

custom_toast.dart

소스 코드

전체 코드

```
1  import 'package:flutter/material.dart';
2  import 'package:flutter_riverpod/flutter_riverpod.dart';
3
4  import '../app_colors.dart';
5  import '../app_styles.dart';
6  import 'toast_provider.dart';
7
8  class CustomToast extends ConsumerWidget {
9    const CustomToast({super.key});
10
11    @override
12    Widget build(BuildContext context, WidgetRef ref) {
13      final toastState = ref.watch(toastProvider);
14
15      if (!toastState.isVisible || toastState.message == null) {
16        return const SizedBox.shrink(); // 보이지 않을 때는 빈 위젯
17      }
18
19      return Positioned(
20        top: 57,
21        left: 0,
22        right: 0,
23        child: IgnorePointer( // 토스트 위의 UI 터치 가능하도록
24          child: Material(
25            color: Colors.transparent,
26            child: Center(
27              child: Container(
28                // alignment: Alignment.center,
29                padding: EdgeInsets.symmetric(vertical: 23, horizontal: 70),
30                decoration: BoxDecoration(
31                  borderRadius: BorderRadius.circular(20),
32                  border: BoxBorder.all(color: AppColors.D32F2F, width: 1),
33                  color: Color(0xffffde3e3)),
34                child: Text( toastState.message ?? "",
35                  textAlign: TextAlign.center,
36                  style: AppStyles.tsStep1Toast),
37              ),
38            ),
39
40            // child: Container(
41            //   width: double.maxFinite,
42            //   margin: const EdgeInsets.symmetric(horizontal: 20.0),
43            //   padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 10.0),
44            //   decoration: BoxDecoration(
45            //     color: const Color(0xFFFFFFFF),
```

```

46      //      borderRadius: BorderRadius.circular(6.0),
47      //      border: Border.all(color: AppColors.TEXT_20, width: 1)
48      //    ),
49      //    child: Row(
50      //      mainAxisAlignment: MainAxisAlignment.min,
51      //      mainAxisAlignment: MainAxisAlignment.center,
52      //      children: <widget>[
53      //        Flexible(
54      //          child: Text(
55      //            toastState.message ?? "",
56      //            style: Theme.of(context).textTheme.displayLarge,
57      //            textAlign: TextAlign.center,
58      //          ),
59      //        ),
60      //      ],
61      //    ),
62      //  ),
63    ),
64  ),
65 );
66 }
67 }

```

1 파일 개요

파일명: `lib/common/utils/custom_toast.dart`

역할: Flutter에서 사용하는 **커스텀 토스트(Custom Toast)** 위젯 정의

주요 목적:

- 앱 내 공통 **Toast 메시지 UI** 제공
- Riverpod 상태(`toastProvider`)를 구독하여 **토스트 표시 여부와 메시지 결정**
- 토스트 표시 시 **UI 터치 차단 없이 시각적 알림만 제공**

사용 맥락 예시:

- 사용자에게 간단한 알림 메시지 표시
- 시스템 상태, 오류, 성공/실패 알림
- 기존 FlutterToast와 다르게 위젯 트리에서 직접 관리 가능

2 주요 기능

1. Riverpod 상태 구독

- `ref.watch(toastProvider)` 로 토스트 상태 감시
- 상태 변경 시 UI 자동 갱신

2. 표시 여부 조건

- `isVisible` 이 `false` 이거나 `message` 가 `null` 이면 토스트 숨김
- `SizeBox.shrink()` 사용으로 화면 차지하지 않음

3. UI 구성

- `Positioned` 로 화면 상단 고정(`top: 57`)
- 좌우 끝까지 펼치기(`left:0, right:0`)
- `IgnorePointer` 사용하여 토스트 뒤 UI 터치 가능

4. 토스트 스타일

- `Material(color: Colors.transparent)`
- `Center` + `Container` 사용
- 패딩: `vertical:23, horizontal:70`
- 테두리: `borderRadius: 20, border: AppColors.D32F2F, width:1`
- 배경색: `Color(0xffffde3e3)`
- 텍스트 스타일: `AppStyles.tsStep1Toast`

5. 커스터마이징 가능

- 메시지 문자열 `toastState.message` 반영
- 상태관리 기반으로 표시/숨김 제어 가능

3 구조 분석

```
1 CustomToast (ConsumerWidget)
2   └─ build(context, ref)
3       └─ ref.watch(toastProvider) → toastState
4       └─ if (!toastState.isVisible || toastState.message == null)
5           └─ return SizedBox.shrink()
6       └─ return Positioned (top:57, left:0, right:0)
7           └─ IgnorePointer
8               └─ Material (color: transparent)
9                   └─ Center
10                       └─ Container
11                           └─ padding: EdgeInsets.symmetric(vertical:23,
12 horizontal:70)
13                           └─ decoration: BoxDecoration(borderRadius, border,
color)
                           └─ Text(toastState.message, style:
AppStyles.tsStep1Toast, textAlign: center)
```

특징:

- UI가 간단하고 시각적 요소만 담당
- 상태 관리(Riverpod) 기반으로 표시 여부 결정
- `IgnorePointer` 로 터치 이벤트 무시

4 동작 흐름

1. `CustomToast` 위젯 트리에서 렌더링

2. `ref.watch(toastProvider)` 로 상태 확인
 3. `toastState.isVisible == false` 이면 빈 위젯 반환 → 화면에 표시 안 함
 4. `toastState.isVisible == true && message != null` 이면 토스트 표시
 5. `Positioned` 로 화면 상단 배치, `IgnorePointer` 로 터치 이벤트 무시
 6. `Container` + `Text` 로 스타일 적용
-

5 장점

- **상태 기반 제어**: Riverpod과 연동되어 메시지 자동 갱신
 - **UI 터치 간섭 없음**: `IgnorePointer` 적용
 - **재사용성 높음**: 어디서나 위젯 트리에 포함 가능
 - **스타일 일관성**: `BoxDecoration`과 `AppStyles` 적용
-

6 단점 / 개선점

1. **위치 고정**
 - `top: 57` → 화면 크기나 반응형 환경에서 제한적
 - 개선: `MediaQuery` 기반 동적 위치 계산 가능
 2. **길이/크기 고정**
 - `Padding`이 고정, 길이가 긴 메시지 처리 어려움
 - 개선: `Flexible` 또는 `wrap` 적용
 3. **애니메이션 없음**
 - 나타났다 사라지는 효과 없음
 - 개선: `AnimatedOpacity`, `SlideTransition` 등 적용
 4. **다중 토스트**
 - 동시에 여러 메시지 처리 불가
 - 개선: Toast Queue 구조로 여러 메시지 순차 처리
 5. **커스터마이징 부족**
 - 색상, 폰트, 위치, duration 등 외부에서 설정 불가
 - 개선: 매개변수로 스타일 옵션 전달
-

7 개선된 구조 예시 (개념)

```
1 CustomToast(  
2   top: MediaQuery.of(context).padding.top + 20,  
3   padding: EdgeInsets.symmetric(vertical: 20, horizontal: 50),  
4   backgroundColor: Colors.black.withOpacity(0.8),  
5   textStyle: TextStyle(color: Colors.white, fontSize: 16),  
6   duration: Duration(seconds: 2),  
7 )
```

- 위치, 스타일, 지속 시간 등을 외부에서 조절 가능
- 반응형 화면과 다중 메시지 지원 가능