Add path provider package from internet.

More information on this site:

<https://www.youtube.com/watch?v=Hqqz2BaPUis>

[Read and write files | Flutter](https://docs.flutter.dev/cookbook/persistence/reading-writing-files)

Code:

import 'dart:async';  
import 'dart:io';  
  
import 'package:flutter/material.dart';  
import 'package:path\_provider/path\_provider.dart';  
  
void main() {  
 runApp(  
 MaterialApp(  
 title: 'Reading and Writing Files',  
 home: FlutterDemo(storage: CounterStorage()),  
 ),  
 );  
}  
  
class CounterStorage {  
 Future<String> get \_localPath async {  
 final directory = await getApplicationDocumentsDirectory();  
  
 return directory.path;  
 }  
  
 Future<File> get \_localFile async {  
 final path = await \_localPath;  
 return File('$path/counter.txt');  
 }  
  
 Future<String> readCounter() async {  
 try {  
 final file = await \_localFile;  
  
 // Read the file  
 final contents = await file.readAsString();  
  
 return contents.toString();  
 } catch (e) {  
 // If encountering an error, return 0  
 return e.toString();  
 }  
 }  
  
 Future<File> writeCounter(String counter) async {  
 final file = await \_localFile;  
  
 // Write the file  
 return file.writeAsString('$counter');  
 }  
}  
  
class FlutterDemo extends StatefulWidget {  
 const FlutterDemo({super.key, required this.storage});  
  
 final CounterStorage storage;  
  
 @override  
 State<FlutterDemo> createState() => \_FlutterDemoState();  
}  
  
class \_FlutterDemoState extends State<FlutterDemo> {  
 TextEditingController \_counter = TextEditingController();  
 late String state;  
 String emp = 'file is empty';  
 String read1 = '';  
 String read2 = '';  
  
 @override  
 void initState() {  
 super.initState();  
 widget.storage.readCounter().then((String value) {  
 setState(() {  
 state = value;  
 });  
 });  
 }  
  
 Future<File> writeData() async {  
 setState(() {  
 state = \_counter.text;  
 \_counter.text = '';  
 });  
 return widget.storage.writeCounter(state);  
 }  
  
 Future<String> readData() async {  
 read1 = await widget.storage.readCounter();  
 setState(() {  
 read2 = read1;  
 });  
 return read2;  
 }  
  
 String whattowrite() {  
 if (read2.isEmpty) {  
 return 'nothing saved yet!';  
 } else {  
 return read2;  
 }  
 }  
  
 // Future<File> \_incrementCounter() {  
 // setState(() {  
 // \_counter;  
 // });  
 //  
 // // Write the variable as a string to the file.  
 // return widget.storage.writeCounter(\_counter.toString());  
 // }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: const Text('Reading and Writing Files'),  
 ),  
 body: Center(  
 child: Column(  
 children: [  
 TextField(  
 controller: \_counter,  
 keyboardType: TextInputType.*text*,  
 ),  
 TextButton(  
 onPressed: () {  
 writeData();  
 readData();  
 },  
 child: Text(  
 'Save Data',  
 style: TextStyle(color: Colors.*black*),  
 )),  
 Text(  
 '${read2.isEmpty ? 'Folder is empty!' : read2}',  
 style: TextStyle(color: Colors.*black*),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}