



العطاء الرقمي  
Attaa Digital



# تطوير التطبيقات باستخدام Flutter

Eng. Mouaz M. Al-Shahmeh  
Google Dev Library Contributor

Google Developers

## Ask Me Anything

m.m.shahmeh@gmail.com


g.dev/mouaz\_m\_shahmeh



### LINKS

 [github.com/Eng-Mouaz-M-AlShahmeh](https://github.com/Eng-Mouaz-M-AlShahmeh)

 [linkedin.com/in/mouaz-shahmeh](https://linkedin.com/in/mouaz-shahmeh)


 [stackoverflow.com/users/18449528](https://stackoverflow.com/users/18449528)


 [twitter.com/mouaz\\_m\\_shahmeh](https://twitter.com/mouaz_m_shahmeh)




### Mouaz M. Shahmeh

Developer at AlRaedah  
He/Him

[g.dev/mouaz\\_m\\_shahmeh](https://g.dev/mouaz_m_shahmeh) 

 Public


**CITY/TOWN**


 Riyadh Saudi Arabia

**BIO**

Software Engineer and Developer.  
General Director Developer of real-estate  
Tadawl App. Founder of TQDR Pay system.

**STATS**

 62 • Badges earned

 9 • Pages saved

# أهداف اللقاء

# أهداف اللقاء

1

بناء تطبيقات بشفرة كود  
واحدة بلغة البرمجة DART  
ولمنصات عديدة

2

بناء تطبيقات تفاعلية ومتجاوبة  
لمقاسات الشاشات

3

التعرف على البرمجة الآمنة  
للقيم الفارغة

4

بناء تطبيقات جميلة وسريعة  
وبأداء أصلي بتقنية Flutter





# أهداف اللقاء

إدارة البيانات في التطبيق  
بفاعلية والتعامل مع قواعد  
بيانات علائقية وغير علائقية

7

استخدام تنقل فعال بين  
شاشات التطبيق

5

برمجة التطبيقات بأكواد نظيفة  
عن طريق إدارة الحالة

8

اختبار التطبيقات بشكل حقيقي  
وتجهيزها للنشر في متاجر  
التطبيقات

6



# محاوِر اللقاء

# محاوِر اللقاء

تشغيل أول تطبيق Flutter

3

مقدمة في Flutter

1

Declarative vs Imperative

4

تهيئة بيئة التطوير

2



# محاوِر اللقاء

State Management

7

التصميم المتجاوب

5

المفاتيح - التنقل

8

Null Safety

6



# محاوِر اللقاء

أنماط بناء التطبيقات

11

خيارات قواعد البيانات

9

المشروع - ملخص ومراجع

12

اختبارات الأداء

10



1

# Flutter مقدمة في

# مقدمة في Flutter

## What is Flutter?

Flutter is Google's portable UI toolkit for crafting beautiful, natively compiled applications for mobile, web, and desktop from a single codebase. Flutter works with existing code, is used by developers and organizations around the world, and is free and open source.



# مقدمة في Flutter

تطوير التطبيقات الأصلي:  
اندرويد (Kotlin - Java)  
ابل (Objective C - Swift)

تطبيقات ثلاثية الأبعاد (Unity)

Hot Reload - Hot Restart

One Code Base (Dart)

بيئات تطوير منافسة  
cross-platform  
(KMM - Xamarin - React Native)

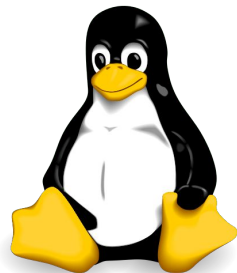




# مقدمة في Flutter



android



Windows 11

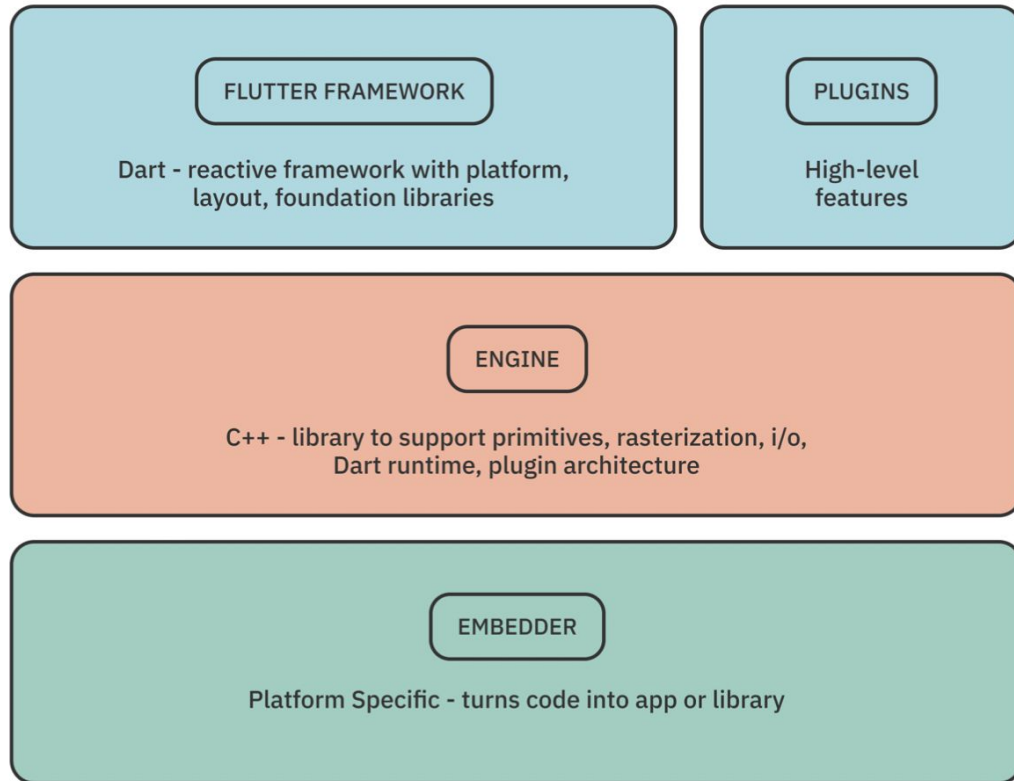
تطبيقات الجوال (اندرويد - ابل)

