Date : 250721~250727

# 📅 개인 일정 관리 앱 - 7일 완성 가이드

## \*\*Day 1: 프로젝트 기본 구조 + Navigator\*\*

\*\*복습 개념\*\*: Navigator 안전장치 (canPop, maybePop)

\*\*구현 목표\*\*: 앱의 기본 구조와 화면 간 네비게이션

### \*\*🗓️ Day 1 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 3개 기본 화면 구성 (홈, 일정추가, 일정상세)

- Navigator 안전장치 적용한 화면 이동

- AppBar와 기본 레이아웃 완성

\*\*2️⃣ 기존 코드 연결점\*\*

- 새 프로젝트 생성부터 시작

- main.dart에 MaterialApp 설정

- 각 화면별 StatefulWidget 생성

\*\*3️⃣ 구현 순서\*\*

1. 프로젝트 생성 및 기본 설정

2. 홈 화면 (ScheduleListScreen) 생성

3. 일정 추가 화면 (AddScheduleScreen) 생성

4. 일정 상세 화면 (ScheduleDetailScreen) 생성

5. Navigator.canPop() 체크 후 이동하는 안전한 네비게이션 적용

\*\*4️⃣ 주의사항\*\*

- Arguments 전달 시 null 체크 필수

- 뒤로가기 시 canPop() 확인 후 pop() 실행

- AppBar leading 커스터마이징 시 Navigator 안전장치 적용

\*\*5️⃣ 완성 기준\*\*

- 3개 화면 + 기본 네비게이션 동작

- 안전한 뒤로가기 기능

- 각 화면에 기본 AppBar와 Scaffold 구조

---

## \*\*Day 2: 일정 추가 Form + 유효성 검증\*\*

\*\*복습 개념\*\*: Form 검증 + Null Safety

\*\*구현 목표\*\*: 일정 데이터 입력 폼과 안전한 검증 시스템

### \*\*🗓️ Day 2 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 일정 추가 Form 완성 (제목, 날짜, 시간, 메모)

- 입력값 검증 시스템 구현

- Null Safety 적용한 안전한 데이터 처리

\*\*2️⃣ 기존 코드 연결점\*\*

- AddScheduleScreen에 Form 위젯 추가

- TextEditingController들을 StatefulWidget의 State 클래스에 선언

- dispose()에서 리소스 정리

\*\*3️⃣ 구현 순서\*\*

1. Form과 GlobalKey<FormState> 설정

2. TextFormField들 구성 (제목, 메모)

3. DatePicker와 TimePicker 연동

4. 검증 로직 구현 (빈 값, 날짜 유효성)

5. 안전한 데이터 수집 및 전달

\*\*4️⃣ 주의사항\*\*

- Controller는 State 클래스에서 관리

- dispose()에서 반드시 controller.dispose() 호출

- null 체크 후 데이터 처리: `if (value != null && value.isNotEmpty)`

\*\*5️⃣ 완성 기준\*\*

- 완전한 입력 폼 (제목, 날짜, 시간, 메모)

