'home/home_screen.dart';  
import 'my_carrot/my_carrot_screen.dart';  
import 'near_me/near_me_screen.dart';  
import 'neighborhood_life/neighborhood_life_screen.dart';
```

◆ import 구문 확인

TIP MainScreens 위젯에서 import 구문 확인하기

다른 파일들에 있는 코드들을 현재 파일에서 사용하고 싶다면 import 구문을 작성하고 파일의 경로를 작성해 주면 됩니다. 하지만 매번 작성하게 된다면 번거로운 작업이 될 수 있습니다. IDE(개발 환경 도구)에 자동완성 기능을 이용해 import 구문을 완성하면 편리합니다.



◆ 경고 문구 없애기

TIP cupertino 경고 문구 없애기

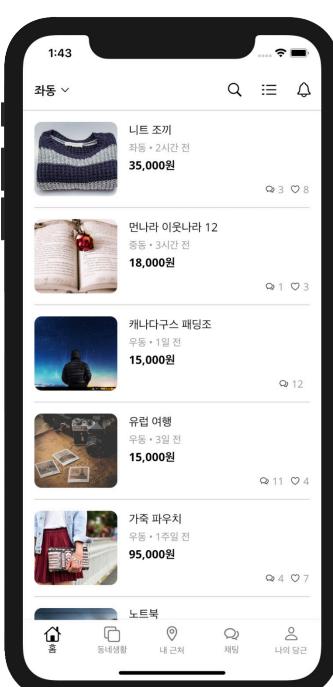
위 사진 cupertino.dart 문구 아래 경고 표시가 확인됩니다. 커서를 문구 가운데에 놓으면 왼쪽 상단에 표시되는 전구 모양을 눌러 주세요.

Save 'cupertino' to perfect-level dictionary를 선택하면 'cupertino'에 대한 경고 표시가 사라지게 됩니다.

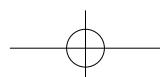
01 _ 3 홈 화면 만들기

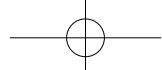
해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_03에 공개되어 있습니다.

이번 장을 완료하면 아래와 같은 화면을 만들 수 있습니다.



◆ 훌 완성화면

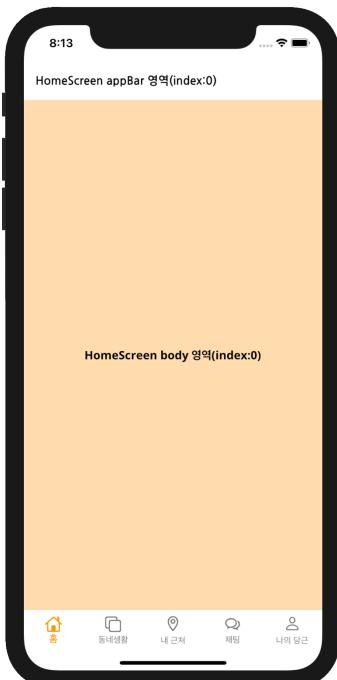




스크롤 가능한 콘텐츠를 만드는 것은 UI 개발에 필수적인 부분입니다. 이 장에서는 스크롤 가능한 ListView.separated 위젯을 이용해서 스크롤 가능한 콘텐츠를 만들어봅시다.

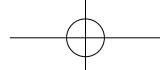
작업 순서

- ❶ HomeScreen 위젯 기본 코드 입력하기
- ❷ AppBar 만들기
- ❸ 화면에 사용할 샘플 데이터 만들기
- ❹ 독립된 파일로 위젯 만들기
- ❺ HomeScreen 위젯 완성하기
- ❻ HomeScreen 위젯 기본 코드 입력하기



◆ 홈 기본 코드 입력

HomeScreen 위젯은 부모 위젯 IndexedStack(메인화면) 위젯의 index 0번째 하위 항목입니다. 앱 뼈대 만들기에서 작업했던 lib / screens / home 폴더에 home_screen.dart 파일을 열고 코드를 입력해 봅시다.



lib / screens / home / home_screen.dart

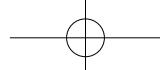
```
class HomeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // ❶
    return Scaffold(
      appBar: AppBar(
        title: Text('HomeScreen appBar 영역(index:0)'),
      ),
    // ❷
    body: Container(
      color: Colors.orange[100],
      child: Center(
        // ❸
        child: Text(
          'HomeScreen body 영역(index:0)',
          style: textTheme().headline2,
        ),
      ),
    );
  }
}
```

- ❶ appBar 와 body 영역을 나눌 수 있게 Scaffold 위젯을 사용합니다.
- ❷ 위젯의 영역을 표시하기 위해 Container 위젯의 color 속성을 사용합니다. 잠시 후에 스크롤이 가능한 위젯으로 교체 할 예정입니다.
- ❸ theme.dart 파일에서 만들었던 textTheme 메서드를 사용해서 Text 스타일을 지정했습니다.

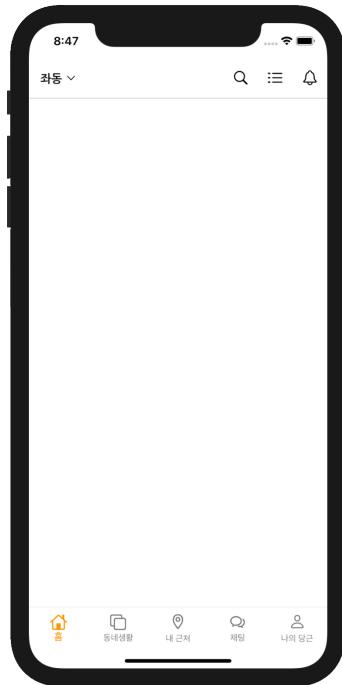
TIP

```
import 'package:carrot_market_ui/theme.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
```

import 구문도 잊지 말고 완성해 주세요. 가능한 지면 활용과 편의상 import 구문을 표시하지 않습니다. 자동완성 기능을 사용해서 완성해 봅시다.



AppBar 만들기

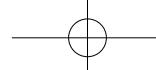


◆ AppBar 위젯

이 프로젝트 앱에서 AppBar는 화면마다 각각 따로 만들어서 학습하겠습니다. Flutter의 AppBar 위젯을 만드는 방법에 익숙해져 봅시다.

```
lib / screens / home / home_screen.dart

class HomeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        // ❶
        title: Row(
          children: [
            const Text('좌동'),
            const SizedBox(width: 4.0),
            const Icon(
              CupertinoIcons.chevron_down,
              size: 15.0,
            ),
          ],
        ),
        // ❷
      ),
    );
  }
}
```



```

actions: [
    IconButton(icon: const Icon(CupertinoIcons.search), onPressed: () {}),
    IconButton(
        icon: const Icon(CupertinoIcons.list_dash), onPressed: () {}),
    IconButton(icon: const Icon(CupertinoIcons.bell), onPressed: () {})
],
// ③
bottom: const PreferredSize(
    preferredSize: Size.fromHeight(0.5),
    child: Divider(thickness: 0.5, height: 0.5, color: Colors.grey),
),
),
// ④
body: Container(),
);
}
}

```

① title 속성에 Text 위젯만이 아닌 여러 위젯을 활용해서 만들 수 있습니다.

② actions 속성은 title 위젯 다음 행에 표시할 위젯 목록입니다.

③ AppBar 하단에 라인을 표시하기 위해 bottom 속성을 사용합니다.

④ body 영역에 기본 코드로 입력했던 부분을 제거합니다.

TIP AppBar에 자동으로 스타일이 적용된 이유는?

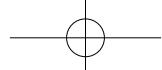
lib / main.dart

```

... 생략
class CarrotMarketUI extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: 'carrot_market_ui',
            debugShowCheckedModeBanner: false,
            home: MainScreens(),
            // ①
            theme: theme(),
        );
    }
}

```

① MaterialApp의 theme 속성에 우리가 만든 lib / theme.dart 파일의 theme() 메서드를 사용했기 때문입니다.



화면에 사용할 샘플 데이터 만들기

HomeScreen 위젯을 완성하기 위해서 우리는 Scaffold의 body에 사용자가 스크롤 할 수 있는 리스트가 필요합니다. 그러기 위해 ListView 위젯에 사용할 데이터를 만드는 방법부터 배워 봅시다.

① Product 모델 클래스 만들기

모든 기능을 갖춘 앱에서는 웹서버에서 JSON 기반 API를 호출하여 데이터를 로드 합니다. 그리고 JSON 기반의 문자열 데이터를 해당 언어의 오브젝트(클래스)로 변환하는데 이때 필요한 클래스를 모델 클래스라고 합니다.

“ 데이터를 Object로 변환하는 과정을 JSON Parsing이라고 합니다. Parsing은 구문을 분석한다는 뜻입니다. 즉 JSON 데이터를 분석하여 Dart Object로 변환합니다. Dart Object로 변환하는 이유는 Dart 프로그래밍을 할 때 활용하기 편하기 때문입니다. JSON은 단순 하나의 문자열이기 때문에 다루기가 불편하고 어렵습니다.

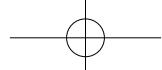
우리는 통신을 할 웹서버가 없기 때문에 임시 데이터를 만들어 학습합니다. Product 클래스를 만들어봅시다. lib / models 폴더에 product.dart 파일을 만들고 다음과 같이 코드를 입력해 주세요.

```
lib / models / product.dart

class Product {
    String title;
    String author;
    String address;
    String urlToImage;
    String publishedAt;
    String price;
    int heartCount;
    int commentsCount;

    Product({
        required this.title,
        required this.author,
        required this.address,
        required this.urlToImage,
        required this.publishedAt,
        required this.price,
        required this.heartCount,
        required this.commentsCount,
    });
}

// 샘플 데이터
....
```

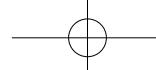


② Product 모델 클래스 샘플 데이터 만들기

스크롤을 사용할 수 있을 만큼에 데이터가 필요합니다. 코드 양이 많다면 github에서 받은 소스코드에서 carrot_market_ui_03에 product.dart 파일의 코드를 복사해서 사용할 수 있습니다.

lib / models / product.dart

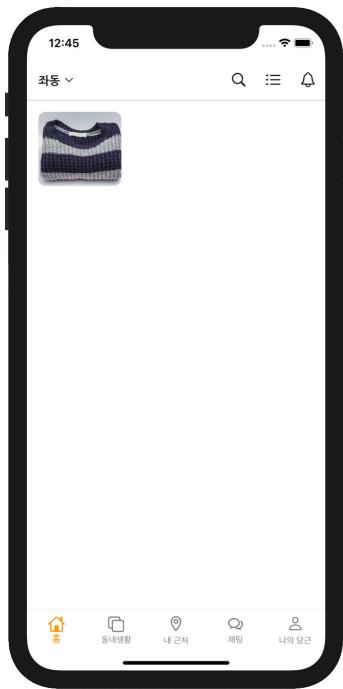
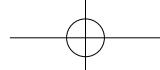
```
// 샘플 데이터
List<Product> productList = [
    Product(
        title: '니트 조끼',
        author: 'author_1',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_7.jpg?raw=true',
        publishedAt: '2시간 전',
        heartCount: 8,
        price: '35000',
        address: '좌동',
        commentsCount: 3),
    Product(
        title: '먼나라 이웃나라 12',
        author: 'author_2',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_6.jpg?raw=true',
        publishedAt: '3시간 전',
        heartCount: 3,
        address: '중동',
        price: '18000',
        commentsCount: 1),
    Product(
        title: '캐나다구스 패딩조',
        author: 'author_3',
        address: '우동',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_5.jpg?raw=true',
        publishedAt: '1일 전',
        heartCount: 0,
        price: '15000',
        commentsCount: 12,
    ),
    Product(
        title: '유럽 여행',
        author: 'author_4',
        address: '우동',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_4.jpg?raw=true',
        publishedAt: '3일 전',
        heartCount: 4,
```



```
        price: '15000',
        commentsCount: 11,
    ),
    Product(
        title: '가죽 파우치',
        author: 'author_5',
        address: '우동',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_3.jpg?raw=true',
        publishedAt: '1주일 전',
        heartCount: 7,
        price: '95000',
        commentsCount: 4,
    ),
    Product(
        title: '노트북',
        author: 'author_6',
        address: '좌동',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_2.jpg?raw=true',
        publishedAt: '5일 전',
        heartCount: 4,
        price: '115000',
        commentsCount: 0,
    ),
    Product(
        title: '미개봉 아이패드',
        author: 'author_7',
        address: '좌동',
        urlToImage:
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_product_1.jpg?raw=true',
        publishedAt: '5일 전',
        heartCount: 8,
        price: '85000',
        commentsCount: 3,
    ),
);
];
```

독립된 파일로 위젯 만들기

필요에 따라서 위젯을 별도에 파일로 만들어 둘 수 있습니다. 성능에 있어 고려할 사항도 있지만 유지 보수 및 가독성이 좋은 점이 많습니다. 우리가 만든 위젯을 별도에 파일로 만들어서 사용하는 연습을 해 봅시다.



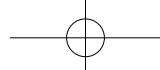
◆ ProductItem 위젯

① ProductItem 위젯 만들기

먼저 lib / screens / home 폴더에 components 폴더를 만들어 주세요. 그리고 components 폴더에 product_item.dart 파일을 생성하고 다음과 같이 코드를 작성해 주세요.

lib / screens / home / components / product_item.dart

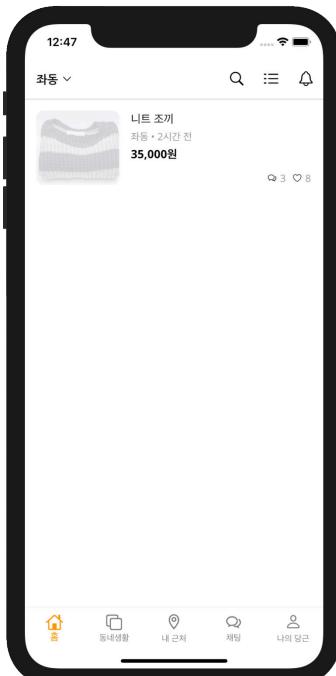
```
class ProductItem extends StatelessWidget {  
    // ❶  
    final Product product;  
    // ❷  
    ProductItem({required this.product});  
  