تطبيقات سطح المكتب (ويندوز - ماك - لينكس)

تطبيقات الويب



# مقدمة في Flutter



# مقدمة في Flutter

FLUTTER FRAMEWORK

UI Theme - Cupertino / Material

Widgets - Styling, Text, Controls

Rendering

Foundation - Painting, Animation, Gestures



# Flutter في مقدمة

## Welcome to Skia: The 2D Graphics Library

[Learn More](#)

Skia is an open source 2D graphics library which provides common APIs that work across a variety of hardware and software platforms. It serves as the graphics engine for Google Chrome and Chrome OS, Android, Flutter, and many other products.

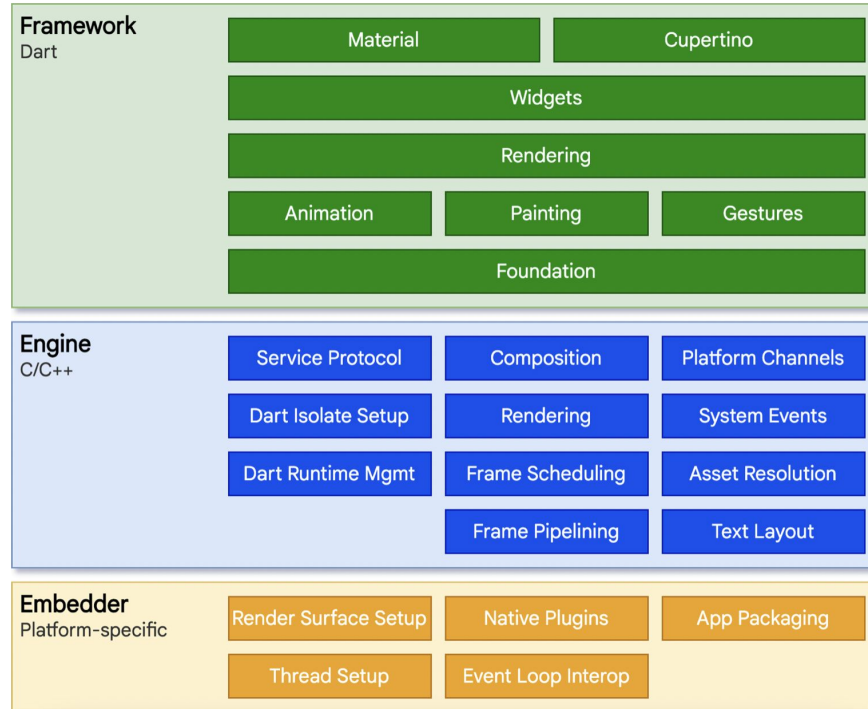


### Supported Platforms

Windows 7, 8, 8.1, 10  
macOS 10.10.5 or later  
iOS 11 or later  
Android 4.1 (JellyBean) or later  
Ubuntu 18.04+, Debian 10+, openSUSE  
15.2+, or Fedora Linux 32+

