



mobile

programming

project

모바일프로그래밍 프로젝트

mobile programming project__

```
18 <p style="text-align:left;">Left</p>
19 <p style="text-align:center;">Center</p>
20 <p style="text-align:right;">Right</p>
```

-

-

-

24 학 과 : 컴퓨터공학과

25 이 름 : 오진영

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26



mobile

programming

project

9

10

11

12

13

14

프로젝트 개요 ; Project outline __

15

16

프로젝트 기능 ; Project function __

17

18

앱 화면 구성 ; Screen Configuration __

20

21

앱 구조 및 구현 방식 ; Structure & Implementation Method __

22

23

Q&A ; Q&A __

24

25

26



mobile

programming

project

Project outline __

프로젝트 명

MyTimetable – 나만의 시간표 관리 앱

특징

직관적인 시간표 입력 기능

시간표 목록 조회 기능

시간표 삭제 가능

로컬 DB 저장



mobile

programming

project

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

Development Schedule

2025. 05.12

2025. 05.19

2025. 05.26

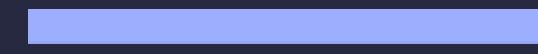
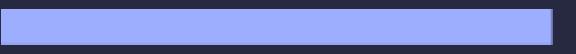
2025. 06.02

└ 기획 및 UI 설계

└ 기본 기능 개발

└ 고급 기능 추가

└ 테스트 및 발표 준비





mobile

programming

project

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

project function __

시간표 입력



요일, 시간, 과목명을 직접 입력 및 저장

화면에서 Spinner와 입력창을 통해
시간표를 추가

시간표 조회



RecyclerView로 정리된 시간표 목록 확인

메인화면에서 입력된 시간표들이
리스트로 정리되어 요일, 시간, 과목을
한눈에 볼 수 있게 구성



mobile

programming

project

9

10

11

12

project function __

13

14

15

삭제 기능

16



리스트 항목을 길게 눌러 삭제

17

18

19

삭제 전에는 확인ダイ얼로그가 떠서
실수 삭제 방지

20

21

22

23

24

25

26

영구 저장



SQLite를 사용하여 앱 종료 후에도 유지

앱 종료해도 데이터 유지
별도의 로그인 없이도 개인 시간표 관리



mobile

programming

project

9

10

11

앱 메인 화면 – MainActivity: 시간표 목록 조회

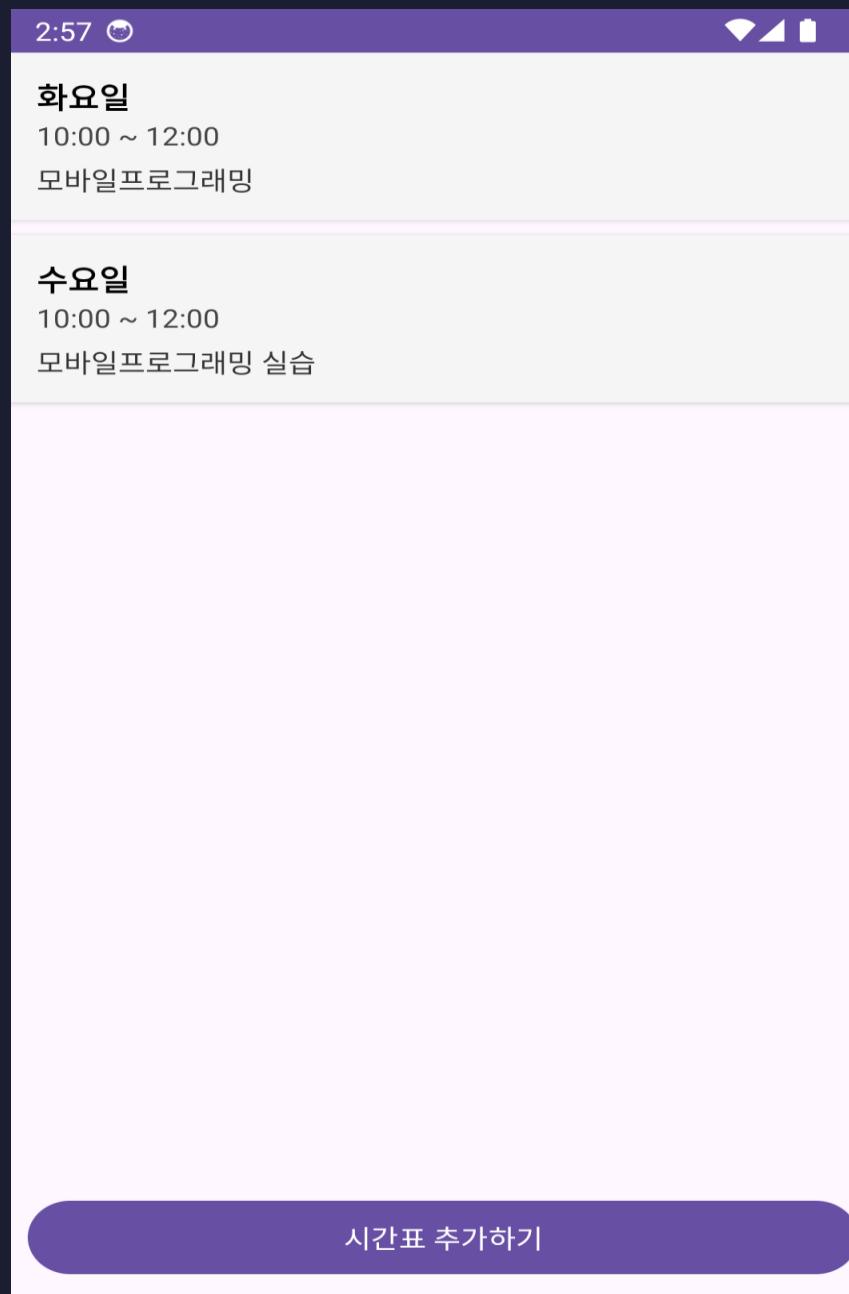
12

13

14



15



16

17

18

19

20

21

22

23

24

25

26



- 앱의 메인 화면
- 사용자가 입력한 시간표 목록이 요일, 시간
과목 순으로 출력
- RecyclerView를 이용해 리스트 형태로 정리
- 하단의 시간표 추가하기 버튼을 누르면 입력
화면으로 이동



