Pasos para integrar SQLDelight en tu aplicación Kotlin Multiplatform

# 1. Configurar el proyecto

Asegúrate de que tu proyecto esté configurado correctamente para Kotlin Multiplatform. Tu `build.gradle.kts` principal debe incluir las configuraciones necesarias.

```kotlin

plugins {

kotlin("multiplatform")

id("com.squareup.sqldelight")

}

kotlin {

android()

iosX64()

iosArm64()

iosSimulatorArm64()

sourceSets {

val commonMain by getting {

dependencies {

implementation("com.squareup.sqldelight:runtime:1.5.3")

}

}

val androidMain by getting {

dependencies {

implementation("com.squareup.sqldelight:android-driver:1.5.3")

}

}

val iosMain by getting {

dependencies {

implementation("com.squareup.sqldelight:native-driver:1.5.3")

}

}

}

}

sqldelight {

database("DoctorNowDatabase") {

packageName = "com.example.doctornow"

}

}

```

# 2. Crear esquemas de base de datos

Crea un archivo `.sq` para definir el esquema de tu base de datos. Por ejemplo, `DoctorNowDatabase.sq`:

```sql

CREATE TABLE Doctor (

id INTEGER NOT NULL PRIMARY KEY,

name TEXT NOT NULL,

specialty TEXT NOT NULL

);

CREATE TABLE Appointment (

id INTEGER NOT NULL PRIMARY KEY,

doctorId INTEGER NOT NULL,

date TEXT NOT NULL,

time TEXT NOT NULL,

FOREIGN KEY(doctorId) REFERENCES Doctor(id)

);

insertDoctor:

INSERT INTO Doctor(name, specialty) VALUES (?, ?);

insertAppointment:

INSERT INTO Appointment(doctorId, date, time) VALUES (?, ?, ?);

selectAllDoctors:

SELECT \* FROM Doctor;

selectAppointmentsForDoctor:

SELECT \* FROM Appointment WHERE doctorId = ?;

```

# 3. Generar el código de SQLDelight

SQLDelight generará las clases necesarias para interactuar con la base de datos. Asegúrate de sincronizar tu proyecto después de agregar el archivo `.sq`.

# 4. Inicializar la base de datos en tu aplicación

En el código común (`commonMain`), crea una función para inicializar la base de datos:

```kotlin

import com.example.doctornow.DoctorNowDatabase

import com.squareup.sqldelight.db.SqlDriver

expect class DatabaseDriverFactory {

fun createDriver(): SqlDriver

}

class Database(databaseDriverFactory: DatabaseDriverFactory) {

val database = DoctorNowDatabase(databaseDriverFactory.createDriver())

val doctorQueries = database.doctorQueries

val appointmentQueries = database.appointmentQueries

}

```

En `androidMain`, implementa `DatabaseDriverFactory`:

```kotlin

import android.content.Context

import com.squareup.sqldelight.android.AndroidSqliteDriver

import com.squareup.sqldelight.db.SqlDriver

actual class DatabaseDriverFactory(private val context: Context) {

actual fun createDriver(): SqlDriver {

return AndroidSqliteDriver(DoctorNowDatabase.Schema, context, "doctornow.db")

}

}

```

En `iosMain`, implementa `DatabaseDriverFactory`:

```kotlin

import com.squareup.sqldelight.db.SqlDriver

import com.squareup.sqldelight.native.driver.NativeSqliteDriver

actual class DatabaseDriverFactory {

actual fun createDriver(): SqlDriver {

return NativeSqliteDriver(DoctorNowDatabase.Schema, "doctornow.db")

}

}

```

# 5. Usar la base de datos

Ahora puedes usar la base de datos en tu aplicación:

```kotlin

class DoctorRepository(private val database: Database) {

fun insertDoctor(name: String, specialty: String) {

database.doctorQueries.insertDoctor(name, specialty)

}

fun getDoctors(): List<Doctor> {

return database.doctorQueries.selectAllDoctors().executeAsList()

}

fun insertAppointment(doctorId: Long, date: String, time: String) {

database.appointmentQueries.insertAppointment(doctorId, date, time)

}

fun getAppointmentsForDoctor(doctorId: Long): List<Appointment> {

return database.appointmentQueries.selectAppointmentsForDoctor(doctorId).executeAsList()

}

}

```

6. Integrar con tu lógica de UI

Por último, integra tu repositorio con la lógica de tu UI en las distintas plataformas. Asegúrate de llamar a las funciones del repositorio para interactuar con la base de datos.

Resumen

Con esta configuración, podrás tener una base de datos local que funciona en ambas plataformas (Android e iOS) sin necesidad de un backend. Si tienes más preguntas o necesitas más detalles en algún punto específico, házmelo saber.