    @override  
    Widget build(BuildContext context) {  
        // ❸  
        return Container(  
            height: 135.0,  
            padding: const EdgeInsets.all(16.0),  
            child: Row(  
                children: [  
                    // ❹  
                    ClipRRect(  
                        borderRadius: BorderRadius.circular(10.0),  
                        child: Image.network(product.imageUrl),  
                    ),  
                    const SizedBox(width: 10.0),  
                    Column(  
                        mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
                        crossAxisAlignment: CrossAxisAlignment.start,  
                        children: [
```



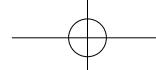
```
        child: Image.network(
            product.urlToImage,
            width: 115,
            height: 115,
            fit: BoxFit.cover,
        ),
    ),
    const SizedBox(width: 16.0),
    // ❶
    // ProductDetail(product: product)
],
),
);
}
}
```

- ❶ 모델 클래스 Product 타입의 변수를 선언합니다.
- ❷ 생성자입니다. 이 위젯을 생성할 때 Product 타입의 객체를 사용한다고 명시합니다.
- ❸ ProductItem 위젯의 전체 세로 크기를 Container 위젯의 height 속성을 사용해서 명시합니다.
- ❹ 이미지의 모서리에 곡선 효과를 주기 위해 사용합니다.
- ❺ Row 위젯의 두 번째 항목이 될 위젯도 별도의 파일로 만들어서 사용합니다. 다음 차례에서 ProductDetail 위젯을 완성하고 주석을 풀어 주세요.

❷ ProductDetail 위젯 만들기



◆ ProductDetail 위젯



lib / screens / home / components 폴더에 product_detail.dart 파일을 생성하고 다음과 같이 코드를 입력해 주세요.

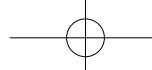
```
lib / screens / home / components / product_detail.dart

class ProductDetail extends StatelessWidget {
    final Product product;

    const ProductDetail({required this.product});

    @override
    Widget build(BuildContext context) {
        // ❶
        return Expanded(
            child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                    Text(product.title, style: textTheme().bodyText1),
                    const SizedBox(height: 4.0),
                    Text('${product.address} · ${product.publishedAt}'),
                    const SizedBox(height: 4.0),
                    // ❷
                    Text(
                        '${numberFormat(product.price)}원',
                        style: textTheme().headline2,
                    ),
                    const Spacer(),
                    Row(
                        mainAxisAlignment: MainAxisAlignment.end,
                        children: [
                            // ❸
                            Visibility(
                                visible: product.commentsCount > 0,
                                child: _buildIcons(
                                    product.commentsCount,
                                    CupertinoIcons.chat_bubble_2,
                                ),
                            ),
                            const SizedBox(width: 8.0),
                            Visibility(
                                visible: product.heartCount > 0,
                                child: _buildIcons(
                                    product.heartCount,
                                    CupertinoIcons.heart,
                                ),
                            ),
                        ],
                    ),
                ],
            );
    }
}
```

29 플로터로 앱 만들기(실전)



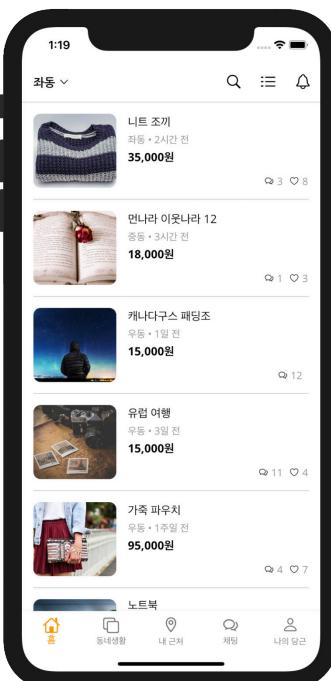
```

    }
    // ④
    String numberFormat(String price) {
        final formatter = NumberFormat('#,###');
        return formatter.format(int.parse(price));
    }
    // ⑤
    Widget _buildIcons(int count, IconData iconData) {
        return Row(
            children: [
                Icon(iconData, size: 14.0),
                const SizedBox(width: 4.0),
                Text('$count'),
            ],
        );
    }
}

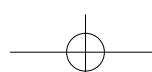
```

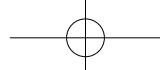
- ❶ Expanded의 부모 위젯은 ProductItem의 Row 위젯입니다. 가로 방향으로 최대 확장 가능한 넓이만큼 늘어납니다.
- ❷ numberFormat 함수를 사용해서 숫자에 콤마(,)를 추가하였습니다.
- ❸ 데이터 상태에 따라 위젯을 감추거나 보여줘야 할 때 활용할 수 있는 위젯입니다.
- ❹ pubspec.yaml 파일에서 추가한 int: ^0.17.0 라이브러리를 사용해서 NumberFormat 기능을 만들어 줍니다.
- ❺ 반복적인 작업을 메서드화 시켰습니다.

HomeScreen 위젯 완성하기



◆ 헴 body

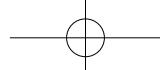




lib / screens / home / home_screen.dart

```
class HomeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: Row(
                    children: [
                        const Text('좌동'),
                        const SizedBox(width: 4.0),
                        const Icon(
                            CupertinoIcons chevron_down,
                            size: 15.0,
                        ),
                    ],
                ),
            ),
            actions: [
                IconButton(icon: const Icon(CupertinoIcons.search), onPressed: () {}),
                IconButton(
                    icon: const Icon(CupertinoIcons.list_dash), onPressed: () {}),
                IconButton(icon: const Icon(CupertinoIcons.bell), onPressed: () {})
            ],
            bottom: const PreferredSize(
                preferredSize: Size.fromHeight(0.5),
                child: Divider(thickness: 0.5, height: 0.5, color: Colors.grey),
            ),
        );
        // ①
        body: ListView.separated(
            separatorBuilder: (context, index) => const Divider(
                height: 0,
                indent: 16,
                endIndent: 16,
                color: Colors.grey,
            ),
            itemBuilder: (context, index) {
                // ②
                return ProductItem(
                    product: productList[index],
                );
            },
            // ③
            itemCount: productList.length,
        ), // end of ListView.separated
    );
}
}
```

- ① 하단에 구분선이 있는 리스트 위젯을 만들 때 사용할 수 있는 위젯입니다. Divider 위젯의 indent, endIndent 속성을 이용해서 선의 시작과 끝을 설정할 수 있습니다.
- ② 독립된 파일로 분리한 ProductItem 위젯을 사용합니다. 오류 표시가 나온다면 product_item.dart 파일을 import 해주세요.
- ③ 리스트에 표시할 데이터의 개수를 알려 줍니다.



TIP

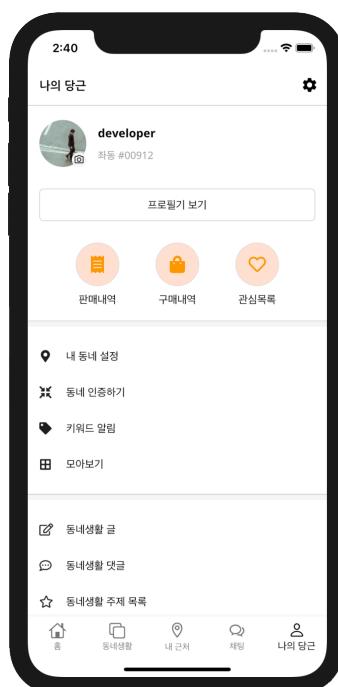
lib / screens / main_screens.dart 파일의 BottomNavigationBar 위젯에 속성을 추가해 봅시다.

```
bottomNavigationBar: BottomNavigationBar(  
    backgroundColor: Colors.white,  
    type: BottomNavigationBarType.fixed,  
    currentIndex: _selectedIndex,  
    // ①  
    selectedItemColor: Colors.black,  
    // ②  
    unselectedItemColor: Colors.black54,  
    onTap: (index) {  
        setState(() {  
            _selectedIndex = index;  
        });  
    },  
    items: [  
        ... 생략  
    ],  
,
```

① 활성화된 아이콘의 색상을 정의합니다.

② 비활성화된 아이콘의 색상을 정의합니다.

01_14 나의당근 화면 만들기

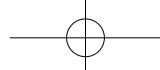


◆ 나의 당근

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_04에 공개되어 있습니다.

이번 장을 완료하면 아래와 같은 화면을 만들 수 있습니다.

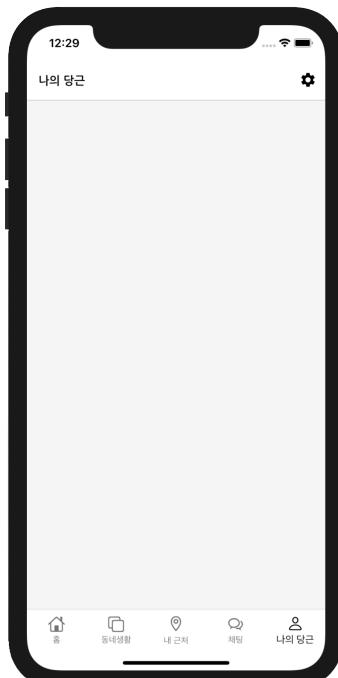
앱 프로파일 화면은 대부분의 앱에서 거의 필수적인 요소입니다. 이 장에서는 Card 위젯을 사용해 콘텐츠를 만들면서 학습해 보겠습니다.



작업 순서

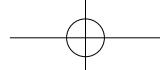
- ① MyCarrotScreen 위젯 기본 코드 입력하기
- ② 나의당근 화면 Header 위젯 만들기
- ③ 모델 클래스 만들기
- ④ 하단 Card 메뉴 위젯 만들기
- ⑤ MyCarrotScreen 위젯 완성하기

나의 당근 화면 기본 코드 입력하기



◆ 나의 당근 기본 화면

MyCarrotScreen 위젯은 부모 위젯 IndexedStack(메인화면) 위젯의 index 4번째 하위 항목입니다. IndexedStack 하위 항목을 만드는 순서는 학습에 있어 효율적인 차례로 진행하겠습니다. 앱 뼈대 만들기에서 작업했던 lib / screens / my_carrot 폴더에 my_carrot_screen.dart 파일을 열고 코드를 입력해 봅시다.



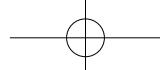
lib / screens / my_carrot / my_carrot_screen.dart