mobile

programming

project

9

10

11

12

13

14

15

16

17

18

19

20

21

22

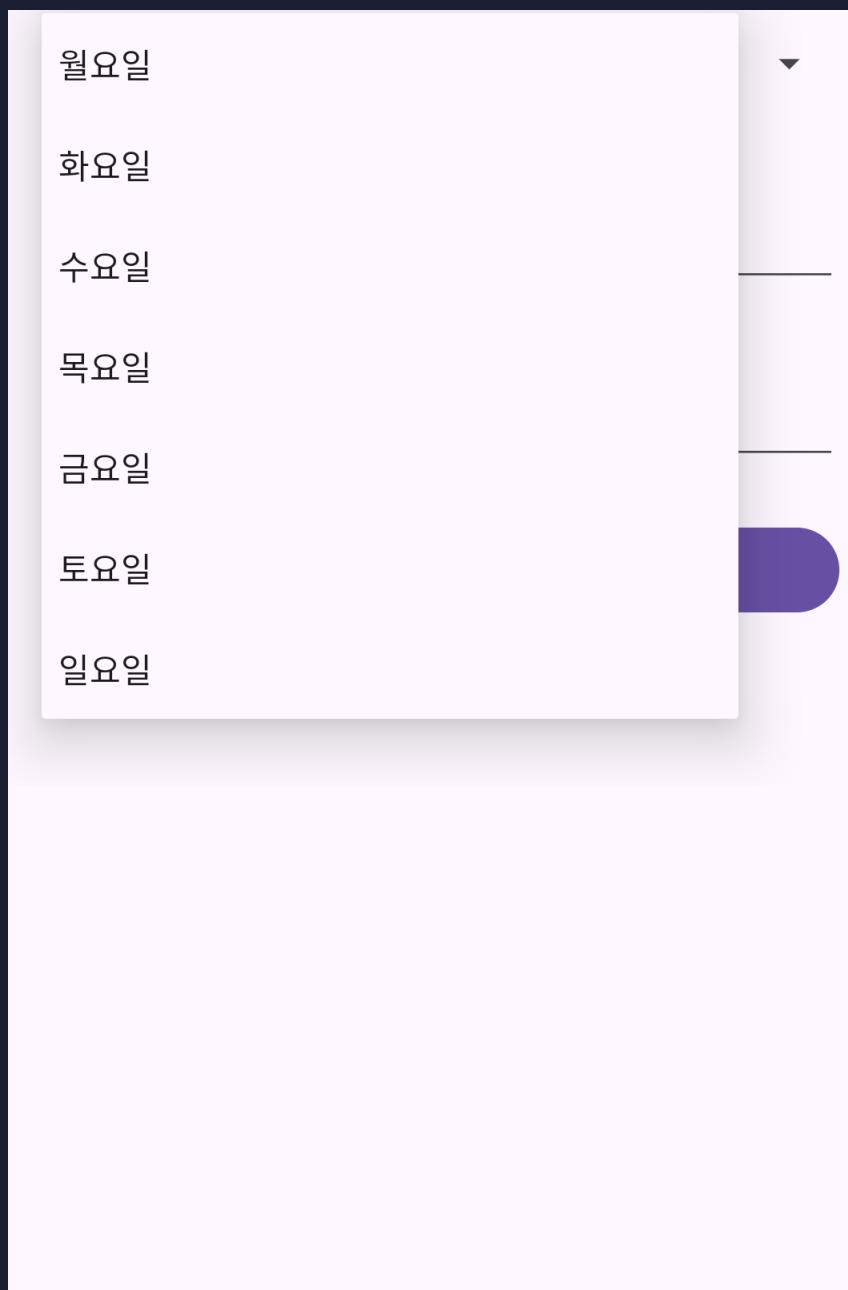
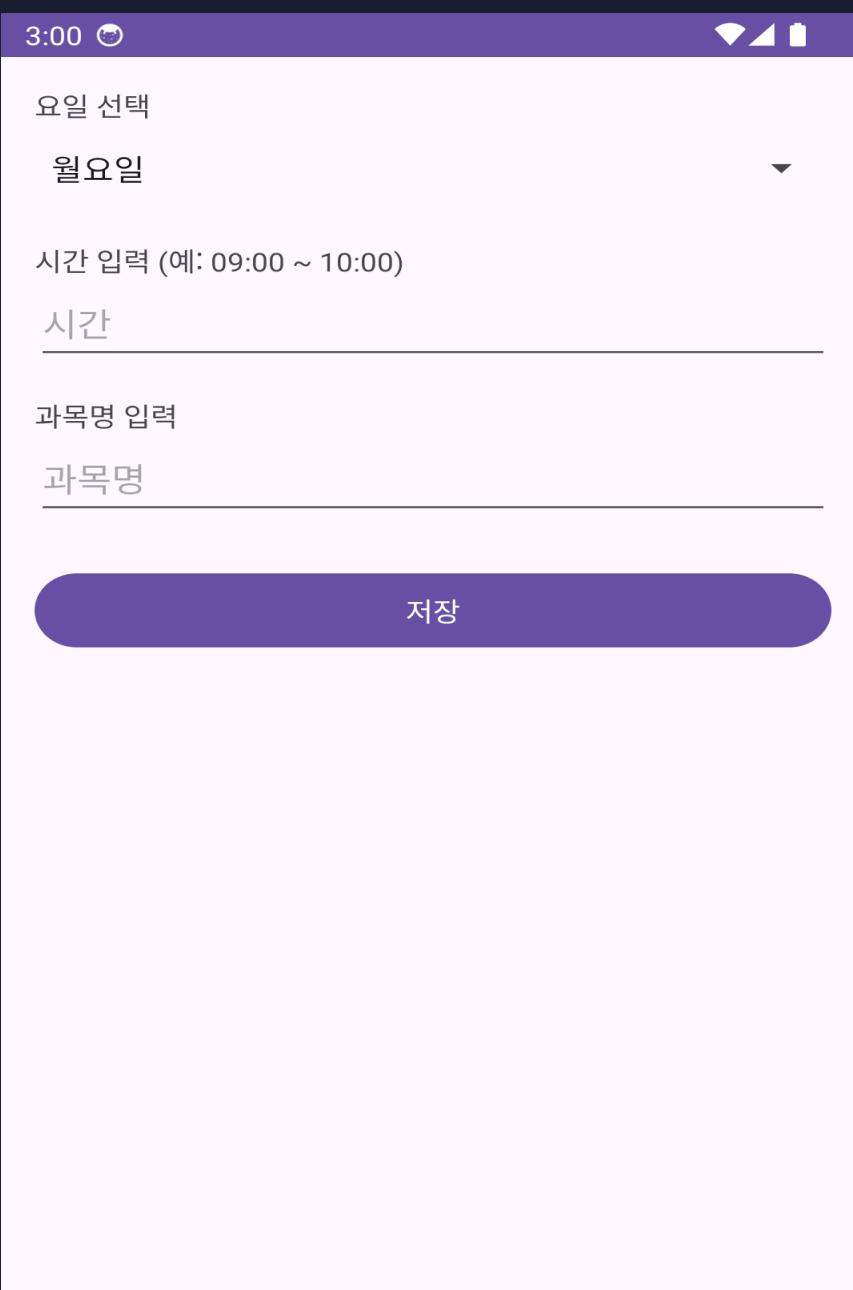
23