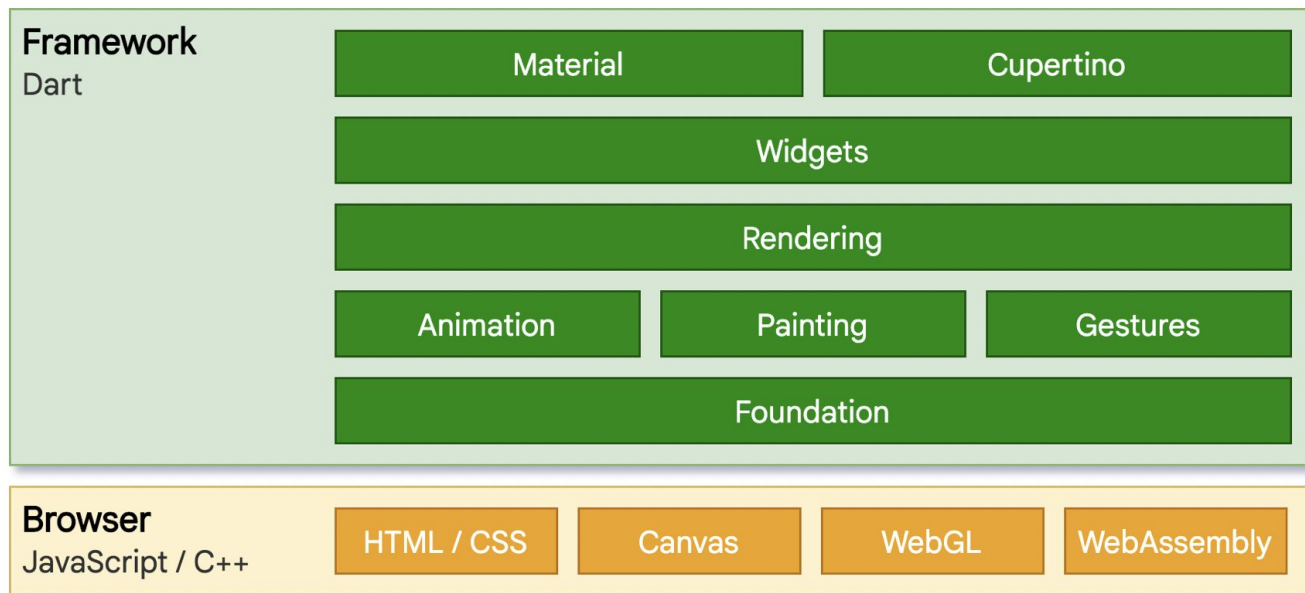


# مقدمة في Flutter

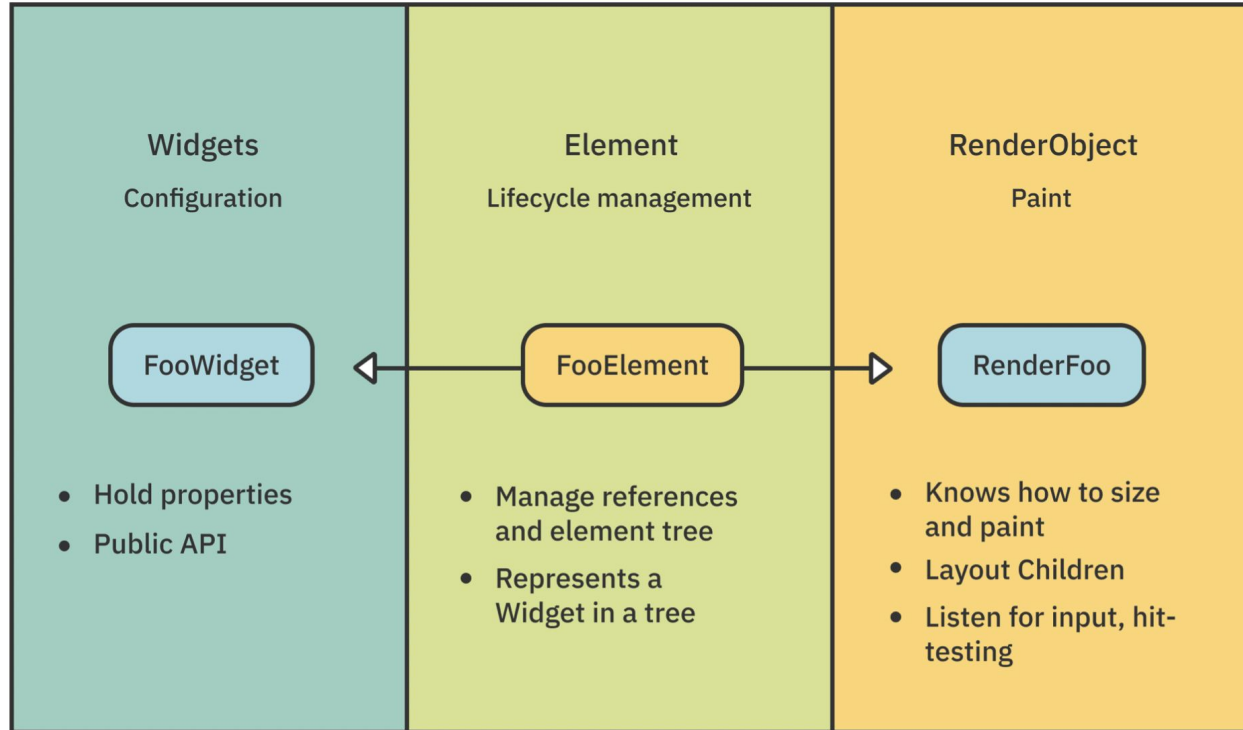


# مقدمة في Flutter

The web version of the architectural layer diagram is as follows:



