Exploring Flutter

A powerful Tool for Cross-Platform App Development

GitHub Repo

GitHub PDF

Franklin Developer Lunch & Learn

Meetup – Franklin, TN, USA

By Richard Hoehn – June 2024

Flutter

About Me

Living in Franklin, TN
since 98

•

Grew up in Switzerland
Zurich Area

•

Machinist by Trade
CNC Programmer

▼

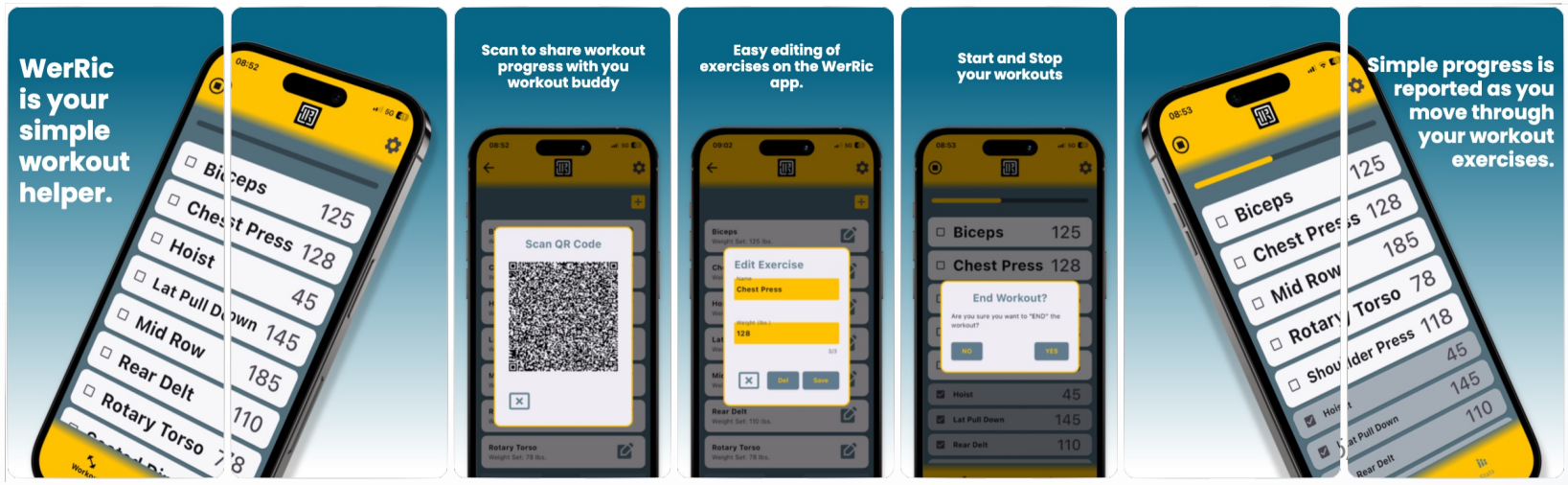
PLC – Ladder Logic

▼

Embedded C – ARM & AVR

▼

App & Web Development



Introduction & Agenda

Introduction – Why I like Flutter…

- A **cross-platform** UI development Platform | Framework | SDK – **Code once!**
- Language **Dart** that is easy to use and **Declarative** in nature, but still strongly typed
- Easy to use Tool-Chain and Good Documentation
- *and Finally* **Open-Source** with a large community supporting module development

Agenda ~ 40min

- What is Flutter, the Architecture, and the SDK Setup
- Building an App, Widgets (*both Stateless and Stateful*), and the use of State Mgmt.
- Firebase and how Google backend support enabled quick development
- *and finally,* Review of Unique Features, Demo, and Conclusions

What is Flutter?

An open-source UI software development kit (SDK) created by Google – Released end of 2018.

•

Cross-platform app development allows developers to use one programming language and one codebase to build an application for multiple platforms.

