Glancing Room Database

Wonyoung Choi

Room

“Room provides an abstraction layer over SQLite to allow fluent database access while harnessing the full power of SQLite”

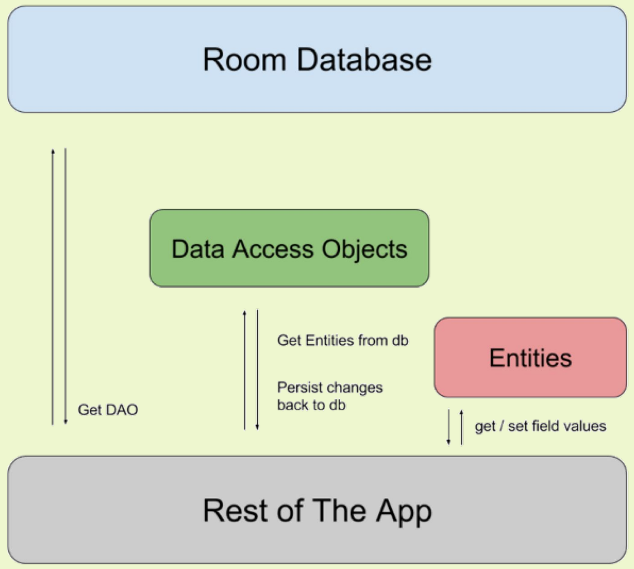
android

Room

- Persistence Library in Jetpack
- Better approach for data persistence
- Easier to work with SQLite
- Decreasing boilerplate code
- Verifying SQL Queries at compile time

Main Components

- Entity
- Dao
- Room Database



Entity

- Representing a table within the database
- *@Entity* annotation to build a table
- Specifying columns by annotations

```kotlin
@Entity(tableName = "persons")
data class Person {
    @PrimaryKey val id:Int,
    @ColumnInfo(name="first_name") val firstName:String?,
    @ColumnInfo(name="last_name") val lastName:String?,
    val address:String?
}
```

android

Dao

- Containing methods used for accessing the database
- Equivalent to *Cursor* class in SQLiteDatabase class

```
@Dao
interface PersonDao {
    @Query("SELECT * FROM person")
    fun personList():List<Person>

    @Insert
    fun insertPerson(person:Person)

    @Update
    fun updatePerson(person:Person)

    @Delete
    fun deletePerson(person:Person)
}
```

Database

- Containing database holder
- *Serving as the main access point*

```
@Database(entities = arrayOf(Person::class), version = 1)
abstract class AppDatabase : RoomDatabase() {
    abstract fun personDao(): PersonDao
    // some pre-configurations…
}
```

```
val db = Room.databaseBuilder(
            applicationContext,
            AppDatabase::class.java, "database-name"
        ).build()
```

```kotlin
val userDao = db.userDao()
val users: List<User> = userDao.getAll()
```