Flutter - 뽀모도로 앱 따라 만들기

상태	플러터

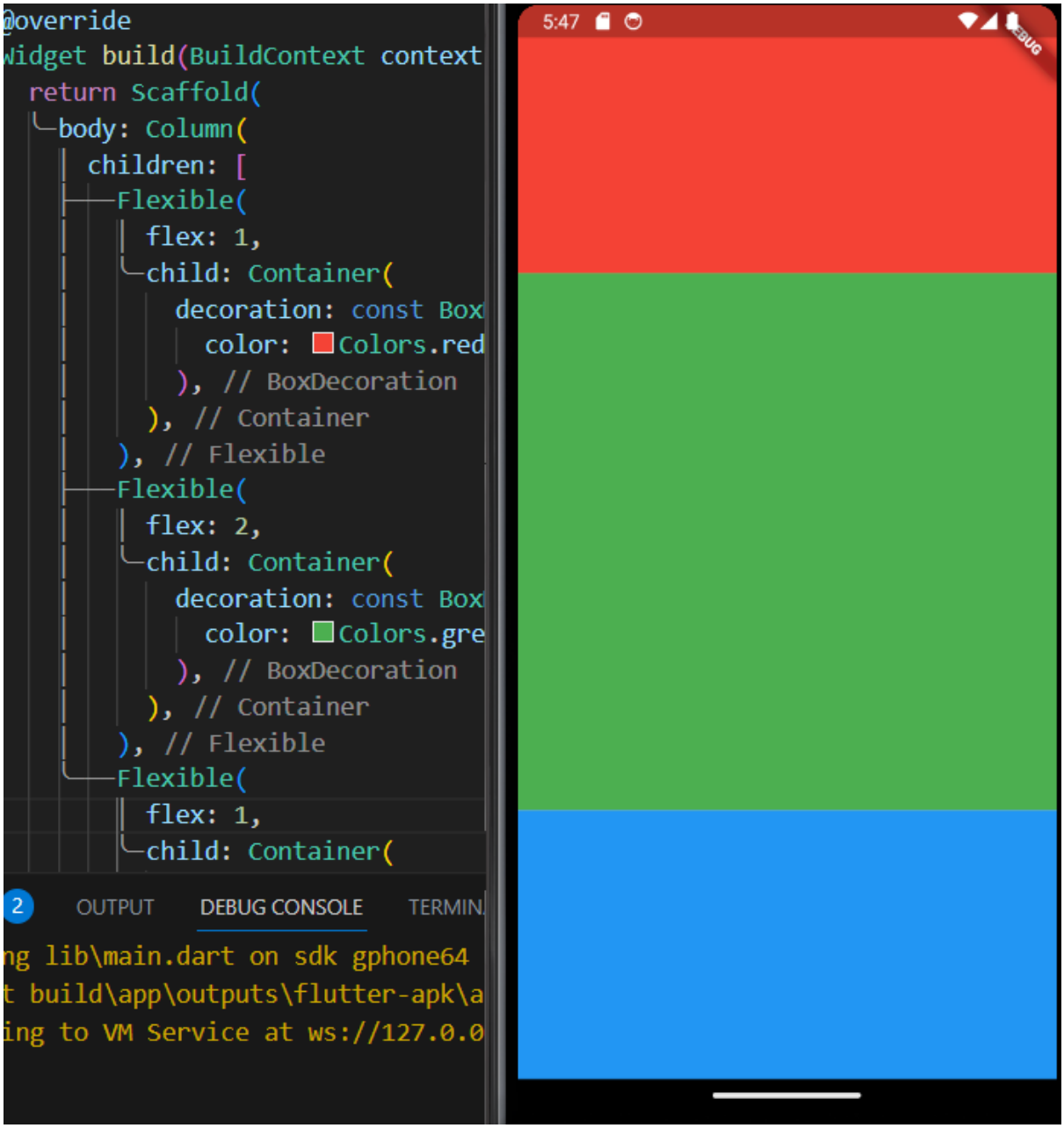
*

```
return Scaffold(
      body: Column(
        children: [Flexible(child: child)],
      ),
    );
```

Flexible의 역할 : 하드코딩되는 값을 만들게 해준다. 높이 200, 너비 100픽셀 뭐 이런게 아니라, UI를 비율에 기반하여 유연하게 만들 수 있게 해준다. 이걸 원해왔어…

```
children: [
          Flexible(
            flex: 1,
            child: Container(
              decoration: const BoxDecoration(
                color: Colors.red,
              ),
            ),
          ),
          Flexible(
            flex: 2,
            child: Container(
              decoration: const BoxDecoration(
                color: Colors.green,
              ),
            ),
          ),
          Flexible(
            flex: 1,
            child: Container(
              decoration: const BoxDecoration(
                color: Colors.blue,
              ),
            ),
          ),
        ],
```

여기서 flex가 비율!



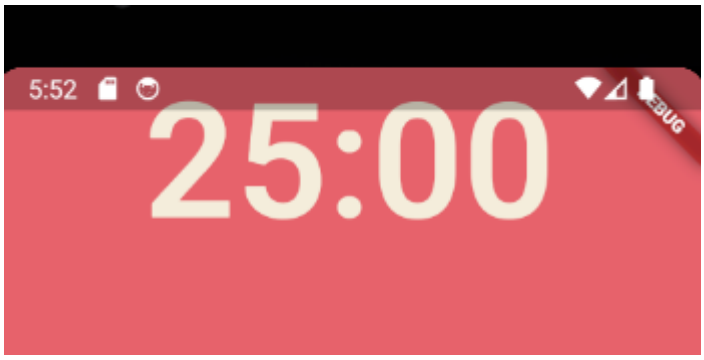
지금 보면 빨강 파랑이 1만큼의 비율을, 초록이 2만큼의 비율을 가지고 있다.

{ 전체 화면 / (1 + 2 + 1) } * 2 만큼의 비율을 초록이 차지하고 있는 거겠지~

지난 날 복습

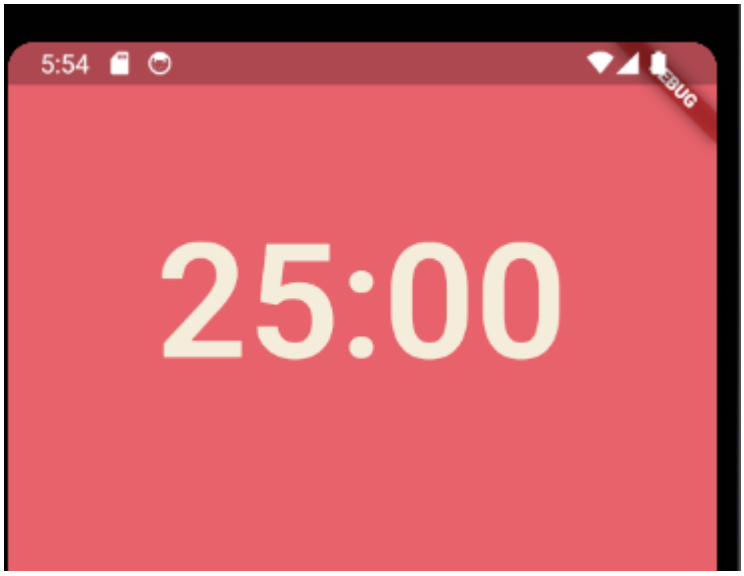
context를 타고타고 색깔들을 가져옵니다.

```
urn Scaffold(
ackgroundColor: Theme.of(context).colorScheme.background,
ody: Column(
 children: [
  Flexible(
    flex: 1,
    child: Container(    Unnecessary instance of 'Container'.⏎Tr
      child: Text(
        "25:00",
        style: TextStyle(
          color: Theme.of(context).cardColor,
          fontSize: 89,
          fontWeight: FontWeight.w600
```



