

# Persistence in Dart/Flutter

# Persistence

- Relational
  - Sqlflite
  - Moor
- Firebase
- Non SQL
  - Hive
  - ObjectBox
  - Isar
  - Couchbase lite



# sqlite 2.0.2

Published 2 months ago • [tekartik.com](#) Null safety

[SDK](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [MACOS](#)

👍 2.7K

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

## sqlite

pub v2.0.2 build failing codecov 57%

SQLite plugin for [Flutter](#). Supports iOS, Android and MacOS.

- Support transactions and batches
- Automatic version management during open
- Helpers for insert/query/update/delete queries
- DB operation executed in a background thread on iOS and Android

Other platforms support:

- Linux/Windows/DartVM support using [sqlite\\_common\\_ffi](#)
- Web [is not supported](#).

Usage example:

- [notepad\\_sqlite](#): Simple flutter notepad working on iOS/Android/Windows/linux/Mac

## Getting Started

In your flutter project add the dependency:

```
dependencies:  
  ...  
  sqlite:
```

For help getting started with Flutter, view the [online documentation](https://pub.dev/packages/sqlite).

## Usage example

# sqlite 2.0.2

Published 2 months ago • [tekartik.com](#) Null safety

[SDK](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [MACOS](#)

👍 2.7K

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

## sqlite

pub v2.0.2 build failing codecov 57%

SQLite plugin for [Flutter](#). Supports iOS, Android and MacOS.

- Support transactions and batches
- Automatic version management during open
- Helpers for insert/query/update/delete queries
- DB operation executed in a background thread on iOS and Android

Other platforms support:

- Linux/Windows/DartVM support using [sqlite\\_common\\_ffi](#)
- **Web is not supported.**

Usage example:

- [notepad\\_sqlite](#): Simple flutter notepad working on iOS/Android/Windows/linux/Mac

## Getting Started

In your flutter project add the dependency:

```
dependencies:  
  ...  
  sqlite:
```

For help getting started with Flutter, view the [online documentation](https://pub.dev/packages/sqlite).

## Usage example

## moor\_flutter 4.1.0

Published 7 days ago • [simonbinder.eu](#) Null safety[SDK](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [MACOS](#) 308[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

## moor\_flutter

**Important notice:** Moor has been renamed to Drift. This package will continue to be supported for a while, but we encourage users to migrate to `drift` and `drift_sqflite`. Automated tooling to help you make the switch is available! For more information, see [the documentation](#). Thanks for your understanding!

Please see the homepage of [drift](#) for details on how to use this package.

- ▼ **Smart linting:** Moor generates type-safe code based on your tables and queries. If you make a mistake in your queries, Moor will find it at compile time and provide helpful and descriptive lints.
- **Reactive:** Turn any SQL query into an auto-updating stream! This includes complex queries across many tables.
- **Cross-Platform support:** Moor works on Android, iOS, macOS, Windows, Linux and the web. [This template](#) is a Flutter todo app that works on all platforms.
- **Battle tested and production ready:** Moor is stable and well tested with a wide range of unit and

[https://pub.dev/packages/moor\\_flutter](https://pub.dev/packages/moor_flutter)

<https://moor.simonbinder.eu/>

<https://moor.simonbinder.eu/docs/getting-started/>

If you have any questions, feedback or ideas, feel free to [create an issue](#). If you enjoy this project, I'd

verbose than JPA –  
code generation

dependency injection :(

# moor\_flutter 3.1.0

Published Jul 6, 2020 • [simonbinder.eu](#)



## moor\_flutter 4.1.0

Published 7 days ago • [simonbinder.eu](#) Null safety

[SDK](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [MACOS](#)

308

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

## moor\_flutter

**Important notice:** Moor has been renamed to Drift. This package will continue to be supported for a while, but we encourage users to migrate to `drift` and `drift_sqflite`. Automated tooling to help you make the switch is available! For more information, see [the documentation](#). Thanks for your understanding!

Please see the homepage of [drift](#) for details on how to use this package.



se than JPA –  
generation

ncy injection :(

<https://moor.simonbinder.eu/>  
<https://moor.simonbinder.eu/docs/getting-started/>

If you have any questions, feedback or ideas, feel free to [create an issue](#). If you enjoy this project, I'd

# Drift: Persistence library for Dart

[Learn more](#) [Get from pub](#) 

With a fluent query api, a powerful sql analyzer, auto-updating streams and much more, drift makes persistence fun. Scroll down to learn about its key features, or visit the [getting started guide](#) for a step-by-step guide on using drift.