# مقدمة في Flutter



# مقدمة في Flutter

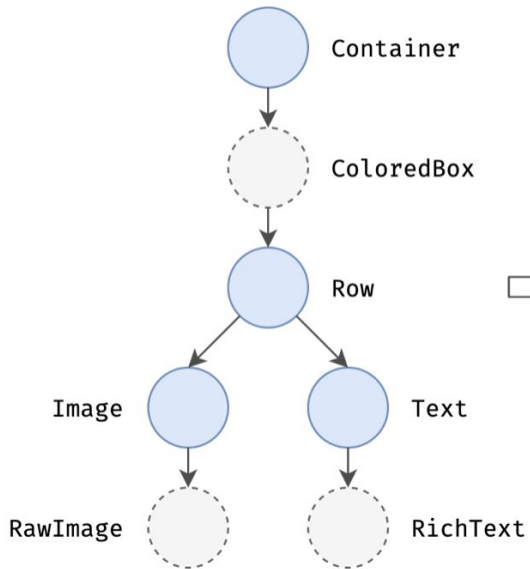
①	User input	Responses to input gestures (keyboard, touchscreen, etc.)	
②	Animation	User interface changes triggered by the tick of a timer	
③	Build	App code that creates widgets on the screen	
④	Layout	Positioning and sizing elements on the screen	RENDERING
⑤	Paint	Converting elements into a visual representation	
⑥	Composition	Overlaying visual elements in draw order	
⑦	Rasterize	Translating output into GPU render instructions	