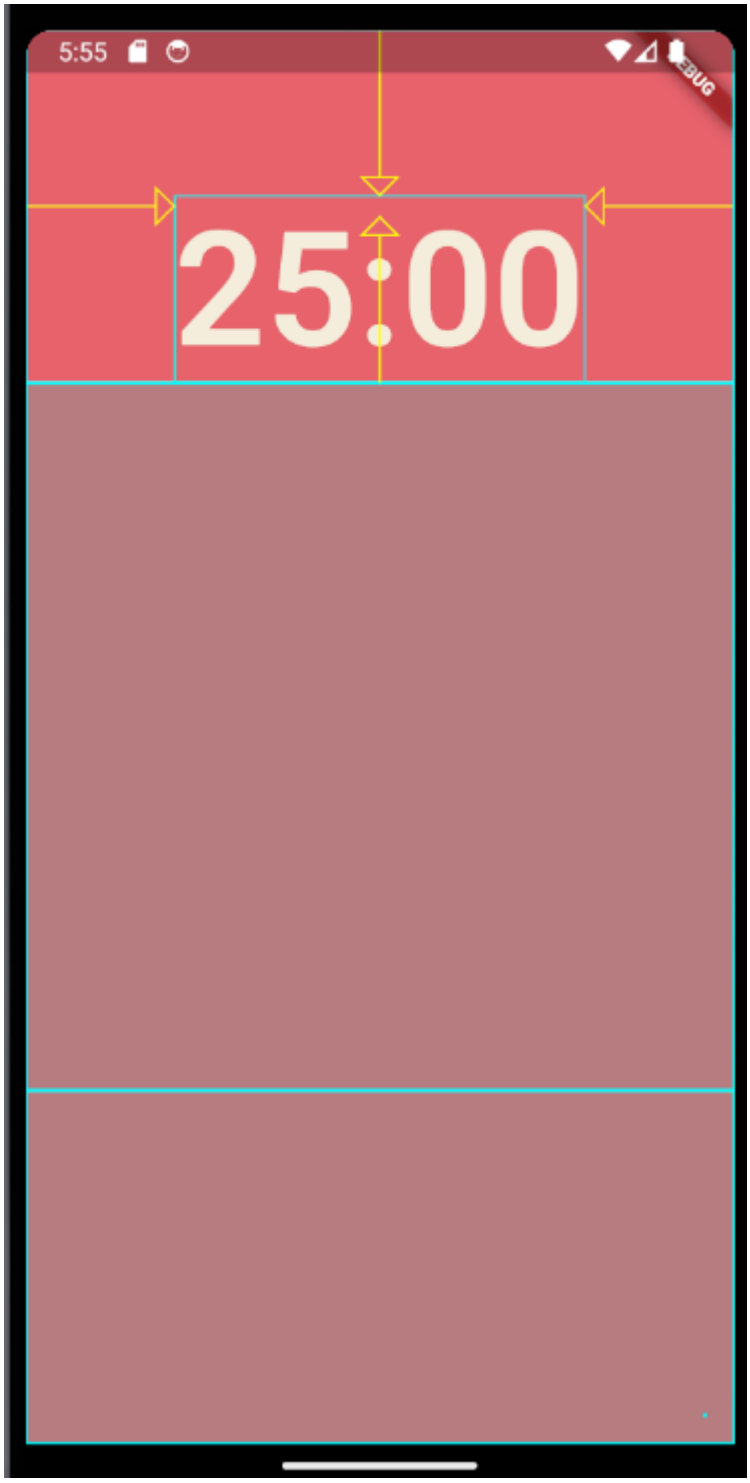
그런데 이런… 윗부분이 가림. 이거는 Container의 alignment property로 해결할 수 있다.

```
child: Container(
              alignment: Alignment.bottomCenter,
              child: Text(
                "25:00",
```



우와~~~~~

이게… 어떻게 이런거지? 했더니 ㅋㅋㅋㅋ 이미 flexible로 나눠놔서 이렇게 됐던 거였어 아주 좋아요~~~~~



▼ 두 번째 플렉스. 버튼 쓰는 법을 복습합니다.

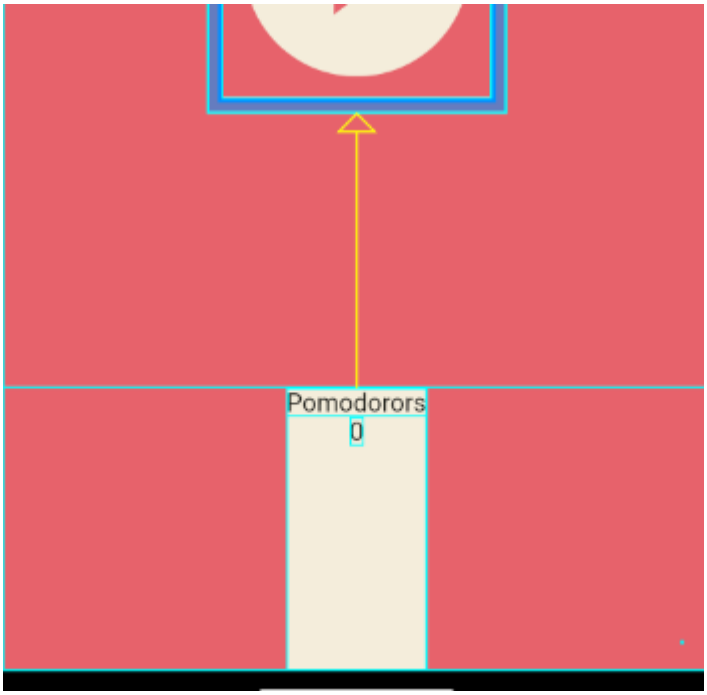
```
Flexible(
            flex: 3,
```

```
            child: Center(
              child: IconButton(
                iconSize: 150,
                color: Theme.of(context).cardColor,
                onPressed: () {},
                icon: const Icon(
                  Icons.play_circle_filled_outlined,
                ),
              ),
            ),
          ),
```

세 번째 플렉스 감.

```
Flexible(
            flex: 1,
            child: Container(
              decoration: BoxDecoration(
                color: Theme.of(context).cardColor,
              ),
              child: const Column(
                children: [
                  Text("Pomodorors"),
                  Text("0"),
                ],
              ),
            ),
          ),
```