24

25

26

시간표 추가 화면 - AddScheduleActivity



- 사용자가 새로운 시간표 항목 입력
- 曜일은 Spinner로 선택
- 시간과 과목은 직접 입력
- 저장 버튼 누르면 메인화면으로 돌아감



mobile

programming

project

9

10

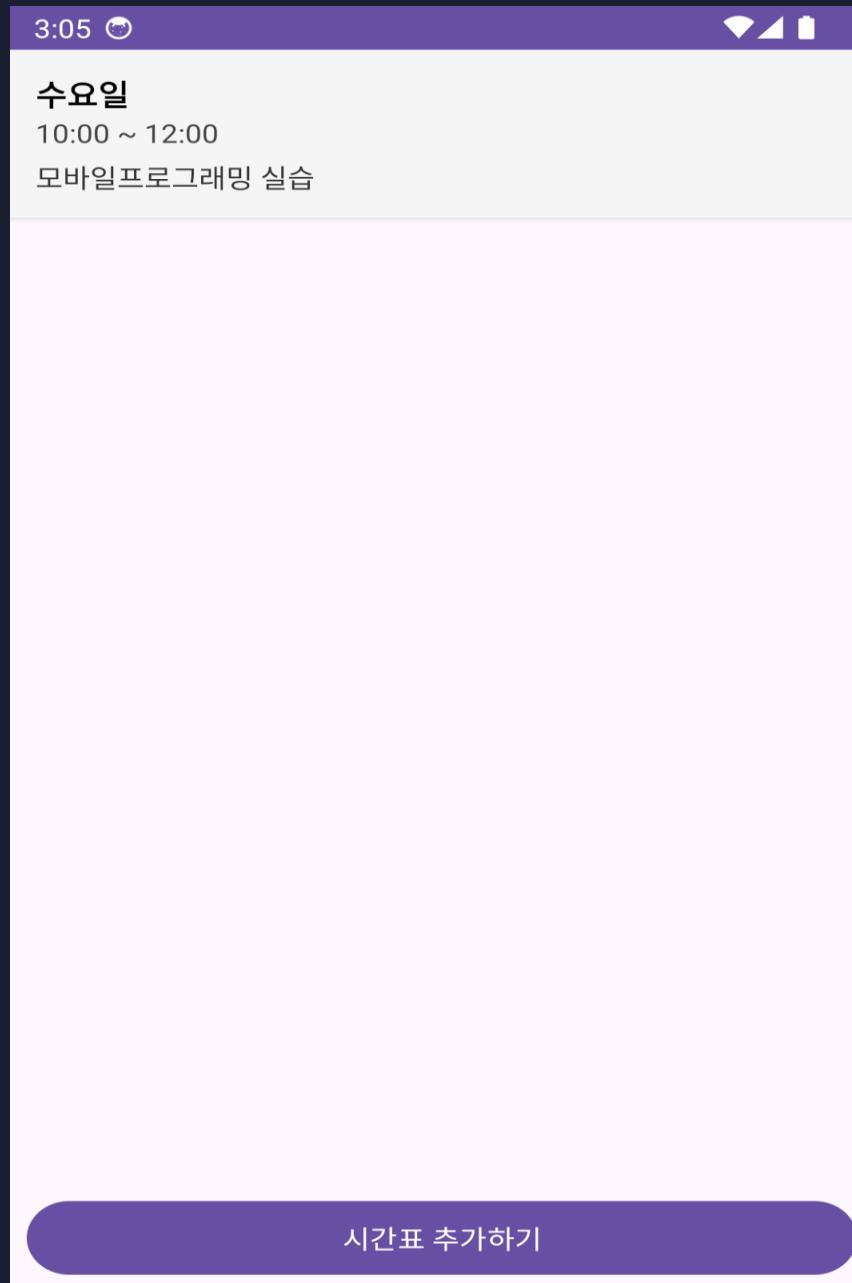
11

삭제 기능 화면 - 롱클릭 시 다이얼로그

12



13



14

15

16

17

18

19

20

21

22

23

24

25

26

- 메인화면에서 시간표 항목 길게 누르면 삭제 여부를 묻는 다이얼로그 생성
- 삭제를 누르면 해당 항목이 삭제되고 리스트 자동 갱신

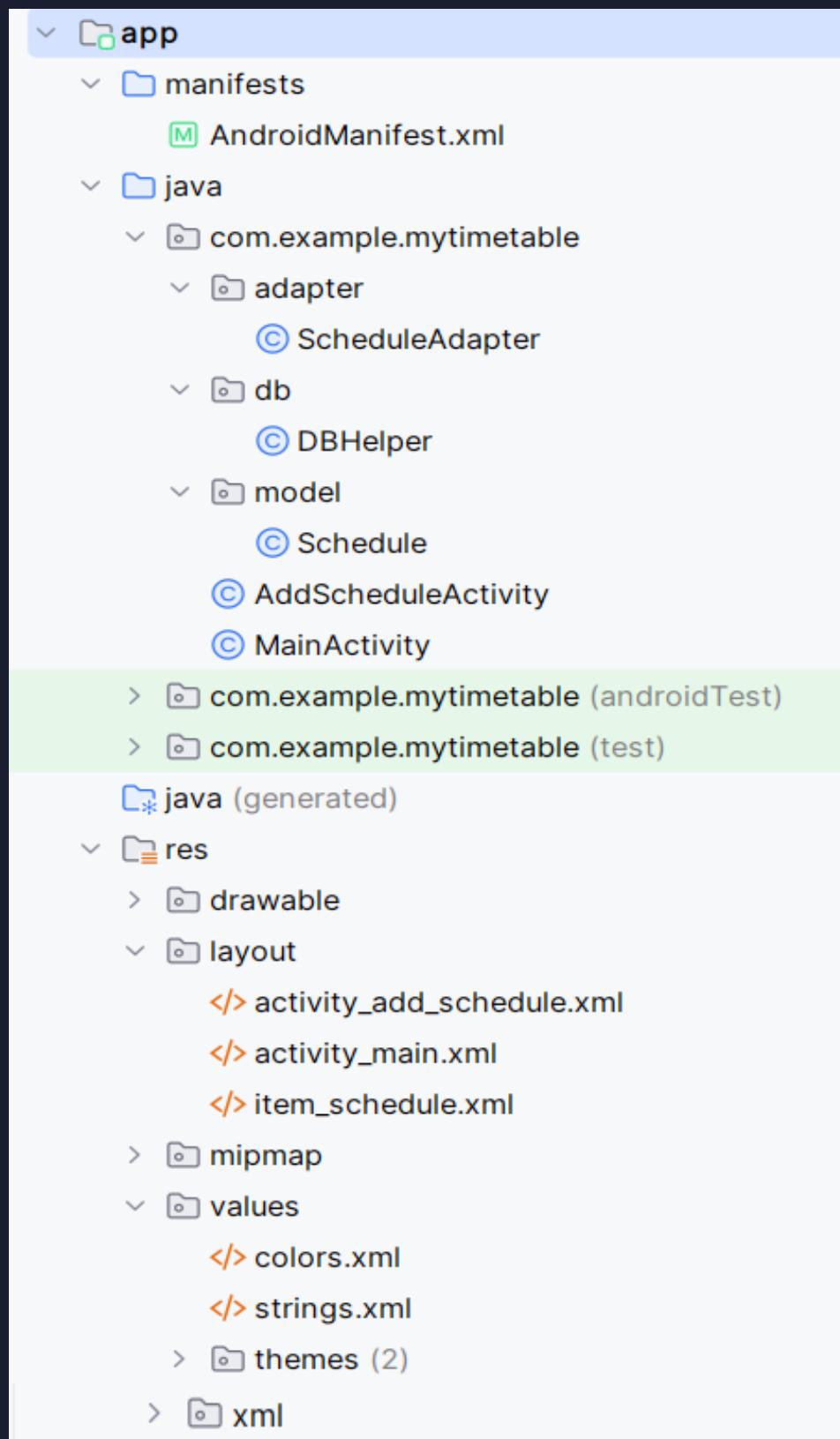


mobile

programming

project

앱 구조



1. **AndroidManifest.xml**
2. **strings.xml**
3. **activity_add_schedule.xml**
4. **activity_main.xml**
5. **item_schedule.xml**
6. **Schedule.java (데이터 클래스)**
7. **DBHelper.java (DB 저장/삭제)**
8. **ScheduleAdapter.java (리스트 출력/삭제)**
9. **AddScheduleActivity.java (입력 로직)**
10. **MainActivity.java (전체 제어)**

앱 전반구조 -> 데이터 흐름 -> 로직 구조

9



mobile

programming

project

10

AndroidManifest.xml

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.mytimetable">
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="@string/app_name"
9          android:roundIcon="@mipmap/ic_launcher_round"
10         android:supportsRtl="true"
11         android:theme="@style/Theme.MyTimetable">
12
13         <activity
14             android:name=".MainActivity"
15             android:exported="true">
16             <intent-filter>
17                 <action android:name="android.intent.action.MAIN" />
