dbhelper.dart main.dart

lib > main.dart > MyHomePage > _query

```dart
import 'package:flutter/material.dart';
import 'dbhelper.dart';
final dbHelper = DatabaseHelper();
Run | Debug | Profile
Future<void> main() async {WidgetsFlutterBinding.ensureInitialized();
  await dbHelper.init();runApp(const MyApp());}
class MyApp extends StatelessWidget{const MyApp({super.key});
  @override Widget build(BuildContext context) {
    return MaterialApp(title: 'SQFlite Demo',
    theme: ThemeData(primarySwatch: Colors.blue,),
    home:const MyHomePage(),);}} // MaterialApp
class MyHomePage extends StatelessWidget {
  const MyHomePage({super.key});
  @override Widget build(BuildContext context) {
  return Scaffold(appBar: AppBar(title: const Text('sqflite'),),
    body: Center(child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[ElevatedButton(onPressed:_insert,
            child: const Text('insert'),), // ElevatedButton
            const SizedBox(height:10),ElevatedButton(onPressed: _query,
            child: const Text('query'),), // ElevatedButton
          const SizedBox(height: 10),ElevatedButton(onPressed: _update,
            child: const Text('update'),), // ElevatedButton
          const SizedBox(height: 10),ElevatedButton(onPressed: _delete,
            child: const Text('delete'),),],),),);} // ElevatedButton //
  void _insert() async{Map<String, dynamic> row={
    DatabaseHelper.columnName:'Bob',DatabaseHelper.columnAge:23};
    final id = await dbHelper.insert(row);
    debugPrint('inserted row id: $id');}
  void _query() async{final allRows=await dbHelper.queryAllRows();
    debugPrint('query all rows:');
    for (final row in allRows) {
      debugPrint(row.toString());}}
  void _update() async {
    Map<String, dynamic> row = {
      DatabaseHelper.columnId: 1,
      DatabaseHelper.columnName: 'Mary',
      DatabaseHelper.columnAge: 32};
    final rowsAffected = await dbHelper.update(row);
    debugPrint('updated $rowsAffected row(s)');}
  void _delete() async {
    final id = await dbHelper.queryRowCount();
    final rowsDeleted = await dbHelper.delete(id);
    debugPrint('deleted $rowsDeleted row(s): row $id');}}
```

dbhelper.dart main.dart

lib > dbhelper.dart > DatabaseHelper > init

```dart
import 'package:path/path.dart';
import 'package:sqflite/sqflite.dart';
import 'package:path_provider/path_provider.dart';
class DatabaseHelper {
  static const _databaseName = "MyDatabase.db";
  static const _databaseVersion = 1;
  static const table = 'my_table';
  static const columnId = '_id';
  static const columnName = 'name';
  static const columnAge = 'age';
  late Database _db;
  Future<void> init() async {
    final documentsDirectory = await getApplicationDocumentsDirectory();
    final path = join(documentsDirectory.path, _databaseName);
    _db = await openDatabase(path, version: _databaseVersion,
    onCreate: _onCreate,);}
  Future _onCreate(Database db, int version) async {
    await db.execute('''CREATE TABLE $table (
      $columnId INTEGER PRIMARY KEY,
      $columnName TEXT NOT NULL,
      $columnAge INTEGER NOT NULL)''');}
  Future<int> insert(Map<String, dynamic> row) async {
    return await _db.insert(table, row);}
  Future<List<Map<String, dynamic>>> queryAllRows() async {
    return await _db.query(table);}
  Future<int> queryRowCount() async {
    final results = await _db.rawQuery('SELECT COUNT(*) FROM $table');
    return Sqflite.firstIntValue(results) ?? 0;}
  Future<int> update(Map<String, dynamic> row) async {
    int id = row[columnId];return await _db.update(
      table,row,where:'$columnId=?', whereArgs: [id],);}
  Future<int> delete(int id)async{
    return await _db.delete(table,where:'$columnId=?',
      whereArgs: [id],);}}
```

