〈 Recommendations

Architecture design patterns

Architecture > Design patterns

If you've already read through the architecture guide page, or if you're comfortable with Flutter and the MVVM pattern, the following articles are for you.

These articles aren't about high-level app architecture, rather they're about solving specific design problems that improve your application's code base regardless of how you've architected your app. That said, the articles do assume the MVVM pattern laid out on the previous pages in the code examples.

Optimistic state

user experience | asynchronous dart

Improve the perception of responsiveness of an application by implementing optimistic state.

Persistent storage architecture: Key-value data

data | shared-preferences | dark mode

Save application data to a user's on-device key-value store.

Persistent storage architecture: SQL

data | SQL

Save complex application data to a user's device with SQL.

Offline-first support

data | user experience | repository pattern

Implement offline-first support for one feature in an application.

The command pattern

mvvm | asynchronous dart | state

Simplify view model logic by implementing a Command class.

Error handling with Result objects

error handling | services

Improve error handling across classes with Result objects.

〈 Recommendations