```
class MyCarrotScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      // ❶
      backgroundColor: Colors.grey[100],
      appBar: AppBar(
        title: const Text('나의 당근'),
        actions: [
          IconButton(icon: const Icon(Icons.settings), onPressed: () {}),
        ],
        bottom: const PreferredSize(
          preferredSize: Size.fromHeight(0.5),
          child: Divider(thickness: 0.5, height: 0.5, color: Colors.grey),
        ),
      ),
      body: ListView(
        children: [
          // ❷
          //MyCarrotHeader(),
          const SizedBox(height: 8.0),
          // ❸
          //CardIconMenu(iconMenuList: iconMenu1),
          const SizedBox(height: 8.0),
          // ❹
          //CardIconMenu(iconMenuList: iconMenu2),
          const SizedBox(height: 8.0),
          // ❺
          //CardIconMenu(iconMenuList: iconMenu3),
        ],
      ),
    );
  }
}
```

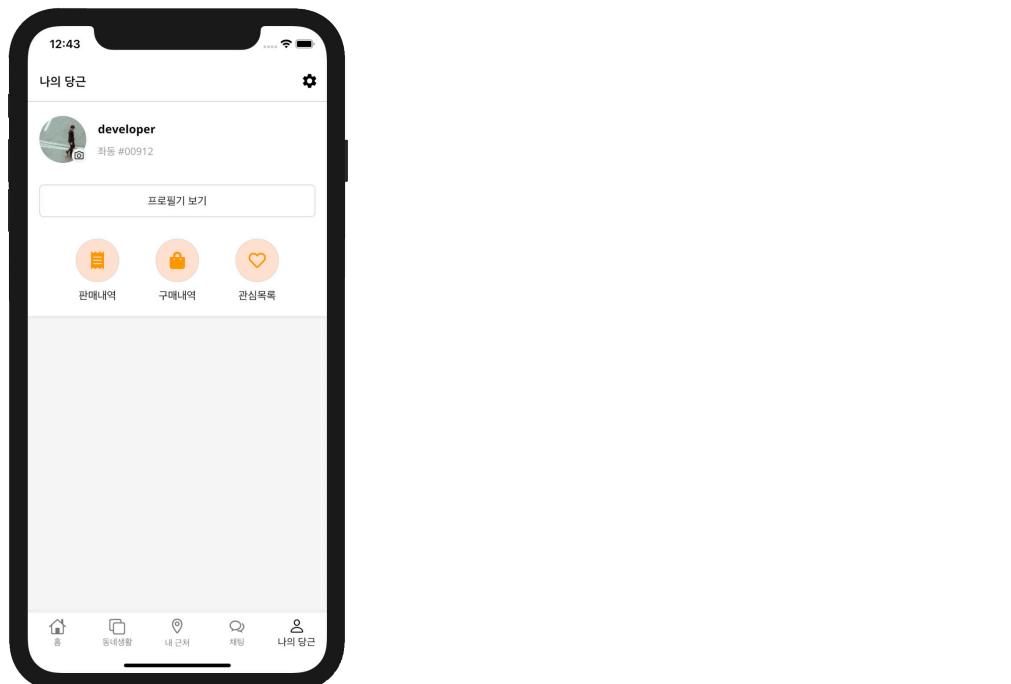
❶ Scaffold를 기본 배경색을 white로 지정했습니다. 하지만 개별적으로 색상을 지정해서 사용할 수 있습니다.

❷ 나의 당근 화면 상단 부분을 별도의 파일로 분리해서 위젯을 만들겠습니다. 기본적인 위젯의 코드만 작성하고 주석을 풀어 실제 위젯을 보면서 작업해 주세요.

❸ ❹ ❺ 번과 함께 Card 위젯을 사용해서 화면 하단 부분을 별도의 파일로 분리해서 위젯을 만들겠습니다.

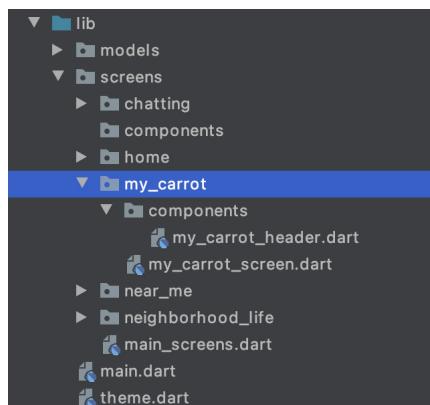


나의당근 화면 Header 위젯 만들기

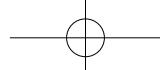


◆ 나의당근 Header 위젯

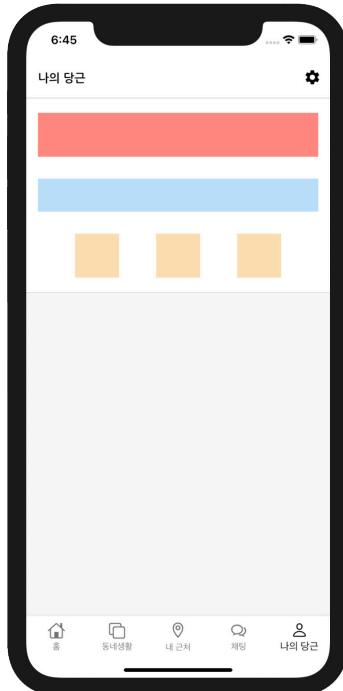
lib / screens / my_carrot 폴더에 components 폴더를 생성하고 my_carrot_header.dart 파일을 만들어 주세요.



◆ 나의당근 폴더 및 파일 생성



① MyCarrotHeader 기본 코드 만들기



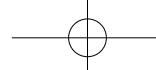
◆ 기본 레이아웃 구성

단계를 나누어서 작업해 봅시다. 1.4.2 절에서 MyCarrotHeader 위젯의 완성된 사진을 보고 대략적인 레이아웃 구조를 먼저 만들어봅시다.

```
ib / screens / my_carrot / components / my_carrot_header.dart

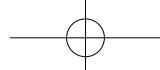
class MyCarrotHeader extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // ❶
    return Card(
      elevation: 0.5,
      margin: EdgeInsets.zero,
      // ❷
      shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(0.0)),
      // ❸
      child: Padding(
        padding: const EdgeInsets.symmetric(vertical: 20, horizontal: 16),
        // ❹
        child: Column(
          children: [
            // ❺
            _buildProfileRow(),

```

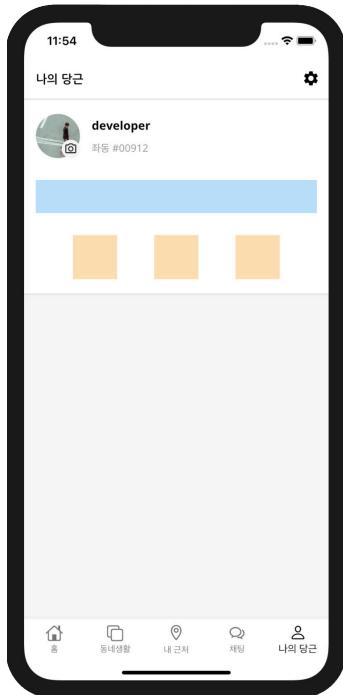


```
const SizedBox(height: 30),  
// ⑥  
_buildProfileButton(),  
const SizedBox(height: 30),  
// 7  
Row(  
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
    children: [  
        _buildRoundTextButton('판매내역', FontAwesomeIcons.receipt),  
        _buildRoundTextButton('구매내역', FontAwesomeIcons.shoppingBag),  
        _buildRoundTextButton('관심목록', FontAwesomeIcons.heart),  
    ],  
)  
],  
,  
),  
);  
}  
// ⑤  
Widget _buildProfileRow() {  
    return Container(color: Colors.redAccent[100], height: 60);  
}  
// ⑥  
Widget _buildProfileButton() {  
    return Container(color: Colors.blue[100], height: 45);  
}  
// ⑦  
Widget _buildRoundTextButton(String title, IconData iconData) {  
    return Container(color: Colors.orange[100], height: 60, width: 60);  
}
```

- ❶ Card 위젯은 입체감과 모서리에 곡선이 필요한 위젯을 만들 때 사용하는 위젯입니다. Card 위젯 자체에는 크기를 지정할 수 없고 부모 위젯의 제약과 자식 위젯의 크기에 따라 크기가 결정됩니다. Card 위젯은 기본적으로 margin을 가지고 있습니다. margin 값은 여기에서는 필요가 없기 때문에 EdgeInsets.zero를 설정해 줍니다.
- ❷ Card 위젯의 모서리 곡선을 없애기 위해 사용합니다.
- ❸ Card 위젯 세로와 가로의 패딩 값을 설정합니다.
- ❹ Card 위젯의 내부 컨텐츠의 레이아웃 구조를 Column으로 만들어 줍니다.
- ❺ Container로 레이아웃을 먼저 그려 줍니다. 잠시 후에 Row 위젯 와 그 자식 위젯들로 교체합니다.
- ❻ 잠시 후에 InkWell 위젯으로 감싸고 위젯을 완성합니다.
- ❼ 잠시 후에 Column 위젯으로 교체 위젯들을 완성해 봅시다.



② _buildProfileRow 메서드 완성하기

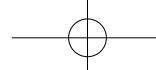


◆ _buildProfileRow위젯

StatelessWidget 안에서 반복적인 작업이나 가독성을 위해 위젯을 메서드로 분리해서 사용할 수 있습니다. 4.2.1의 5번에서 정의했던 _buildProfileRow 메서드를 완성해 봅시다.

ib / screens / my_carrot / components / my_carrot_header.dart

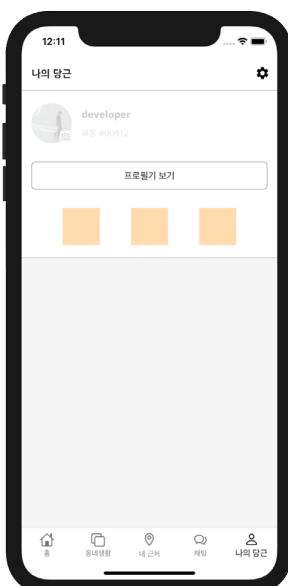
```
... 생략
Widget _buildProfileRow() {
  return Row(
    children: [
      // ❶
      Stack(
        children: [
          SizedBox(
            width: 65,
            height: 65,
            child: ClipRRect(
              borderRadius: BorderRadius.circular(32.5),
              child: Image.network(
                'https://placeimg.com/200/100/people',
                fit: BoxFit.cover,
              ),
            ),
          ),
          Positioned(
            bottom: 0,
            right: 0,
```



```
child: Container(
    width: 20,
    height: 20,
    decoration: BoxDecoration(
        borderRadius: BorderRadius.circular(15),
        color: Colors.grey[100]),
    child: Icon(
        Icons.camera_alt_outlined,
        size: 15,
    ),
),
),
),
],
),
),
SizedBox(width: 16),
Column(
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
        Text('developer', style:textTheme().headline2),
        SizedBox(height: 10),
        Text('좌동 #00912'),
    ],
),
),
);
}
... 생략
```

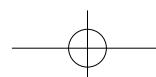
① Stack 위젯과 Positioned 위젯을 활용해서 이미지 위에 다른 위젯을 쌓고 꾸며 줄 수 있습니다.

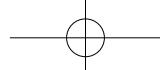
③ _buildProfileButton 메서드 완성하기



◆ _buildProfileButton위젯

39 플로터로 앱 만들기(실전)





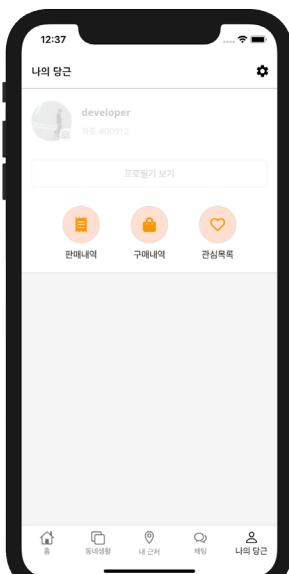
Flutter에서는 버튼 위젯을 다양하게 지원 하지만 Container와 BoxDecoration 위젯을 사용해서 우리가 원하는 모양의 버튼을 만들어 줄 수 있습니다. _buildProfileButton 메서드를 완성해 봅시다.

ib / screens / my_carrot / components / my_carrot_header.dart

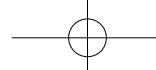
```
Widget _buildProfileButton() {  
    // ❶  
    return InkWell(  
        onTap: () {},  
        child: Container(  
            decoration: BoxDecoration(  
                border: Border.all(  
                    color: Color(0xFFD4D5DD),  
                    width: 1.0,  
                ),  
                borderRadius: BorderRadius.circular(6.0),  
            ),  
            height: 45,  
            child: Center(  
                child: Text(  
                    '프로필기 보기',  
                    style: textTheme().subtitle1,  
                ),  
            ),  
        );  
}
```

❶ InkWell 위젯을 활용해서 onTap 기능을 만들어 줄 수 있습니다.

❷ _buildRoundTextButton 메서드 완성하기



◆ _buildRoundTextButton위젯



lib / screens / my_carrot / components / my_carrot_header.dart

```
Widget _buildRoundTextButton(String title, IconData iconData) {
    return Column(
        children: [
            Container(
                width: 60,
                height: 60,
                decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(30.0),
                    color: Color.fromRGBO(255, 226, 208, 1),
                    border: Border.all(color: Color(0xFFD4D5DD), width: 0.5)),
                child: Icon(
                    iconData,
                    color: Colors.orange,
                ),
            ),
            SizedBox(height: 10),
            Text(
                title,
                style: textTheme().subtitle1,
            )
        ],
    );
}
```

모델 클래스 만들기

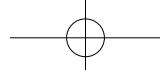
나의 당근 화면에서 사용할 샘플 데이터를 만들어 봅시다. lib / models 폴더에 icon_menu.dart 파일을 만들어 주세요.

lib / models / icon_menu.dart

```
class IconMenu {
    // ①
    final String title;
    // ②
    final IconData iconData;

    IconMenu({required this.title, required this.iconData});

    // 화면에 사용할 데이터
    final List<IconMenu> iconMenu1 = [
        IconMenu(title: '내 동네 설정', iconData: FontAwesomeIcons.mapMarkerAlt),
        IconMenu(title: '동네 인증하기', iconData: FontAwesomeIcons.compressArrowsAlt),
        IconMenu(title: '키워드 알림', iconData: FontAwesomeIcons.tag),
    ];
}
```



```

IconMenu(title: '모아보기', IconData: FontAwesomeIcons.borderAll)
];