Drift is an easy to use, reactive persistence library for Flutter apps. Define tables in Dart or SQL and enjoy a fluent query API, auto-updating streams and more!



# drift 1.5.0

Published 8 days ago • [simonbinder.eu](#) Null safety

[SDK](#) [DART](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [LINUX](#) [MACOS](#) [WEB](#) [WINDOWS](#)

383

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

## Drift

Proudly Sponsored by [Stream](#) ❤️



[Try the Flutter Chat Tutorial](#)

Drift is a reactive persistence library for Flutter and Dart, built on top of `sqlite`. Drift is

- **Flexible:** Drift let's you write queries in both SQL and Dart, providing fluent apis for both languages. You can filter and order results or use joins to run queries on multiple tables. You can even use complex sql features like `WITH` and `WINDOW` clauses.
- **Feature rich:** Drift has builtin support for transactions, schema migrations, complex filters and expressions, batched updates and joins. We even have a builtin IDE for SQL!
- **Modular:** Thanks to builtin support for daos and `import` s in sql files, drift helps you keep your database code simple.
- **Safe:** Drift generates typesafe code based on your tables and queries. If you make a mistake in your queries, drift will find it at compile time and provide helpful and descriptive lints.
- **Fast:** Even though drift lets you write powerful queries, it can keep up with the performance of key-value stores. Drift is the only major persistence library with builtin threading support, allowing you to run database code across isolates with zero additional effort.

D

With  
much  
featu

Drift  
De



# SPEC

```
import 'package:drift/drift.dart';

// assuming that your file is called filename.dart. This will give you
// first, but it's needed for drift to know about the generated code
part 'filename.g.dart';

// this will generate a table called "todos" for us. The rows of the
// be represented by a class called "Todo".
class Todos extends Table {
  IntColumn get id => integer().autoIncrement();
  TextColumn get title => text().withLength(min: 6, max: 32());
  TextColumn get content => text().named('body');
  IntColumn get category => integer().nullable();
}

// This will make drift generate a class called "Category" to represent
// this table. By default, "Categorie" would have been used because
// strips away the trailing "s" in the table name.
@DataClassName('Category')
class Categories extends Table {
  IntColumn get id => integer().autoIncrement();
  TextColumn get description => text();
}

// this annotation tells drift to prepare a database class that uses
// tables we just defined. We'll see how to use that database class
@DriftDatabase(tables: [Todos, Categories])
class MyDatabase extends _MyDatabase {
}
```



# Generating the code

Drift integrates with Dart's `build` system, so you can generate all the code needed with `flutter pub run build_runner build`. If you want to continuously rebuild the generated code where you change your code, run `flutter pub run build_runner watch` instead. After running either command once, the drift generator will have created a class for your database and data classes for your entities. To use it, change the `MyDatabase` class as follows:



# USE

```
// These imports are only needed to open the database
import 'dart:io';

import 'package:drift/native.dart';
import 'package:path_provider/path_provider.dart';
import 'package:path/path.dart' as p;

@DriftDatabase(tables: [Todos, Categories])
class MyDatabase extends _MyDatabase {
  // we tell the database where to store the data with this constructor
  MyDatabase() : super(_openConnection());

  // you should bump this number whenever you change or add a table definition
  // Migrations are covered later in the documentation.
  @override
  int get schemaVersion => 1;
}

LazyDatabase _openConnection() {
  // the LazyDatabase util lets us find the right location for the file
  return LazyDatabase(() async {
    // put the database file, called db.sqlite here, into the document
    // for your app.
    final dbFolder = await getApplicationDocumentsDirectory();
    final file = File(p.join(dbFolder.path, 'db.sqlite'));
    return NativeDatabase(file);
  });
}
```

<https://drift.simonbinder.eu/docs/getting-started/>



```
// inside the database class, the `todos` getter has been created by drift.
@DriftDatabase(tables: [Todos, Categories])
class MyDatabase extends _$MyDatabase {

    // the schemaVersion getter and the constructor from the previous page
    // have been omitted.

    // loads all todo entries
    Future<List<Todo>> get allTodoEntries => select(todos).get();

    // watches all todo entries in a given category. The stream will automatically
    // emit new items whenever the underlying data changes.
    Stream<List<Todo>> watchEntriesInCategory(Category c) {
        return (select(todos)..where((t) => t.category.equals(c.id))).watch();
    }
}
```