18                 <category android:name="android.intent.category.LAUNCHER" />
19             </intent-filter>
20         </activity>
21
22         <!-- 시간표 추가 화면 -->
23         <activity android:name=".AddScheduleActivity" />
24     </application>
25
26 </manifest>

```

- <activity> 등록 (앱 화면 정의)

- 앱 실행시 처음 열리는 MainActivity를 등록하는 코드
- Intent-filter를 통해 앱의 진입점이라는 것을 알림
- Android 12 이상은 Android:exported="true"를 명시해야

앱이 실행

- AddScheduleActivity 등록

시간표를 입력하는 화면인 AddSchedule이 등록돼 있어야,

MainActivity에서 화면 전환 가능



mobile

programming

project

9

10

strings.xml

12

```
1 <resources>
2   ① <string name="app_name">MyTimetable</string>
3   ② <string name="btn_save">저장</string>
4   ③ <string-array name="days_of_week">
5     <item>월요일</item>
6     <item>화요일</item>
7     <item>수요일</item>
8     <item>목요일</item>
9     <item>금요일</item>
10    <item>토요일</item>
11    <item>일요일</item>
12  </string-array>
13
14
15 </resources>
```

13

14

15

16

17

18

19

20

21

22

23

24

25

26

1. 앱에서 사용하는 모든 문자열을 한 곳에서 관리

- 앱 전반에서 사용하는 텍스트를 모아놓은 리소스 파일
- UI 요소의 텍스트를 직접 코드에 쓰는 대신, 여기서 참조하여

유지보수 용이하게 개발

2. 버튼 텍스트 재사용 가능하게 정의

- 버튼에 표시되는 텍스트를 여기서 정의하여 화면마다 일관성을 유지

3.曜일 선택용 배열 정의

- Spinner에曜일을 선택하게 만들기 위해 string-array를 정의하여 UI에서 바로 사용할 수 있게 개발

9



mobile

programming

project

10

activity_add_schedule.xml – (1)