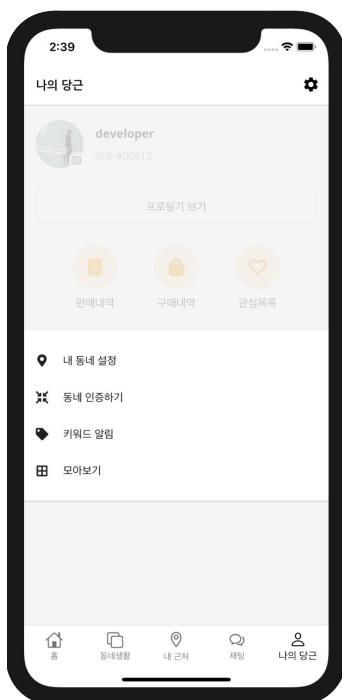
final List<IconMenu> iconMenu2 = [
  IconMenu(title: '동네생활 글', IconData: FontAwesomeIcons.edit),
  IconMenu(title: '동네생활 댓글', IconData: FontAwesomeIcons.commentDots),
  IconMenu(title: '동네생활 주제 목록', IconData: FontAwesomeIcons.star)
];

final List<IconMenu> iconMenu3 = [
  IconMenu(title: '비즈프로필 관리', IconData: FontAwesomeIcons.store),
  IconMenu(title: '지역광고', IconData: FontAwesomeIcons.bullhorn)
];

```

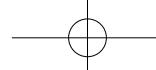
- ❶ 메뉴 위젯으로 만들 title로 사용합니다.
❷ 메뉴 위젯을 만들 IconData으로 사용합니다.

하단 Card 메뉴 위젯 만들기



◆ Card 메뉴 위젯

lib / screens / my_carrot / components 폴더에 card_icon_menu.dart 파일을 만들어 주세요.



lib / screens / my_carrot / components / card_icon_menu.dart

```
class CardIconMenu extends StatelessWidget {
    // ①
    final List<IconMenu> iconMenuList;

    CardIconMenu({required this.iconMenuList});

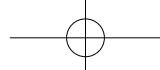
    @override
    Widget build(BuildContext context) {
        // ②
        return Card(
            elevation: 0.5,
            margin: EdgeInsets.zero,
            shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(0.0)),
            child: Padding(
                padding: const EdgeInsets.all(16.0),
                child: Column(
                    // ③
                    children: List.generate(
                        iconMenuList.length,
                        (index) => _buildRowIconItem(
                            iconMenuList[index].title, iconMenuList[index].iconData),
                    ),
                ),
            ),
        );
    }

    Widget _buildRowIconItem(String title, IconData iconData) {
        return Container(
            height: 50,
            child: Row(
                children: [
                    Icon(iconData, size: 17),
                    const SizedBox(width: 20),
                    Text(title, style: textTheme().subtitle1)
                ],
            ),
        );
    }
}
```

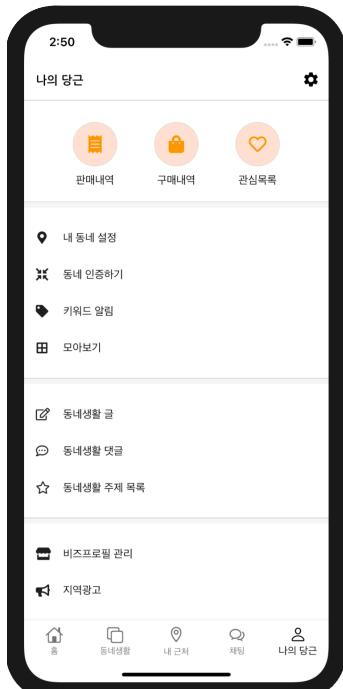
① List<IconMenu> 타입을 가진 멤버 변수에 선언합니다.

② Card 위젯을 활용하고 둘근 모서리 효과를 없애 줍니다.

③ List.generate()는 리스트를 만들어 주는 생성자입니다. length의 길이만큼 0부터 index -1까지 범위의 각 인덱스를 오름차순으로 호출하여 만든 값으로 리스트 생성합니다. 멤버 변수 iconMenuList에 들어오는 값으로 _buildRowIconItem 메서드를 호출시켜 위젯을 만들어 줍니다.



MyCarrotScreen 위젯 완성하기

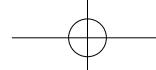


◆ MyCarrotScreen위젯

my_carrot_screen.dart 파일을 열고 주석 해제 후 import 구문도 완성해 주세요.

```
lib / screens / my_carrot / my_carrot_screen.dart

class MyCarrotScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      appBar: AppBar(
        title: const Text('나의 당근'),
        actions: [
          IconButton(icon: const Icon(Icons.settings), onPressed: () {}),
        ],
        bottom: const PreferredSize(
          preferredSize: Size.fromHeight(0.5),
          child: Divider(thickness: 0.5, height: 0.5, color: Colors.grey),
        ),
      ),
      body: ListView(
        children: [
          MyCarrotHeader(),
          const SizedBox(height: 8.0),
          // ❶
          CardIconMenu(iconMenuList: iconMenu1),
        ],
      ),
    );
  }
}
```



```

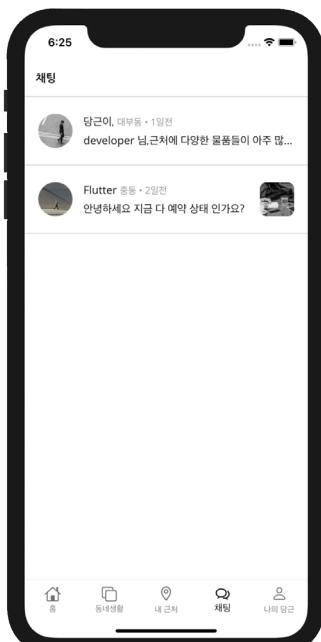
const SizedBox(height: 8.0),
// ②
CardIconMenu(iconMenuList: iconMenu2),
const SizedBox(height: 8.0),
// ③
CardIconMenu(iconMenuList: iconMenu3),
],
),
);
}
}
}

```

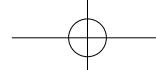
- ① 우리가 만든 CardIconMenu 위젯을 사용합니다. 모델 클래스 만들에서 작업했던 List<IconMenu> 타입의 iconMenu1로 선언한 리스트 데이터를 넣어 주세요.
- ② List<IconMenu> 타입의 iconMenu2로 선언한 리스트 데이터를 넣어 줍니다.
- ③ List<IconMenu> 타입의 iconMenu3로 선언한 리스트 데이터를 넣어 줍니다.

01 _ 5 채팅화면 만들기

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_05에 공개되어 있습니다.
이번 장을 완료하면 아래와 같은 화면을 만들 수 있습니다.



◆ 채팅화면



작업 순서

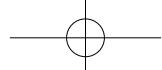
- ① 모델 클래스 및 샘플 데이터 만들기
- ② 재사용 위젯 만들기
- ③ CarttingScreen 위젯 기본 코드 입력하기
- ④ ChatContainer 위젯 만들기
- ⑤ CarttingScreen 위젯 완성하기

모델 클래스 및 샘플 데이터 만들기

lib / models / chat_message.dart

```
class ChatMessage {  
    final String sender;  
    final String profileImage;  
    final String location;  
    final String sendDate;  
    final String message;  
    final String? imageUri;  
  
    ChatMessage({  
        required this.sender,  
        required this.profileImage,  
        required this.location,  
        required this.sendDate,  
        required this.message,  
        this.imageUri,  
    });  
}  
  
// 샘플 데이터  
List<ChatMessage> chatMessageList = [  
    ChatMessage(  
        sender: '당근이',  
        profileImage: 'https://placeimg.com/200/100/people/grayscale',  
        location: '대부동',  
        sendDate: '1일전',  
        message: 'developer 님, 근처에 다양한 물품들이 아주 많이 있습니다.',  
    ),  
    ChatMessage(  
        sender: 'Flutter ',  
        profileImage: 'https://placeimg.com/200/100/people',  
        location: '중동',  
        sendDate: '2일전',  
        message: '안녕하세요 지금 다 예약 상태인가요?',  
        imageUri: 'https://placeimg.com/200/100/tech/grayscale'  
    );
```

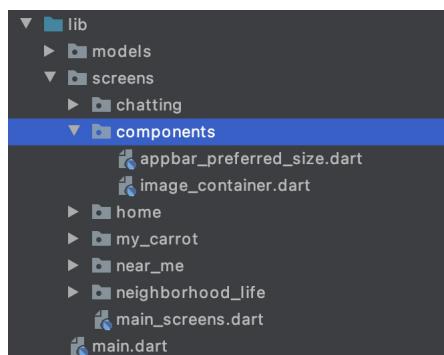
46



재사용 위젯 만들기

앱 뼈대 만들기에서 lib / screens / components 폴더를 만들었습니다. 이 폴더에는 하나의 화면에서만 사용하는 위젯이 아닌 여러 화면에서 사용하는 위젯들을 관리하는 폴더입니다. 여기서는 두 개의 재사용 가능한 위젯을 만들어보고 사용해 봅시다.

먼저 lib / screens / components 폴더에 `appbar_preferred_size.dart` 파일과 `image_container.dart` 파일을 만들어 주세요.



◆ 폴더 및 파일 만들기

① appBarBottomLine 메서드 만들기

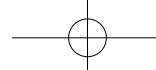
우리는 반복적으로 사용할 수 있는 위젯들은 StatelessWidget 또는 StatefulWidget을 상속받은 class로 만들어 별도의 파일로 만드는 방법을 배웠습니다. 하지만 꼭 Class 위젯으로 만들지 않고 별도의 파일에 전역 메서드를 만들어서 사용할 수 있습니다.

이번 앱 예제에서는 각각의 화면마다 Appbar의 bottom 속성을 사용해 구분선을 만들어 주고 있습니다. 반복적인 작업이니 이 부분도 별도에 파일로 만들어서 사용해 봅시다. `appbar_preferred_size.dart` 파일 열고 다음과 같이 작업해 봅시다.

```
lib / screens / components / appBar_preferred_size.dart

import 'package:flutter/material.dart';

PreferredSize appBarBottomLine() {
    var height = 0.5;
    // ①
    return PreferredSize(
        preferredSize: Size.fromHeight(height),
        child: Divider(
            thickness: height,
            height: height,
```



```
        color: Colors.grey,  
    ),  
);  
}
```

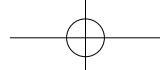
❶ Appbar의 bottom 속성에는 PreferredSize 위젯을 사용해야 합니다. PreferredSize 위젯은 자식 위젯에게 어떤 제약도 부과하지 않고 부모 위젯에게 공간을 차지하는 크기만을 알려주는 위젯입니다.

❷ ImageContainer 위젯 만들기

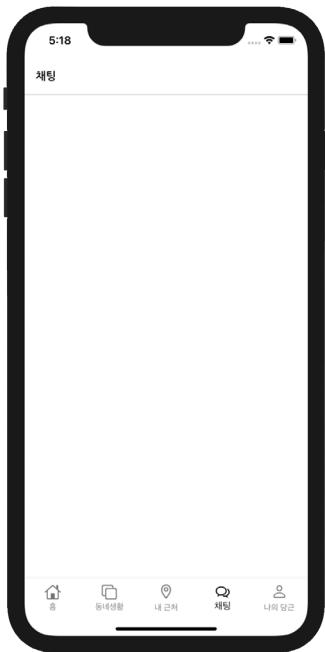
이미지를 그려주는 위젯도 별로의 파일로 분리해서 만들어봅시다. image_container.dart 파일을 열고 다음과 같이 코딩해 주세요.

lib / screens / components / image_container.dart

```
import 'package:flutter/material.dart';  
  
class ImageContainer extends StatelessWidget {  
    final double borderRadius;  
    final String imageUrl;  
    final double width;  
    final double height;  
  
    const ImageContainer({  
        Key? key,  
        required this.borderRadius,  
        required this.imageUrl,  
        required this.width,  
        required this.height,  
    }) : super(key: key);  
  
    @override  
    Widget build(BuildContext context) {  
        return ClipRRect(  
            borderRadius: BorderRadius.circular(borderRadius),  
            child: Image.network(  
                imageUrl,  
                width: width,  
                height: height,  
                fit: BoxFit.cover,  
            ),  
        );  
    }  
}
```



CarttingScreen 위젯 기본 코드 입력하기

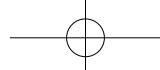


◆ 채팅 기본 화면

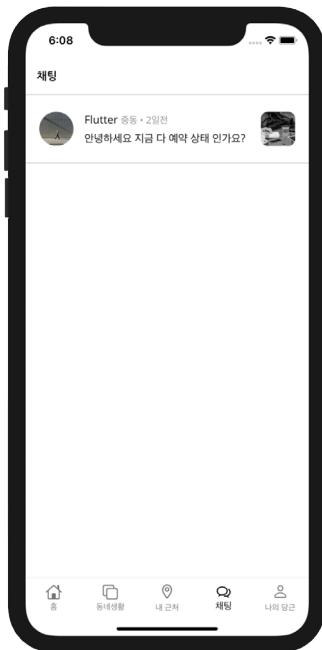
```
lib / screens / chatting / chatting_screen.dart
```