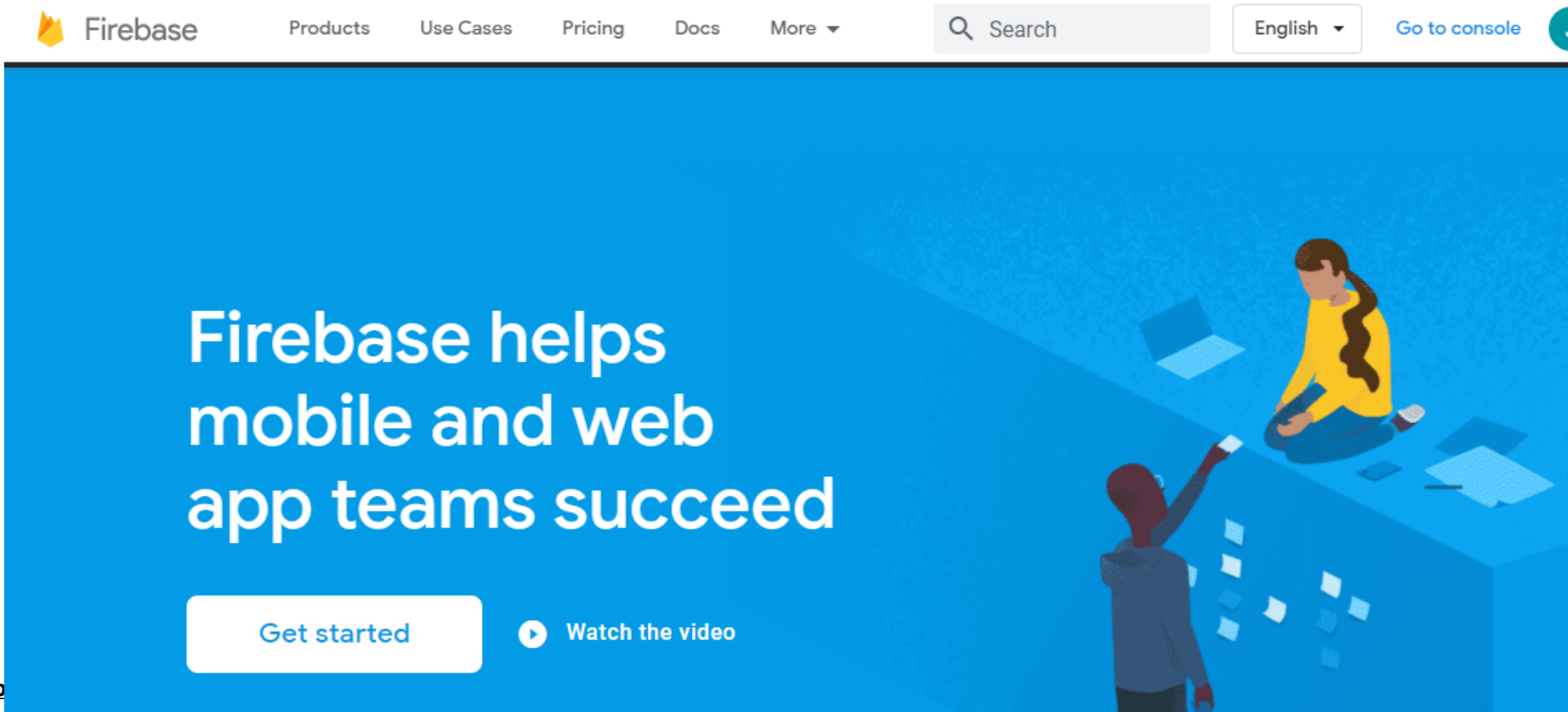
[https://drift.simonbinder.eu/docs/getting-started/writing\\_queries/](https://drift.simonbinder.eu/docs/getting-started/writing_queries/)



# Firebase

# Firestore

- Not only a “database”
- Front end for accessing multiple google web resources



Get started

Watch the video



## Realtime Database

Pay to scale

iOS   C++ 

Realtime Database is Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in realtime. We recommend Cloud Firestore instead of Realtime Database for most developers starting a new project.



## Cloud Firestore

Pay to scale

iOS  

Store and sync data between users and devices - at global scale - using a cloud-hosted, NoSQL database. Cloud Firestore gives you live synchronization and offline support along with efficient data queries. Its integration with other Firebase products enables you to build truly serverless apps.



## Firebase ML BETA

Pay to scale

iOS 

Bring powerful machine learning features to your mobile app. Firebase ML helps you deploy custom ML models optimized for on-device inference, which reduces your initial app installation size and lets you more easily make updates. You can also use AutoML Vision Edge to train your own custom image classification models, or access Cloud AI Vision APIs for a more turn-key solution.