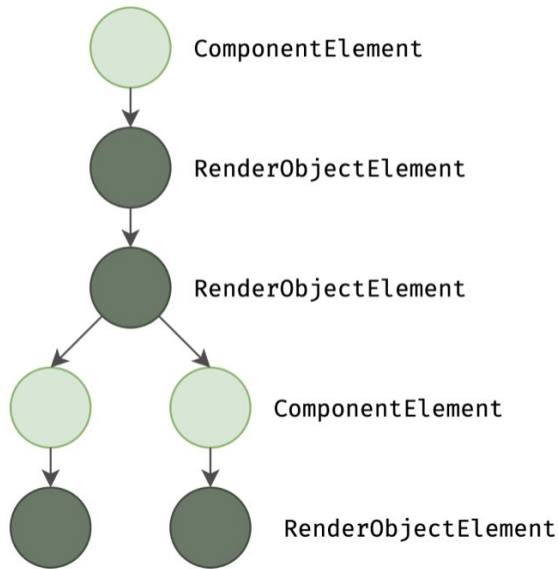


# مقدمة في Flutter

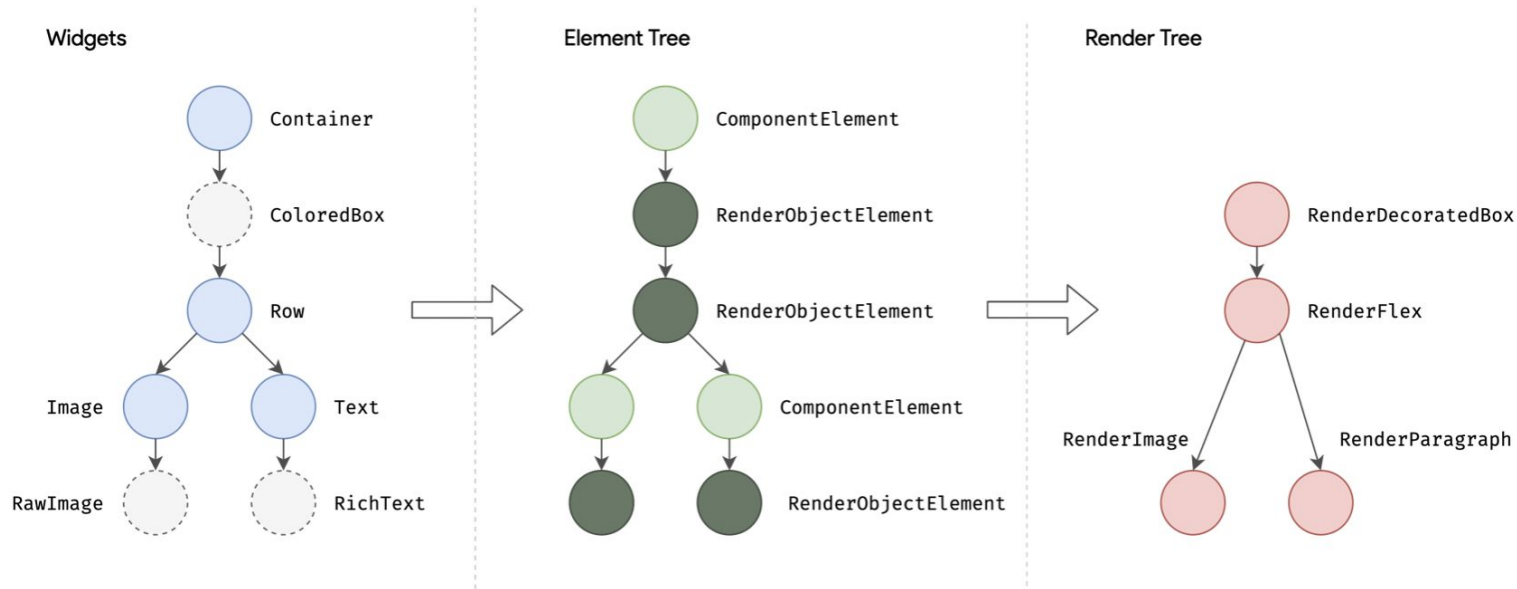
Widgets



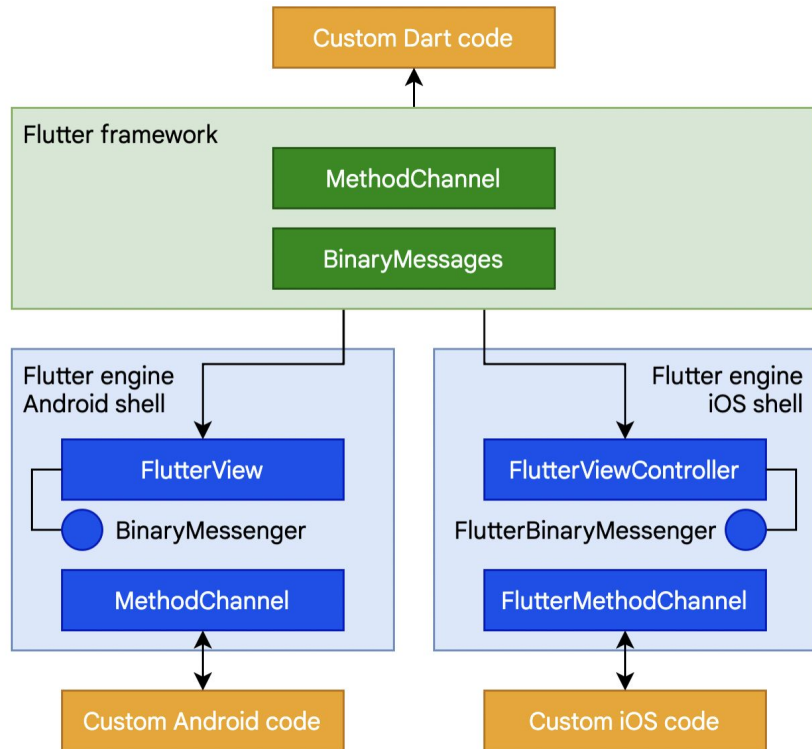
Element Tree



# مقدمة في Flutter

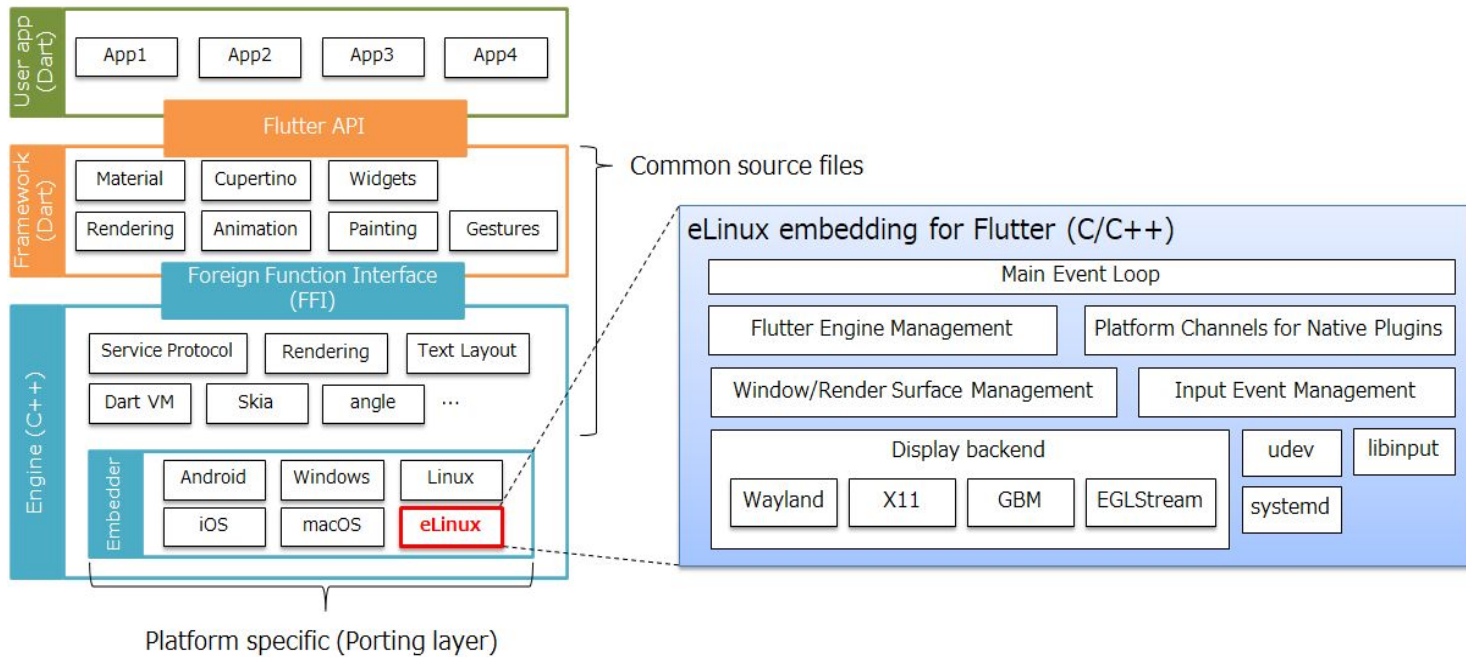


# مقدمة في Flutter



# مقدمة في Flutter

جاري العمل في Flutter الآن في  
أنظمة Embedder



# مقدمة في Flutter

جاري العمل في Flutter الآن في أنظمة  
Embedder

Tested devices

Board / SoC	Vendor	OS / BSP	Backend	Status
Jetson Nano	NVIDIA	JetPack 4.3	Wayland	✓
Jetson Nano	NVIDIA	JetPack 4.3	DRM	✓ (#1)
Raspberry Pi 4 Model B	Raspberry Pi Foundation	Ubuntu 20.10	Wayland	✓
Raspberry Pi 4 Model B	Raspberry Pi Foundation	Ubuntu 20.10	DRM	✓ (#9)
i.MX 8MQuad EVK	NXP	Sumo (kernel 4.14.98)	Wayland	✓
i.MX 8M Mini EVKB	NXP	Zeus (kernel 5.4.70)	Wayland	✓
RB5 Development Kit	Qualcomm	Ubuntu 18.04.05	DRM	✓
Zynq	Xilinx	-	-	Not tested
Desktop (x86_64)	Intel	Ubuntu 20.04	Wayland	✓
Desktop (x86_64)	Intel	Ubuntu 20.04	DRM	✓
Desktop (x86_64)	Intel	Ubuntu 20.04	X11	✓
QEMU (x86_64)	QEMU	AGL (Automotive Grade Linux) koi	Wayland	✓
