

TP4_Task_1

By Khertish Lobine

03/02/24

Université des Mascareignes

Faculty of Information and Communication
Technology

Department of Applied Computer Science

Course : PMH

To Mr. Shiam Beehary

1. Adding the dependencies

The `sqlite` package provides classes and functions to interact with a SQLite database. The `path` package provides functions to define the location for storing the database on disk.

To add the packages as a dependency, run `flutter pub add`:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\Dev\db_test_app> flutter pub add sqlite path
Resolving dependencies...
  flutter_lints 2.0.3 (3.0.1 available)
  lints 2.1.1 (3.0.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.8.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (from transitive dependency to direct dependency) (1.9.0 available)
+ sqlite 2.3.1
+ sqlite_common 2.5.0+2
+ synchronized 3.1.0+1
  test_api 0.6.1 (0.7.0 available)
```

2. Define the data dog model and opening the database

Define the path to the database file using `getDatabasesPath()` from the `sqlite` package, combined with the `join` function from the `path` package. Open the database with the `openDatabase()` function from `sqlite`.

```
lib > models.dart > Dog
1  class Dog {
2    final int id;
3    final String name;
4    final int age;
5
6    const Dog({
7      required this.id,
8      required this.name,
9      required this.age,
10   });
11 }
```

```

Run | Debug | Profile
void main() async {
  // Avoid errors caused by flutter upgrade.
  // Importing 'package:flutter/widgets.dart' is required.
  WidgetsFlutterBinding.ensureInitialized();
  // Open the database and store the reference.
  final database = openDatabase(
    // Set the path to the database. Note: Using the `join` function from the
    // `path` package is best practice to ensure the path is correctly
    // constructed for each platform.
    join(await getDatabasesPath(), 'doggie_database.db'),
  );
}

```

3. Creating the tables for dog

```

// When the database is first created, create a table to store dogs.
onCreate: (db, version) {
  // Run the CREATE TABLE statement on the database.
  return db.execute(
    'CREATE TABLE dogs(id INTEGER PRIMARY KEY, name TEXT, age INTEGER)',
  );
},
// Set the version. This executes the onCreate function and provides a
// path to perform database upgrades and downgrades.
version: 1,
);

```

4. Inserting dogs into the database by first converting dogs into a map and using the insert() to store the map

```

lib > models.dart > Dog > toString
4   final int age;
5
6   const Dog({
7     required this.id,
8     required this.name,
9     required this.age,
10  });
11
12  // Convert a Dog into a Map. The keys must correspond to the names of the
13  // columns in the database.
14  Map<String, dynamic> toMap() {
15    return {
16      'id': id,
17      'name': name,
18      'age': age,
19    };
20  }
21
22  // Implement toString to make it easier to see information about
23  // each dog when using the print statement.
24  @override
25  String toString() {
26    return 'Dog{id: $id, name: $name, age: $age}';
27  }
28 }

```

```
lib > db_test.dart > main
43     },
44   }
45
46   // Create a Dog and add it to the dogs table
47   var fido = const Dog(
48     id: 0,
49     name: 'Fido',
50     age: 35,
51   );
52
53   await insertDog(fido);
```

5. The list is then retrieved

```
lib > db_test.dart > main > dogs
46   // A method that retrieves all the dogs from the dogs table.
47   Future<List<Dog>> dogs() async {
48     // Get a reference to the database.
49     final db = await database;
50
51     // Query the table for all The Dogs.
52     final List<Map<String, dynamic>> maps = await db.query('dogs');
53
54     // Convert the List<Map<String, dynamic> into a List<Dog>.
55     return List.generate(maps.length, (i) {
56       return Dog(
57         id: maps[i]['id'] as int,
58         name: maps[i]['name'] as String,
59         age: maps[i]['age'] as int,
60       ); // Dog
61     }); // List.generate
62   }
```

7. A dog is then updated in the database

Using whereArgs to pass arguments to a where statement instead of using string interpolation, such as where: "id = \${dog.id}". This helps safeguard against SQL injection attacks.

```
lib > db_test.dart > main > updateDog
61     }); // List.generate
62   }
63
64   Future<void> updateDog(Dog dog) async {
65     // Get a reference to the database.
66     final db = await database;
67
68     // Update the given Dog.
69     await db.update(
70       'dogs',
71       dog.toMap(),
72       // Ensure that the Dog has a matching id.
73       where: 'id = ?',
74       // Pass the Dog's id as a whereArg to prevent SQL injection.
75       whereArgs: [dog.id],
76     );
77   }
```

```
lib > db_test.dart > main
91     // Update Fido's age and save it to the database.
92     fido = Dog(
93       id: fido.id,
94       name: fido.name,
95       age: fido.age + 7,
96     );
97     await updateDog(fido);
98
99     // Print the updated results.
100    print(await dogs()); // Prints Fido with age 42.
101  }
```

8. A dog is then deleted from the database

```

lib > db_test.dart > main > deleteDog
//
78
79 Future<void> deleteDog(int id) async {
80   // Get a reference to the database.
81   final db = await database;
82
83   // Remove the Dog from the database.
84   await db.delete(
85     'dogs',
86     // Use a `where` clause to delete a specific dog.
87     where: 'id = ?',
88     // Pass the Dog's id as a whereArg to prevent SQL injection.
89     whereArgs: [id],
90   );
91 }

```

```

lib > db_test.dart > main
114   print(await dogs()); // Prints Fido
115
116   // Delete Fido from the database.
117   await deleteDog(fido.id);
118
119   // Print the list of dogs (empty).
120   print(await dogs());
121 }

```

9. Output of the code should contain

```

PS D:\Dev\db_test_app> flutter run lib/db_test.dart
Launching lib/db_test.dart on sdk gphone64 x86 64 in debug mode...
Running Gradle task 'assembleDebug'... 5.4s
✓ Built build\app\outputs\flutter-apk\app-debug.apk.
Installing build\app\outputs\flutter-apk\app-debug.apk... 1,785ms
I/flutter (12252): [Dog{id: 0, name: Fido, age: 35}]
I/flutter (12252): [Dog{id: 0, name: Fido, age: 42}]
I/flutter (12252): []
Syncing files to device sdk gphone64 x86 64... 32ms

Flutter run key commands.
r Hot reload.

```