<https://firebase.google.com/products>

# Firestore is often used to refer to ...

- The Realtime Database - the Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in real time.
- Firestore - a cloud-hosted, NoSQL database - “Store and sync data between users and devices - at global scale –
- Has integration with other Firebase products

**Some resources tend to still refer to firebase but in fact are implicitly using Firestore.** Google recommends Cloud Firestore instead of Realtime Database for most developers starting a new project.

# FlutterFire

The official Firebase plugins for Flutter

[Get Started »](#)

[GitHub »](#)

Plugin	Version	pub.dev	Firebase	View Source	Mobile	Web	MacOS
AdMob	 			<a href="#">firebase_admob</a>	✓	✗	✗
Analytics	 			<a href="#">firebase_analytics</a>	✓	✓	✗
Authentication	 			<a href="#">firebase_auth</a>	✓	✓	✓
Cloud Firestore	 			<a href="#">cloud_firestore</a>	✓	✓	✓
				<a href="#">cloud_functions</a>	✓	✓	✓
Cloud Messaging	 			<a href="#">firebase_messaging</a>	✓	✗	✗

<https://firebase.flutter.dev/>



Flutter support for Windows desktop is now in alpha (along with macOS and Linux support)! For more information, see [this free article](#) and [the desktop page](#).

Get started ▾

Samples &amp; tutorials ▾

Development ▴

▸ User interface

▾ Data &amp; backend

▸ State management

Networking &amp; http

JSON and serialization

**Firestore**

▸ Accessibility &amp; internationalization

▸ Platform integration

▸ Packages &amp; plugins

▸ Add Flutter to existing app

▸ Tools &amp; techniques

▸ Migration notes

Testing &amp; debugging ▾

Performance &amp; optimization ▾

Deployment ▾

Resc

# Firestore

[Docs](#) > [Development](#) > [Data & backend](#) > Firestore

Firestore is a Backend-as-a-Service (BaaS) app development platform that provides hosted backend services such as a realtime database, cloud storage, authentication, crash reporting, machine learning, remote configuration, and hosting for your static files.

Firestore supports Flutter. For more information, see:

- The [FlutterFire](#) site
- [Getting started with Firestore and Flutter](#)
- [Firestore for Flutter codelab](#)
- [Multi Platform Firestore Flutter](#)
- [Use Firestore to host your Flutter app on the web](#)

Also, the Flutter community has created docs and videos that you might find useful. Here are a few:

- [Building chat app with Flutter and Firestore](#)
- [Using Firestore as a backend to your Flutter app](#) (video)
- [Live Coding Firestore Authentication with Flutter](#) (video)
- [Flutter & Firestore Auth 01](#) (video)
- [Flutter: Firestore Tutorial Part 1 - Auth and Sign in](#) (video)

<https://flutter.dev/docs/development/data-and-backend/firestore>





## Documentation

[Overview](#)[Guides](#)[Reference](#)[Samples](#)[Libraries](#)

Guides

Get started with Firebase ▸

▸ Add Firebase to an app

▸ Add Firebase to a game

Add Firebase to a server

▾ Use Firebase with a framework

Flutter

Set up projects  
programmatically β

Manage your Firebase projects ▾

Prototype and test with  
Emulator Suite β ▾

Analytics ▾

Extensions β ▾

DEVELOP

Authentication ▾

Realtime Database ▾

Cloud Firestore ▾

Storage ▾

Hosting

Cloud Functions ▾

Google is committed to advancing racial equity for Black communities. [See how.](#)

Firebase &gt; Docs &gt; Guides

[Send feedback](#)

Contents ▾

Prerequisites

Step 1: Create a Firebase project

Step 2: Register your app with Firebase

Step 3: Add a Firebase configuration file

Step 4: Add FlutterFire plugins

...

# Add Firebase to your Flutter app



Follow this guide to add Firebase products to a Flutter app.

★ **Note:** Firebase supports frameworks like Flutter on a best-effort basis. These integrations are not covered by Firebase Support and may not have full feature parity with the [official Firebase SDKs](#).



iOS

Android

## Prerequisites

<https://firebase.google.com/docs/flutter/setup>

- Set up a physical iOS device or the iOS simulator for running your app.





Subscribe

## Firestore CRUD in Flutter



Firestore CRUD in Flutter - A complete guide



Watch later



Share

COMPLETE

CRUD

FIREBASE AND FLUTTER

This tutorial will cover the basics of CRUD in flutter



Join me on Slack



View Code



Download Code



Written by **Dane Mackier**

Lead Developer and Software Architect

<https://www.filledstacks.com/post/firestore-crud-in-flutter/>

📅 26 January 2020

🕒 18 minutes







## Fast, Enjoyable & Secure NoSQL Database







 tests no status
 coverage 95%
 pub.dev v2.0.4
 license Apache-2.0

Hive is a lightweight and blazing fast key-value database written in pure Dart. Inspired by [Bitcask](#).