```

12 <TextView android:text="요일 선택" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
13
14 <Spinner
15     android:id="@+id/spinner_day"
16     android:layout_width="match_parent"
17     android:layout_height="wrap_content"
18     android:entries="@array/days_of_week"
19     android:minHeight="48dp"/>
20
21 <TextView android:text="시간 입력 (예: 09:00 ~ 10:00)" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
22 <EditText
23     android:id="@+id/edit_time"
24     android:layout_width="match_parent"
25     android:layout_height="wrap_content"
26     android:hint="시간"
27     android:minHeight="48dp"/>

```

시간표 항목을 입력하는 화면

1. 요일 선택 Spinner

- 드롭다운 형태로 요일 선택
- Strings.xml에 정의한 days_of_week 배열 사용

2. 시간 입력 EditText

- 사용자가 원하는 시간대를 직접 입력
- hint 속성으로 입력 예시 제공
- 추후 TimePicker와 연동 가능

10

activity_add_schedule.xml - (2)

11

```
13 <TextView android:text="과목명 입력" android:lay
14 <EditText
15     android:id="@+id/edit_subject"
16     android:layout_width="match_parent"
17     android:layout_height="wrap_content"
18     android:hint="과목명"
19     android:minHeight="48dp"/>
```

12

13

```
21 <Button
22     android:id="@+id/btn_save"
23     android:layout_width="match_parent"
24     android:layout_height="wrap_content"
25     android:text="저장"
26     android:layout_marginTop="24dp"/>
```

14

15

16

17

18

19

20

21

22

23

24

25

26

3. 과목 입력 EditText

- 과목명을 직접 입력
- 필수 입력 요소로 사용됨 (AddScheduleActivity.java에서
빈 값 체크 있음)

4. 저장 버튼

- 사용자가 입력을 마친 후 데이터 저장
- AddScheduleActivity.java에서 버튼 클릭시 SQLite에
저장되고 앱 메인화면으로 돌아감



mobile

programming

project

9

10

activity_main.xml

11

12

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recycler_schedule"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_above="@+id/btn_add"/>

    <Button
        android:id="@+id/btn_add"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="시간표 추가하기"
        android:layout_alignParentBottom="true"
        android:layout_margin="8dp"/>

</RelativeLayout>
```

13

14

15

16

17

18

19

20

21

22

23

24

25

26

- 앱의 메인화면으로 사용자가 입력한 시간표 목록을 확인 가능
- RecyclerView를 통해 리스트 형태로 출력
- RelativeLayout을 사용해 하단 버튼과 리스트가 충돌 없이 배치

1. RecyclerView 사용

- 입력된 시간표를 리스트 형태로 출력
- Adapter와 연결되어 실시간 데이터 반영

2. 추가 버튼(하단 고정)

- 사용자가 새 시간표를 추가할 수 있도록 하단에 배치
- 버튼 클릭시 AddScheduleActivity로 이동

11 Item_schedule.xml

```
13 <TextView  
14     android:id="@+id/text_day"  
15     android:layout_width="wrap_content"  
16     android:layout_height="wrap_content"  
17     android:text="월요일"  
18     android:textSize="16sp"  
19     android:textStyle="bold"  
20     android:textColor="#000000" />
```

```
14 <TextView  
15     android:id="@+id/text_time"  
16     android:layout_width="wrap_content"  
17     android:layout_height="wrap_content"  
18     android:text="09:00 ~ 10:00"  
19     android:textSize="14sp"  
20     android:textColor="#444444" />
```

```
22 <TextView  
23     android:id="@+id/text_subject"  
24     android:layout_width="wrap_content"  
25     android:layout_height="wrap_content"  
26     android:text="자료구조"  
27     android:textSize="14sp"  
28     android:textColor="#333333"  
29     android:layout_marginTop="4dp" />
```

- 시간표 목록에 표시되는 각 한 줄(하나의 수업 정보)의 모양 정의

1. 요일 표시 TextView

첫번째 줄은 수업이 있는 요일을 굵은 글씨로 표시하여 어떤 요일인지
빠르게 구분할 수 있도록 구성

2. 시간 표시 TextView

두번째 줄에는 수업 시간을 출력하며, 가독성을 고려해 글자 크기 설정

3. 과목명 표시 TextView

마지막 줄에는 수업 과목명을 표시하며, 위쪽과의 간격을 두어 구분감

9



mobile

programming

project

10

Schedule.java – (1)

11

```

12 package com.example.mytimetable.model;

13
14 public class Schedule {
15     private int id;
16     private String day;
17     private String time;
18     private String subject;

19     public Schedule(int id, String day, String time, String subject) {
20         this.id = id;
21         this.day = day;
22         this.time = time;
23         this.subject = subject;
24     }

25     public int getId() { return id; }
26     public String getDay() { return day; }
27     public String getTime() { return time; }
28     public String getSubject() { return subject; }

```

19

20

21

22

23

24

25

26

- 앱에서 사용하는 시간표 데이터 구조(모델 클래스)
- 시간표의 각 항목을 하나의 객체로 표현
- 요일, 시간, 과목명, 고유한 ID 저장

1. 클래스가 정의된 위치

- model 패키지에 위치
- 앱에서 사용하는 데이터 구조만 따로 정리해 유지보수와 가독성 높임

2. 멤버 변수 정의

- 시간표 항목 하나를 표현하기 위해 4개의 멤버 변수 사용
- Id는 각 항목을 구분하는 고유 값
- day, time, subject는 각각 요일, 시간, 과목 정보 저장

9



mobile

programming

project

10

Schedule.java – (2)

```

12 package com.example.mytimetable.model;

13
14 8 usages
15 public class Schedule {
16     2 usages
17     private int id;
18     2 usages
19     private String day;
20     2 usages
21     private String time;
22     2 usages
23     private String subject;
24
25     1 usage
26     public Schedule(int id, String day, String time, String subject) {
27         this.id = id;
28         this.day = day;
29         this.time = time;
30         this.subject = subject;
31     }
32
33     1 usage
34     public int getId() { return id; }
35     1 usage
36     public String getDay() { return day; }
37     1 usage
38     public String getTime() { return time; }
39     1 usage
40     public String getSubject() { return subject; }
41 }
```

