btn_number.dart

소스 코드

전체 코드

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

class NumberButton extends StatelessWidget {
  final int number;
  final VoidCallback onPressed;
  final double width;
  final double height;

  const NumberButton({
    Key? key,
    required this.number,
    required this.onPressed,
    this.width = 100.0,
    this.height = 100.0,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      width: width,
      height: height,
      decoration: BoxDecoration(
        color: Colors.blueGrey[50], // 버튼 배경색 (선택 사항)
        borderRadius: BorderRadius.circular(12.0), // 버튼 모서리 둥글기 (선택 사항)
        boxShadow: [ // 살짝 그림자 효과 (선택 사항)
          BoxShadow(
            color: Colors.grey.withAlpha(30),
            spreadRadius: 1,
            blurRadius: 3,
            offset: Offset(0, 1),
          ),
        ],
      ),
      child: Material( // InkWell 효과를 위해 Material 위젯으로 감싸기
        color: Colors.transparent, // Material의 색상은 투명하게 하여 Container 색상이 보이도록
        borderRadius: BorderRadius.circular(12.0), // InkWell 효과가 Container 모양을 따르도록
        child: InkWell(
          borderRadius: BorderRadius.circular(12.0), // 튀어나가지 않도록 동일한 borderRadius 적용
          onTap: onPressed,
          splashColor: Colors.blueAccent.withOpacity(0.3), // 눌렀을 때 퍼지는 효과 색상
          highlightColor: Colors.blueAccent.withOpacity(0.1), // 누르고 있을 때 배경 하이라이트 색상
```

```
          child: Center(
            child: Text(
              number.toString(),
              style: TextStyle(
                fontSize: 32,
                fontWeight: FontWeight.bold,
                color: Colors.black87,
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

1. 구조 분석

`NumberButton`은 다음 역할을 수행하는 stateless UI 컴포넌트:

- 정해진 크기(width/height) 안에 숫자 텍스트 표시
- 터치 이벤트(onPressed) 콜백 처리
- `Container` + `Material` + `InkWell` 구조로 ripple 효과 제공
- borderRadius, boxShadow 등을 포함한 스타일 프리셋

구조 계층:

```
Container
 └ Material
      └ InkWell
           └ Center
                └ Text(number)
```

---

2. 문제점 및 개선 포인트

✔ (1) Container + Material + InkWell 패턴은 적절하지만, Material을 최상단에 두면 더 명확함

현재는 **Container → Material → InkWell**,
권장 패턴은 **Material → Ink → DecoratedBox** 혹은 **Material → InkWell → Ink**.

지금 구조도 동작하지만 ripple clip이 미세하게 어긋날 수 있음.

✔ (2) Color, Style 하드코딩

공통 스타일(AppStyles, AppColors)을 사용하는 프로젝트라면 통일 필요.

✔ (3) 접근성(A11Y) 고려 없음

semanticLabel 추가 가능.

✔ (4) width/height 기본값이 너무 큼

재사용성을 높이려면 optional하고, parent constraints 기반 렌더링 권장.

---

3. 공식문서 스타일 리팩토링 버전

Flutter 공식 문서의 Material 버튼 컴포넌트 스타일에 맞춘 구조로 다시 구성.

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

class NumberButton extends StatelessWidget {
  final int number;
  final VoidCallback onPressed;
  final double width;
  final double height;

  const NumberButton({
    super.key,
    required this.number,
    required this.onPressed,
    this.width = 100.0,
    this.height = 100.0,
  });

  @override
  Widget build(BuildContext context) {
    return Material(
      color: Colors.blueGrey[50],
      borderRadius: BorderRadius.circular(12.0),
      elevation: 1,
      child: InkWell(
        onTap: onPressed,
        borderRadius: BorderRadius.circular(12.0),
        splashColor: Colors.blueAccent.withOpacity(0.25),
        highlightColor: Colors.blueAccent.withOpacity(0.08),
        child: SizedBox(
          width: width,
          height: height,
          child: Center(
            child: Text(
              number.toString(),
              style: const TextStyle(
                fontSize: 32,
                fontWeight: FontWeight.bold,
                color: Colors.black87,
              ),
            ),
          ),
        ),
```

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

---

4. 리팩토링 특징

- Material을 최상단으로 올려 ripple 영역과 clip 단순화
- Container 대신 SizedBox 사용하여 레이아웃 명확화
- elevation 사용하여 그림자를 Material 시스템에 맞게 처리
- borderRadius는 Material + InkWell 양쪽에 적용하여 ripple clip 정상화
- splash/hightlight 색상 투명도 조절