이렇게만 했더니



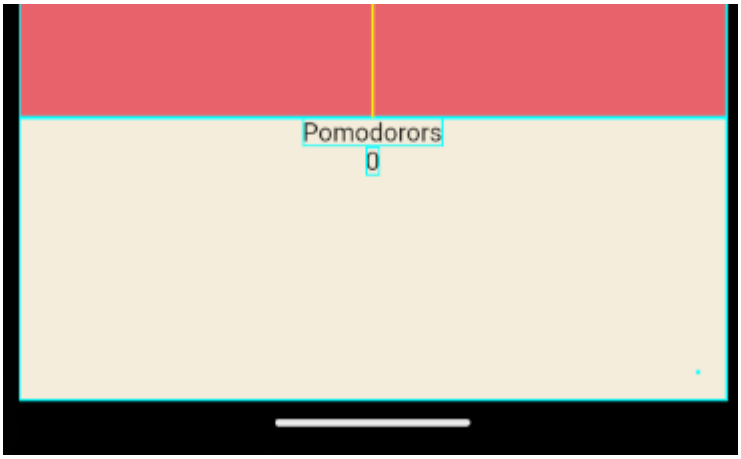
옹졸해…

저 공간을 다 차지하게 하는 법은? container를 row로 감싸준다!!

그리고 row에 감싸진 container를 한 번 더 (row 제외해야함! 안에 있는 것만!!)

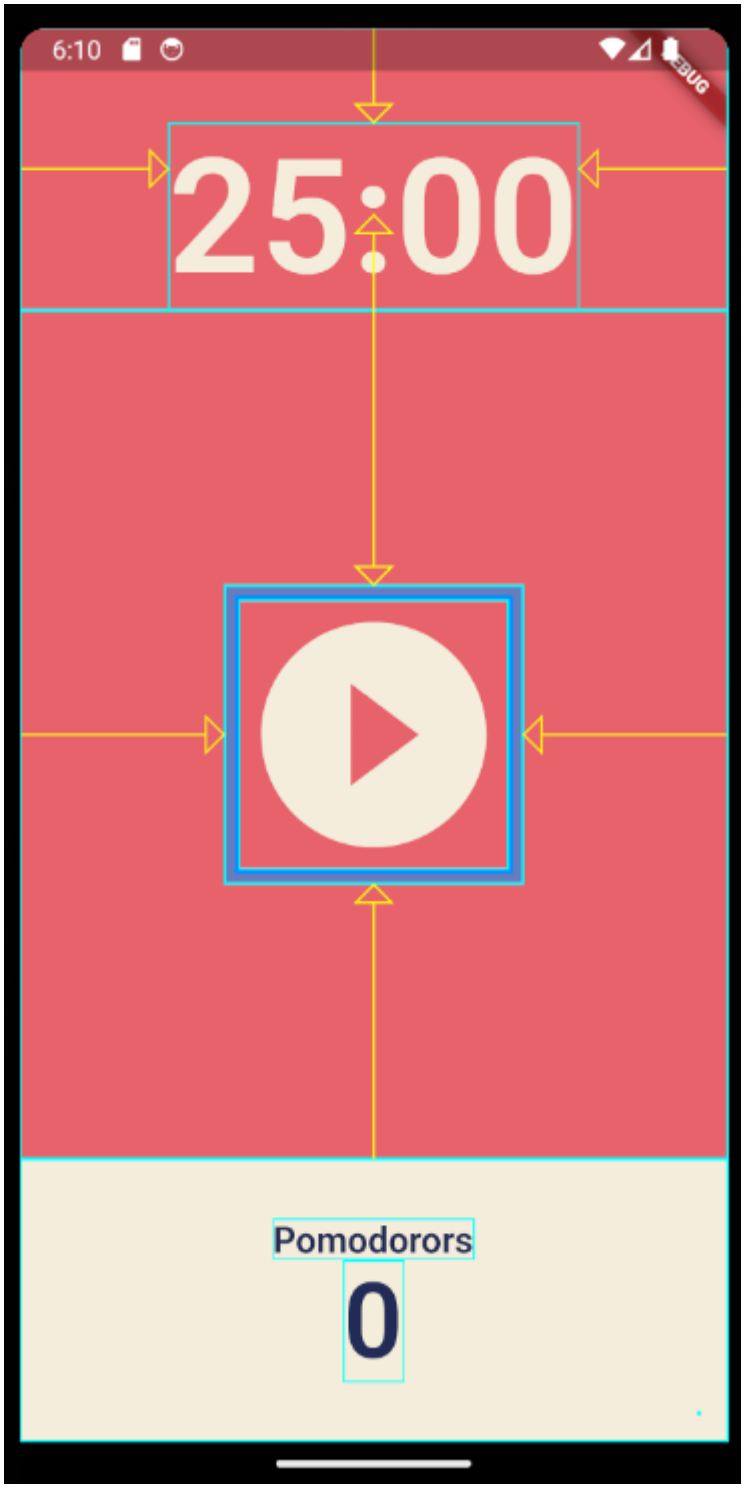
expanded로 감싸준다.