3. 생성자 정의

생성자를 통해 객체를 만들 때 네 가지 값을 한번에 넣을 수 있도록 설계

4. Getter 메서드

외부 클래스에서는 이 객체의 값을 직접 접근하지 않고, getter 메서드를 통해
값을 읽을 수 있도록 캡슐화를 적용



mobile

programming

project

9

10

11

DBHelper.java – (1)

```
12 ① public class DBHelper extends SQLiteOpenHelper {  
13  
14 ②     1 usage  
15     public static final String DB_NAME = "timetable.db";  
16     1 usage  
17     public static final int DB_VERSION = 1;  
18  
19 ③     2 usages  
20     >     public DBHelper(Context context) { super(context, DB_NAME, null, DB_VERSION); }  
21  
22     @Override  
23     public void onCreate(SQLiteDatabase db) {  
24         db.execSQL(  
25             "CREATE TABLE schedule (" +  
26                 "id INTEGER PRIMARY KEY AUTOINCREMENT, " +  
27                 "day TEXT, " +  
28                 "time TEXT, " +  
29                 "subject TEXT")  
30     };  
31  
32     1 usage  
33     public void deleteSchedule(int id) {  
34         SQLiteDatabase db = this.getWritableDatabase();  
35         db.delete("schedule", "id=?", new String[]{String.valueOf(id)});  
36         db.close();  
37     }  
38
```

- SQLite 데이터베이스를 생성하고 관리하는 핵심 클래스
- 시간표 정보를 저장하는 schedule 테이블 생성
- 데이터를 삭제할 수 있는 기능 제공

1. SQLiteOpenHelper 상속

- 안드로이드에서 제공하는 SQLiteOpenHelper 클래스를 상속받아 데이터베이스 생성과 버전 관리 담당

2. 데이터베이스 이름과 버전

- 데이터베이스 파일의 이름은 timetable.db로 지정, 현재는 초기 버전인 1

3. 테이블 생성 – onCreate()

- 앱이 처음 실행될 때 onCreate() 메서드가 호출되어 schedule 테이블 생성
- 각 항목은 id, day, time, subject로 구성, id는 자동 증가하는 기본 키



mobile

programming

project

9

10

11

DBHelper.java – (2)

12

```
(4) 13 public void deleteSchedule(int id) {  
14     SQLiteDatabase db = this.getWritableDatabase();  
15     db.delete("schedule", "id=?", new String[]{String.valueOf(id)});  
16     db.close();  
17 }
```

18

19

no usages

20

```
(5) 21 @Override  
22 23 public void onUpgrade(SQLiteDatabase db, int oldV, int newV) {  
24     db.execSQL("DROP TABLE IF EXISTS schedule");  
25     onCreate(db);  
26 }
```

4. 삭제 기능 구현

- 사용자가 항목을 삭제할 때 호출되는 메서드
- 해당 id에 해당하는 데이터를 DB에서 삭제

5. 버전 변경시 onUpgrade() 처리

- 데이터베이스 버전이 바뀔 경우 기존 테이블을 삭제하고 새로 생성해 앱이 최신 구조를 유지할 수 있도록 개발

26



mobile

programming

project

9

10

ScheduleAdapter.java – (1)

```
12 ① public class ScheduleAdapter extends RecyclerView.Adapter<ScheduleAdapter.ViewHolder> {  
13  
14     3 usages  
15     private List<Schedule> scheduleList;  
16  
17     3 usages  
18     private OnItemDeleteListener onItemDeleteListener;  
19  
20     1 usage  
21     public ScheduleAdapter(List<Schedule> list) { this.scheduleList = list; }  
22  
23  
24 ② 4 usages  
25     public static class ViewHolder extends RecyclerView.ViewHolder {  
26         2 usages  
27         TextView day, time, subject;  
28  
29         1 usage  
30         public ViewHolder(View view) {  
31             super(view);  
32             day = view.findViewById(R.id.text_day);  
33             time = view.findViewById(R.id.text_time);  
34             subject = view.findViewById(R.id.text_subject);  
35         }  
36     }  
37 }
```

- 시간표 데이터를 리스트 형태로 출력해주는 RecyclerView의 어댑터 클래스
 - 각 시간표 항목을 화면에 표시하고, 풍클릭시 삭제 다이얼로그를 띄워서 삭제 요청 처리
1. RecyclerView.Adapter 상속
 - RecyclerView.Adapter를 상속해서 리스트 화면에서 데이터를 어떻게 표시할지 정의
 2. ViewHolder 패턴
 - ViewHolder는 시간표 한 줄에서 사용하는 UI 요소들을 저장하는 클래스
 - 매번 뷰를 새로 생성하지 않고 재사용함으로써 성능을 높임
 3. onCreateViewHolder – 레이아웃 연결
 - item_schedule.xml 레이아웃을 읽어와서 각 시간표 항목을 어떻게 보여줄지 정의

ScheduleAdapter.java – (2)

```

12
13     @Override
14     public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
15         View view = LayoutInflater.from(parent.getContext())
16             .inflate(R.layout.item_schedule, parent, attachToRoot: false);
17         return new ViewHolder(view);
18     }
19
20
21     @Override
22     public void onBindViewHolder(ViewHolder holder, int position) {
23         Schedule schedule = scheduleList.get(position);
24
25         holder.day.setText(schedule.getDay());
26         holder.time.setText(schedule.getTime());
27         holder.subject.setText(schedule.getSubject());
28
29         // ✅ 롱클릭 시 삭제ダイアログ
30         holder.itemView.setOnLongClickListener(v -> {
31             AlertDialog.Builder builder = new AlertDialog.Builder(holder.itemView.getContext());
32             builder.setTitle("삭제 확인")
33                 .setMessage("이 시간표를 삭제할까요?")
34                 .setPositiveButton( text: "삭제", (dialog, which) -> {
35                     if (onItemDeleteListener != null) {
36                         onItemDeleteListener.onDelete(schedule.getId());
37                     }
38                 })
39                 .setNegativeButton( text: "취소", listener: null)
40                 .show();
41
42             return true;
43         });
44     }
45

```

3. onCreateViewHolder – 레이아웃 연결

- item_schedule.xml 레이아웃을 읽어와서 각 시간표 항목을 어떻게 보여줄지 정의

4. onBindViewHolder – 데이터 바인딩

- 데이터의 요일, 시간, 과목 정보를 각각의 TextView에 연결해서 화면에 표시

5. 롱클릭시 삭제ダイアログ 구현

- 사용자가 리스트 항목을 길게 누르면 삭제 여부를 묻는ダイアログ가 표시되고, 확인을 누르면 외부에서 등록된 삭제 리스너를 통해 DB에서 데이터를 삭제



mobile

programming

project

9

10

11

ScheduleAdapter.java – (3)

