





# Code

## Main.dart

import 'dart:math';  
import 'package:flutter/material.dart';  
import 'package:madlab12/controllers/fileController.dart';  
import 'package:madlab12/controllers/sqfliteController.dart';  
import 'package:shared\_preferences/shared\_preferences.dart';  
  
import 'models/RandomNumber.dart';  
  
void main() async {  
 WidgetsFlutterBinding.*ensureInitialized*();  
 await sqfliteController.*initialize*();  
 runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 debugShowCheckedModeBanner: false,  
 home: MyHomePage(),  
 );  
 }  
}  
  
Future<int> getSharedPrefsInt() async{  
 final prefs = await SharedPreferences.*getInstance*();  
 return prefs.getInt("integerValue");  
}  
  
Future removeSharedPrefsInt() async {  
 final prefs = await SharedPreferences.*getInstance*();  
 await prefs.remove("integerValue");  
}  
  
Future saveSharedPrefsInt(int newInt) async {  
 final prefs = await SharedPreferences.*getInstance*();  
 await prefs.setInt("integerValue", newInt);  
}  
  
class MyHomePage extends StatefulWidget {  
 @override  
 \_MyHomePageState createState() => \_MyHomePageState();  
}  
class \_MyHomePageState extends State<MyHomePage> {  
 int randomNumber;  
  
 @override  
 void initState() {  
 super.initState();  
 randomNumber = Random().nextInt(1000);  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: SafeArea(  
 child: Center(  
 child: SingleChildScrollView(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Text("Generated random number", style: TextStyle(fontSize: 28),),  
 Text(randomNumber.toString(), style: TextStyle(fontSize: 28),),  
 SizedBox(height: 30,),  
 Text("Shared preferences number", style: TextStyle(fontSize: 28),),  
 FutureBuilder(  
 future: getSharedPrefsInt(),  
 builder: (context, snapshot){  
 if (snapshot.hasData){  
 return Text(snapshot.data.toString(), style: TextStyle(fontSize: 28),);  
 }  
 else {  
 if (snapshot.hasError) print(snapshot.error);  
 return SizedBox.shrink();  
 }  
 },  
 ),  
 SizedBox(height: 30,),  
 Text("SQLite number", style: TextStyle(fontSize: 28),),  
 FutureBuilder(  
 future: sqfliteController.*getRandomNumbers*(),  
 builder: (context, snapshot){  
 if (snapshot.hasData && snapshot.data.isNotEmpty){  
 return Text(snapshot.data.last.value.toString(), style: TextStyle(fontSize: 28),);  
 }  
 else {  
 if (snapshot.hasError) print(snapshot.error);  
 return SizedBox.shrink();  
 }  
 },  
 ),  
 SizedBox(height: 30,),  
 Text("File number", style: TextStyle(fontSize: 28),),  
 FutureBuilder(  
 future: fileController.*readFromFile*(),  
 builder: (context, snapshot){  
 if (snapshot.hasData && snapshot.data != -1){  
 return Text(snapshot.data.toString(), style: TextStyle(fontSize: 28),);  
 }  
 else {  
 if (snapshot.hasError) print(snapshot.error);  
 return SizedBox.shrink();  
 }  
 },  
 ),  
 SizedBox(height: 50,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
 children: [  
 RaisedButton(  
 color: Colors.*orange*,  
 textColor: Colors.*white*,  
 child: Text("Load all"),  
 onPressed: (){setState(() {});},  
 ),  
 RaisedButton(  
 color: Colors.*green*,  
 textColor: Colors.*white*,  
 child: Text("Random"),  
 onPressed: (){setState(() {randomNumber = Random().nextInt(1000);});},  
 ),  
 ],  
 ),  
 SizedBox(height: 10,),  
 RaisedButton(  
 color: Colors.*blue*,  
 textColor: Colors.*white*,  
 child: Text("Save to SharedPrefs"),  
 onPressed: () async {  
 await saveSharedPrefsInt(randomNumber);},  
 ),  
 SizedBox(height: 10,),  
 RaisedButton(  
 color: Colors.*deepPurple*,  
 textColor: Colors.*white*,  
 child: Text("Save to SQLite"),  
 onPressed: () async {  
 await sqfliteController.*insertInDB*(RandomNumber(randomNumber));},  
 ),  
 SizedBox(height: 10,),  
 RaisedButton(  
 color: Colors.*pink*,  
 textColor: Colors.*white*,  
 child: Text("Save to Cache File"),  
 onPressed: () async {  
 await fileController.*saveToFile*(randomNumber);},  
 ),  
 RaisedButton(  
 color: Colors.*red*,  
 textColor: Colors.*white*,  
 child: Text("Delete All"),  
 onPressed: () async {  
 await fileController.*deleteFileValue*();  
 await sqfliteController.*deleteAllRows*();  
 await removeSharedPrefsInt();  
 setState(() {  
  