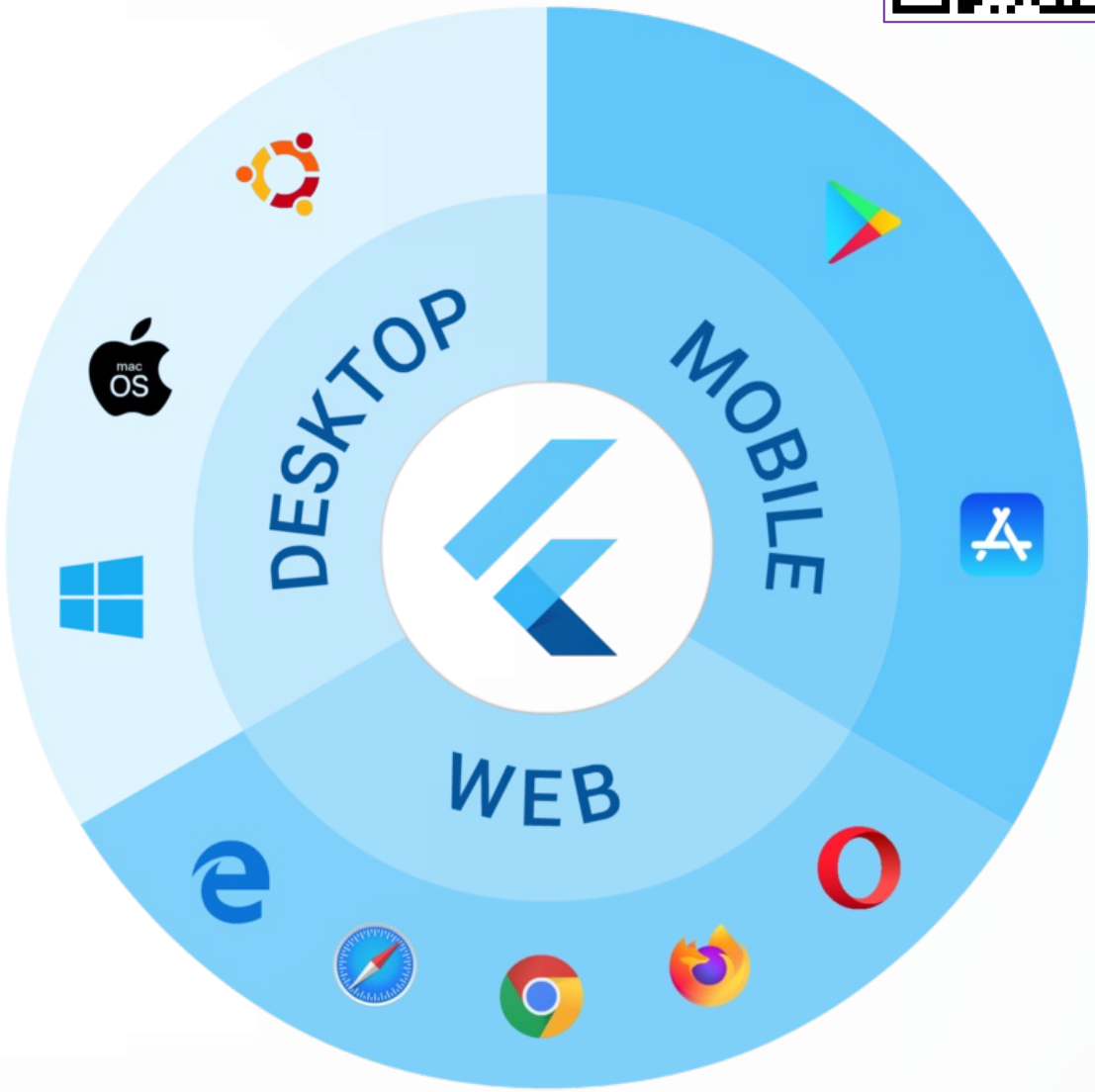
Your Code is **Native Compiled** to

Mobile (iOS, Android)

Web

Desktop (Windows, macOS, Linux)

From a Single codebase!



https://medium.com/swlh/flutter-2020-state-of-cross-platform-814f1d8ff16

Flutter

What is Flutter?

Dart – Programing Language

Released by Google - 2011

Optimized for UI Development

- Dart is designed with a focus on UI development due to it's declarative nature.
- It's expansive standard library and easy syntax simplify the development of complex and responsive UIs.

Ahead-of-Time (AOT) & Just-in-Time (JIT) Compilation

- AOT compilation enables Flutter apps to be compiled into optimized, native machine code, ensuring fast startup times and high performance.
- JIT compilation allows for hot reload, enabling developers to see code changes in real-time without restarting the app.

Worldwide, Jun 2024 :

Rank	Change	Language	Shar	
1		Python	29.06 %	
2		Java	15.97 %	
3		JavaScript	8.7 %	-0.6 %
4		C#	6.73 %	-0.0 %
5		C/C++	6.4 %	-0.0 %
6	↑	R	4.75 %	+0.3 %
7	↓	PHP	4.57 %	-0.5 %
8		TypeScript	3.0 %	-0.1 %
9		Swift	2.76 %	+0.3 %
10		Rust	2.5 %	+0.4 %
11		Objective-C	2.39 %	+0.3 %
12		Go	2.25 %	+0.3 %
13		Kotlin	1.98 %	+0.1 %
14		Matlab	1.47 %	-0.2 %
15	↑↑↑	Dart	1.02 %	+0.1 %
16		Ruby	1.0 %	-0.1 %
17	↑↑	VBA	0.96 %	+0.0 %
18	↓	Powershell	0.88 %	-0.1 %
19	↓↓↓↓	Ada	0.87 %	-0.3 %
20		Scala	0.61 %	-0.1 %
21		Lua	0.6 %	+0.0 %
22	↑	Abap	0.44 %	-0.1 %
23	↓	Visual Basic	0.34 %	-0.2 %
24		Julia	0.27 %	-0.1 %
25	↑	Perl	0.24 %	-0.1 %
26	↑↑	Haskell	0.12 %	-0.1 %
27	↓↓	Groovy	0.07 %	-0.3 %
28	↓	Cobol	0.05 %	-0.2 %

https://pypl.github.io/PYPL.html

Flutter

What is Flutter?
Dart – Declarative

Declarative Style:
The UI is built using a tree of widgets that describe the structure and appearance of the app at any given state.

State Management:
The state is managed using the setState method to trigger a rebuild of the UI with the new state.

Readability:
The code is easy to read and understand because it focuses on what the UI should look like, not how to construct it step by step.

main.dart

```dart
class _GreetingScreenState extends State<GreetingScreen> {
  bool _isHello = true;

  void _toggleGreeting() {
    setState(() {
      _isHello = !_isHello;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Declarative Programming')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              _isHello ? 'Hello, World!' : 'Goodbye, World!',
              style: TextStyle(fontSize: 24),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: _toggleGreeting,
              child: Text('Toggle Greeting'),
            ),
          ],
        ),
      ),
    );
  }
}
```

Flutter

What is Flutter?
Comparisons

Aspect	Flutter	React Native	Xamarin
Language and Framework	**Dart**, developed by Google	**JavaScript** and **React**, developed by Facebook	**C#** and **.NET**, developed by Microsoft
Performance	Compiles to native ARM code High performance	Uses JavaScript bridge May have performance issues	Compiles to native code Good performance
Development Experience	Hot reload for instant updates	Hot reload for real-time changes	Hot reload and live reload, more complex management
Community and Ecosystem	Growing community, strong Google backing	Large mature community, extensive resources	Established community, strong Microsoft support
Learning Curve	Easy to learn Dart	Familiar JavaScript Complex React learning	Steeper due to C# and .NET framework complexity
Cost and Licensing	Open-source and free	Open-source and free	Open-source and free Enterprise features may require a subscription

Flutter

What is Flutter?

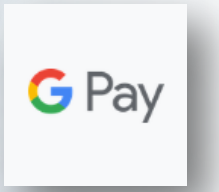
Popular Apps by Flutter

BMW

My BMW App was launched in July 2020 and has established itself in 47 countries on five continents as a universal interface providing the user a seamless experience between mobile phone and vehicle.

eBay Motors

eBay Motors ships a new version of their app to both app stores every week. Their apps share 98.3% of their code, the eBay Motors app has a single source of truth – which means one set of meetings, one set of designs, one backlog queue, and one team to manage.



