8장. DBMS을 이용한 데이터 영속화

8.1 SQLite을 이용한 영속화

- SQLite (www.sqlite.org)는 오픈소스로 만들어진 관계형 데이터베이스
- data/data/[package_name]/databases 에 저장

8.1.1. SQLiteDatabase 클래스

- SQL 문 수행 은 이 클래스의 함수를 이용

```
SQLiteDatabase db=openOrCreateDatabase("memodb", MODE_PRIVATE, null);
```

- execSQL(String sql): insert, update 등 select 문이 아닌 나머지 SQL 수행
- rawQuery(String sql, String[] selectionArgs): select SQL 수행

```
db.execSQL("insert into tb_memo (title, content) values (?,?)", new String[]{title, content});
```

```
Cursor cursor= db.rawQuery("select title, content from tb_memo  order by _id desc limit 1", null);
```

- Cursor는 선택된 행(row)의 집합 객체
- moveToNext(): 순서상으로 다음 행 선택
- moveToFirst(): 가장 첫 번째 행 선택
- moveToLast(): 가장 마지막 행 선택
- moveToPrevious(): 순서상으로 이전 행 선택

8.1 SQLite을 이용한 영속화

```
while (cursor.moveToNext()){
   titleView.setText(cursor.getString(0));
   contentview.setText(cursor.getString(1));
}
```

8.1.2. SQLiteOpenHelper 클래스

- 테이블 생성이나 스키마 변경 등의 작업

```
public class DBHelper extends SQLiteOpenHelper {

   public static final int DATABASE_VERSION = 1;

   public DBHelper(Context context) {
      super(context, "memodb", null, DATABASE_VERSION);
   }
   @Override
   public void onCreate(SQLiteDatabase db) {
      //.....
   }
   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion,
                         int newVersion) {
      //.....
   }
}
```

8.1 SQLite을 이용한 영속화

- onCreate( ): 앱이 설치된 후 SQLiteOpenHelper가 최초로 이용되는 순간 한 번 호출
- onUpgrade( ): 데이터베이스 버전이 변경될 때마다 호출

```
DBHelper helper = new DBHelper(this);
SQLiteDatabase db = helper.getWritableDatabase();
```

8.1.3. insert( ), query( ), update( ), delete( ) 함수 이용

- insert(String table, String nullColumnHack, ContentValues values)
- update(String table, ContentValues values, String whereClause, String[] whereArgs)
- delete(String table, String whereClause, String[] whereArgs)
- query(String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy, String limit)

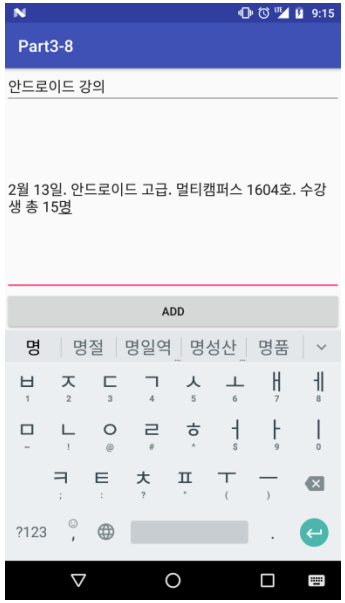
```
ContentValues values=new ContentValues();
values.put("name", "kkang");
values.put("phone", "0100000");
db.insert("USER_TB", null, values);
```

```
Cursor c=db.query("USER_TB", new String[]{"name", "phone"},  "ID=?", new String[]{"kkang"},  null, null,null);
```

Step by Step 8-1 – SQLite 실습

- SQLite를 이용해 데이터 영속화하는 실습
- 간단한 메모장 기능을 가정해서 작성
- 유저가 입력한 글을 데이터베이스에 저장하고 저장된 데이터를 select 해서 화면에 출력

1. Module 생성
2. ReadDBActivity 생성
3. layout xml 파일 복사
4. DBHelper 클래스 작성
5. MainActivity 작성
6. ReadDBActivity 작성
7. 실행



8.2 Realm을 이용한 데이터 영속화

8.2.1. Realm 소개

- https://realm.io에서 오픈소스로 만들어지고 있는 데이터베이스
- Realm은 자바 객체를 해석해 그 객체의 데이터를 그 대로 저장, 획득

8.2.2. Realm 사용 설정

- 프로젝트 수준의 그레이들 파일에 의존성 (dependency)을 설정

```
dependencies {
    classpath 'com.android.tools.build:gradle:2.2.3'
    classpath "io.realm:realm-gradle-plugin:2.2.0"
}
```

- 모듈의 gradle 파일

```
apply plugin: 'realm-android'
```

8.2.3. Realm 사용

- Realm이 관리할 VO(Value-Object) 클래스

```
public class MemoVO extends RealmObject {
    public String title;
    public String content;
}
```

8.2 Realm을 이용한 데이터 영속화

- Realm 객체를 획득

```
Realm.init(this);
Realm mRealm=Realm.getDefaultInstance();
```

- 저장

```
mRealm.executeTransaction(new Realm.Transaction() {
    @Override
    public void execute(Realm realm) {
        // Add a person
        MemoVO vo = realm.createObject(MemoVO.class);
        vo.title=title;
        vo.content=content;
    }
});
```

- 데이터를 획득

```
MemoVO vo = mRealm.where(MemoVO.class).equalTo("title", "hello").findFirst();
```

8.2 Realm을 이용한 데이터 영속화

- RealmQuery

- findAll()
- findAllSorted(String fieldName)
- findAllSorted(String[] fieldNames, Sort[] sortOrders)
- findAllSorted(String fieldName, Sort sortOrder)
- findAllSorted(String fieldName1, Sort sortOrder1, String fieldName2, Sort sortOrder2)
- findFirst()

```
mRealm.where(MemoVO.class).findAllSorted("title", Sort.DESCENDING);
```

- 조건을 명시할 때 equalTo( ) 함수 이외에 between( ), beginsWith( ), endsWith( ), isNotNull ( ), in ( ), isNull ( ), lessThan ( ), lessThanOrEqualTo( ), contains ( ), like ( ) 등의 다양한 함수를 제 공

- RealmResults

```
RealmResults<MemoVO> results = mRealm.where(MemoVO.class) .equalTo("title", "Tiger").findAll();
```

8.2 Realm을 이용한 데이터 영속화

- 데이터를 삭제

```
MemoVO vo = mRealm.where(MemoVO.class).equalTo("title", "hello").findFirst();
vo.deleteFromRealm();
```

- 모든 데이터를 삭제

```
mRealm.delete(MemoVO.class);
```

Step by Step 8-2 – Realm 실습

SQLite로 진행했던 실습을 그대로 Realm으로 테스트

1. Activity 생성
2. RealmReadActivity 생성
3. layout xml 복사
4. Realm 설정
5. MemoVO.java 작성
6. Lab8_2Activity.java 작성
7. RealmReadActivity.java 작성
8. Lab8_2Activity.java 실행