```
class ChattingScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('채팅'),
        // ❶
        bottom: appBarBottomLine(),
      ),
      // ❷
      body: ListView(
        children: List.generate(
          chatMessageList.length,
          // ❸
          (index) => Container(),
        ),
      );
    }
}
```

- ❶ AppBar의 bottom 속성에 우리가 만든 appBarBottomLine 메서드를 사용합니다.
- ❷ 앞서 홈 화면 만들기에서 ListView.separated 위젯을 사용해서 리스트 항목을 만들고 하단에 Divider 위젯을 활용해서 구분선을 만드는 방법을 배웠습니다. 이번에는 ListView 위젯을 사용해서 만드는 방법을 배워 보겠습니다.
- ❸ Container 위젯은 잠시 후에 다른 위젯으로 교체될 예정입니다.



ChatContainer 위젯 만들기

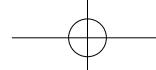


◆ ChatContainer 위젯

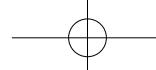
lib / screens / chatting에 components 폴더를 생성하고 chat_container.dart 파일을 만들어 주세요. 재사용 위젯 만들기에서 작업했던 ImageConatiner 위젯도 활용해 봅시다.

lib / screens / chatting / components / chat_container.dart

```
class ChatContainer extends StatelessWidget {  
  const ChatContainer({  
    Key? key,  
    required this.chatMessage,  
  }) : super(key: key);  
  // ❶  
  final ChatMessage chatMessage;  
  
  @override  
  Widget build(BuildContext context) {  
    return Container(  
      // ❷  
      decoration: const BoxDecoration(  
        border: Border(bottom: BorderSide(color: Colors.grey, width: 0.5)),  
      ),  
      height: 100,  
      child: Padding(  
        padding: const EdgeInsets.all(20),
```



```
child: Row(
  children: [
    // 3
    ImageContainer(
      width: 50,
      height: 50,
      borderRadius: 25,
      imageUrl: chatMessage.profileImage,
    ),
    const SizedBox(width: 16),
    Expanded(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
          const Spacer(),
          // 4
          Text.rich(
            TextSpan(children: [
              TextSpan(
                text: chatMessage.sender,
                style: textTheme().bodyText1,
              ),
              TextSpan(text: chatMessage.location),
              TextSpan(text: ' • ${chatMessage.sendDate}'),
            ]),
          ),
          const Spacer(),
          Text(
            chatMessage.message,
            style: textTheme().bodyText1,
            overflow: TextOverflow.ellipsis,
          ),
          const Spacer(),
        ],
      ),
    ),
    Visibility(
      // 5
      visible: chatMessage.imageUri != null,
      child: Padding(
        padding: const EdgeInsets.only(left: 8.0),
        child: ClipRRect(
          borderRadius: BorderRadius.circular(10),
          child: ImageContainer(
            width: 50,
            height: 50,
            borderRadius: 8,
          ),
        ),
      ),
    ),
  ],
)
```



```
        imageUrl: chatMessage.imageUri ?? '',
        ),
        ),
        ),
        ),
        ],
        ),
        );
    }
}
```

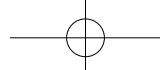
- ❶ ChatContainer 위젯을 생성할 때 우리가 만든 모델 클래스 ChatMessage 객체를 넘겨받습니다.
- ❷ Container의 하단 부분에 decoration 속성을 사용해서 구분 선을 만들어 줄 수 있습니다. 부모 위젯에서 ListView.separated 위젯을 사용해서 만들 수 있지만 다양한 방법을 학습해 보는 게 좋습니다.
- ❸ 재사용 위젯 만들기에서 작업했던 ImageContainer 위젯을 사용합니다.
- ❹ Text, rich 위젯은 문단 단위로 텍스트를 꾸며 줄 수 있습니다.
- ❺ Visibility 위젯을 활용해 chatMessage.imageUri가 null 아닐 경우 위젯을 보여줍니다.

CarttingScreen 위젯 완성하기

lib / screens / chatting / chatting_screen.dart

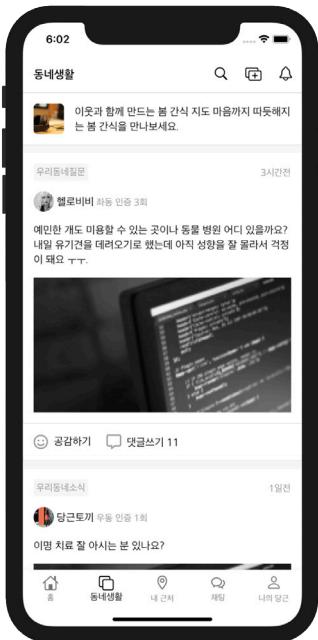
```
class ChattingScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: Text('채팅'),
                bottom: appBarBottomLine(),
            ),
            body: ListView(
                children: List.generate(
                    chatMessageList.length,
                    // ❶
                    (index) => ChatContainer(chatMessage: chatMessageList[index]),
                ),
            ),
        );
    }
}
```

- ❶ lib / models / chat_message.dart 파일에서 만든 데이터와 ChatContainer을 사용해서 위젯을 완성합니다.



01 _ 6 동네생활 화면 만들기

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_06에 공개되어 있습니다.
이번 장을 완료하면 만들 수 있는 화면입니다.

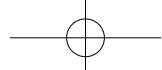


◆ 동네생활 화면

이번 장에서는 스프레드 연산자(spread operator)와 List.generate 생성자를 활용해서 위젯을 만드는 방법을 배워 보겠습니다.

작업 순서

- ① 모델 클래스 및 샘플 데이터 만들기
- ② NeighborhoodLifeScreen 기본 코드 입력하기
- ③ Header 위젯 만들기
- ④ Body 위젯 만들기
- ⑤ NeighborhoodLifeScreen 위젯 완성하기



모델 클래스 및 샘플 데이터 만들기

동네생활 화면에 사용할 모델 클래스를 만들어 봅시다.

lib / models 폴더에 neighborhood_life.dart 파일을 만들어 주세요.

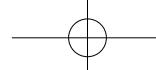
① 모델 클래스 만들기

lib / models / neighborhood_life.dart

```
class NeighborhoodLife {  
    final String category;  
    final String profileImgUri;  
    final String userName;  
    final String location;  
    final String content;  
    final String contentImgUri;  
    final int commentCount;  
    final int authCount;  
    final String date;  
  
    NeighborhoodLife({  
        required this.category,  
        required this.profileImgUri,  
        required this.userName,  
        required this.location,  
        required this.content,  
        required this.contentImgUri,  
        required this.commentCount,  
        required this.authCount,  
        required this.date,  
    });  
}  
  
// 샘플 데이터 1  
...  
// 샘플 데이터 2  
...
```

② 샘플 데이터 만들기

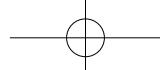
github에서 받은 소스 코드에서 carrot_market_ui_06에 neighborhood_life.dart 파일의 코드를 복사해서 사용해도 됩니다.



lib / models / neighborhood_life.dart

```
// 샘플 데이터 1
String lifeTitle = '이웃과 함께 만드는 봄 간식 지도 마음까지 따듯해지는 봄 간식을 만나보세요。';

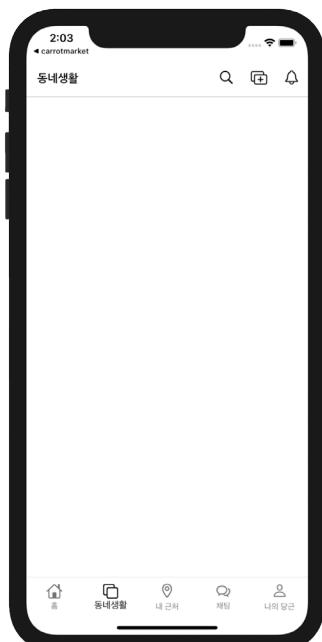
// 샘플 데이터 2
List<NeighborhoodLife> neighborhoodLifeList = [
    NeighborhoodLife(
        category: '우리동네질문',
        profileImgUri: 'https://placeimg.com/200/100/people/grayscale',
        userName: '헬로비비',
        location: '좌동',
        content: '예민한 개도 미용할 수 있는 곳이나 동물 병원 어디 있을까요?\n' +
            '내일 유기견을 데려오기로 했는데 아직 성향을 잘 몰라서 걱정이 돼요 ㅜㅜ。',
        contentImgUri: 'https://placeimg.com/200/100/tech/grayscale',
        commentCount: 11,
        authCount: 3,
        date: '3시간전',
    ),
    NeighborhoodLife(
        category: '우리동네소식',
        profileImgUri: 'https://placeimg.com/200/100/people',
        userName: '당근토끼',
        location: '우동',
        content: '이명 치료 잘 아시는 분 있나요?',
        contentImgUri: 'https://placeimg.com/200/100/animal/grayscale',
        commentCount: 2,
        authCount: 1,
        date: '1일전',
    ),
    NeighborhoodLife(
        category: '분실',
        profileImgUri: 'https://placeimg.com/200/100/nature/grayscale',
        userName: 'flutter',
        location: '중동',
        content: '롯데 3차 니나도 롯데캐슬 방향으로 재래시장 앞쪽 지나 혹시 에어팟 오른쪽 주우신 분 있나요ㅜㅜ',
        contentImgUri: '',
        commentCount: 11,
        authCount: 8,
        date: '1일전',
    ),
    NeighborhoodLife(
        category: '우리동네질문',
        profileImgUri: 'https://placeimg.com/200/100/any',
        userName: '구름나드리',
        location: '우동',
```



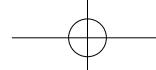
```
        content: '밤부터 새벽까지 하던 토스트 아저씨 언제 다시 오나요ㅠㅠ',
        contentImgUri: '',
        commentCount: 0,
        authCount: 7,
        date: '3일전',
    ),
    NeighborhoodLife(
        category: '우리동네질문',
        profileImgUri: 'https://placeimg.com/200/100/people/grayscale',
        userName: '아는형',
        location: '만덕동',
        content: '아니 이 시간에 마이크 들고 노래하는 사람은 정상인가요?',
        contentImgUri: 'https://placeimg.com/200/100/tech',
        commentCount: 11,
        authCount: 2,
        date: '5일전',
    ),
];

```

NeighborhoodLifeScreen 위젯 기본 코드 입력하기



◆ 동네생활 기본 화면



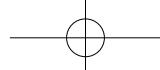
앱 뼈대 만들기에서 작업했던 neighborhood_life_screen.dart 파일을 열고 기본 코드를 입력해 봅시다.

lib / screens / neighborhood_life / neighborhood_life_screens.dart

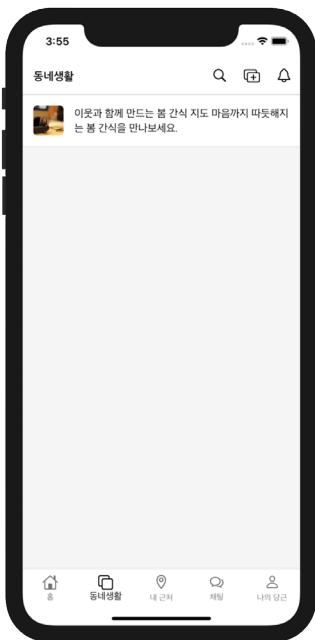
```
class NeighborhoodLifeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Scaffold(
            backgroundColor: Colors.grey[100],
            appBar: AppBar(
                title: Text('동네생활'),
                actions: [
                    IconButton(icon: Icon(CupertinoIcons.search), onPressed: () {}),
                    IconButton(
                        icon: Icon(CupertinoIcons.plus_rectangle_on_rectangle),
                        onPressed: () {}),
                    IconButton(icon: Icon(CupertinoIcons.bell), onPressed: () {}),
                ],
                bottom: appBarBottomLine(),
            ),
            body: ListView(
                children: [
                    // ❶
                    // LifeHeader(),
                    // ❷
                    // Padding(
                    //     padding: const EdgeInsets.only(bottom: 12.0),
                    //     child: LifeBody(
                    //         neighborhoodLife: neighborhoodLifeList[0],
                    //     ),
                    // ),
                    ],
                ),
            );
    }
}
```

❶ 잠시 후에 상단 부분 LifeHeader 위젯을 만들어 봅니다. 기본 코드 작성 후 주석을 해제할 예정입니다.

❷ 잠시 후에 만들 LifeBody 위젯입니다. 기본 코드 작성 후 주석을 해제할 예정입니다.



LifeHeader 위젯 만들기

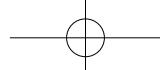


◆ LifeHeader 위젯

동네생활 상단 부분 위젯을 만들어봅시다. 먼저 lib / screens / neighborhood_life에 components 폴더를 생성하고 life_header.dart 파일을 만들어 주세요.

lib / screens / neighborhood_life / components / life_header.dart

```
class LifeHeader extends StatelessWidget {  
    @override  
    Widget build(BuildContext context) {  
        return Card(  
            margin: EdgeInsets.only(bottom: 12.0),  
            elevation: 0.5,  
            // ❶  
            shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(0.0)),  
            child: Padding(  
                padding: const EdgeInsets.all(16.0),  
                child: Row(  
                    children: [  
                        // ❷  
                        ImageContainer(  
                            borderRadius: 6.0,  
                            imageUrl: 'https://placeimg.com/200/100/any',  
                            width: 45.0,  
                            height: 45.0),  
                    ],  
                ),  
            ),  
        );  
    }  
}  
  
class ImageContainer extends StatelessWidget {  
    final BorderRadius borderRadius;  
    final String imageUrl;  
    final double width;  
    final double height;  
  
    ImageContainer({  
        required this.imageUrl,  
        required this.width,  
        required this.height,  
        this.borderRadius = BorderRadius.circular(6.0),  
    });  
  