Google Pay

1.1 million lines of code instead of 1.7 million from previous native versions. At the same time, the team estimates that they've saved about 60-70% of their engineers' time because a single codebase.

Nubank

Nubank is the largest independent digital bank outside of Asia, giving over 48 million users financial banking tools. Flutter provides more UI consistency.

Flutter Architecture

Flutter Framework – *Developers Work Here*

It is a very core part of Flutter app architecture. Flutter framework has got the material parts, widgets, or the Cupertino, and it becomes the foundation element of application development.

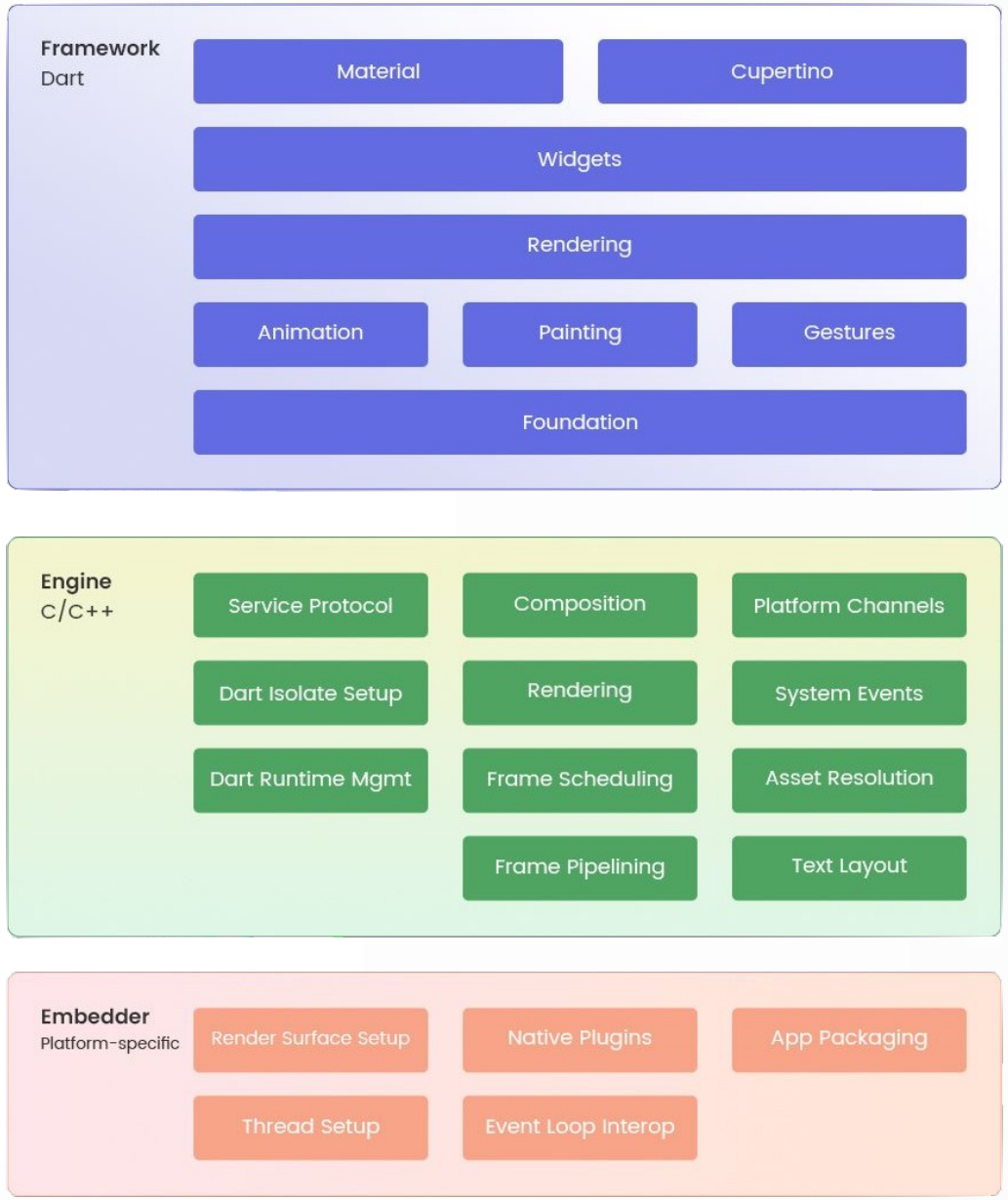
Flutter Engine – *Google Team Works Here*

Written in C++, this Flutter engine tackles and handles the heavy liftings in software development like the network requests, input, and output, along with managing the complex translations of rendering.

Embedder – *Specific to Deployment OS*

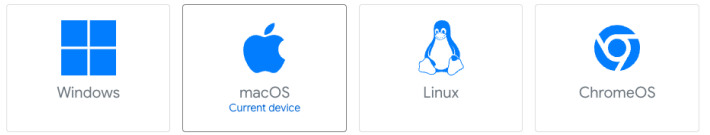
The embedder layers undertake the heavy lifting necessary to translate how Flutter language works with the particular OS.

Flutter user interfaces are developed, composited, and painted by Flutter itself and not translated into a similar OS Widget.



https://radixweb.com/blog/overview-of-flutter-architecture

Setting Up Flutter



System Requirements

- Windows, Mac, and Linux – All need to be 64bit
- Disk – You will need ~2GB
- GIT by default (*.gitignore added*)