Row(Expanded(Container))



*

플러터 업뎃되면서 headline1이 displayLarge로 바뀜. 근데 저장하면 자동으로 수정해줘야가.



▼ 여기까지 코드

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

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
```

```
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Theme.of(context).colorScheme.background,
      body: Column(
        children: [
          Flexible(
            flex: 1,
            child: Container(
              alignment: Alignment.bottomCenter,
              child: Text(
                "25:00",
                style: TextStyle(
                  color: Theme.of(context).cardColor,
                  fontSize: 89,
                  fontWeight: FontWeight.w600,
                ),
              ),
            ),
          ),
          Flexible(
            flex: 3,
            child: Center(
              child: IconButton(
                iconSize: 150,
                color: Theme.of(context).cardColor,
                onPressed: () {},
                icon: const Icon(
                  Icons.play_circle_filled_outlined,
                ),
              ),
            ),
          ),
          Flexible(
            flex: 1,
            child: Row(
              children: [
                Expanded(
                  child: Container(
                    decoration: BoxDecoration(
                      color: Theme.of(context).cardColor,
                    ),
                    child: Column(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Text(
                          "Pomodorors",
                          style: TextStyle(
                            fontSize: 20,
                            fontWeight: FontWeight.w600,
                            color:
                                Theme.of(context).textTheme.displayLarge!.color,
                          ),
                        ),
```

```
                        Text(
                          "0",
                          style: TextStyle(
                            fontSize: 58,
                            fontWeight: FontWeight.w600,
                            color:
                                Theme.of(context).textTheme.displayLarge!.color,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
              ),
            ],
          ),
        ),
      ],
    ),
  );
 }
}
```

Timer

timer는 dart의 표준 라이브러리에 포함되어있고, 이를 통해 정해진 간격에 한 번씩 함수를 실행할 수 있다.

```
late Timer timer;
```

이렇게 먼저 써줘야한다. 나중에~, property를 사용하기 전에 반드시 초기화한다고 약속해 주는 것이다.

```
void onStartPressed() {
    timer = Timer.periodic(duration, () { })
}
```

(함수는 임의로 적은 것)

duration은 주기이다. periodic을 쓰면 이 마다 한 번씩 실행시킨다고 하는 것.

```
void onTick(Timer timer) {
    setState(() {
```

```
      totalSeconds -= 1;
    });
  }

  void onStartPressed() {
    timer = Timer.periodic(
      const Duration(seconds: 1),
      onTick,
    );
  }
```

1초마다 실행한다는 onTick함수 저거… 사용할 때는 괄호를 넣지 않는다! 기억하기 괄호는 지금 당장 실행한다는 거니까.

그리고 Timer timer 써주는 것 잊지말것~~~ ; 도!!

버튼을 누를 때마다 실행되는 함수와 아이콘을 변경해보아요

```
Flexible(
            flex: 3,
            child: Center(
              child: IconButton(
                iconSize: 150,
                color: Theme.of(context).cardColor,
                onPressed: isRunning ? onPausePressed : onStartPressed,
                icon: Icon(
                  isRunning
                      ? Icons.pause_circle_outline
                      : Icons.play_circle_outlined,
                ),
              ),
            ),
          ),
```

메서드도 새로 만들었어요~

```
void onStartPressed() {
    timer = Timer.periodic(
      const Duration(seconds: 1),
      onTick,
    );
    setState(() {
      isRunning = true;
    });
  }

  void onPausePressed() {
    timer.cancel();
```

```
    setState(() {
      isRunning = false;
    });
  }
```

timer.cancel 하면 타이머가 멈춤!

이제 카운트가 다 세어지고 0이 되면 뽀모도로 몇 번 돌았는지 세는 카운트가 1 올라가고, 또 카운트가 1500으로 초기화되게 해보자.