12

```
13 @Override  
14 public int getItemCount() { return scheduleList.size(); }  
15 // ✅ 인터페이스 정의  
16 ⑥ 2 usages  
17 public interface OnItemDeleteListener {  
18     1 usage  
19         void onDelete(int id);  
20     }  
21 // ✅ 리스너 등록 메서드  
22 1 usage  
23 public void setOnItemDeleteListener(OnItemDeleteListener listener) {  
24     this.onItemDeleteListener = listener;  
25 }  
26 }
```

6. 삭제 리스너 인터페이스 정의

- 삭제 처리는 Adapter 내부에서 직접하지 않고, 외부(MainActivity)에서 처리할 수 있도록 인터페이스를 정의해 구조를 분리



AddScheduleActivity.java – (1)

```

spinnerDay = findViewById(R.id.spinner_day);          ①
editTime = findViewById(R.id.edit_time);
editSubject = findViewById(R.id.edit_subject);
btnSave = findViewById(R.id.btn_save);

dbHelper = new DBHelper( context: this);            ②

btnSave.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String day = spinnerDay.getSelectedItem().toString();
        String time = editTime.getText().toString();
        String subject = editSubject.getText().toString();

        if (time.isEmpty() || subject.isEmpty()) {
            Toast.makeText( context: AddScheduleActivity.this, text: "모든 항목을 입력하세요", Toast.LENGTH_SHORT).show();
            return;
        }

        SQLiteDatabase db = dbHelper.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put("day", day);
        values.put("time", time);
        values.put("subject", subject);

        db.insert( table: "schedule", nullColumnHack: null, values);
        db.close();

        Toast.makeText( context: AddScheduleActivity.this, text: "저장되었습니다!", Toast.LENGTH_SHORT).show();
    }
})

```

- 시간표 항목을 입력하는 화면

- 사용자가 요일, 시간, 과목명을 입력한 뒤 저장 버튼을 누르면,

입력한 정보가 SQLite에 저장되고 메인화면으로 돌아감

1. UI 요소 초기화

- 레이아웃 XML 파일(activity_add_schedule.xml)에 있는 Spinner, EditText, Button을 Java 코드에서 연결해서 사용할 수 있도록 초기화

2. DBHelper 초기화

- DB에 데이터를 저장하기 위해 DBHelper 클래스를 객체로 생성해 사용

AddScheduleActivity.java – (2)

```

spinnerDay = findViewById(R.id.spinner_day);
editTime = findViewById(R.id.edit_time);
editSubject = findViewById(R.id.edit_subject);
btnSave = findViewById(R.id.btn_save);

dbHelper = new DBHelper(context: this);

btnSave.setOnClickListener(new View.OnClickListener() { ③
    @Override
    public void onClick(View view) {
        String day = spinnerDay.getSelectedItem().toString();
        String time = editTime.getText().toString();
        String subject = editSubject.getText().toString();

        if (time.isEmpty() || subject.isEmpty()) { ④
            Toast.makeText(context: AddScheduleActivity.this, text: "모든 항목을 입력하세요", Toast.LENGTH_SHORT).show();
            return;
        }

        SQLiteDatabase db = dbHelper.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put("day", day);
        values.put("time", time);
        values.put("subject", subject);

        db.insert(table: "schedule", nullColumnHack: null, values);
        db.close();

        Toast.makeText(context: AddScheduleActivity.this, text: "저장되었습니다!", Toast.LENGTH_SHORT).show(); ⑥
        // 메인화면으로 돌아가기
        finish();
    }
}

```

3. 저장 버튼 클릭 이벤트 처리

- 사용자가 저장 버튼을 클릭하면 동작할 코드를 정의

4. 입력값 검증 및 저장 처리

- 입력값이 비어 있을 경우 경고 메시지를 띄우고 저장하지 않도록 예외 처리

5. SQLite에 데이터 저장

- ContentValues에 입력된 데이터를 담아서 schedule 테이블에 insert() 메서드로 저장

6. 저장 완료 후 종료 및 Toast 출력

- 저장이 완료되면 사용자에게 메시지를 보여주고, finish()를 통해 메인 화면으로 자동 복귀



mobile

programming

project

9

10

MainActivity.java – (1)

11

12

13

14

15

16

```
recyclerView.setLayoutManager(new LinearLayoutManager(context: this));
recyclerView.setAdapter(adapter);
```

17

18

19

```
private void loadSchedules() {
    scheduleList.clear();

    SQLiteOpenHelper db = dbHelper.getReadableDatabase();
    Cursor cursor = db.rawQuery(sql: "SELECT * FROM schedule", selectionArgs: null);

    while (cursor.moveToNext()) {
        int id = cursor.getInt(cursor.getColumnIndexOrThrow(s: "id"));
        String day = cursor.getString(cursor.getColumnIndexOrThrow(s: "day"));
        String time = cursor.getString(cursor.getColumnIndexOrThrow(s: "time"));
        String subject = cursor.getString(cursor.getColumnIndexOrThrow(s: "subject"));

        Schedule schedule = new Schedule(id, day, time, subject);
        scheduleList.add(schedule);
    }
}
```

20

21

22

23

24

25

26

전체 흐름을 제어하는 중심 액티비티, 앱의 메인 화면

저장된 시간표를 RecyclerView로 출력

새로운 시간표를 추가하거나 삭제하는 기능 전체를 제어

1. RecyclerView 설정 + Adapter 연결

- 시간표 항목을 리스트 형태로 보여주기 위해 RecyclerView를 사용
- 데이터를 화면에 출력하기 위해 ScheduleAdapter를 연결