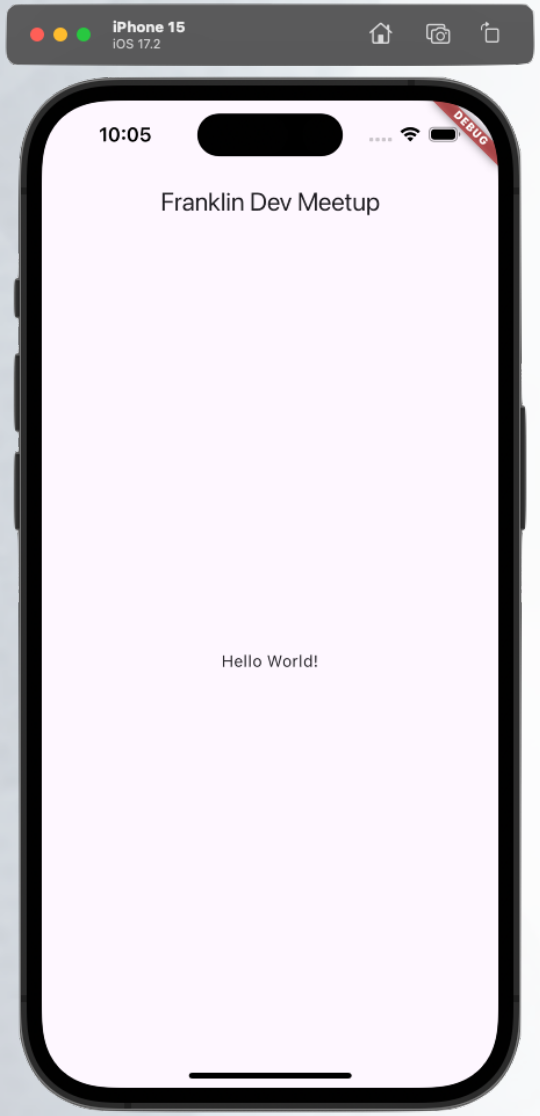
```
(base) ➜  meetup.franklindev.flutter git:(main) ✗ flutter create -e app
Developer identity "Apple Development: Richard Hoehn (GYCQF4QSM4)" selected for iOS code signing
Creating project app...
Resolving dependencies in `app`...
Downloading packages...
Got dependencies in `app`.
Wrote 128 files.

All done!
You can find general documentation for Flutter at: https://docs.flutter.dev/
Detailed API documentation is available at: https://api.flutter.dev/
If you prefer video documentation, consider: https://www.youtube.com/c/flutterdev

In order to run your empty application, type:

  $ cd app
  $ flutter run

Your empty application code is in app/lib/main.dart.

(base) ➜  meetup.franklindev.flutter git:(main) cd app
(base) ➜  app git:(main)
```

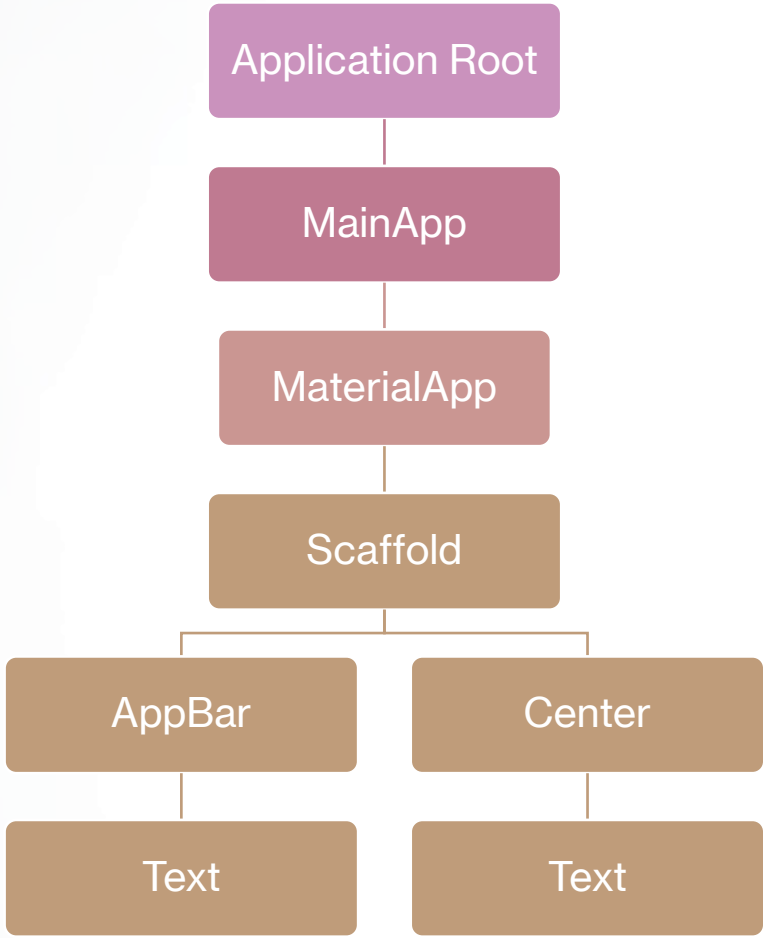
Flutter Doctor

- Environment Check
- Diagnostic on Miss-Configuration
- Cross Platform Support on different OS Developer machines

```
(base) ➜  app git:(main) flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.22.0, on macOS 14.5 23F79 darwin-arm64, locale en-US)
[✓] Android toolchain - develop for Android devices (Android SDK version 33.0.2)
[✓] Xcode - develop for iOS and macOS (Xcode 15.4)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2022.1)
[✓] VS Code (version 1.89.1)
[✓] Connected device (4 available)
[✓] Network resources

• No issues found!
(base) ➜  app git:(main)
```

Widget Tree – All Things are Widgets





main.dart

```dart
import 'package:flutter/material.dart';

void main() {
  runApp(const MainApp());
}

class MainApp extends StatelessWidget {
  const MainApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Franklin Dev Meetup'),
        ),
        body: const Center(
          child: Text('Hello World!'),
        ),
      ),
    );
  }
}
```