```
void onTick(Timer timer) {
    if (totalSeconds == 0) {
      setState(() {
        // 0이 되었으니 초기화~ 하고 뽀모도로 1 증가
        totalPomodorors += 1;
        isRunning = false;
        totalSeconds = twentyFiveMinutes;
      });
      timer.cancel();
    } else {
      setState(() {
        totalSeconds -= 1;
      });
    }
  }
```

그냥 if문 하나 쓰세용

뽀모도로 25분 같은 변수… 개발할 때는 10초로 해두고 그러잖아, 그거 실수 안하려면 아래처럼 아예 상수 변수를 만들어 두는 게 좋다.

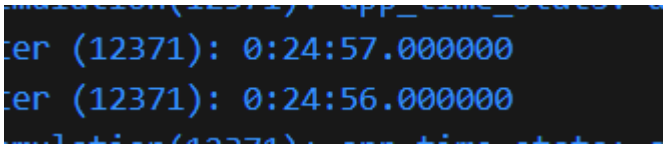
```
static const twentyFiveMinutes = 1500;
int totalSeconds = twentyFiveMinutes; // 25분을 초로 환산 1500
```

이제 이 1500초를 분 단위로 보여줘볼까~~

```
String format(int seconds) {
    // 초를 분으로 포맷팅해서 보여주자.
    var duration = Duration(seconds: seconds);
    print(duration);
```

```
    return "$seconds";
  }
```

duration을 사용하니까

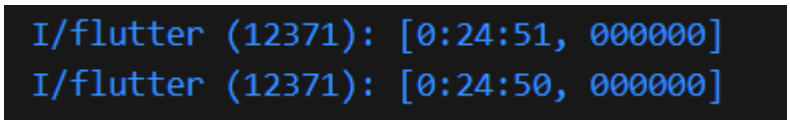


이렇게 나오네요~~

그럼 이걸 변형해봅시다 우리가 원하는 방식으로 출력할 수 있게.

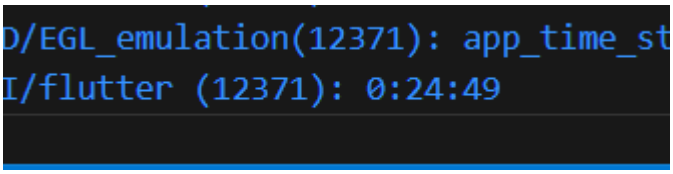
```
duration.toString().split(".")
```

이걸 쓰면 됨!



쨘 그럼 첫번째를 추출하면 되겠네.

```
duration.toString().split(".").first
```



first라니… 인덱스로는 못 하나?

```
duration.toString().split(".")[0]
```

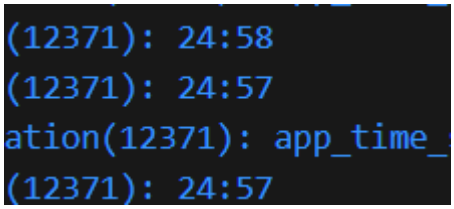
되네용 굳

그럼 앞에 0: 를 떼어버리자

```
duration.toString().split(".")[0].substring(2, 7)
```

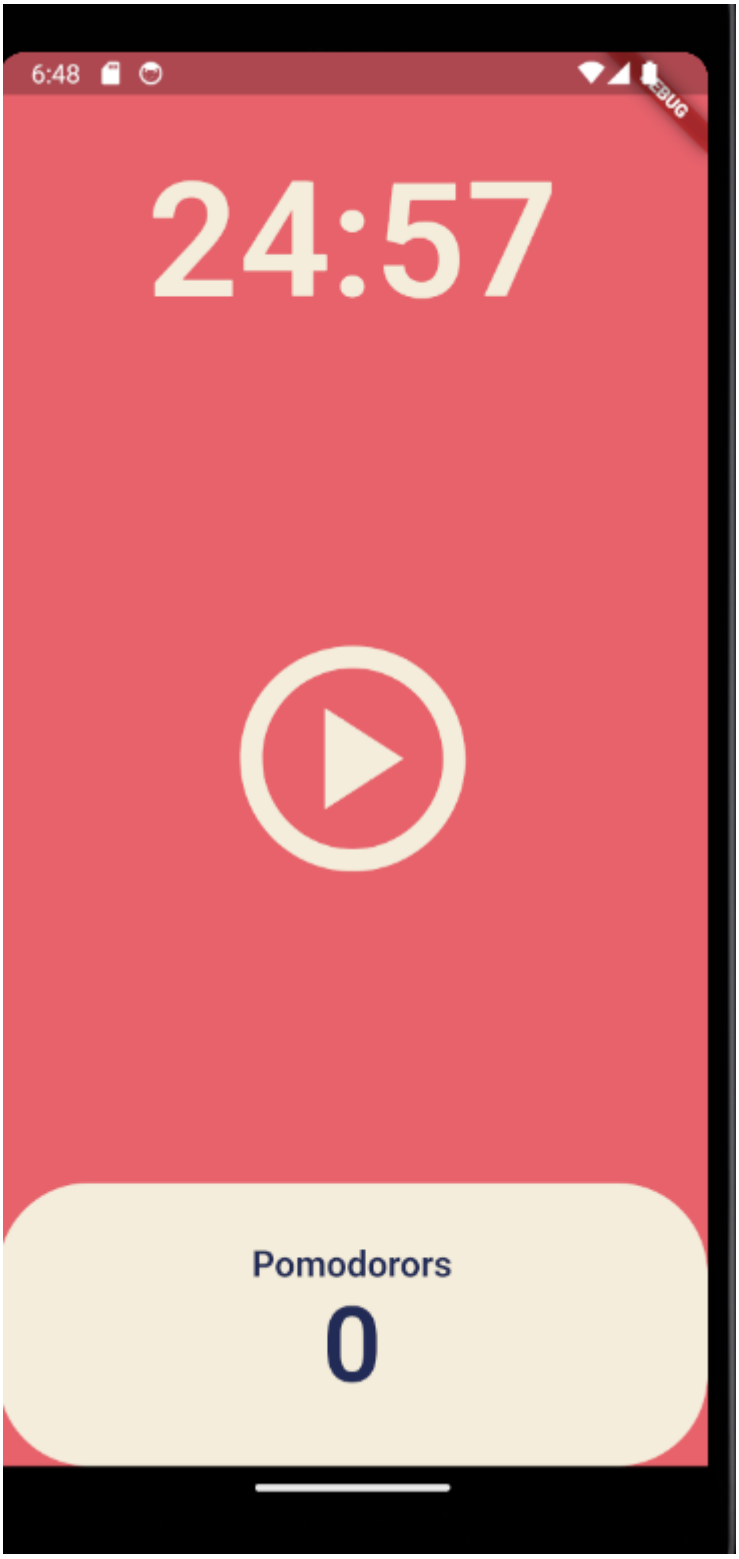
두번째 인덱스부터 7번째 인덱스까지 다 잘라

근데 걍 이러면 split 할 필요 없는거 아닌가? 그치만… 이런거 저런거 보여주고 싶으셨겠지