- 유효성 검증 동작

- 안전한 데이터 수집 및 홈 화면으로 전달

---

## \*\*Day 3: 상태 관리 + 일정 목록 표시\*\*

\*\*복습 개념\*\*: StatefulWidget 생명주기 + Map 데이터 구조

\*\*구현 목표\*\*: 일정 목록 관리와 화면 업데이트

### \*\*🗓️ Day 3 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 일정 데이터 모델 클래스 생성

- 상태 관리로 일정 목록 관리

- ListView로 일정 목록 표시

\*\*2️⃣ 기존 코드 연결점\*\*

- ScheduleListScreen의 State에 List<Map<String, dynamic>> 선언

- Navigator에서 받은 데이터를 목록에 추가

- setState()로 화면 업데이트

\*\*3️⃣ 구현 순서\*\*

1. Schedule 데이터 모델 정의 (Map 형태)

2. State에 schedules 리스트 선언

3. 새 일정 추가 시 안전한 Map 업데이트

4. ListView.builder로 목록 표시

5. 각 일정 항목 터치 시 상세 화면 이동

\*\*4️⃣ 주의사항\*\*

- Map 업데이트 시 null 체크: `if (newData != null) schedules.add(newData)`

- initState()에서 초기 데이터 설정

- ListView에서 index 범위 체크

\*\*5️⃣ 완성 기준\*\*

- 일정 목록 표시

- 새 일정 추가 시 목록 업데이트

- 일정 항목 터치 시 상세 화면 이동

---

## \*\*Day 4: 데이터 영구 저장 + 불러오기\*\*

\*\*복습 개념\*\*: SharedPreferences + JSON 파싱

\*\*구현 목표\*\*: 앱 재시작 후에도 데이터 유지

### \*\*🗓️ Day 4 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- SharedPreferences로 일정 데이터 저장

- JSON 인코딩/디코딩으로 복잡한 데이터 처리

- 앱 시작 시 저장된 데이터 불러오기

\*\*2️⃣ 기존 코드 연결점\*\*

- pubspec.yaml에 shared\_preferences 추가

- State의 initState()에서 데이터 로드

- 일정 추가/수정/삭제 시마다 저장

\*\*3️⃣ 구현 순서\*\*

1. shared\_preferences 패키지 추가

2. 저장 함수: List<Map> → JSON → SharedPreferences

3. 로드 함수: SharedPreferences → JSON → List<Map>

4. initState()에서 자동 로드

5. 데이터 변경 시마다 자동 저장

\*\*4️⃣ 주의사항\*\*

- JSON 파싱 시 try-catch로 에러 처리

- null 체크 후 데이터 처리: `if (jsonString != null)`

- List<Map<String, dynamic>>과 JSON 변환 주의

\*\*5️⃣ 완성 기준\*\*

- 앱 재시작 후 데이터 유지

- 일정 추가 시 자동 저장

- 안전한 JSON 파싱 동작

---

## \*\*Day 5: 수정/삭제 기능 + 완료 표시\*\*

\*\*복습 개념\*\*: 비동기 처리 패턴 + forEach 안전한 사용

\*\*구현 목표\*\*: 완전한 CRUD 기능 구현

### \*\*🗓️ Day 5 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 일정 수정 기능 구현

- 일정 삭제 기능 구현

- 완료/미완료 토글 기능

\*\*2️⃣ 기존 코드 연결점\*\*

- 상세 화면에 수정/삭제 버튼 추가

- 목록에서 스와이프 또는 길게 눌러 삭제

- 체크박스로 완료 상태 관리

\*\*3️⃣ 구현 순서\*\*

1. 일정 수정 화면 (기존 Form 재사용)

2. Dismissible 위젯으로 스와이프 삭제

3. 완료 상태 체크박스 추가

4. 데이터 업데이트 후 SharedPreferences 저장

5. ScaffoldMessenger로 사용자 피드백

\*\*4️⃣ 주의사항\*\*

- forEach 대신 안전한 반복문 사용

- 삭제 전 확인 다이얼로그 표시

- 상태 변경 후 즉시 저장

\*\*5️⃣ 완성 기준\*\*

- 완전한 CRUD 기능

- 완료/미완료 상태 관리

- 사용자 친화적인 피드백

---

## \*\*Day 6: 날짜별 필터링 + 애니메이션\*\*

\*\*복습 개념\*\*: Widget 크기 계산 + Container vs Padding

\*\*구현 목표\*\*: 앱 고도화 및 사용자 경험 개선

### \*\*🗓️ Day 6 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 날짜별 일정 필터링 기능

- 부드러운 애니메이션 효과

- 성능 최적화된 위젯 사용

\*\*2️⃣ 기존 코드 연결점\*\*

- 홈 화면에 날짜 선택 위젯 추가

- where() 메서드로 필터링

- AnimatedContainer나 Hero 애니메이션 적용

\*\*3️⃣ 구현 순서\*\*

1. DatePicker로 날짜 선택 기능

2. whereQ 메서드로 날짜별 필터링

3. AnimatedContainer로 부드러운 전환

4. Container vs Padding 성능 최적화

5. 필터 상태 표시 UI

\*\*4️⃣ 주의사항\*\*

- 단순 여백은 Padding, 복합 기능은 Container 사용

- 날짜 비교 시 DateTime 형식 통일

- 애니메이션 과도하지 않게 적용

\*\*5️⃣ 완성 기준\*\*

- 날짜별 필터링 동작

- 부드러운 애니메이션 효과

- 최적화된 성능

---

## \*\*Day 7: 정리 & 자기화\*\*

\*\*목표\*\*: 프로젝트 마무리 및 학습 정리

\*\*작업\*\*: 코드 정리, 문서 작성, 학습 노트 정리

### \*\*🗓️ Day 7 진행 가이드\*\*

\*\*1️⃣ 오늘의 목표\*\*

- 전체 코드 리팩토링

- README 작성

- 학습한 개념들 정리

\*\*2️⃣ 작업 내용\*\*

1. 코드 주석 추가 및 정리

2. 사용하지 않는 코드 제거

3. README.md 작성

4. 프로젝트를 통해 배운 점 정리

5. 다음 프로젝트 계획

\*\*3️⃣ 정리할 학습 포인트\*\*

- Navigator 안전장치 사용법

- SharedPreferences + JSON 패턴

- StatefulWidget 생명주기 관리

- Form 검증 및 Null Safety

- Map 데이터 안전한 업데이트

\*\*4️⃣ 완성 기준\*\*

- 깔끔한 코드 구조

- 완성된 README

- 학습 노트 정리

---

## \*\*🎯 학습 성과 평가\*\*

- \*\*기술적 완성도\*\*: 모든 CRUD 기능 동작 🚀

- \*\*개념 적용도\*\*: 5개 복습 개념 완전 활용 💪

- \*\*코드 품질\*\*: 안전하고 읽기 쉬운 구조 📚

- \*\*학습 정리\*\*: 경험을 지식으로 전환 ✍️

## \*\*💡 이번 프로젝트의 목적\*\*

- \*\*통합적 사고\*\*: 여러 개념을 자연스럽게 조합하는 능력 개발

- \*\*실무 경험\*\*: 실제 앱 개발 프로세스 완전 체험

- \*\*점진적 성장\*\*: 매일 발전하는 앱을 통한 성취감과 동기 부여

- \*\*안전한 코딩\*\*: Null Safety와 에러 처리의 중요성 체득