    @override  
    Widget build(BuildContext context) {  
        return Container(  
            width: width,  
            height: height,  
            decoration: BoxDecoration(  
                image: DecorationImage(image: NetworkImage(imageUrl)),  
                borderRadius: borderRadius,  
            ),  
        );  
    }  
}
```



```

const SizedBox(width: 16.0),
// ❸
Expanded(
  child: Text(
    lifeTitle,
    style: textTheme().bodyText1,
    maxLines: 2,
    overflow: TextOverflow.ellipsis,
  ),
)
],
),
);
}
}
}

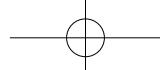
```

- ❶ Card 위젯을 활용해서 만들고 하단에 둉근 모서리 효과를 제거합니다.
- ❷ 재사용 위젯으로 만들었던 ImageContainer 위젯을 활용합니다.
- ❸ 부모 위젯인 Row 위젯의 ImageContainer 위젯을 제외한 남는 공간을 확장합니다. 1.6.2절 예제에서 // 1 주석을 풀고 life_header.dart 파일을 import 해주세요.

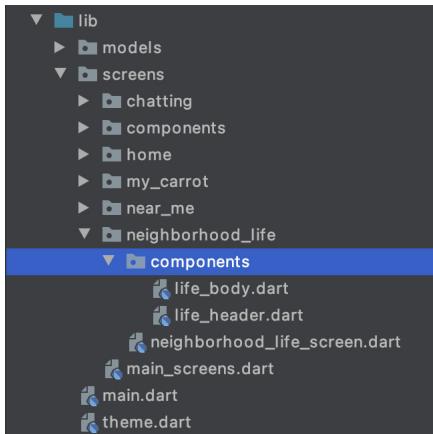
LifeBody 위젯 만들기



◆ LifeBody위젯

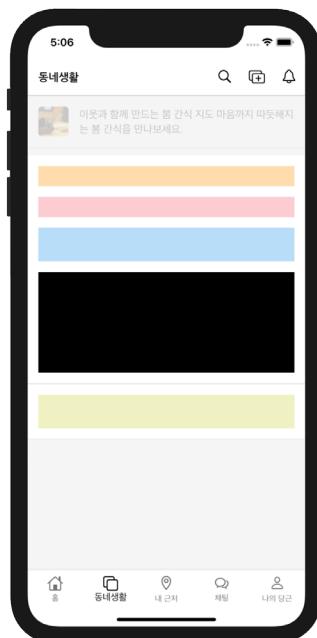


동네생활 중간 부분의 위젯을 만들어 봅시다. components 폴더에 life_body.dart 파일을 만들어 주세요.



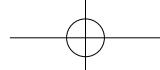
◆ 동네생활 폴더 및 파일

① LifeBody 기본 코드 만들기



◆ LifeBody기본위젯

위젯 구조가 조금 복잡할 수 있습니다. 전체적인 레이아웃 먼저 그려주고 작업을 진행하겠습니다.



lib / screens / neighborhood_life / components / life_body.dart

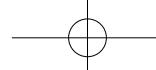
```
class LifeBody extends StatelessWidget {
    // ❶
    final NeighborhoodLife neighborhoodLife;

    const LifeBody({Key? key, required this.neighborhoodLife}) : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Container(
            decoration: BoxDecoration(
                color: Colors.white,
                // ❷
                border: Border(
                    bottom: BorderSide(width: 0.5, color: Color(0xFFD4D5DD)),
                ),
            ),
            child: Column(
                children: [
                    _buildTop(),
                    _buildWriter(),
                    _buildWriting(),
                    _buildImage(),
                    // ❸
                    Divider(
                        height: 1,
                        thickness: 1,
                        color: Colors.grey[300],
                    ),
                    // ❹
                    _buildTail(neighborhoodLife.commentCount)
                ],
            ),
        );
    }

    Padding _buildTop() {
        return Padding(
            padding: const EdgeInsets.symmetric(
                vertical: 16,
                horizontal: 16,
            ),
            child: Container(color: Colors.orange[100], height: 30),
        );
    }

    Padding _buildWriter() {
        return Padding(
            padding: const EdgeInsets.symmetric(horizontal: 16),
            child: Container(color: Colors.red[100], height: 30),
        );
    }
}
```



```

Padding _buildWriting() {
  return Padding(
    padding: const EdgeInsets.symmetric(horizontal: 16, vertical: 16),
    child: Container(color: Colors.blue[100], height: 50),
  );
}

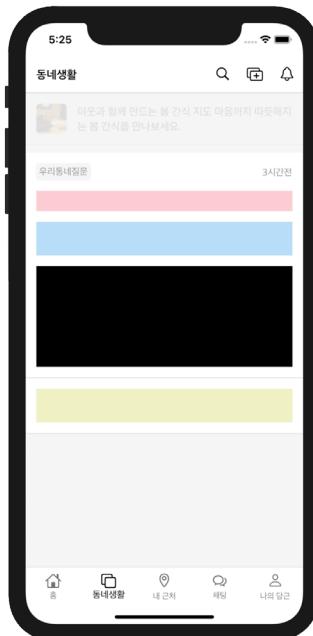
Padding _buildTail(int commentCount) {
  return Padding(
    padding: const EdgeInsets.only(left: 16, right: 16, bottom: 16),
    child: Container(color: Colors.black, height: 150),
  );
}

Padding _buildTail(int commentCount) {
  return Padding(
    padding: const EdgeInsets.all(16),
    child: Container(color: Colors.lime[100], height: 50),
  );
}
}

```

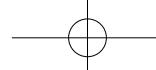
- ① lib / models / neighborhood_life.dart 파일에 만든 NeighborhoodLife 클래스를 선언합니다.
 - ② LifeBody 위젯의 최상위 위젯은 Container 위젯으로 하고 하단에 border 속성을 사용하여 구분선을 그려줍니다.
 - ③ Divider 위젯에 양쪽 끝에 패딩을 주지 않습니다.
 - ④ NeighborhoodLife 객체에 넘겨받는 commentCount 값을 메서드 인자 값으로 넣어 줍니다. 잠시 후에 사용하게 됩니다.
- 1.6.2절 예제에서 // 2 주석을 풀고 life_body.dart 파일을 import 해주세요.

② _buildTop 위젯 만들기



◆ _buildTop위젯

Chapter 01 • 당근마켓 UI 만들어보기 62



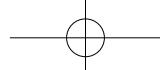
_buildTop 메서드를 다음과 같이 수정해 주세요.

lib / screens / neighborhood_life / components / life_body.dart

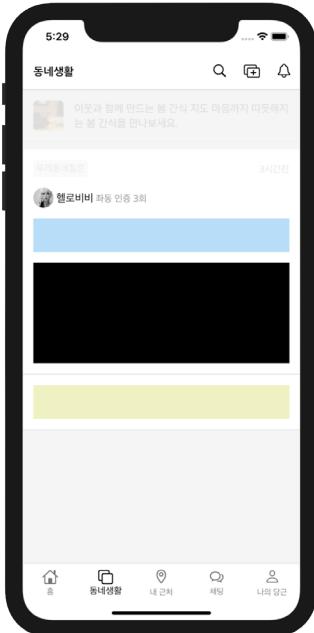
```
Padding _buildTop() {
    return Padding(
        padding: const EdgeInsets.symmetric(
            vertical: 16,
            horizontal: 16,
        ),
        child: Row(
            // ❶
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
                Container(
                    padding: EdgeInsets.all(4),
                    decoration: BoxDecoration(
                        // ❷
                        shape: BoxShape.rectangle,
                        borderRadius: BorderRadius.circular(4),
                        color: Color.fromRGBO(247, 247, 247, 1),
                    ),
                    child:
                        Text(neighborhoodLife.category, style: textTheme().bodyText2),
                ),
                Text(
                    neighborhoodLife.date,
                    style: textTheme().bodyText2,
                ),
            ],
        ),
    );
}
```

❶ 자식 위젯을 시작과 끝에 배치합니다. 시작과 끝 사이의 위젯의 나머지 공간은 모두 균일하게 배분합니다.

❷ Box의 모양을 직사각형으로 만들어 줍니다.



③ _buildWriter 위젯 만들기

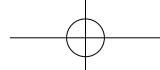


◆ _buildWriter위젯

_buildWriter 메서드를 다음과 같이 수정해 주세요.

lib / screens / neighborhood_life / components / life_body.dart

```
Padding _buildWriter() {
    return Padding(
        padding: const EdgeInsets.symmetric(horizontal: 16),
        child: Row(
            children: [
                ImageContainer(
                    width: 30,
                    height: 30,
                    borderRadius: 15,
                    imageUrl: neighborhoodLife.profileImgUri,
                ),
                Text.rich(
                    TextSpan(
                        children: [
                            TextSpan(
                                text: '${neighborhoodLife.userName}',
                                style: textTheme().bodyText1),
                            // ❶
                            TextSpan(text: ' ${neighborhoodLife.location}'),
                            TextSpan(text: ' 인증 ${neighborhoodLife.authCount}회')
                        ]
                    )
                )
            ]
        )
    );
}
```



```
        ],
        ),
        )
    ],
    ),
);
}
```

- ❶ theme.dart 파일에서 bodyText2 속성의 값을 GoogleFonts.openSans(fontSize: 14.0, color: Colors.grey)로 지정했기 때문에 따로 값을 사용하지 않아도 됩니다.

④ _buildWriter 위젯 만들기

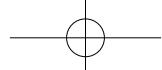


◆ _buildWriter위젯

_buildWriting 메서드를 다음과 같이 수정해주세요.

lib / screens / neighborhood_life / components / life_body.dart

```
Padding _buildWriting() {
    return Padding(
        padding: const EdgeInsets.symmetric(horizontal: 16, vertical: 16),
        // ❶
        child: Align(
            alignment: Alignment.centerLeft,
            child: Text(
                neighborhoodLife.content,
                style: textTheme().bodyText1,
```



```

        maxLines: 3,
        overflow: TextOverflow.ellipsis,
        textAlign: TextAlign.start,
    ),
),
);
}
}

```

① Text 위젯을 centerLeft로 정렬하기 위해 사용합니다.

5 _buildImage 위젯 만들기



◆ _buildImage위젯

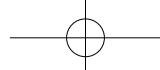
_buildImage 메서드를 다음과 같이 수정해주세요.

```

lib / screens / neighborhood_life / components / life_body.dart

Visibility _buildImage() {
    return Visibility(
        // ❶
        visible: neighborhoodLife.contentImgUri != '',
        child: Padding(
            padding: EdgeInsets.only(left: 16, right: 16, bottom: 16),
            child: Image.network(
                neighborhoodLife.contentImgUri,
                height: 200,

```



```
        width: double.infinity,
        fit: BoxFit.cover,
    ),
),
);
}
}
```

- ① 넘겨받는 객체(NeighborhoodLife)에 contentImageUri에 값이 있다면 위젯을 표시합니다.

⑥ _buildTail 위젯 만들기

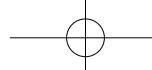


◆ _buildTail위젯

_buildTail() 메서드를 다음과 같이 수정해주세요.

lib / screens / neighborhood_life / components / life_body.dart

```
Padding _buildTail(int commentCount) {
    return Padding(
        padding: const EdgeInsets.all(16),
        child: Row(
            children: [
                Icon(
                    FontAwesomeIcons.smile,
                    color: Colors.grey,
                    size: 22,
                ),
            ],
        ),
    );
}
```

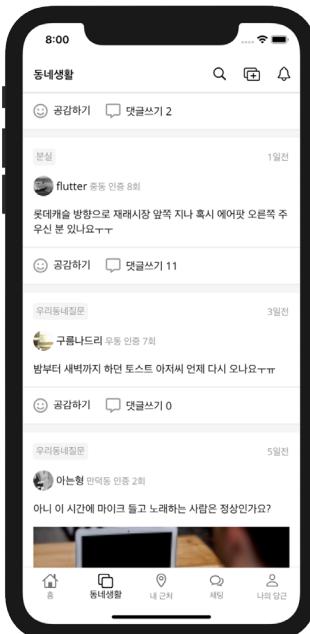