### Documentation & Samples

If you need queries, multi-isolate support or links between objects check out [Isar Database](#).

## Features

-  Cross platform: mobile, desktop, browser
-  Great performance (see [benchmark](#))
-  Simple, powerful, & intuitive API
-  Strong encryption built in
-  NO native dependencies
-  Batteries included

## Getting Started

Check out the [Quick Start](#) documentation to get started.

## Usage

You can use Hive just like a map. It is not necessary to await `Futures`.

<https://github.com/hivedb/hive>

```
box.put('name', 'David');
```



Hive supports all primitive types, `List`, `Map`, `DateTime`, `BigInt` and `Uint8List`. Any object can be stored using [TypeAdapters](#).

Dart
Format
Reset
▶ Run

```

1 import 'package:hive/hive.dart';
2
3 void main() async {
4   //Hive.init('somePath') -> not needed in browser
5
6   var box = await Hive.openBox('testBox');
7
8   box.put('name', 'David');
9
10  print('Name: ${box.get('name')}');
11 }
  
```

Console

no issues

## Usage

You can use Hive just like a map. It is not necessary to await `Futures`.

<https://github.com/hivedb/hive>

```
box.put('name', 'David');
```



# Flutter save data to local storage with Hive NoSQL database package

Hive is a lightweight and blazing fast key-value database written in pure Dart.



Hasan Basri Bayat

Follow



Sep 26 · 3 min read



<https://itnext.io/flutter-save-data-to-local-storage-with-hive-nosql-database-package-8a0de834f313>



### What is Hive?

Hive is an advanced NoSQL local database.

### What is Box?

All data stored in Hive is organized in boxes. A box can be compared to a table in SQL, but it does not have a structure and can contain anything.

Boxes can also be encrypted to store sensitive data.

Flutter  
with  
package

Hive is a pure Dart. Hive can store both primitive data and objects.



Hasan Basri Bayat

Follow



Sep 26 · 3 min read



<https://itnext.io/flutter-save-data-to-local-storage-with-hive-nosql-database-package-8a0de834f313>



# Isar Database





Hasan Basri Bayat

Mar 6 · 5 min read ★ · [Listen](#)



Save



## The Fastest Local Database in Dart/Flutter

ObjectBox is a gamechanger with its performance and AutoSync option

Hive is great for key-value operations but when it comes to storing objects with advanced queries in that case you'll need **better packages**.

**And they are;**

1. ObjectBox
2. Isar (which is an improved version of Hive) (same author)

I'll give you a minimal introduction for `ObjectBox` for now but I'll also cover `Isar` later so subscribe and stay tuned!

<https://itnext.io/the-fastest-local-database-in-dart-flutter-a65ff5b29fee>



424



2





Hasan Basri Bayat

Mar 6 · 5 min read ★ · [Listen](#)



[Save](#)



## The Fastest Local Database

ObjectBox is a gamechanging local database option

Hive is great for key-value or simple queries, but for advanced queries in that case...

And they are;

1. ObjectBox

2. Isar (which is an improved version of Hive) (same author)

And they are;

1. ObjectBox

2. Isar (which is an improved version of Hive) (same author)

I'll give you a minimal introduction for `ObjectBox` for now but I'll also cover `Isar` later so subscribe and stay tuned!