QEMU (x86_64)	QEMU	AGL (Automotive Grade Linux) koi	DRM	✓

# مقدمة في Flutter

جاري العمل في Flutter الآن في أنظمة Embedder

TOYOTA AGL مثال TIZEN DEVICES

# TIZEN™

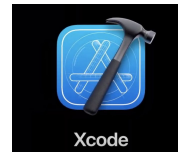


2

# تهيئة بيئة التطوير



# تهيئة بيئة التطوير



مواصفات جهاز جيدة

تحميل Flutter SDK

تحميل برنامج أكواد

ابل (cocoapods) - اندرويد (JDK Java)

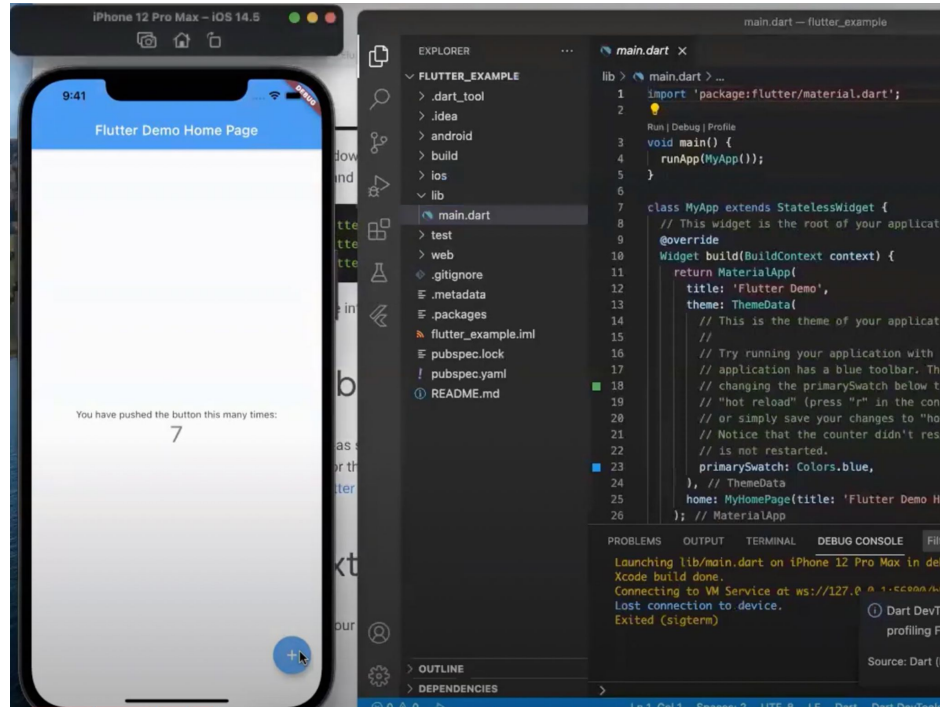




3

# تشغيل أول تطبيق

# تشغيل أول تطبيق





4



# Declarative VS Imperative

# Declarative VS Imperative

$$\text{UI} = f(\text{state})$$

The layout  
on the screen

Your  
build  
methods

The application state

## Enum enhancement

Before: have to use extensions

```
enum Water {
  frozen,
  ...
  boiling;
}

extension Members on Water {
  int waterToTemp(Water water) {
    switch (water) {
      case Water.frozen:
        return 32;
    }
  }
}

String convertToString() =>
  "The $name water is ${waterToTemp(this)} F.";
}

void main() { print(Water.frozen.convertToString()); }
```