2. SQLite DB에서 데이터 불러오기

- DB에서 시간표 데이터를 가져오는 loadSchedules() 메서드를 통해
앱 실행 시마다 최신 데이터를 불러와 리스트에 표시



mobile

programming

project

MainActivity.java – (2)

```
9  
10  
11  
12    adapter.setOnItemDeleteListener(id -> {  
13        dbHelper.deleteSchedule(id); // DB에서 삭제  
14        LoadSchedules();           // 목록 새로고침  
15    });  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26
```

```
17  
18    btnAdd.setOnClickListener(new View.OnClickListener() {  
19        @Override  
20        public void onClick(View view) {  
21            // 시간표 추가 액티비티로 이동  
22            Intent intent = new Intent(packageContext: MainActivity.this, AddScheduleActivity.class);  
23            startActivity(intent);  
24        }  
25    });  
26
```

```
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
5510  
5511  
5512  
5513  
5514  
5515  
5516  
5517  
5518  
5519  
5520  
5521  
5522  
5523  
5524  
5525  
5526  
5527  
5528  
5529  
5530  
5531  
5532  
5533  
5534  
5535  
5536  
5537  
5538  
5539  
55310  
55311  
55312  
55313  
55314  
55315  
55316  
55317  
55318  
55319  
55320  
55321  
55322  
55323  
55324  
55325  
55326  
55327  
55328  
55329  
55330  
55331  
55332  
55333  
55334  
55335  
55336  
55337  
55338  
55339  
55340  
55341  
55342  
55343  
55344  
55345  
55346  
55347  
55348  
55349  
55350  
55351  
55352  
55353  
55354  
55355  
55356  
55357  
55358  
55359  
55360  
55361  
55362  
55363  
55364  
55365  
55366  
55367  
55368  
55369  
55370  
55371  
55372  
55373  
55374  
55375  
55376  
55377  
55378  
55379  
55380  
55381  
55382  
55383  
55384  
55385  
55386  
55387  
55388  
55389  
55390  
55391  
55392  
55393  
55394  
55395  
55396  
55397  
55398  
55399  
553100  
553101  
553102  
553103  
553104  
553105  
553106  
553107  
553108  
553109  
553110  
553111  
553112  
553113  
553114  
553115  
553116  
553117  
553118  
553119  
553120  
553121  
553122  
553123  
553124  
553125  
553126  
553127  
553128  
553129  
553130  
553131  
553132  
553133  
553134  
553135  
553136  
553137  
553138  
553139  
553140  
553141  
553142  
553143  
553144  
553145  
553146  
553147  
553148  
553149  
553150  
553151  
553152  
553153  
553154  
553155  
553156  
553157  
553158  
553159  
553160  
553161  
553162  
553163  
553164  
553165  
553166  
553167  
553168  
553169  
553170  
553171  
553172  
553173  
553174  
553175  
553176  
553177  
553178  
553179  
553180  
553181  
553182  
553183  
553184  
553185  
553186  
553187  
553188  
553189  
553190  
553191  
553192  
553193  
553194  
553195  
553196  
553197  
553198  
553199  
553200  
553201  
553202  
553203  
553204  
553205  
553206  
553207  
553208  
553209  
553210  
553211  
553212  
553213  
553214  
553215  
553216  
553217  
553218  
553219  
553220  
553221  
553222  
553223  
553224  
553225  
553226  
553227  
553228  
553229  
553230  
553231  
553232  
553233  
553234  
553235  
553236  
553237  
553238  
553239  
553240  
553241  
553242  
553243  
553244  
553245  
553246  
553247  
553248  
553249  
553250  
553251  
553252  
553253  
553254  
553255  
553256  
553257  
553258  
553259  
553260  
553261  
553262  
553263  
553264  
553265  
553266  
553267  
553268  
553269  
553270  
553271  
553272  
553273  
553274  
553275  
553276  
553277  
553278  
553279  
553280  
553281  
553282  
553283  
553284  
553285  
553286  
553287  
553288  
553289  
553290  
553291  
553292  
553293  
553294  
553295  
553296  
553297  
553298  
553299  
553300  
553301  
553302  
553303  
553304  
553305  
553306  
553307  
553308  
553309  
553310  
553311  
553312  
553313  
553314  
553315  
553316  
553317  
553318  
553319  
553320  
553321  
553322  
553323  
553324  
553325  
553326  
553327  
553328  
553329  
553330  
553331  
553332  
553333  
553334  
553335  
553336  
553337  
553338  
553339  
5533310  
5533311  
5533312  
5533313  
5533314  
5533315  
5533316  
5533317  
5533318  
5533319  
55333110  
55333111  
55333112  
55333113  
55333114  
55333115  
55333116  
55333117  
55333118  
55333119  
553331110  
553331111  
553331112  
553331113  
553331114  
553331115  
553331116  
553331117  
553331118  
553331119  
5533311110  
5533311111  
5533311112  
5533311113  
5533311114  
5533311115  
5533311116  
5533311117  
5533311118  
5533311119  
55333111110  
55333111111  
55333111112  
55333111113  
55333111114  
55333111115  
55333111116  
55333111117  
55333111118  
55333111119  
553331111110  
553331111111  
553331111112  
553331111113  
553331111114  
553331111115  
553331111116  
553331111117  
553331111118  
553331111119  
5533311111110  
5533311111111  
5533311111112  
5533311111113  
5533311111114  
5533311111115  
5533311111116  
5533311111117  
5533311111118  
5533311111119  
55333111111110  
55333111111111  
55333111111112  
55333111111113  
55333111111114  
55333111111115  
55333111111116  
55333111111117  
55333111111118  
55333111111119  
553331111111110  
553331111111111  
553331111111112  
553331111111113  
553331111111114  
553331111111115  
553331111111116  
553331111111117  
553331111111118  
553331111111119  
5533311111111110  
5533311111111111  
5533311111111112  
5533311111111113  
5533311111111114  
5533311111111115  
5533311111111116  
5533311111111117  
5533311111111118  
5533311111111119  
55333111111111110  
55333111111111111  
55333111111111112  
55333111111111113  
55333111111111114  
55333111111111115  
55333111111111116  
55333111111111117  
55333111111111118  
55333111111111119  
553331111111111110  
553331111111111111  
553331111111111112  
553331111111111113  
553331111111111114  
553331111111111115  
553331111111111116  
553331111111111117  
553331111111111118  
553331111111111119  
5533311111111111110  
5533311111111111111  
5533311111111111112  
5533311111111111113  
5533311111111111114  
5533311111111111115  
5533311111111111116  
5533311111111111117  
5533311111111111118  
5533311111111111119  
5533311
```



mobile

programming

project

감사합니다!