<https://itnext.io/the-fastest-local-database-in-dart-flutter-a65ff5b29fee>



424

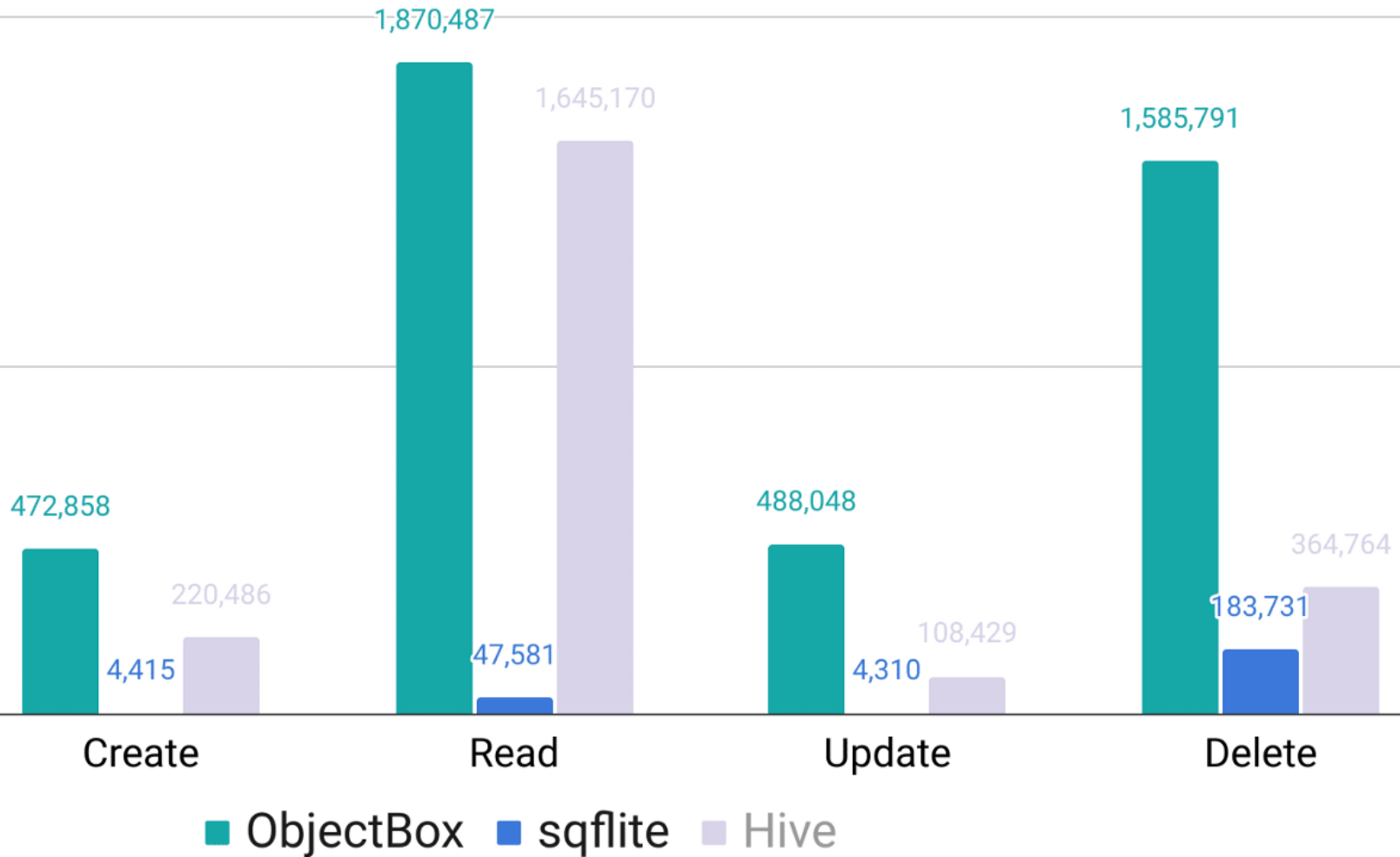


2





## CRUD: objects per second (more is better)



## isar 2.2.1

Published 31 days ago • isar.dev Null safety

SDK DART FLUTTER PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS

417

Readme Changelog Example Installing Versions Scores



## Isar Database

tests passing pub.dev v2.2.1 license Apache-2.0 Follow 1.3k

Quickstart • Documentation • Sample Apps • Support &amp; Ideas • Pub.dev

Isar [ee-zahr]:

1. River in Bavaria, Germany.
2. Database that will make your life easier.

## Features

- ❤️ Made for Flutter. Easy to use, no config, no boilerplate
- 🚀 Highly scalable The sky is the limit (pun intended)
- 🔍 Feature rich. Composite & multi-entry indexes, query modifiers, JSON support etc.
- ⌚ Asynchronous. Parallel query operations & multi-isolate support by default
- 🐙 Open source. Everything is open source and free forever!

Isar can do much more (and we are just getting started)

<https://pub.dev/packages/isar>

- 📱 Multipatform. IOS, Android, Desktop and FULL WEB SUPPORT!
- ✍️ ACID semantics. Rely on consistency

417 130 93%  
LIKES PUB POINTS POPULARITY

Publisher

isar.dev

Metadata

Fast Cross Platform  
Database for Flutter Apps.  
Supports indexes, FTS,  
queries etc.