https://10015.io/tools/code-to-image-converter

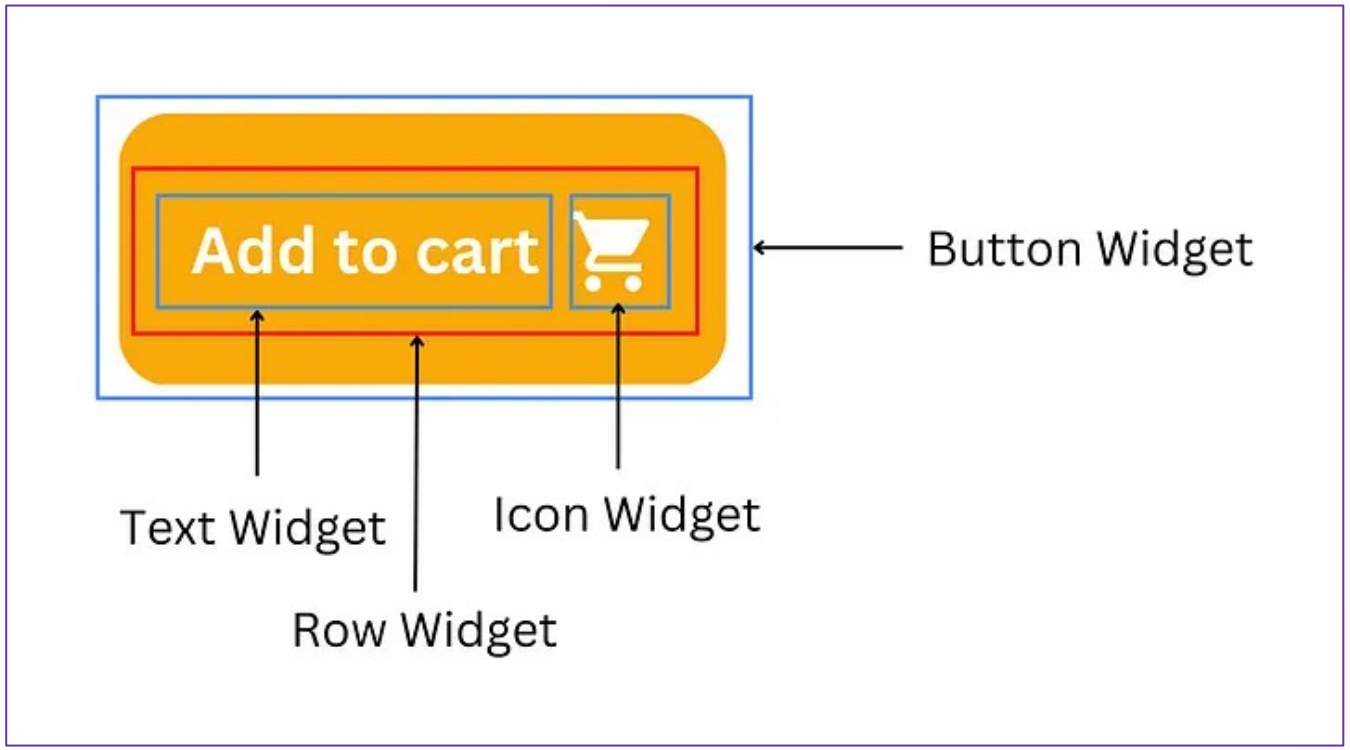
Flutter

Widget Tree

Nesting of Widgets

main.dart

```dart
class ButtonWidget extends StatelessWidget {
  const ButtonWidget({super.key});

  @override
  Widget build(BuildContext context) {
    return ElevatedButton(
      onPressed: () {
        print('Button Pressed');
      },
      child: const Row(
        children: [
          Text('Add to cart'),
          Icon(Icons.shopping_cart),
        ],
      ),
    );
  }
}
```



https://medium.com/gytworkz/understanding-flutter-widgets-and-architecture-98ee1f209b96

Flutter

Widget Tree
Stateful & Stateless Widgets

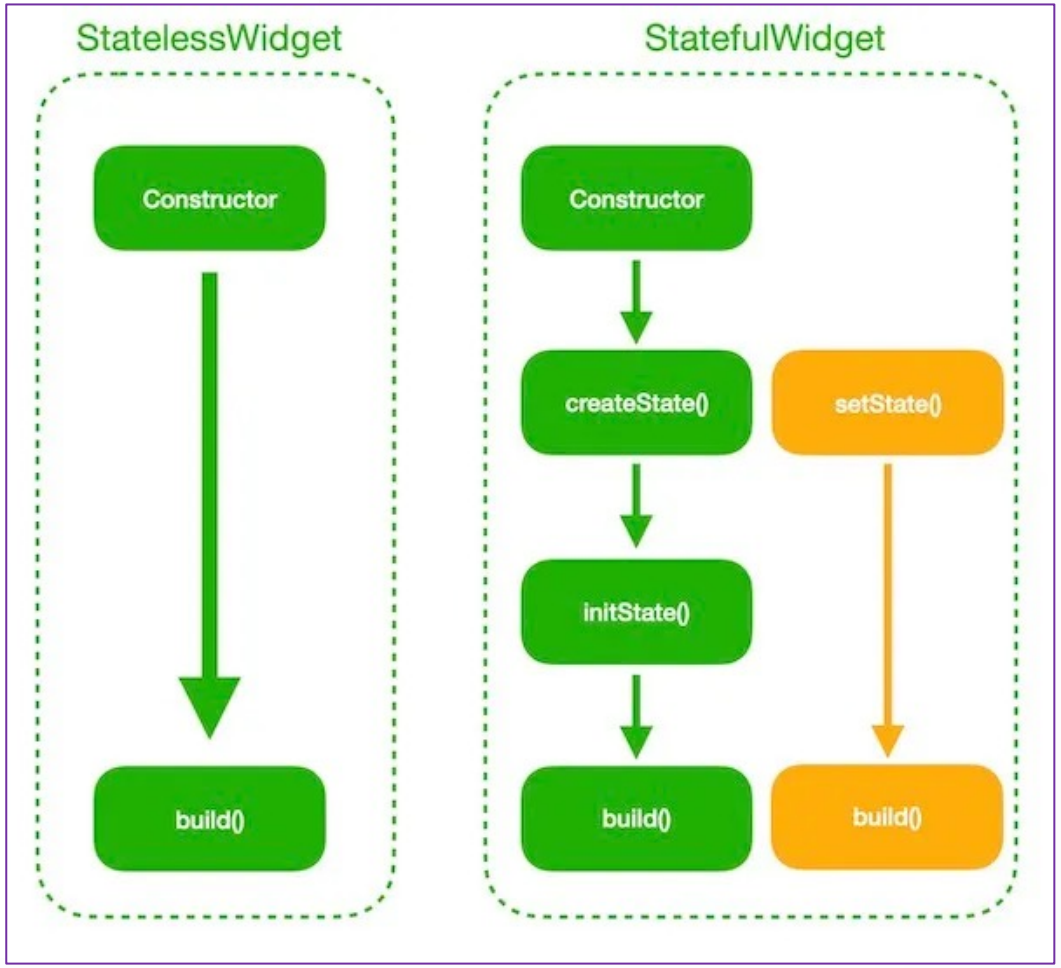
Flutter has 2 Types of Widgets
Stateless & Stateful

Stateless – *Immutable*
…widgets do not contain state hence they can be updated only when it's parent changes.