```

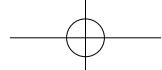
        SizedBox(width: 8),
        Text(
            '공감하기',
            style: TextStyle(fontSize: 16, color: Colors.black),
        ),
        SizedBox(width: 22),
        Icon(
            FontAwesomeIcons.commentAlt,
            color: Colors.grey,
            size: 22,
        ),
        SizedBox(width: 8),
        Text(
            "${"댓글쓰기"} $commentCount",
            style: TextStyle(fontSize: 16, color: Colors.black),
        ),
    ],
),
),
);
}
}

```

NeighborhoodLifeScreen 위젯 완성하기



◆ 동네생활 위젯 완성

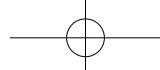


ListView 위젯 안에 스프레드 연산자와 Name 생성자(이름을 지정해준 생성자)를 사용해서 우리가 만든 LifeBody 위젯을 사용해 봅시다. neighborhood_life_screens.dart 파일을 열고 다음과 같이 코드를 수정해주세요.

lib / screens / neighborhood_life / neighborhood_life_screens.dart

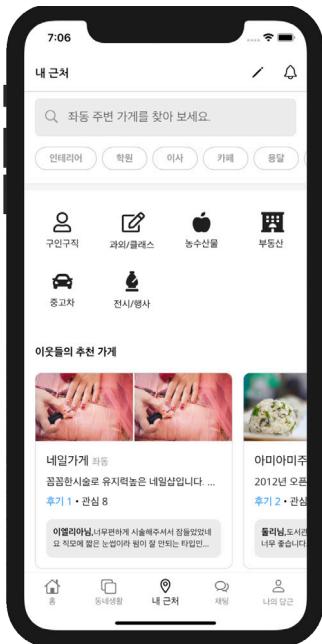
```
class NeighborhoodLifeScreen extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Scaffold(
            backgroundColor: Colors.grey[100],
            appBar: AppBar(
                title: Text('동네생활'),
                actions: [
                    IconButton(icon: Icon(CupertinoIcons.search), onPressed: () {}),
                    IconButton(
                        icon: Icon(CupertinoIcons.plus_rectangle_on_rectangle),
                        onPressed: () {}),
                    IconButton(icon: Icon(CupertinoIcons.bell), onPressed: () {}),
                ],
                bottom: appBarBottomLine(),
            ),
            body: ListView(
                children: [
                    LifeHeader(),
                    // ❶
                    ...List.generate(
                        neighborhoodLifeList.length,
                        (index) => Padding(
                            padding: const EdgeInsets.only(bottom: 8.0),
                            child: LifeBody(
                                neighborhoodLife: neighborhoodLifeList[index],
                            ),
                        ),
                    ),
                ],
            );
    }
}
```

❶ List.generate 생성자는 neighborhoodLifeList.length 길이만큼 반복문을 돌면서 데이터의 집합(Collection) 중 하나인 List(데이터의 순서가 있고 중복 허용) 형의 자료구조를 생성합니다. 즉 LifeBody 자식 위젯을 가지는 Padding 위젯을 List(자료구조)로 만들어 줍니다. 그리고 우리는 스프레드 연산자를 사용해서 만들어진 위젯을 하나씩 꺼내면서 List를 훌뿌려줍니다.



01 _ 7 내 근처 화면 만들기

해당 소스 코드는 https://github.com/flutter-coder/flutter-ui-book2/tree/master/carrot_market_ui/carrot_market_ui_07에 공개되어 있습니다.

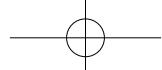


◆ 내 근처 화면

이번 프로젝트 앱의 마지막 화면입니다. 이번 장에서는 `TextField` 위젯을 만드는 방법을 알아보고 `Wrap` 위젯과 수평 방향으로 스크롤 되는 위젯을 사용해서 콘텐츠를 만들어봅시다.

작업 순서

- ① 모델 클래스 및 샘플 데이터 만들기
- ② `NearMeScreen` 위젯 기본 코드 입력하기
- ③ `TextField` 위젯 만들기
- ④ 수평 방향으로 스크롤 되는 위젯 만들기
- ⑤ `Wrap` 위젯 사용해 보기
- ⑥ `StoreItem` 위젯 만들기
- ⑦ `NearMeScreen` 위젯 완성하기



모델 클래스 및 샘플 데이터 만들기

내 근처 화면에 사용할 모델 클래스와 샘플 데이터들을 만들어 봅시다. lib / models 폴더에 recommend_store.dart 파일을 만들어 주세요.

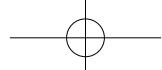
① 모델 클래스 만들기

lib / models / recommend_store.dart

```
class RecommendStore {  
    String storeName;  
    String location;  
    String description;  
    int commentCount;  
    int likeCount;  
    String comment;  
    String commentUser;  
    List storeImages;  
  
    RecommendStore({  
        required this.storeName,  
        required this.location,  
        required this.description,  
        required this.commentCount,  
        required this.likeCount,  
        required this.comment,  
        required this.commentUser,  
        required this.storeImages,  
    });  
}  
// 샘플데이터  
...  
// 샘플데이터  
...
```

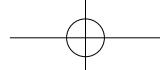
② 샘플 데이터 만들기

github에서 받은 소스코드에서 carrot_market_ui_07에 recommend_store.dart 파일의 코드를 복사해서 사용할 수 있습니다.



lib / models / recommend_store.dart

```
// 샘플데이터
final List searchKeyword = ['인테리어', '학원', '이사', '카페', '용달', '네일', '에어콘'];
// 샘플데이터
List<RecommendStore> recommendStoreList = [
    RecommendStore(
        storeName: '네일가게',
        location: '좌동',
        description: '꼼꼼한시술로 유지력높은 네일샵입니다. 좌동에 위치하고 있습니다.',
        commentCount: 1,
        likeCount: 8,
        commentUser: '이엘리아님',
        comment: '너무편하게 시술해주셔서 잠들었었네요 직모에 짧은 눈썹이라 펌이 잘 안되는 타입인데 너무 이쁘게 됐네요',
        storeImages: [
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_1_1.jpg?raw=true',
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_1_2.jpg?raw=true',
        ]),
    RecommendStore(
        storeName: '아미아미주먹밥',
        location: '우동',
        description: '2012년 오픈한 해운대도서관 분관쪽에 위치하고 있습니다.',
        commentCount: 2,
        likeCount: 2,
        commentUser: '둘리님',
        comment: '도서관이 근처라 시험기간마다 이용하는데 너무 좋습니다.',
        storeImages: [
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_2_1.jpg?raw=true',
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_2_2.jpg?raw=true',
        ]),
    RecommendStore(
        storeName: '영어원어민 논술',
        location: '중동',
        description: '원어민 영어 고급논술&디베이트&스피치 전문',
        commentCount: 7,
        likeCount: 1,
        commentUser: 'kkglo님',
        comment: '저희 아들은 학원 주입식이 아닌 살아있는 영어 수업을 할 수 있어서 너무 좋네요',
        storeImages: [
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_3_1.jpg?raw=true',
            'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_3_2.jpg?raw=true',
        ]),
    RecommendStore(
        storeName: '밸레빙/코인워시 우동점',
        location: '우동',
        description: '밸래방 / 크린토비아 코인워시 우동점 신설했습니다. 많은 이용 바랍니다.'
    )
]
```



```

commentCount: 11,
likeCount: 5,
commentUser: '코인님',
comment: '처음 방문때 건조기 무료로 서비스 해주셔서 너무 감사 하네요. 앞으로 자주 이용 합니다.',
storeImages: [
  'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_4_1.jpg?raw=true',
  'https://github.com/flutter-coder/ui_images/blob/master/carrot_store_4_2.jpg?raw=true',
])
];

```

NearMeScreen 위젯 기본 코드 입력하기

내 근처 화면은 부모 위젯(main_screens.dart) IndexedStack 위젯의 자식 항목의 index 2번째 항목입니다.

앱 뼈대 만들기에서 작업했던 near_me_screen.dart 파일을 열고 기본 코드를 입력해 봅시다.

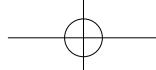
lib / screens / near_me / near_me_screen.dart

```

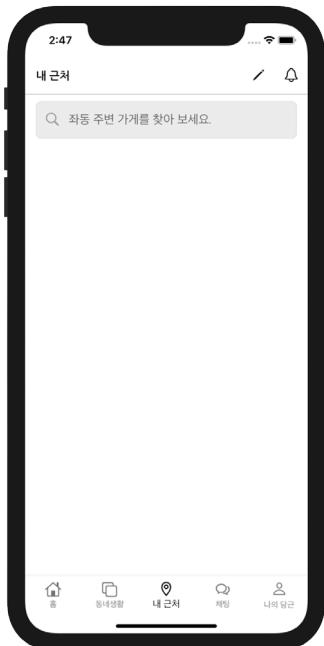
class NearMeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('내 근처'),
        actions: [
          IconButton(icon: const Icon(CupertinoIcons.pencil), onPressed: () {}),
          IconButton(icon: const Icon(CupertinoIcons.bell), onPressed: () {}),
        ],
        bottom: appBarBottomLine(),
      ),
      body: ListView(
        children: [
          const SizedBox(height: 10),
          // ❶ ...
        ],
      );
    }
}

```

❶ 앞으로 만들게 될 위젯들을 해당 자리에 추가할 예정입니다.



TextField 위젯 만들기

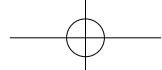


◆ TextField 위젯

여기서는 TextField의 사용방법 전에 꾸미는 방법을 학습하고 “Chapter2 마켓컬리 UI 만들어보기”에서 좀 더 자세하게 학습하겠습니다. lib / screens / near_me 폴더에 components 폴더를 생성하고 search_text_field.dart 파일을 만들어 주세요.

lib / screens / near_me / components / search_text_field.dart

```
class SearchTextField extends StatelessWidget {  
    @override  
    Widget build(BuildContext context) {  
        return Container(  
            child: TextField(  
                cursorColor: Colors.grey,  
                // ❶  
                decoration: InputDecoration(  
                    // ❷  
                    disabledBorder: _buildOutlineInputBorder(),  
                    // ❸  
                    enabledBorder: _buildOutlineInputBorder(),  
                    // ❹  
                    focusedBorder: _buildOutlineInputBorder(),  
                    filled: true,  
                    fillColor: Colors.grey[200],  
                    // ❺  
                ),  
            ),  
        );  
    }  
}
```



```

prefixIcon: Icon(
  CupertinoIcons.search,
  color: Colors.grey,
),
// ⑥
contentPadding:
  const EdgeInsets.only(left: 0, bottom: 15, top: 15, right: 0),
hintText: '좌동 주변 가게를 찾아 보세요.',
hintStyle: TextStyle(fontSize: 18),
),
),
),
);
}
}

OutlineInputBorder _buildOutlineInputBorder() {
return OutlineInputBorder(
// ⑦
borderSide: const BorderSide(width: 0.5, color: Color(0xFFD4D5DD)),
// ⑧
borderRadius: BorderRadius.circular(8.0),
);
}
}

```

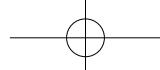
- ① 텍스트 필드의 장식하는 속성입니다. decoration 키의 데이터 타입은 테두리, 레이블, 아이콘 및 스타일을 설정할 수 있는 InputDecoration 객체를 사용합니다.
- ② TextField를 비 활성 상태, enable 속성을 false로 설정했을 때의 border 스타일을 지정하는 속성입니다.
- ③ TextField의 enable 속성을 true로 설정했을 때의 border 스타일을 지정하는 속성입니다.
- ④ TextField에 포커스가 왔을 때 border 스타일을 지정합니다.
- ⑤ 글이 입력되는 content 영역 앞에 Icon 위젯을 사용하기 위해 지정합니다.
- ⑥ 글 입력 영역을 감싸고 있는 container의 padding 값을 지정합니다.
- ⑦ BorderSide 객체를 이용해 border의 색상과 두께를 설정합니다.
- ⑧ TextField Border 영역에 곡선을 주기 위해 사용합니다.

완성한 SearchTextField 위젯을 NearMeScreen 위젯의 ListView 안에 넣어 줍니다.

```
lib / screens / near_me / near_me_screen.dart

... 생략
body: ListView(
  children: [
    const SizedBox(height: 10),
    // ①
    Padding(
      padding: const EdgeInsets.symmetric(horizontal: 16),

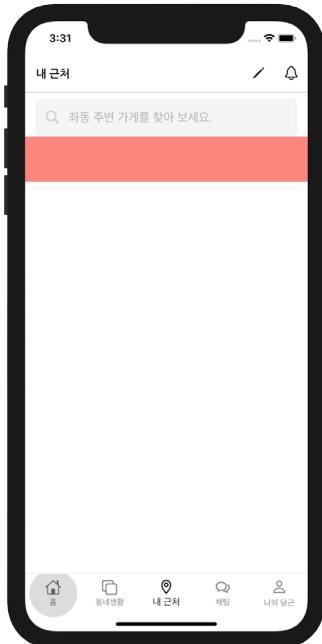
```



```
        child: SearchTextField(),
    ),
],
),
... 생략
```

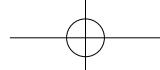
- ① Padding 위젯을 추가하고 child 속성에 SearchTextField 추가해 주세요.

수평 방향으로 스크롤 되는 위젯 만들기

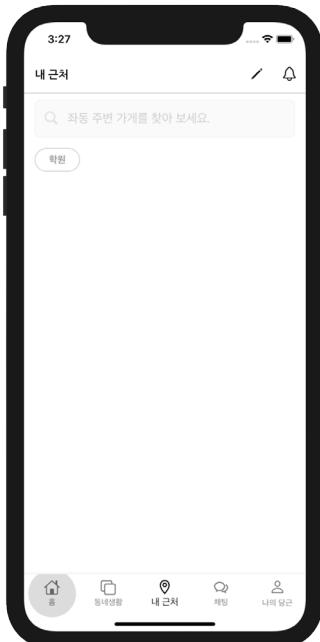


◆ 수평 스크롤 위젯

수평 스크롤 위젯의 부모 위젯은 수직 방향으로 스크롤 되는 ListView입니다. ListView는 수직으로 스크롤 될 때 해당 Axis(방향)으로 나열되는 위젯 각각의 높이를 자식 위젯에게 위임하는 특성이 있습니다. 그렇기 때문에 수평 방향 ListView에 높이를 주지 않으면 높이가 0이기 때문에 화면에 보이지 않습니다. 수평 방향 ListView에 꼭 고정 height 값을 지정해야 합니다. 그럼 다음과 같이 작업을 진행해 봅시다.



① RoundBorderText 위젯 만들기



◆ RoundBorderText 위젯

수평 스크롤 영역 안에 들어갈 위젯을 만들어봅시다.

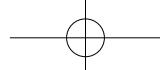
components 폴더에 round_border_text.dart 파일을 만들어 주세요.

lib / screens / near_me / components / round_border_text.dart