```
String format(int seconds) {
    // 초를 분으로 포맷팅해서 보여주자.
    var duration = Duration(seconds: seconds);
    return duration.toString().split(".")[0].substring(2, 7);
  }
```

```
child: Text(
                format(totalSeconds),
                style: TextStyle(
                  color: Theme.of(context).cardColor,
                  fontSize: 89,
                  fontWeight: FontWeight.w600,
                ),
              ),
```



짠~~

코드 챌린지~

재시작하는 버튼을 만들어보래. 작은 카운트 리셋 버튼으로~

빠르게 가보자

```
void onResetPressed() {
    setState(() {
      timer.cancel();
      totalSeconds = twentyFiveMinutes;
      isRunning = false;
    });
  }
```

이 코드를 써줬다.

timer.cancel() 을 안하면 지옥의 타이머 겹침이 시작됨… 1초에 한 번씩 줄어드는게 아니게 돼~~~ 꼭 써줘야한다.

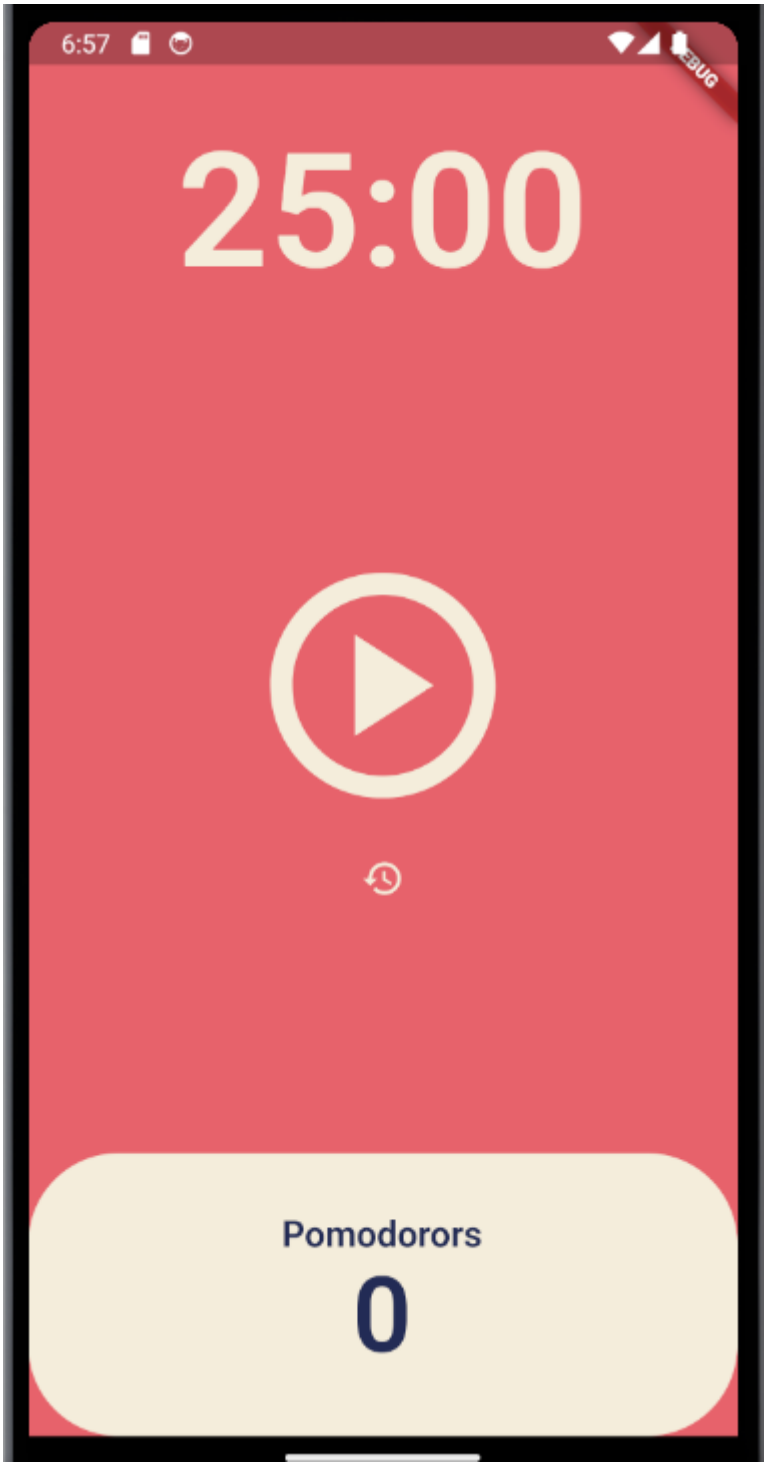
▼ 전체 코드