Stateful – *Mutable*
…widgets can hold the states internally, so they can be updated when its states changes and when it's parent changes.

Why?
…straight forward lifecycle and memory management.

Developers can use stateless widgets for static content and stateful widgets for dynamic content, optimizing both performance and interactivity.



https://medium.com/gytworkz/understanding-flutter-widgets-and-architecture-98ee1f209b96

State Management

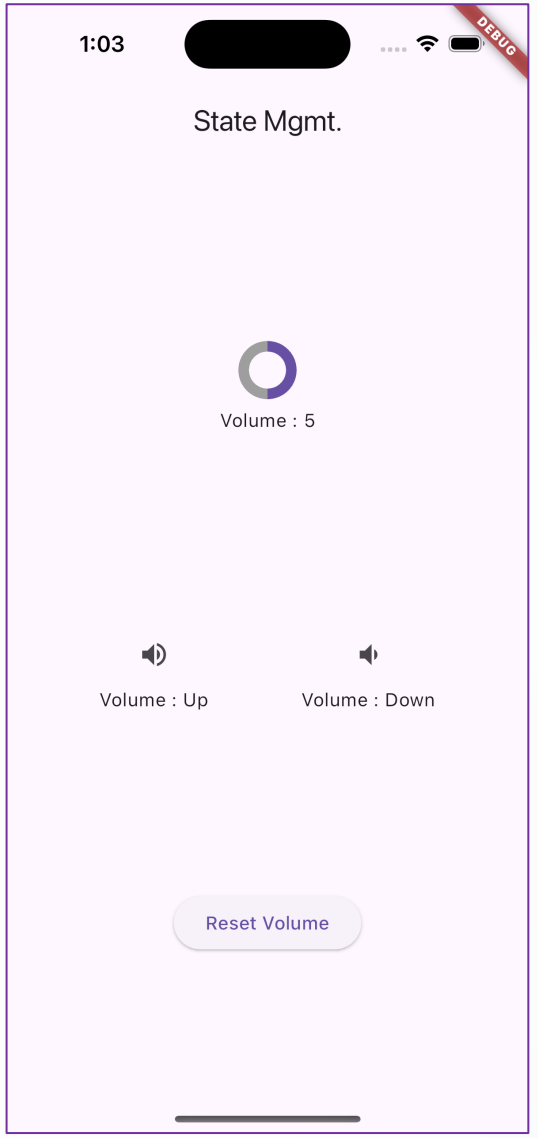
Reactive programming updates the (UI) anytime the underlying data (state) changes.

Centralized State Control:
State management allows you to centralize the control of the application's state, making it easier to manage and update the state consistently across the app.

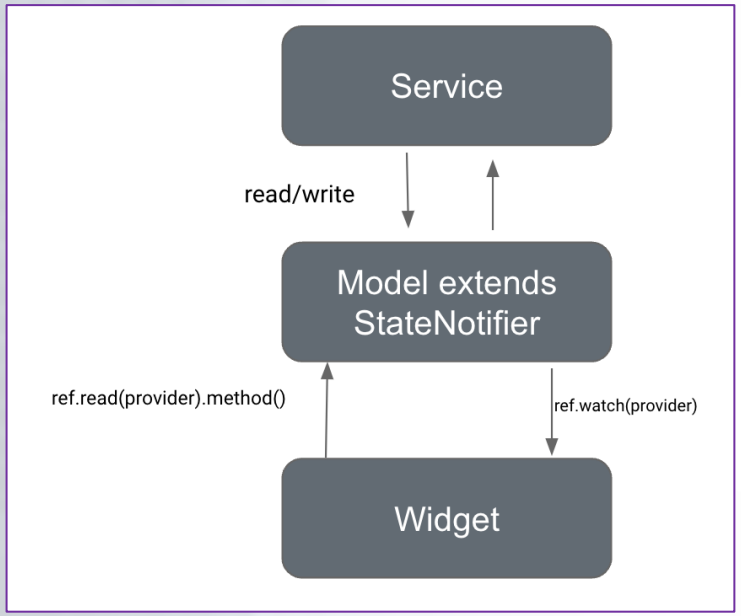
Improved Scalability:
As your application grows, state management helps maintain a clean and organized codebase by separating the UI from the business logic.

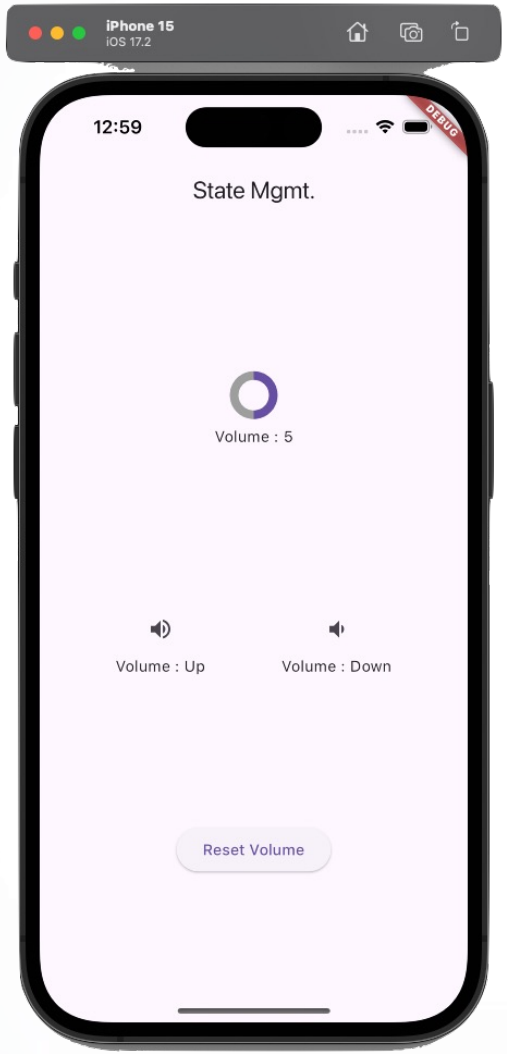
Enhanced Debugging and Testing:
With state management, debugging and testing become more straightforward because the state transitions and changes are controlled and observable.

