08

sqflite

네이티브 연동

내부 데이터베이스 이용하기 - sqflite

• pubspec.yaml

```
dependencies:
  sqflite: ^2.0.2+1
```

데이터베이스 열기 — openDatabase()

- 테이블 내용을 파일에 저장하므로 먼저 데이터베이스 파일을 열어야 합니다. 이때 openDatabase() 함수를 이용

• 데이터베이스 열기

```
var db = await openDatabase('my_db.db');
```

내부 데이터베이스 이용하기 - sqflite

데이터베이스 열기 — openDatabase()

- openDatabase() 함수의 version 매개변숫값은 개발자가 지정하는 데이터베이스 버전
- 버전을 변경할 수 있는데 변경하면 onUpgrade 매개변수에 지정한 함수가 자동으로 호출
- onCreate 매개변수에 등록하는 함수는 앱을 설치한 후 openDatabase()로 처음 데이터베이스를 이용할 때 딱 한 번 호출

• 버전 정보와 작업 지정하기

```
var db = await openDatabase(
    "my_db.db",
    version: 1,
    onCreate: (Database db, int version) async {
    },
    onUpgrade: (Database db, int oldVersion, int newVersion) async {
    }
);
```

내부 데이터베이스 이용하기 - sqflite

쿼리 실행하기 — execute()

- SQL 문을 실행할 때는 Database 객체가 제공하는 execute()와 rawQuery(), rawInsert(), rawUpdate, rawDelete() 함수를 이용
- execute() 함수는 매개변수에 SQL 문을 문자열로 지정하여 모든 SQL 문을 실행
- execute() 함수의 반환값은 없습니다.

• 테이블 만들기

```
await db.execute('''
    CREATE TABLE Test (
      id INTEGER PRIMARY KEY,

      name TEXT,
      value INTEGER,
      num REAL
    )
  ''');
```

내부 데이터베이스 이용하기 - sqflite

쿼리 실행하기 — execute()

- rawQuery(), rawInsert(), rawUpdate, rawDelete() 함수는 각각 SELECT, INSERT, UPDATE, DELETE 문을 실행
  - •Future<List<Map<String, Object?>>> rawQuery(String sql, [List<Object?>? arguments])
  - •Future<int> rawUpdate(String sql, [List<Object?>? arguments])
  - •Future<int> rawInsert(String sql, [List<Object?>? arguments])
  - •Future<int> rawDelete(String sql, [List<Object?>? arguments])

• 데이터 조회하기

```
List<Map> list = await db.rawQuery('SELECT * FROM Test');
```

• 데이터 삽입하기

```
db.rawInsert(
    'INSERT INTO Test(name, value, num) VALUES("some name", 1234, 456.789)');
```

내부 데이터베이스 이용하기 - sqflite

트랜잭션 처리 — transaction()

- 데이터베이스의 삽입, 수정, 삭제 등의 작업을 트랜잭션으로 처리한다면 Database의 transaction() 함수를 이용
- Transaction 객체를 이용해 rawInsert(), rawUpdate(), rawDelete() 함수를 실행하면 트랜젝션으로 처리

• 트랜잭션 처리하기

```
await db.transaction((txn) async {
  await txn.rawInsert(
      'INSERT INTO Test(name, value, num) VALUES("some name", 1234,
       456.789)');
  await txn.rawInsert(
      'INSERT INTO Test(name, value, num) VALUES(?, ?, ?)',
      ['another name', 12345678, 3.1416]);
});
```

내부 데이터베이스 이용하기 - sqflite

데이터 저장하기 — query(), insert(), update(), delete()

- Database 객체는 query(), insert(), update(), delete() 함수도 제공
- 이 함수를 이용하면 SQL 문을 직접 작성하지 않아도 됩니다.
  - Future<List<Map<String, Object?>>> query(String table, {bool? distinct, List<String>? columns, String? where, List<Object?>? whereArgs, String? groupBy, String? having, String? orderBy, int? limit, int? offset})
  - • Future<int> insert(String table, Map<String, Object?> values, {String? nullColumnHack, ConflictAlgorithm? conflictAlgorithm})
  - • Future<int> update(String table, Map<String, Object?> values, {String? where, List<Object?>? whereArgs, ConflictAlgorithm? conflictAlgorithm})
  - •Future<int> delete(String table, {String? where, List<Object?>? whereArgs})

내부 데이터베이스 이용하기 - sqflite

데이터 저장하기 — query(), insert(), update(), delete()

• 테이블의 데이터를 추상화한 클래스 구현하기

```
class User {
  int? id;
  String? name;
  String? address;

  // 객체 -> 데이터베이스
  Map<String, Object?> toMap() {
    var map = <String, Object?> {
      "name": name,
      "address": address
    };
    if (id != null) {
      map["id"] = id;
    }
    return map;
  }

  User();
  User.fromData(this.name, this.address);

  // 데이터베이스 -> 객체
  User.fromMap(Map<String, Object?> map) {
    id = map["id"] as int;
    name = map['name'] as String;
    address = map['address'] as String;
  }
}
```

내부 데이터베이스 이용하기 - sqflite

데이터 저장하기 — query(), insert(), update(), delete()

• 클래스로 데이터베이스의 데이터 다루기

```
// 삽입
await db.insert("User", user.toMap());
// 수정
db.update("User", user.toMap(), where: 'id=?', whereArgs: [lastId]);
// 삭제
await db.delete('User', where: 'id=?', whereArgs: [lastId]);
// 쿼리
List<Map> maps = await db.query(
  'User',
  columns: ['id', 'name', 'address'],
);
List<User> users = List.empty(growable: true);
maps.forEach((element) {
  users.add(User.fromMap(element as Map<String, Object?>));
});
```

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare