플러터 생산성 앱 - 구현 방향 가이드

📋 프로젝트 세팅

초기 설정

- [ ] Flutter 프로젝트 생성하기
- [ ] lib 폴더에 screens, widgets, models 폴더 구조 만들기
- [ ] pubspec.yaml에서 앱 이름 설정하기

🔥 1단계: 메인 대시보드 화면

학습 목표: ElevatedButton + BuildContext + 기본 push 네비게이션

화면 구조 만들기

- [ ] StatelessWidget으로 DashboardScreen 클래스 만들기
- [ ] Scaffold 위젯으로 기본 화면 구조 잡기
- [ ] AppBar 위젯으로 상단바 만들고 제목 설정하기

ElevatedButton으로 메인 메뉴 4개 만들기

- [ ] **"할일 목록" 버튼**: ElevatedButton 위젯 사용
  - backgroundColor 속성으로 파란색 배경 설정
  - foregroundColor 속성으로 흰색 텍스트 설정
  - elevation 속성으로 그림자 효과 주기
  - padding 속성으로 버튼 크기 키우기
  - textStyle 속성으로 폰트 굵기와 크기 설정
  - onPressed에는 일단 print문으로 클릭 확인
- [ ] **"통계 보기" 버튼**: ElevatedButton 위젯 사용
  - onPressed를 null로 설정해서 비활성화 상태 만들기
  - disabledBackgroundColor 속성으로 비활성화 색상 설정
  - disabledForegroundColor 속성으로 비활성화 텍스트 색상 설정
- [ ] **"설정" 버튼**: OutlinedButton 위젯 사용
  - 기본 스타일 그대로 사용
  - onPressed에 print문으로 클릭 확인
- [ ] **"프로필" 버튼**: TextButton 위젯 사용
  - 기본 스타일 그대로 사용
  - onPressed에 print문으로 클릭 확인

레이아웃 구성하기

- [ ] Column 위젯으로 버튼들을 세로로 배치
- [ ] MainAxisAlignment.center로 버튼들을 화면 중앙에 배치
- [ ] SizedBox 위젯으로 버튼 사이 간격 조정
- [ ] Container나 SizedBox로 버튼 너비 통일하기

main.dart에서 연결하기

- [ ] MaterialApp의 home 속성에 DashboardScreen 연결

테스트하기

- [ ] 앱 실행해서 4개 버튼이 올바르게 보이는지 확인
- [ ] 각 버튼 클릭시 콘솔에 메시지 출력되는지 확인
- [ ] "통계 보기" 버튼이 클릭되지 않는지 확인

🔥 2단계: 할일 목록 화면

학습 목표: OutlinedButton + MaterialStateProperty.all + pop 메서드

데이터 모델 만들기

- [ ] Todo 클래스 만들기 (id, title, category, priority, isCompleted 속성)

화면 구조 만들기

- [ ] StatefulWidget으로 TodoListScreen 클래스 만들기
- [ ] List<Todo> 타입으로 더미 할일 데이터 3-5개 만들기
- [ ] Scaffold + AppBar로 기본 구조 만들기

할일 목록 UI 만들기

- [ ] ListView.builder 위젯으로 할일 목록 표시
- [ ] Card 위젯으로 각 할일 항목 감싸기
- [ ] ListTile이나 Column으로 할일 제목, 카테고리, 우선순위 표시

완료 토글 버튼 만들기

- [ ] OutlinedButton 위젯 사용
- [ ] ButtonStyle과 MaterialStateProperty.all() 사용해서 스타일링
  - backgroundColor를 완료 상태에 따라 다르게 설정
  - foregroundColor를 완료 상태에 따라 다르게 설정
- [ ] onPressed에서 setState로 완료 상태 토글
- [ ] Icon 위젯으로 체크/미체크 아이콘 표시