React Native Equivalent:
This would be Redux or MobX



State Management
Riverpod Example





main.dart

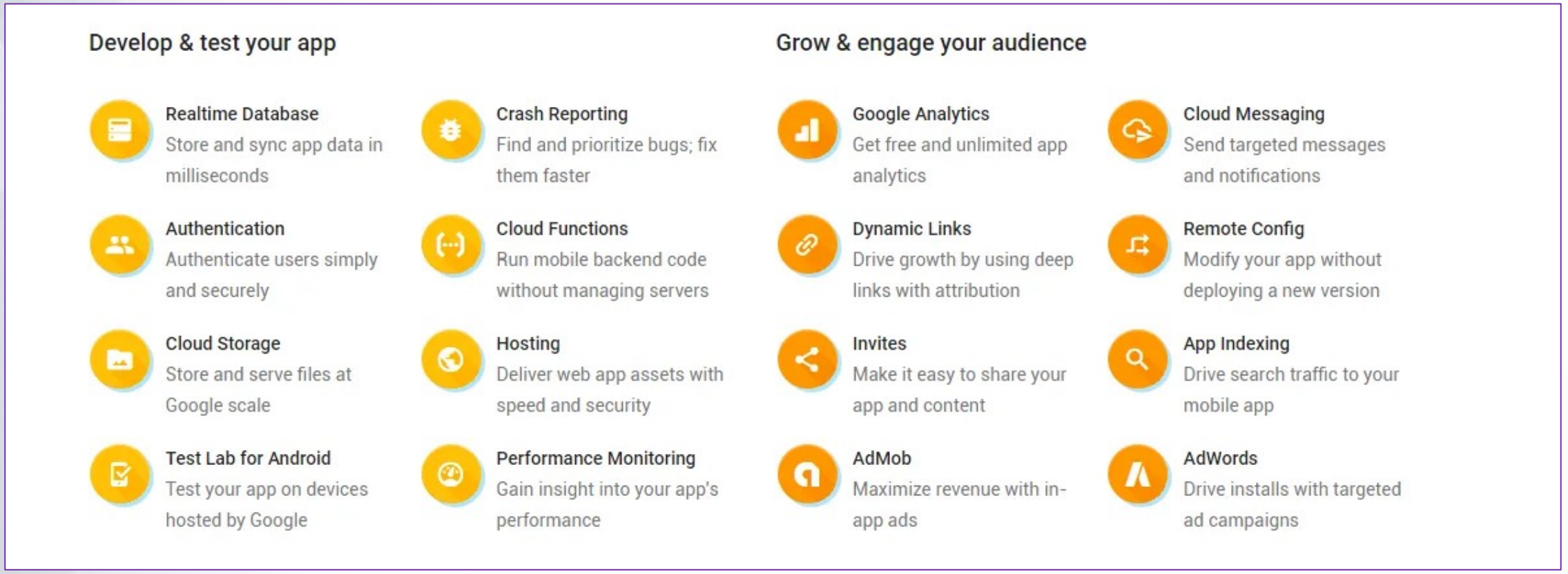
```dart
import 'package:flutter_riverpod/flutter_riverpod.dart';

// Define a VolumeNotifier class that extends StateNotifier<int>
class VolumeNotifier extends StateNotifier<int> {
  VolumeNotifier() : super(5);

  // Method to increment the counter
  void increment() {
    if (state < 10) {
      state++;
    }
  }

  // Method to decrement the counter
  void decrement() {
    if (state > 0) {
      state--;
    }
  }

  // Method to reset the counter
  void reset() {
    state = 5;
  }
}

// Create a provider for the VolumeNotifier
final volumeProvider = StateNotifierProvider<VolumeNotifier, int>((ref) {
  return VolumeNotifier();
});
```

Firebase

Provides a set of **Tools & Services** that work together, simplifying the development process.

Offers **real-time data synchronization** and communication

Supports **apps of all sizes**, from small projects to large-scale applications (*Note: $$$ pricing like AWS*)



Flutter

Firebase
Four (4) Key Modules I use

Authentication

Firebase Authentication provides an easy way to handle user authentication. It supports various authentication methods such as email/password, Google, Facebook, and phone number authentication. This allows you to securely authenticate users without having to build your own backend authentication system.

Realtime Database and Firestore

Firebase offers two cloud-hosted database solutions: Realtime Database and Firestore. Realtime Database is a NoSQL database that stores data in a **JSON** format and synchronizes data in real-time across all clients. Firestore, on the other hand, is a more flexible and scalable NoSQL database that supports complex querying and offline data access..

Cloud Functions

Firebase Cloud Functions (*AWS Lambda*) allow you to run backend code in response to events triggered by Firebase features and HTTPS requests. This serverless architecture helps you execute code without managing servers, enabling you to extend the functionality of your Flutter app with custom backend logic.