Homepage

Repository (GitHub)

View/report issues

Documentation

API reference

License

Apache-2.0 (LICENSE)

Dependencies

ffi, js, meta

More

Packages that depend on  
isar

## isar 2.2.1

Published 31 days ago

[SDK](#)
[DART](#)
[FLUTTER](#)
[WEB](#)
[WINDOWS](#)
 417

[Readme](#)
[Changes](#)

Scores

417

LIKES

130

PUB POINTS

93%

POPULARITY

```
part 'post.g.dart';

@Collection()
class Post {
  int id = Isar.autoIncrement;

  late String title;

  late DateTime date;
}
```






```
final dir = await getApplicationSupportDirectory(); // path_provider package
final isar = await Isar.open(
  schemas: [PostSchema],
  directory: dir.path,
  inspector: true, // if you want to enable the inspector for debug builds
);
```

```
final posts = await isar.posts.filter()
  .titleContains('awesome', caseSensitive: false)
  .sortByDateDesc()
  .limit(10)
  .findAll();
```

Isar [ee-zee]

1. River in Bavaria, Germany.
2. Database

## Features

-  Made for Flutter
-  Highly scalable
-  Feature rich
-  Asynchronous
-  Open source

```
final newPost = Post()..title = 'Amazing new database';



await isar.writeTxn((isar) {
  newPost.id = await isar.posts.put(newPost); // insert
});

final existingPost = await isar.get(newPost.id!); // get

await isar.writeTxn((isar) {
  await isar.posts.delete(existingPost.id!); // delete
});
```

Isar can do much more (and we are just getting started)

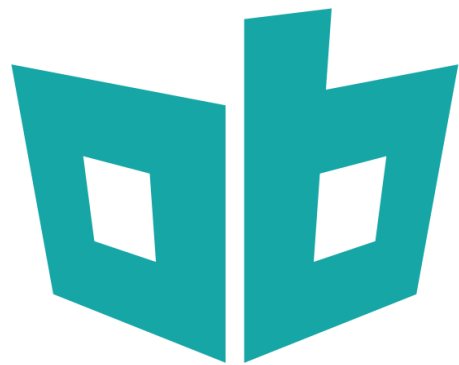
<https://pub.dev/packages/isar>

-  Multiplatform. iOS, Android, Desktop and FULL WEB SUPPORT!!
-  ACID semantics. Rely on consistency

[Publisher](#)
[isar.dev](#)
[Metadata](#)

Fast Cross Platform  
Database for Flutter Apps.  
Supports indexes, FTS,  
queries etc.

[Homepage](#)
[Repository \(GitHub\)](#)
[View/report issues](#)
[Documentation](#)
[API reference](#)
[License](#)
[Apache-2.0 \(LICENSE\)](#)
[Dependencies](#)
[ffi, js, meta](#)
[More](#)
[Packages that depend on isar](#)



# OBJECTBOX

# objectbox 1.4.1

Published 20 days ago • [objectbox.io](#) Null safety

[SDK](#) [DART](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [LINUX](#) [MACOS](#) [WINDOWS](#)

481

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)



## Flutter database for Dart-native objects ❤️

pub v1.4.1

Super-fast Flutter database for storing and syncing Dart objects

- 🚩 **High performance** - improving response rates and enabling real-time applications.
- 💎 **ACID compliant** - Atomic, Consistent, Isolated, Durable.
- 🖥️ **Multiplatform** - Android, iOS, macOS, Linux, Windows.
- 🌱 **Scalable** - grows with your app, handling millions of objects with ease.

Easy to use

- 🔗 **Relations** - object links / relationships are built-in.
- 🔍 **Queries** - filter data as needed, even across relations.
- 🛠️ **Statically typed** - compile time checks & optimizations.
- 📄 **Schema migration** - change your model with confidence.

Oh, and there is one more thing...

- 🔄 **Data Sync** - keeps data in sync offline or online, between devices and servers.

Sneak peek - persist Dart objects with ObjectBox

<https://pub.dev/packages/objectbox>

```
class Person {
```

## objectbox 1.4.1

Published 20 days ago • objectbox.io (Null safety)

[SDK](#)
[DART](#)
[FLUTTER](#)
[PLATFORM](#)
[ANDROID](#)
[IOS](#)
[LINUX](#)
[MACOS](#)
[WINDOWS](#)

481

[Readme](#)
[Changelog](#)
[Example](#)
[Installing](#)
[Versions](#)
[Scores](#)

## Flutter data

pub v1.4.1

Super-fast Flutter databa

- High performance
- ACID compliant
- Multiplatform - A
- Scalable - grows

Easy to use

- Relations - objec
- Queries - filter da
- Statically typed
- Schema migratio

Oh, and there is one more

-  [Data Sync](#) - keeps data in sync offline or online, between devices and servers.

