

# toast\_state.dart

## 소스 코드

### 전체 코드

```
1 import 'package:freezed_annotation/freezed_annotation.dart';
2
3 part 'toast_state.freezed.dart';
4 part 'toast_state.g.dart';
5
6 @freezed
7 abstract class ToastState with _$ToastState {
8   const factory ToastState({
9     @Default(false) bool isVisible,
10    String? message,
11  }) = _ToastState;
12
13   factory ToastState.fromJson(Map<String, Object?> json) => _$ToastStateFromJson(json);
14 }
```

## 1 파일 개요

파일명: lib/common/utils/toast\_state.dart

역할: 토스트 메시지 상태(ToastState) 데이터 클래스 정의

주요 목적:

- Riverpod Notifier(Toast)에서 관리하는 토스트 상태 구조화
- 상태를 불변 객체(immutable)로 정의
- 자동으로 copyWith, equals, hashCode 제공 (freezed)
- JSON 직렬화/역직렬화 지원 (fromJson, toJson)

사용 맥락 예시:

- ToastState를 사용해 isVisible과 message 상태 관리
- CustomToast 위젯과 연동하여 화면 표시 여부 결정
- 글로벌 상태 관리에서 토스트 메시지 전달

## 2 주요 기능

### 1. 상태 정의

- isVisible: 토스트 표시 여부 (bool, 기본값: false)
- message: 토스트에 표시할 문자열 (String?, null 허용)

### 2. 불변 객체(Immutable)

- freezed를 사용하여 상태 변경 시 항상 새로운 객체 생성

- `copyWith` 메서드로 부분적 상태 변경 가능

### 3. JSON 직렬화 지원

- `fromJson` / `toJson` 자동 생성
- 상태를 저장하거나 복원할 때 유용

## 3 구조 분석

```
1 ToastState (Freezed Data class)
2   ├── isVisible: bool = false // 토스트 표시 여부
3   ├── message: String?       // 표시할 메시지
4   ├── copyWith(...)          // 불변 상태 갱신
5   ├── equals / hashCode      // 객체 비교
6   ├── toJson()               // JSON 직렬화
7   └── fromJson(Map)          // JSON 역직렬화
```

특징:

- 불변성 보장 → 상태 관리 안전
- 자동 코드 생성 (`freezed_annotation`, `build_runner`)
- 단순 구조 → UI와 Notifier에서 사용하기 적합

## 4 동작 흐름

1. Notifier(`Toast`)에서 초기 상태 생성

```
1 | const ToastState() // isVisible=false, message=null
```

2. 토스트 표시 시 상태 갱신

```
1 | state = state.copyWith(message: "Hello", isVisible: true);
```

3. 상태를 UI 위젯(`CustomToast`)에서 감지

4. 토스트 숨김 시 상태 초기화

```
1 | state = state.copyWith(message: null, isVisible: false);
```

5. 필요 시 JSON 변환 가능

```
1 | final json = state.toJson();
2 | final restored = ToastState.fromJson(json);
```

## 5 장점

- 불변성: 상태 변경 시 안전

- 자동 생성 코드: `copyWith`, `equals`, `hashCode`
- 직렬화 지원: JSON 변환 가능
- 간단한 구조: Notifier와 UI에서 바로 사용 가능

---

## 6 단점 / 개선점

### 1. 추가 속성 제한

- 현재 `message`와 `isVisible`만 존재
- 개선: `backgroundColor`, `duration`, `textStyle` 등 옵션 추가

### 2. UI 관련 정보 없음

- 토스트 위치, 애니메이션 등 UI 속성은 Notifier/위젯에서 관리
- 개선: 상태에 UI 옵션 포함 가능

### 3. `frozen` 의존

- 자동 생성 파일 필요 (`flutter pub run build_runner build`)
- 프로젝트 환경 설정 필요

---

## 7 개선 예시 (개념)

```
1 @frozen
2 abstract class ToastState with _$ToastState {
3   const factory ToastState({
4     @Default(false) bool isVisible,
5     String? message,
6     @Default(2) int duration,           // 표시 시간
7     @Default("center") String position, // 위치: top/center/bottom
8   }) = _ToastState;
9 }
```

- 이렇게 하면 Notifier에서 위치, 지속 시간 등도 함께 관리 가능
- UI 위젯과 상태를 더 긴밀하게 연결 가능