우선순위 변경 버튼들 만들기

- [ ] TextButton 위젯 3개로 "높음", "보통", "낮음" 버튼 만들기

- [ ] Row 위젯으로 가로 배치
- [ ] ButtonStyle과 MaterialStateProperty.all() 사용
  - 선택된 우선순위에 따라 backgroundColor 다르게 설정
  - 선택된 우선순위에 따라 foregroundColor 다르게 설정
- [ ] onPressed에서 setState로 우선순위 변경

뒤로가기 기능 만들기

- [ ] AppBar의 leading 속성에 IconButton 추가
- [ ] Navigator.of(context).pop() 메서드 사용해서 이전 화면으로 돌아가기

네비게이션 연결하기

- [ ] DashboardScreen의 "할일 목록" 버튼 onPressed 수정
- [ ] Navigator.of(context).push() 메서드 사용
- [ ] MaterialPageRoute로 TodoListScreen 연결

테스트하기

- [ ] 대시보드에서 할일 목록으로 이동 확인
- [ ] 완료 토글 버튼 클릭시 색상과 아이콘 변경 확인
- [ ] 우선순위 버튼들 클릭시 색상 변경 확인
- [ ] 뒤로가기 버튼으로 대시보드 복귀 확인

🔥 3단계: 할일 추가 화면

학습 목표: MaterialStateProperty.resolveWith + Shape 속성 + IconButton + async/await

화면 구조 만들기

- [ ] StatefulWidget으로 AddTodoScreen 클래스 만들기
- [ ] TextEditingController로 제목 입력 관리
- [ ] 선택된 카테고리, 우선순위 상태 변수 만들기
- [ ] Form 위젯으로 전체 입력 폼 구성

제목 입력 필드 만들기

- [ ] TextFormField 위젯으로 할일 제목 입력받기
- [ ] validator 속성으로 빈 값 검증하기
- [ ] InputDecoration으로 힌트 텍스트와 테두리 스타일 설정

카테고리 선택 버튼들 만들기 (CircleBorder 학습)

- [ ] OutlinedButton 위젯 3개로 카테고리 버튼 만들기
- [ ] OutlinedButton.styleFrom 메서드 사용
- [ ] shape 속성에 CircleBorder() 적용해서 원형 버튼 만들기

- [ ] 선택 상태에 따라 backgroundColor 다르게 설정
- [ ] Icon과 Text를 Column으로 세로 배치
- [ ] onPressed에서 setState로 카테고리 선택 상태 변경

우선순위 버튼들 만들기 (resolveWith 학습)

- [ ] ElevatedButton 위젯 3개로 우선순위 버튼 만들기
- [ ] ButtonStyle과 MaterialStateProperty.resolveWith() 사용
- [ ] Set<MaterialState> states 매개변수로 버튼 상태 확인
- [ ] MaterialState.pressed 상태일 때 다른 색상 적용
- [ ] 선택된 우선순위에 따라 다른 스타일 적용
- [ ] minimumSize도 resolveWith로 눌렸을 때 크기 변경

AppBar 액션 버튼 만들기

- [ ] AppBar의 actions 속성에 IconButton 추가
- [ ] Icons.save 아이콘 사용
- [ ] onPressed에 저장 기능 연결 (폼 유효성에 따라 활성화/비활성화)

저장 기능 구현하기

- [ ] _saveTodo 메서드 만들기
- [ ] Form의 validate() 메서드로 유효성 검사
- [ ] Todo 객체 생성해서 Navigator.of(context).pop()으로 데이터 전달

FloatingActionButton 추가하기

- [ ] TodoListScreen에 FloatingActionButton 추가
- [ ] onPressed에서 async/await 사용해서 AddTodoScreen으로 이동
- [ ] Navigator.of(context).push()의 반환값 받기
- [ ] 반환된 결과가 null이 아니면 할일 목록에 추가