Now: directly on enum. Can override.

```
enum Water {
  frozen(32),
  ...
  boiling(212);

  final int tempInFahrenheit;
  const Water(this.tempInFahrenheit);

  @override
  String toString() =>
    "The $name water is $tempInFahrenheit F.";
}

void main() { print(Water.frozen); }
```

5

# التصميم المتجاوب

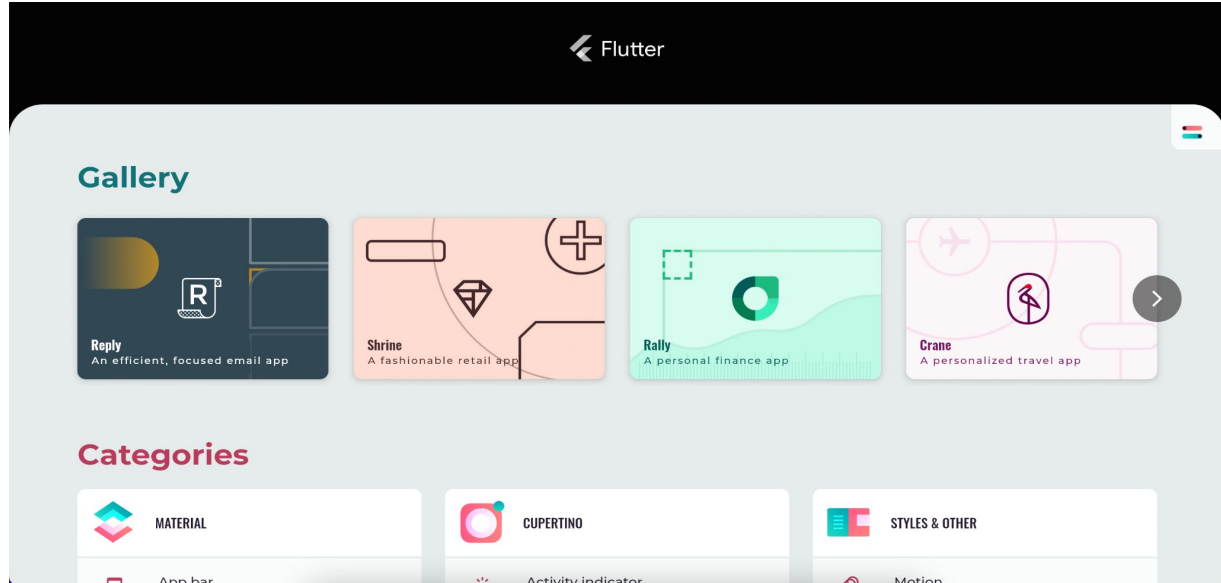
# التصميم المتجاوب

MediaQuery.of(context)

Cupertino (apple)

Material (android) 3

[gallery.flutter.dev](https://gallery.flutter.dev)



6

# Null Safety

# Null Safety

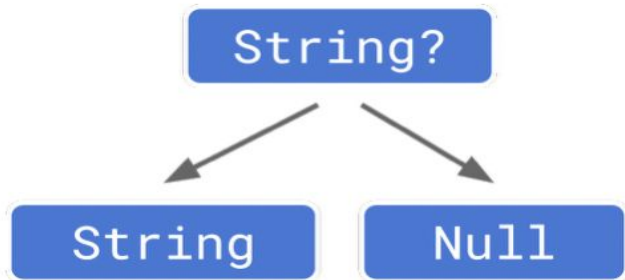
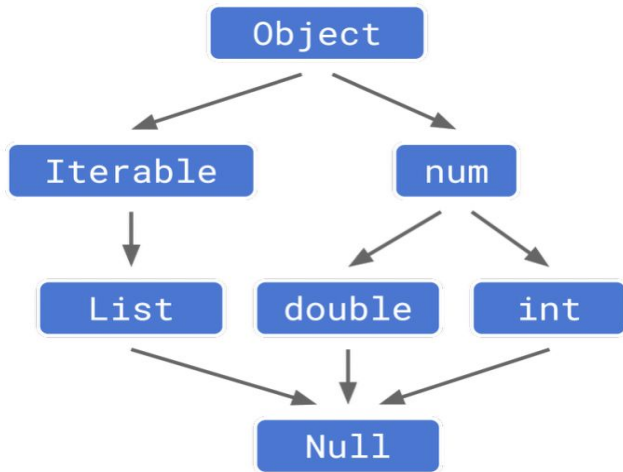
Sound null safety is available in Dart 2.12 and Flutter 2.





# Null Safety

In type theory lingo, the `Null` type was treated as a subtype of all types:



# Null Safety

## Non-nullable types

Object

double

num

int

String

Iterable

bool

List

## Nullable types

Object?

double?

num?

int?