```
import 'dart:async';

import 'package:flutter/material.dart';

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  static const twentyFiveMinutes = 1500;
  int totalSeconds = twentyFiveMinutes; // 25분을 초로 환산 1500
  bool isRunning = false;
  int totalPomodorors = 0;

  late Timer timer;

  void onTick(Timer timer) {
    if (totalSeconds == 0) {
      setState(() {
        // 0이 되었으니 초기화~ 하고 뽀모도로 1 증가
        totalPomodorors += 1;
        isRunning = false;
        totalSeconds = twentyFiveMinutes;
      });
      timer.cancel();
    } else {
      setState(() {
        totalSeconds -= 1;
      });
```

```
  }
}

void onStartPressed() {
  timer = Timer.periodic(
    const Duration(seconds: 1),
    onTick,
  );
  setState(() {
    isRunning = true;
  });
}

void onPausePressed() {
  timer.cancel();
  setState(() {
    isRunning = false;
  });
}

void onResetPressed() {
  setState(() {
    timer.cancel();
    totalSeconds = twentyFiveMinutes;
    isRunning = false;
  });
}

String format(int seconds) {
  // 초를 분으로 포맷팅해서 보여주자.
  var duration = Duration(seconds: seconds);
  return duration.toString().split(".")[0].substring(2, 7);
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Theme.of(context).colorScheme.background,
    body: Column(
      children: [
        Flexible(
          flex: 1,
          child: Container(
            alignment: Alignment.bottomCenter,
            child: Text(
              format(totalSeconds),
              style: TextStyle(
                color: Theme.of(context).cardColor,
                fontSize: 89,
                fontWeight: FontWeight.w600,
              ),
            ),
          ),
        ),
        Flexible(
          flex: 3,
          child: Center(
            child: Column(
```

```
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  IconButton(
                    iconSize: 150,
                    color: Theme.of(context).cardColor,
                    onPressed: isRunning ? onPausePressed : onStartPressed,
                    icon: Icon(
                      isRunning
                          ? Icons.pause_circle_outline
                          : Icons.play_circle_outlined,
                    ),
                  ),
                  IconButton(
                    onPressed: onResetPressed,
                    color: Theme.of(context).cardColor,
                    icon: const Icon(
                      Icons.restore,
                    ),
                  ),
                ],
              ),
            ),
          ),
          Flexible(
            flex: 1,
            child: Row(
              children: [
                Expanded(
                  child: Container(
                    decoration: BoxDecoration(
                      borderRadius: BorderRadius.circular(50),
                      color: Theme.of(context).cardColor,
                    ),
                    child: Column(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Text(
                          "Pomodorors",
                          style: TextStyle(
                            fontSize: 20,
                            fontWeight: FontWeight.w600,
                            color:
                                Theme.of(context).textTheme.displayLarge!.color,
                          ),
                        ),
                        Text(
                          "$totalPomodorors",
                          style: TextStyle(
                            fontSize: 58,
                            fontWeight: FontWeight.w600,
                            color:
                                Theme.of(context).textTheme.displayLarge!.color,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
```

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


6:57
DEBUG
25:00
Pomodorors
0

헤헤