테스트하기

- [ ] + 버튼으로 추가 화면 이동 확인
- [ ] 카테고리 버튼들이 원형으로 표시되고 선택시 색상 변경 확인
- [ ] 우선순위 버튼들이 누를 때 크기와 색상 변경 확인
- [ ] 빈 제목으로 저장시 에러 메시지 확인
- [ ] 정상 저장 후 할일 목록에 새 항목 추가 확인

🔥 4단계: 설정 화면

학습 목표: pushReplacement + pushNamedAndRemoveUntil + PopScope

화면 구조 만들기

- [ ] StatefulWidget으로 SettingsScreen 클래스 만들기
- [ ] 다크모드 상태 변수 (bool 타입) 만들기
- [ ] Scaffold + AppBar 기본 구조

테마 토글 버튼 만들기

- [ ] ElevatedButton.icon 위젯 사용
- [ ] Icons.light_mode, Icons.dark_mode 아이콘 상태에 따라 변경
- [ ] onPressed에서 setState로 다크모드 토글
- [ ] ScaffoldMessenger.of(context).showSnackBar()로 변경 알림

데이터 초기화 버튼 만들기

- [ ] ElevatedButton 위젯 사용하고 빨간색 스타일 적용
- [ ] showDialog로 확인 다이얼로그 만들기
- [ ] AlertDialog 위젯 사용해서 제목, 내용, 액션 버튼들 구성
- [ ] "삭제" 선택시 Navigator.of(context).pushNamedAndRemoveUntil() 사용
- [ ] (route) => false 조건으로 모든 이전 화면 제거

홈으로 돌아가기 버튼 만들기

- [ ] OutlinedButton 위젯 사용
- [ ] Navigator.of(context).pushReplacement() 메서드 사용
- [ ] MaterialPageRoute로 DashboardScreen 연결

PopScope 적용하기

- [ ] PopScope 위젯으로 Scaffold 감싸기
- [ ] canPop을 false로 설정
- [ ] onPopInvoked 콜백에서 확인 다이얼로그 구현
- [ ] showDialog로 나가기 확인 받기
- [ ] 확인시에만 Navigator.of(context).pop() 실행

네비게이션 연결하기

- [ ] DashboardScreen의 "설정" 버튼에 Navigator.push로 연결

테스트하기

- [ ] 설정 화면 이동 확인
- [ ] 테마 토글시 아이콘과 텍스트 변경 및 스낵바 표시 확인
- [ ] 데이터 초기화시 확인 다이얼로그 표시 확인
- [ ] "삭제" 선택시 모든 화면 제거되고 대시보드로 이동 확인
- [ ] 뒤로가기시 확인 다이얼로그 표시 확인
- [ ] "홈으로 돌아가기"로 현재 화면이 대시보드로 교체되는지 확인

🔥 5단계: Named Routes와 프로필 화면

학습 목표: Declarative Routing + arguments 전달 + canPop/maybePop

Named Routes 설정하기

- [ ] main.dart의 MaterialApp에서 routes 속성 설정
- [ ] initialRoute를 '/'로 설정
- [ ] 각 화면별로 라우트 이름과 builder 함수 매핑

ProfileScreen 만들기

- [ ] StatelessWidget으로 ProfileScreen 클래스 만들기
- [ ] 사용자 정보 표시 (더미 데이터 사용)
- [ ] CircleAvatar로 프로필 이미지 표시
- [ ] Card나 ListTile로 이름, 이메일 등 정보 표시

pushNamed로 네비게이션 변경하기

- [ ] 모든 Navigator.push를 Navigator.pushNamed로 변경
- [ ] DashboardScreen의 모든 버튼들을 pushNamed로 수정
- [ ] TodoListScreen의 FloatingActionButton도 pushNamed로 수정

arguments로 데이터 전달하기

- [ ] AddTodoScreen에서 ModalRoute.of(context)?.settings.arguments 사용
- [ ] Map<String, dynamic> 타입으로 arguments 받기
- [ ] 'mode'와 'todo' 키로 추가/편집 모드 구분
- [ ] 편집 모드일 때 기존 데이터로 폼 초기화