```
class RoundBorderText extends StatelessWidget {
    RoundBorderText({Key? key, required this.title, required this.position})
        : super(key: key);
    // ❶
    final String title;
    // ❷
    final int position;

    @override
    Widget build(BuildContext context) {
        // ❸
        var paddingValue = position == 0 ? 16.0 : 8.0;
        return Padding(
            padding: EdgeInsets.only(left: paddingValue),
            child: Container(
                // ❹
                padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 8.0),
                // ❺
                decoration: BoxDecoration(

```



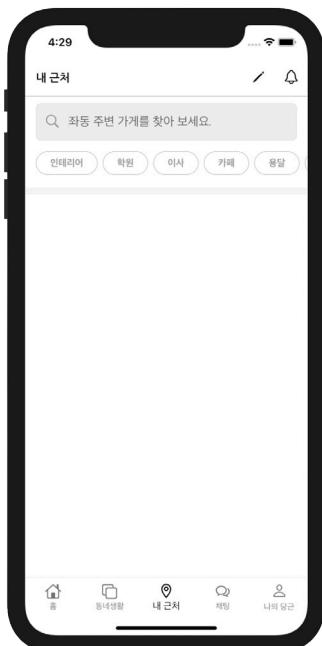
```

        borderRadius: BorderRadius.circular(18.0),
        border: Border.all(width: 0.5, color: Colors.grey),
      ),
      child: Text(title,
        textAlign: TextAlign.center,
        style: TextStyle(fontSize: 14, fontWeight: FontWeight.bold)),
    ),
);
}
}

```

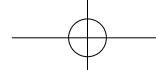
- ❶ 위젯의 title로 사용할 String 타입의 값을 넘겨받습니다.
- ❷ 위젯의 position 값을 넘겨받게 합니다.
- ❸ 위젯의 위치가 0번째일 경우 왼쪽 padding 값을 16.0 아닐 경우 8.0으로 설정합니다. 우리가 만든 SearchTextField 위젯과 왼쪽 정렬을 맞추기 위해 사용합니다.
- ❹ Container와 Text 위젯 사이의 padding 값을 설정합니다.
- ❺ Container의 border 속성과 borderRadius 속성을 사용해서 위젯을 꾸며 줍니다.

② 수평 방향으로 스크롤 되는 위젯 완성하기



◆ 수평 스크롤 위젯

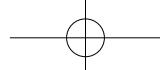
near_me_screen.dart 파일에서 ListView.builder를 활용하여 수평으로 스크롤 되는 위젯들을 나열해 봅니다. 파일을 열고 다음과 같이 코드를 추가해 주세요.



lib / screens / near_me / near_me_screen.dart

```
... 생략
// ❶
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 16),
  child: SearchTextField(),
),
// ❷
SizedBox(
  height: 66,
  child: ListView.builder(
    // ❸
    scrollDirection: Axis.horizontal,
    itemCount: searchKeyword.length,
    itemBuilder: (context, index) {
      return Center(
        child: RoundBorderText(
          title: searchKeyword[index],
          // ❹
          position: index,
        ),
      );
    },
  ),
),
// ❺
Divider(
  color: Colors.grey[100],
  thickness: 10.0,
),
// ❻
... 생략
```

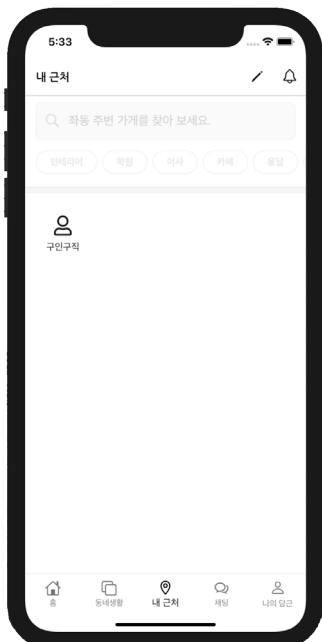
- ❶ 우리가 만든 SearchTextField 위젯입니다. 코드 위치를 알리기 위해 표시하였습니다.
- ❷ SizedBox 위젯을 활용해서 수평 스크롤 되는 영역의 높이 값을 지정해 주어야 합니다.
- ❸ 스크롤 방향 scrollDirection: Axis.horizontal 사용해서 방향을 지정합니다.
- ❹ index 값을 넘겨줍니다. 0번째 위젯임을 알리기 위해 사용합니다. 수평 스크롤 영역(SizedBox) 밖에서 Padding을 사용하게 되면 스크롤 영역이 잘리게 되므로 자연스럽지 못한 UI가 만들어집니다.
- ❺ Divider를 사용해 색상과 굵기 설정을 합니다.
- ❻ 잠시 후 만들게 될 Wrap 위젯의 위치입니다.



Wrap 위젯 사용해 보기

배치하고자 하는 방향에 공간이 부족할 때는 Wrap을 활용할 수 있습니다. Wrap 위젯은 자식을 Row나 Column으로 배치할 수 있고 배치할 공간이 부족해지면 자식 위젯을 다음 줄에 배치합니다. 반응형 웹을 만들 때도 활용할 수 있습니다.

① BottomTitleIcon 위젯 만들기



◆ BottomTitleIcon 위젯

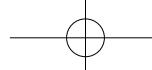
Wrap 위젯 안에 사용될 위젯을 만들어봅시다. lib / screens / near_me / components 폴더에 bottom_title_icon.dart 파일을 만들어 주세요.

```
lib / screens / near_me / components / bottom_title_icon.dart
```

```
class BottomTitleIcon extends StatelessWidget {
    final IconData iconData;
    final String title;

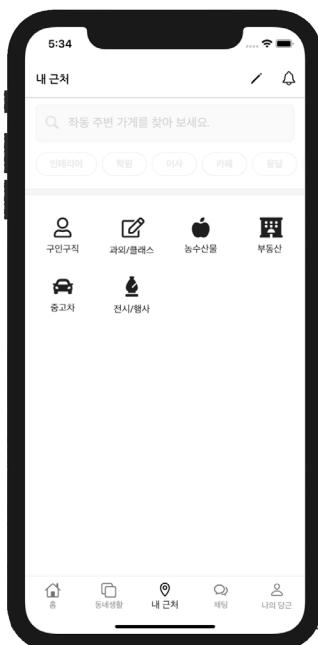
    const BottomTitleIcon(
        {Key? key, required this.iconData, required this.title})
        : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Container(
            width: 80,
            child: Column(
```



```
        children: [
            Icon(iconData, size: 30),
            Padding(
                padding: const EdgeInsets.only(top: 8),
                child: Text(
                    title,
                    style: TextStyle(fontSize: 14, color: Colors.black),
                ),
            ),
            ],
        );
    );
}
```

② Wrap 위젯 완성하기

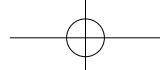


◆ Wrap 위젯

near me screen.dart 파일에서 작업합니다. 파일을 열고 다음과 같이 코드를 추가해 주세요.

lib / screens / near_me / near_me_screen.dart

```
... 생략  
// ⑥  
Padding(  
  padding: const EdgeInsets.only(left: 16, top: 30),  
  child: Wrap(  
    alignment: WrapAlignment.start,  
    spacing: 22.0,
```



```

runSpacing: 30,
children: [
  const BottomTitleIcon(
    title: '구인구직', IconData: FontAwesomeIcons.user),
  const BottomTitleIcon(
    title: '과외/클래스', IconData: FontAwesomeIcons.edit),
  const BottomTitleIcon(
    title: '농수산물', IconData: FontAwesomeIcons.appleAlt),
  const BottomTitleIcon(
    title: '부동산', IconData: FontAwesomeIcons.hotel),
  const BottomTitleIcon(
    title: '중고차', IconData: FontAwesomeIcons.car),
  const BottomTitleIcon(
    title: '전시/행사', IconData: FontAwesomeIcons.chessBishop)
],
),
),
const SizedBox(height: 50),
... 생략

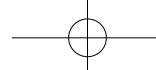
```

- ⑥ Wrap 위젯의 왼쪽과 상단에 padding 값을 사용하고 속성 alignment를 사용하여 정렬을 시작 위치로 설정합니다. spacing 속성은 다음 위젯과의 공간을 띄우고 runSpacing 속성은 다음 줄부터 시작하는 공간을 설정할 수 있습니다.

StoreItem 위젯 만들기



◆ StoreItem 위젯



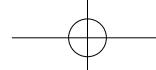
components 폴더에 store_item.dart 파일을 만들어 주세요.

```
lib / screens / near_me / components / store_item.dart

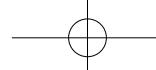
class StoreItem extends StatelessWidget {
    final RecommendStore recommendStore;

    const StoreItem({Key? key, required this.recommendStore}) : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Container(
            decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(10),
                border: Border.all(width: 0.3, color: Colors.grey)),
            width: 289,
            child: Column(
                mainAxisAlignment: MainAxisAlignment.start,
                children: [
                    Row(
                        children: [
                            buildClipRRect(topLeft: 10),
                            const SizedBox(width: 2),
                            buildClipRRect(topRight: 10),
                        ],
                    ),
                    Padding(
                        padding: const EdgeInsets.all(16),
                        child: Column(
                            mainAxisAlignment: MainAxisAlignment.start,
                            children: [
                                Text.rich(
                                    TextSpan(
                                        children: [
                                            TextSpan(
                                                text: '${recommendStore.storeName} ',
                                                style: textTheme().headline1),
                                            TextSpan(text: '${recommendStore.location}'),
                                        ],
                                    ),
                                ),
                                const SizedBox(height: 8),
                                Text(
                                    '${recommendStore.description}',
                                    maxLines: 1,
                                    overflow: TextOverflow.ellipsis,
                                    style: textTheme().subtitle1,
                                )
                            ],
                        ),
                    ),
                ],
            ),
        );
    }
}
```



```
),
const SizedBox(height: 8),
Text.rich(
  TextSpan(
    children: [
      TextSpan(
        text: '후기 ${recommendStore.commentCount}',
        style: TextStyle(fontSize: 15, color: Colors.blue)),
      TextSpan(
        text: ' · 관심 ${recommendStore.likeCount}',
        style: textTheme().subtitle1,
      ),
    ],
  ),
),
Expanded(
  child: Container(
    margin: const EdgeInsets.only(left: 16, right: 16, bottom: 16),
    padding: const EdgeInsets.all(10),
    decoration: BoxDecoration(
      color: Colors.grey[200],
      borderRadius: BorderRadius.circular(10)),
  child: Text.rich(
    TextSpan(
      children: [
        TextSpan(
          text: '${recommendStore.commentUser},',
          style: TextStyle(
            fontSize: 13,
            color: Colors.black,
            fontWeight: FontWeight.bold)),
        TextSpan(
          text: '${recommendStore.comment}',
          style: TextStyle(fontSize: 12, color: Colors.black)),
      ],
    ),
    maxLines: 2,
    overflow: TextOverflow.ellipsis,
  ),
),
),
],
),
```



```

),
);
}

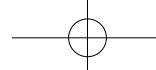
ClipRRect buildClipRRect({double topLeft = 0, double topRight = 0}) {
    return ClipRRect(
        borderRadius: BorderRadius.only(
            topLeft: Radius.circular(topLeft),
            topRight: Radius.circular(topRight)),
        child: Image.network(
            recommendStore.storeImages[0],
            width: 143,
            height: 100,
            fit: BoxFit.cover,
        ),
    );
}
}

```

NearMeScreen 위젯 완성하기



◆ NearMeScreen위젯



StoreItem 위젯을 활용해서 코드를 완성해 봅시다. near_me_screen.dart 파일을 열고 코드를 추가해 주세요.

```
lib / screens / near_me / near_me_screen.dart
...
... 생략
// ❸
Padding(
  padding: const EdgeInsets.only(left: 16.0),
  child: Text('이웃들의 추천 가게', style: textTheme().headline2),
),
const SizedBox(height: 20),
// ❹
Container(
  height: 288,
  child: ListView.builder(
    scrollDirection: Axis.horizontal,
    itemCount: recommendStoreList.length,
    itemBuilder: (context, index) {
      return Padding(
        padding: EdgeInsets.only(left: 16),
        child: StoreItem(
          recommendStore: recommendStoreList[index],
        ),
      );
    },
),
),
// ❺
SizedBox(height: 40)
... 생략
```

❸ Text 위젯과 Padding 위젯을 사용합니다.

❹ 이번에는 수평 방향으로 스크롤 하는 위젯의 영역을 Container로 사용하고 우리가 만든 StoreItem 위젯을 활용합니다.

❺ 스크롤을 최하단으로 내렸을 경우 아래의 여유 공간을 확보하기 위해 사용합니다.