Sneak peek - persist Dart objects with ObjectBox

<https://pub.dev/packages/objectbox>

```
@Entity()
class Person {
  int id;

  String firstName;
  String lastName;

  Person({this.id = 0, required this.firstName, required this.lastName});
}

final store = await openStore();
final box = store.box<Person>();

var person = Person(firstName: 'Joe', lastName: 'Green');

final id = box.put(person); // Create

person = box.get(id)!;      // Read

person.lastName = "Black";
box.put(person);           // Update

box.remove(person.id);     // Delete

// find all people whose name start with letter 'J'
final query = box.query(Person_.firstName.startsWith('J')).build();
final people = query.find(); // find() returns List<Person>
```

```
class Person {
```



Found but did not explore...



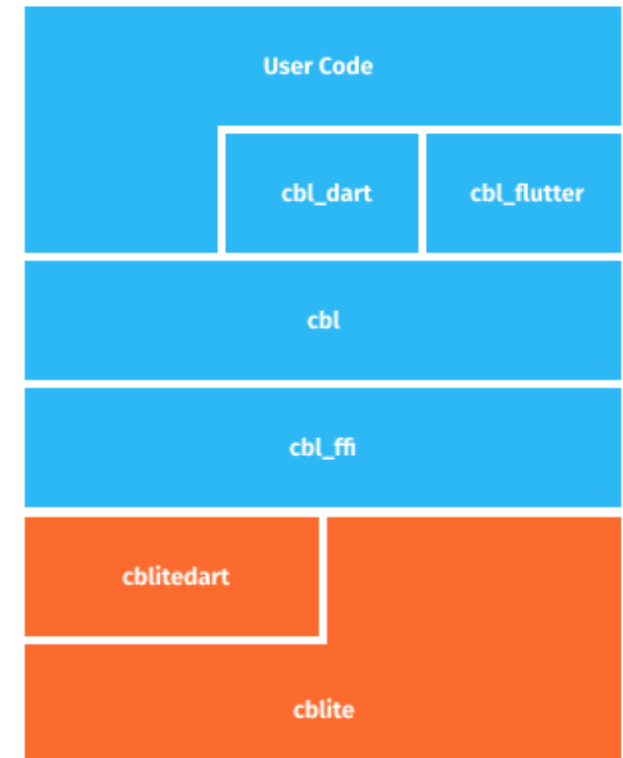
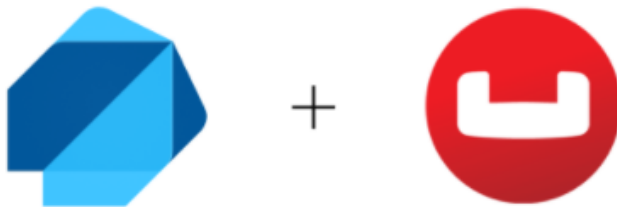
Gabriel Terwesten

Dec 16, 2021 · 6 min read · [Listen](#)



## How to make your Flutter app offline-first with Couchbase Lite

Why I developed Couchbase Lite for Dart + Flutter, how to get started with it and how to implement full-text search locally



Oh wow, it has almost been a year, already. Time flies. In January, I was working on a Flutter app as a side project and wanted to make it offline-first. As many features should work as well as possible, with bad or no connectivity. With mobile network availability always improving, one might think that the offline-

<https://medium.com/flutter-community/how-to-make-your-flutter-app-offline-first-with-couchbase-lite-86bb23780f74>  
responsiveness and user privacy, among other things.

# cbl 1.0.0-beta.11

Published 4 months ago • [gabriel.terwesten.net](#) • Latest: 1.0.5

[UNIDENTIFIED]

25

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

pub v1.0.5 license Apache-2.0 CI passing codecov 91%

Couchbase Lite is an embedded, NoSQL JSON Document Style database, supporting Blobs, Encryption, N1QL Queries, Live Queries, Full-Text Search and Data Sync.

It can be used as a standalone embedded database for mobile or desktop apps. Or it can be combined with [Sync Gateway](#) to synchronize with a central data store.

This package provides a Dart API through which Couchbase Lite can be used on all native platforms which are supported by Dart.

This package is in beta. Use it with caution and [report any issues you see](#).

## Table of contents

- [Features](#)
- [Limitations](#)
- [Getting started](#)
- [Key concepts](#)
  - [Synchronous and Asynchronous APIs](#)
  - [Change listeners](#)
  - [Change streams](#)
  - [Closing resources](#)
- [Usage examples](#)
  - [Open a database](#)
  - [Create a document](#)
  - [Read a document](#)
  - [Update a document](#)
  - [Delete a document](#)
  - [Build a query with N1QL](#)

<https://pub.dev/packages/cbl/versions/1.0.0-beta.11#-features>



# The END