할일 편집 기능 추가하기

- [ ] TodoListScreen의 각 할일 항목에 IconButton 추가
- [ ] Icons.edit 아이콘 사용
- [ ] onPressed에서 pushNamed로 '/add-todo' 이동
- [ ] arguments로 편집 모드와 해당 todo 객체 전달

canPop과 maybePop 실습하기

- [ ] ProfileScreen에 "Can Pop 확인" 버튼 추가
- [ ] ElevatedButton으로 만들고 onPressed에서 Navigator.canPop() 호출
- [ ] ScaffoldMessenger로 결과 표시
- [ ] "Maybe Pop" 버튼도 추가해서 Navigator.maybePop() 테스트

pushReplacementNamed 사용하기

- [ ] SettingsScreen의 "홈으로 돌아가기" 버튼 수정

- [ ] Navigator.pushReplacementNamed()로 '/' 라우트로 이동

테스트하기

- [ ] 모든 화면간 Named Routes 네비게이션 동작 확인
- [ ] 할일 편집시 arguments로 데이터 전달되는지 확인
- [ ] 편집 모드에서 기존 데이터가 폼에 표시되는지 확인
- [ ] canPop, maybePop 버튼들의 동작과 결과 표시 확인

🔥 6단계: 고급 기능 및 마무리

버튼 로딩 상태 구현하기

- [ ] 저장 버튼에 로딩 상태 추가
- [ ] bool 변수로 로딩 상태 관리
- [ ] 로딩 중일 때 CircularProgressIndicator 표시
- [ ] 로딩 중에는 onPressed를 null로 설정

일관된 테마 적용하기

- [ ] ThemeData로 앱 전체 색상 테마 설정
- [ ] colorScheme 속성으로 primary, secondary 색상 정의
- [ ] 모든 버튼이 테마 색상을 따르도록 수정

에러 처리 추가하기

- [ ] try-catch로 네비게이션 에러 처리
- [ ] 폼 validation 메시지 개선
- [ ] 네트워크나 저장 실패시 사용자에게 알림

최종 테스트

- [ ] 전체 앱 플로우 완전히 테스트
- [ ] 모든 버튼 타입이 적절한 곳에 사용되었는지 확인
- [ ] 모든 네비게이션 패턴이 올바르게 동작하는지 확인

🎯 학습 완료 체크리스트

Button 관련 위젯과 속성

- [ ] ElevatedButton 위젯과 styleFrom 메서드 사용법
- [ ] OutlinedButton 위젯과 CircleBorder 적용법
- [ ] TextButton 위젯 기본 사용법
- [ ] IconButton 위젯과 AppBar에서 활용법
- [ ] FloatingActionButton 위젯 사용법

- [ ] ButtonStyle 클래스와 속성들 이해
- [ ] MaterialStateProperty.all() 메서드 사용법
- [ ] MaterialStateProperty.resolveWith() 메서드 고급 활용
- [ ] backgroundColor, foregroundColor 등 색상 속성들
- [ ] elevation, padding, textStyle 등 스타일 속성들

Navigation 관련 메서드와 개념

- [ ] Navigator.of(context) 메서드와 BuildContext 이해
- [ ] push() 메서드와 MaterialPageRoute 사용법
- [ ] pop() 메서드와 데이터 반환하기
- [ ] pushReplacement() 메서드 사용 시점
- [ ] pushNamedAndRemoveUntil() 메서드와 조건 함수
- [ ] canPop()과 maybePop() 메서드 차이점
- [ ] Named Routes 설정과 pushNamed() 사용법
- [ ] RouteSettings와 arguments로 데이터 전달
- [ ] ModalRoute.of(context) 메서드로 arguments 받기
- [ ] PopScope 위젯과 onPopInvoked 콜백 활용

이 가이드로 직접 구현하시면서 개념을 체득하세요! 🚀