main.dart

lib > main.dart > _OnboardingScreenState > build

```dart
import 'package:flutter/material.dart';
Run | Debug | Profile
void main() { runApp(MyApp());}
class MyApp extends StatelessWidget {
  @override Widget build(BuildContext context) {
    return MaterialApp(title: 'Onboarding Demo',
    theme: ThemeData(primarySwatch: Colors.blue,),
      home: OnboardingScreen(),);}} // MaterialApp
class OnboardingScreen extends StatefulWidget {
  @override _OnboardingScreenState createState() => _OnboardingScreenState();}
class _OnboardingScreenState extends State<OnboardingScreen> {
  int _currentPageIndex = 0;final PageController _pageController=PageController(initialPage: 0);
  final List<String> _onboardingItems = ['Welcome to Onboarding!', 'Discover Amazing Features',
  'Learn How to Use the App', 'Get Started Now!',];
  @override Widget build(BuildContext context) {
    return Scaffold(body:Stack(children:[PageView.builder(
      controller:_pageController,itemCount:_onboardingItems.length,
        onPageChanged: (index) {setState(() {_currentPageIndex = index;});},
          itemBuilder: (context, index) {return Padding(padding:
            const EdgeInsets.all(20.0),child: Column(mainAxisAlignment:MainAxisAlignment.center,
              children: [Text( _onboardingItems[index],textAlign:TextAlign.center,style:TextStyle(
                fontSize: 24.0, fontWeight: FontWeight.bold,),),],),);},), // TextStyle // Text // Column
            Positioned(bottom:20.0,left:0,right:0,
              child: Row(mainAxisAlignment: MainAxisAlignment.center,
                children: _buildPageIndicator(),),),Positioned(top: 40.0, right: 20.0, // Row // Position
              child: TextButton(onPressed:(){_navigateToHome();},
                child:Text('Skip',style:TextStyle(fontSize: 18.0,color: Colors.blue,),),),), // TextButt
            Positioned(bottom:20.0,left:20.0,
              child: ElevatedButton(onPressed:(){
                if(_currentPageIndex>0){_pageController.previousPage(
                  duration:Duration(milliseconds:500),curve:Curves.ease,);}},
                child: Text('Back'),),),],), // ElevatedButton // Positioned // Stack
        floatingActionButton: FloatingActionButton(onPressed:(){
          if(_currentPageIndex<_onboardingItems.length-1){
            _pageController.nextPage(duration: Duration(milliseconds: 500),
            curve:Curves.ease,);}else{_navigateToHome();}},child:Icon(Icons.arrow_forward,),),);} // Fl
    List<Widget>_buildPageIndicator(){List<Widget>indicators=[];
      for (int i=0;i<_onboardingItems.length;i++){
        indicators.add(_indicator(i == _currentPageIndex));}
        return indicators;}Widget _indicator(bool isActive) {
      return AnimatedContainer(duration: Duration(milliseconds:300),
        margin: EdgeInsets.symmetric(horizontal: 8.0),height:8.0,
        width:isActive ? 24.0 : 8.0,decoration:BoxDecoration(color:isActive?Colors.blue:Colors.grey,
          borderRadius: BorderRadius.circular(12),),);} // BoxDecoration // AnimatedContainer
    void _navigateToHome() {print('Navigate to Home Screen');}}
```

main.dart

lib > main.dart > _TicTacToeScreenState > _checkWinner

```dart
import 'package:flutter/material.dart';
Run | Debug | Profile
void main() { runApp(TicTacToeApp()); }
class TicTacToeApp extends StatelessWidget {
  @override Widget build(BuildContext context) {
    return MaterialApp(title: 'Tic Tac Toe',
      theme: ThemeData(primarySwatch: Colors.blue,),
      home: TicTacToeScreen(),);} } // MaterialApp
class TicTacToeScreen extends StatefulWidget {
  @override _TicTacToeScreenState createState() => _TicTacToeScreenState();}
class _TicTacToeScreenState extends State<TicTacToeScreen> {
  List<String> _board = [];String _currentPlayer = 'X';
  bool _gameOver = false; // Initialize _gameOver
  @override void initState() {super.initState();
    _initializeBoard(); }void _initializeBoard() { setState(() {
      _board = List.filled(9, '');});} void _onTileTap(int index) {
    if (!_gameOver && _board[index] == '') { setState(() {
        _board[index] = _currentPlayer;
        if (_checkWinner(_currentPlayer)) {
          _showWinnerDialog(_currentPlayer);
          _gameOver = true;} else if (!_board.contains('')) {
          _showDrawDialog();_gameOver = true;
        } else{_currentPlayer = (_currentPlayer == 'X') ? 'O' : 'X';} });} }
  bool _checkWinner(String player) { // Check rows
    for (int i = 0; i < 9; i += 3) {
      if (_board[i] == player && _board[i + 1] == player && _board[i + 2] == player) {
        return true;} }for (int i = 0; i < 3; i++) {
      if (_board[i] == player && _board[i + 3] == player && _board[i + 6] == player) {
        return true;} }
        if ((_board[0] == player && _board[4] == player && _board[8] == player) ||
        (_board[2] == player && _board[4] == player && _board[6] == player)) {
      return true;} return false; } void _showWinnerDialog(String winner) {
    showDialog(context: context,builder: (BuildContext context) {
        return AlertDialog(title: Text('Winner'),
          content: Text('Player $winner wins!'),
          actions: <Widget>[ElevatedButton(onPressed: () {
                Navigator.of(context).pop();_initializeBoard();
                _gameOver = false;},child: Text('Play Again'),),],);}, ); } // ElevatedButton // <Widget>[] // AlertDia
  void _showDrawDialog() { showDialog(
      context: context,builder:(BuildContext context){
        return AlertDialog(title: Text('Draw'),
          content: Text('It\'s a draw!'),
          actions: <Widget>[ElevatedButton(onPressed: () {
                Navigator.of(context).pop();_initializeBoard();
                _gameOver = false;},child: Text('Play Again'),),],);},);} // ElevatedButton // <Widget>[] // AlertDialo
  @override Widget build(BuildContext context){
    return Scaffold(appBar: AppBar(title: Text('Tic Tac Toe'),),
      body: GridView.builder(padding: EdgeInsets.all(16.0),
        itemCount:9,gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
          crossAxisCount:3,crossAxisSpacing:8.0,mainAxisSpacing:8.0,), itemBuilder: (BuildContext context, int index) {
          return GestureDetector(onTap: () => _onTileTap(index),
            child: Container(color: Colors.blue,
              child: Center(child: Text(_board[index],
                style: TextStyle(fontSize: 48.0, color: Colors.white),
                ),),),); },),);}} // Text // Center // Container // GestureDetector // GridView.builder // Scaffold
```