 });  
 },  
 ),  
 ],  
 ),  
 ),  
 ),  
 ), // This trailing comma makes auto-formatting nicer for build methods.  
 );  
 }  
}

## Model

class RandomNumber{  
 int id;  
 int value;  
 DateTime timestamp;  
  
 RandomNumber(this.value){this.timestamp = DateTime.now();}  
  
 RandomNumber.fromMap(Map<String,dynamic> map){  
 id = map['id'];  
 value = map['value'];  
 timestamp = DateTime.fromMillisecondsSinceEpoch(map['timestamp']);  
 }  
  
 Map<String,dynamic> toMap(){  
 return {  
 'id' : this.id,  
 "value" : this.value,  
 "timestamp" : this.timestamp.millisecondsSinceEpoch  
 };  
 }  
  
 @override  
 String toString(){  
 return 'Random Number {id: $id, value: $value, timestamp: ${timestamp.toString()}';  
 }  
}

## SQFLite Controller

import 'package:madlab12/models/RandomNumber.dart';  
import 'package:path/path.dart';  
import 'package:sqflite/sqflite.dart';  
  
class sqfliteController{  
 static Database *database*;  
  
 static Future *initialize*() async {  
 *database* = await openDatabase(  
 join(await getDatabasesPath(), 'random\_number.db'),  
 onCreate: (db, version) {  
 return db.execute('CREATE TABLE randomNumber(id INTEGER PRIMARY KEY, value INTEGER, timestamp INTEGER)',);  
 },  
 version: 1,  
 );  
 }  
  
 static Future<void> *insertInDB*(RandomNumber randomNumber) async {  
 await *database*.insert('randomNumber', randomNumber.toMap(), conflictAlgorithm: ConflictAlgorithm.replace);  
 }  
  
 static Future<List<RandomNumber>> *getRandomNumbers*() async{  
 final List<Map<String, dynamic>> randomNumbersMaps = await *database*.query('randomNumber');  
 return List.generate(randomNumbersMaps.length, (index) => RandomNumber.fromMap(randomNumbersMaps[index]));  
 }  
  
 static Future<void> *updateDB*(RandomNumber randomNumber) async {  
 // '?' are arguments for where clause  
 await *database*.update('randomNumber', randomNumber.toMap(), where: "id = ?", whereArgs: [randomNumber.id], conflictAlgorithm: ConflictAlgorithm.replace);  
 }  
  
 static Future<void> *deleteFromDB*(int id) async {  
 await *database*.delete('randomNumber', where: "id = ?", whereArgs: [id]);  
 }  
  
 static Future<void> *deleteAllRows*() async {  
 await *database*.delete('randomNumber');  
 }  
  
 static Future<void> *closeDB*() async {  
 await *database*.close();  
 }  
}

## File Controller

import 'dart:io';  
import 'package:madlab12/models/RandomNumber.dart';  
import 'package:path\_provider/path\_provider.dart';  
  
class fileController{  
 static Future<int> *readFromFile*() async {  
 var directory = await getTemporaryDirectory();  
 File file = File(directory.path + "/randomNumbers.txt");  
  
 if (file.existsSync()){ return int.*parse*(file.readAsStringSync().trim()); }  
 else{  
 return -1;  
 }  
 }  
  
 static Future<void> *saveToFile*(int randomNumber) async {  
 var directory = await getTemporaryDirectory();  
 File file = File(directory.path + "/randomNumbers.txt");  
 file.writeAsStringSync(randomNumber.toString(), flush: true, mode: FileMode.*write*);  
 }  
  
 static Future<void> *deleteFileValue*() async{  
 var directory = await getTemporaryDirectory();  
 File file = File(directory.path + "/randomNumbers.txt");  
 if (file.existsSync())  
 file.deleteSync();  
 }  
}