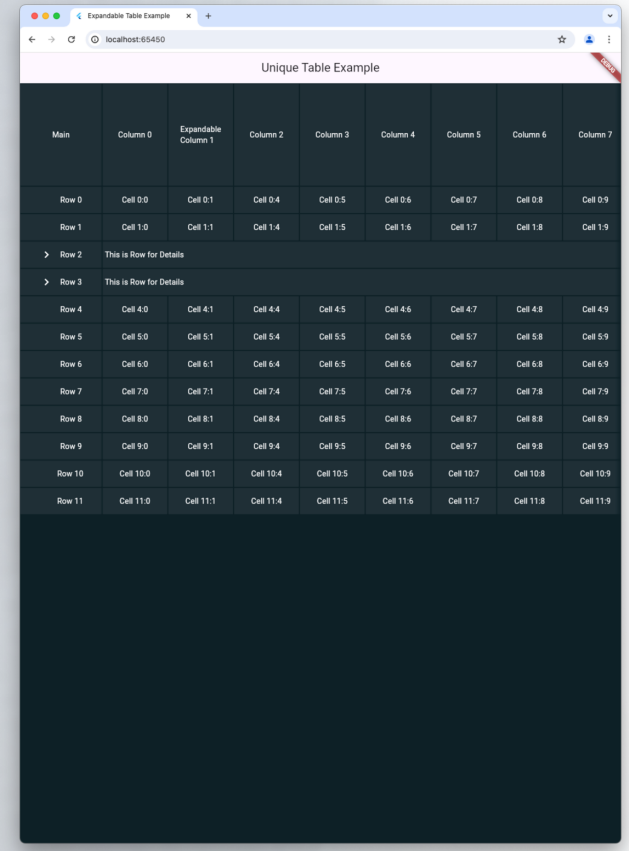
Analytics and Performance Monitoring

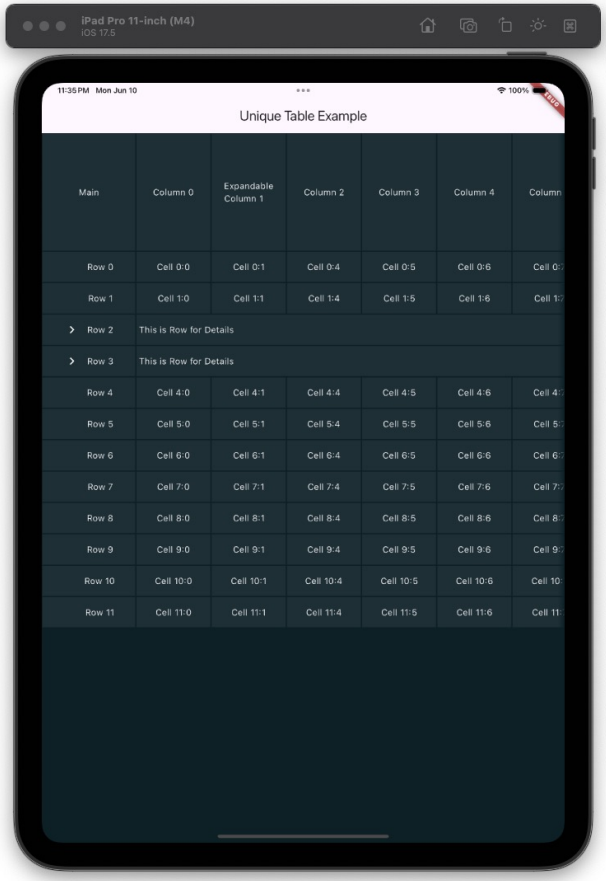
Firebase Analytics provides insights into user behavior and engagement, helping you understand how users interact with your app. Additionally, Firebase Performance Monitoring offers real-time performance data, allowing you to identify and fix performance issues.

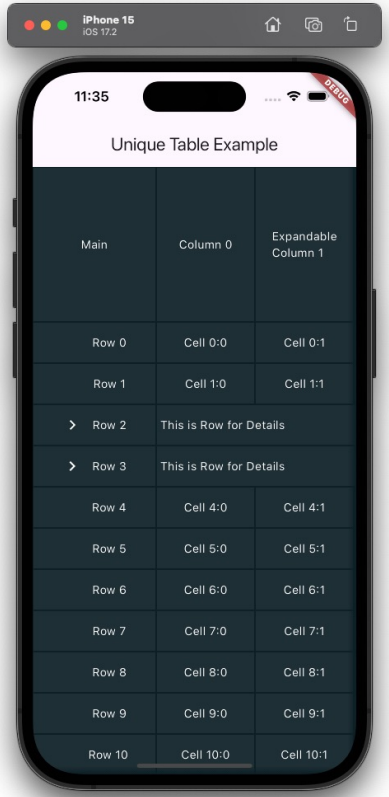
Unique Features of Flutter

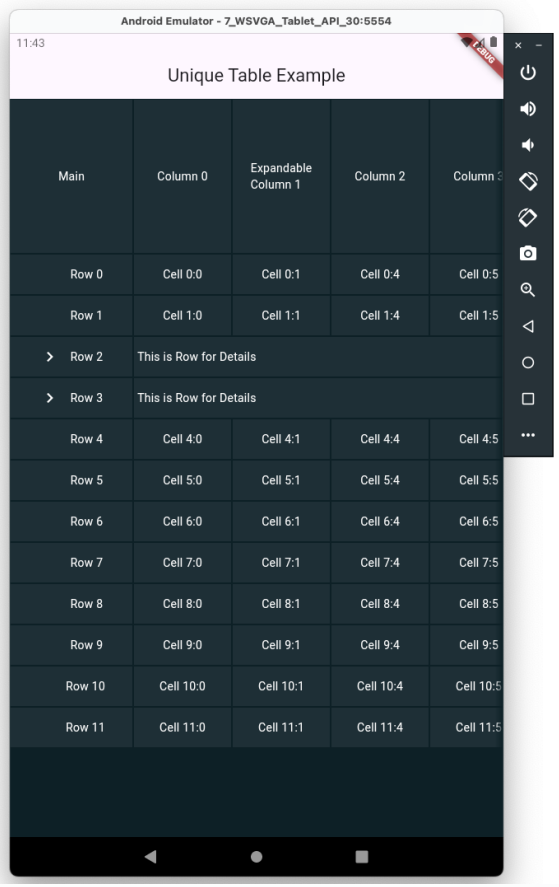
Consistent UI across platforms due to custom rendering engine!

Demo Example!









https://pub.dev/packages/flutter_expandable_table/example

Flutter