main.dart

lib > main.dart > _FileSelectorScreenState > _readFileAsBytes

```dart
import 'dart:html';
import 'dart:typed_data';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
Run | Debug | Profile
void main() {runApp(MyApp());}
class MyApp extends StatelessWidget {
  @override Widget build(BuildContext context) {
    return MaterialApp(title: 'File Size Classifier',
      theme: ThemeData(primarySwatch: Colors.blue,),
      home: FileSelectorScreen(),);}} // MaterialApp
class FileSelectorScreen extends StatefulWidget { @override
  _FileSelectorScreenState createState() => _FileSelectorScreenState();}
class _FileSelectorScreenState extends State<FileSelectorScreen> {
  List<Uint8List> _selectedFiles = [];
  Future<void> _pickFiles() async {
    final input = FileUploadInputElement()..multiple = true;
    input.click();
    input.onChange.listen((e) async {
      final files = input.files;
      if (files != null && files.isNotEmpty) {
        List<Uint8List> filesBytes = [];
        for (var file in files) {filesBytes.add(await _readFileAsBytes(file));}
        filesBytes.sort((a, b) => a.length - b.length);
        setState(() {_selectedFiles = filesBytes;
        });for (Uint8List fileBytes in _selectedFiles) {
          showFileSize(fileBytes);}});}
  Future<Uint8List> _readFileAsBytes(File file) async {
    final reader = FileReader(); reader.readAsArrayBuffer(file);
    await reader.onLoad.first; return reader.result as Uint8List;}
  void showFileSize(Uint8List fileBytes) {
    int sizeInBytes = fileBytes.length;double sizeInMB = sizeInBytes / (1024 * 1024);
    String message = 'Size: ${sizeInMB.toStringAsFixed(2)} MB';
    Fluttertoast.showToast(msg: message, backgroundColor: sizeInMB > 10 ? Colors.red : Colors.green,);
    print(message);}
  @override Widget build(BuildContext context) {
    return Scaffold(appBar: AppBar(title: Text('File Size Classifier'),),
      body: Center(child: Column(mainAxisAlignment: MainAxisAlignment.center,
          children: [ElevatedButton(onPressed: _pickFiles,
              child: Text('Pick Files'),),SizedBox(height: 20), // ElevatedButton
            Text('Uploaded Files:', style: TextStyle(fontWeight: FontWeight.bold),
            ), SizedBox(height: 10), Expanded(child: ListView.builder( // Text
                itemCount: _selectedFiles.length, itemBuilder: (context, index) {return ListTile(title: Text(
                    'File ${index + 1} - Size: ${(_selectedFiles[index].length / (1024 * 1024)).toStringAsFixed(2)} MB'),
                  );},),),],),),);}} // ListTile // ListView.builder // Expanded // Column // Center // Scaffold
```

main.dart

lib > main.dart > _MyHomePageState

```dart
import 'package:flutter/material.dart';
import 'package:geolocator/geolocator.dart';
import 'package:url_launcher/url_launcher.dart';
Run | Debug | Profile
void main() {
  runApp(MyApp());
}class MyApp extends StatelessWidget {
  @override Widget build(BuildContext context) {
    return MaterialApp(title: 'GPS Location App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ), // ThemeData
      home: MyHomePage(),
    );}} // MaterialApp
class MyHomePage extends StatefulWidget {
  @override
  _MyHomePageState createState() => _MyHomePageState();}
class _MyHomePageState extends State<MyHomePage> {
  String _latitude = 'Unknown';
  String _longitude = 'Unknown';
  @override
  void initState() {
    super.initState();
    _getLocation();
  }void _getLocation() async {
    try {
      Position position = await Geolocator.getCurrentPosition(
          desiredAccuracy: LocationAccuracy.high);
      setState(() {
        _latitude = position.latitude.toString();
        _longitude = position.longitude.toString();
      });} catch (e) {
      print('Error: $e');
    } }
  void _openMaps() async {
    String url = 'https://www.google.com/maps/search/?api=1&query=$_latitude,$_longitude';
    if (await canLaunch(url)) {
      await launch(url);
    } else {
      throw 'Could not launch $url';
    } } @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar( title: Text('GPS Location App'),),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text('Latitude: $_latitude',style: TextStyle(fontSize: 18.0),),
            SizedBox(height: 20.0),
            Text('Longitude: $_longitude', style: TextStyle(fontSize: 18.0), ),
            SizedBox(height: 20.0),
            ElevatedButton(onPressed: _openMaps,child: Text('Open in Maps'),),
          ], // <Widget>[]
        ),),);}} // Column // Center // Scaffold
```