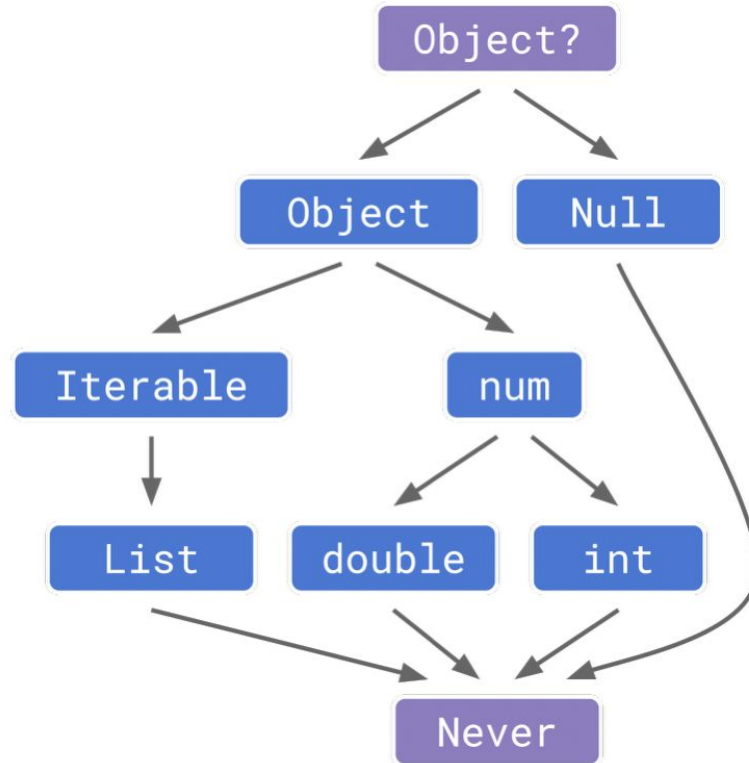
String?

Iterable?

bool?

List?

# Null Safety



# Null Safety

- Types are non-nullable by default and made nullable by adding `?`.
- Optional parameters must be nullable or have a default value. You can use `required` to make named parameters non-optional. Non-nullable top-level variables and static fields must have initializers. Non-nullable instance fields must be initialized before the constructor body begins.
- Method chains after null-aware operators short circuit if the receiver is `null`. There are new null-aware cascade (`?..`) and index (`?[]`) operators. The postfix null assertion “bang” operator (`!`) casts its nullable operand to the underlying non-nullable type.
- Flow analysis lets you safely turn nullable local variables and parameters into usable non-nullable ones. The new flow analysis also has smarter rules for type promotion, missing returns, unreachable code, and variable initialization.
- The `late` modifier lets you use non-nullable types and `final` in places you otherwise might not be able to, at the expense of runtime checking. It also gives you lazy-initialized fields.
- The `List` class is changed to prevent uninitialized elements.





7



# State Management

# State Management

## Provider

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_provider\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_provider_example)

## Riverpod

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_riverpod\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_riverpod_example)

## setState

(StatefulWidget, StatelessWidget, InheritedWidget)

## GetX

## Redux

## Fish - Redux

## BloC / RX

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_bloc\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_bloc_example)

## GetX

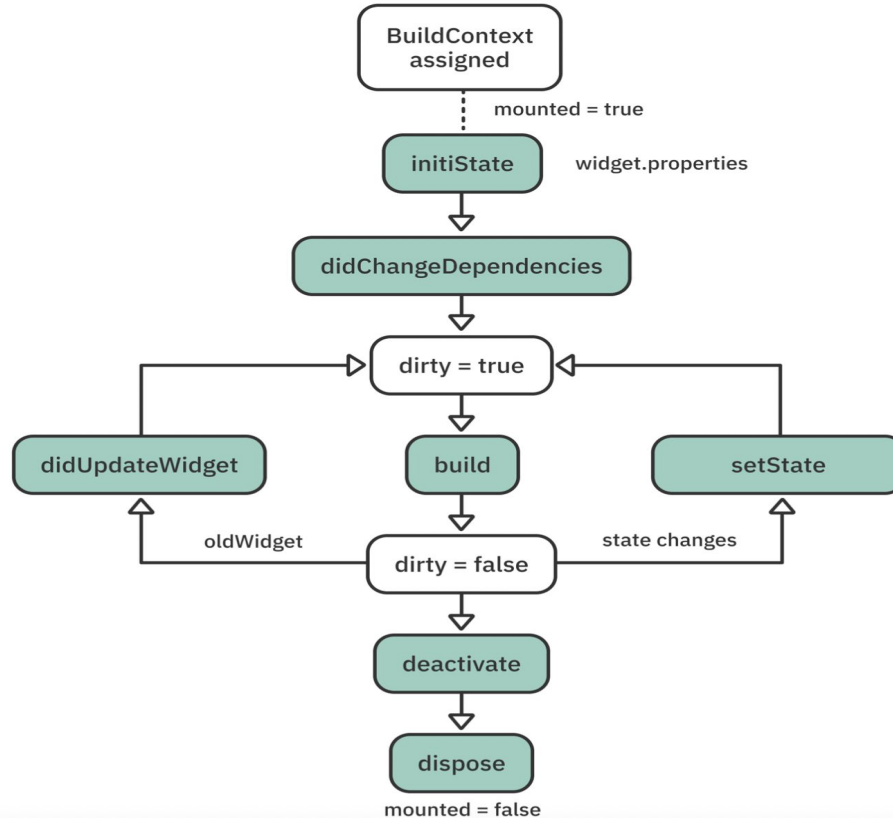
## MobX

## Binder

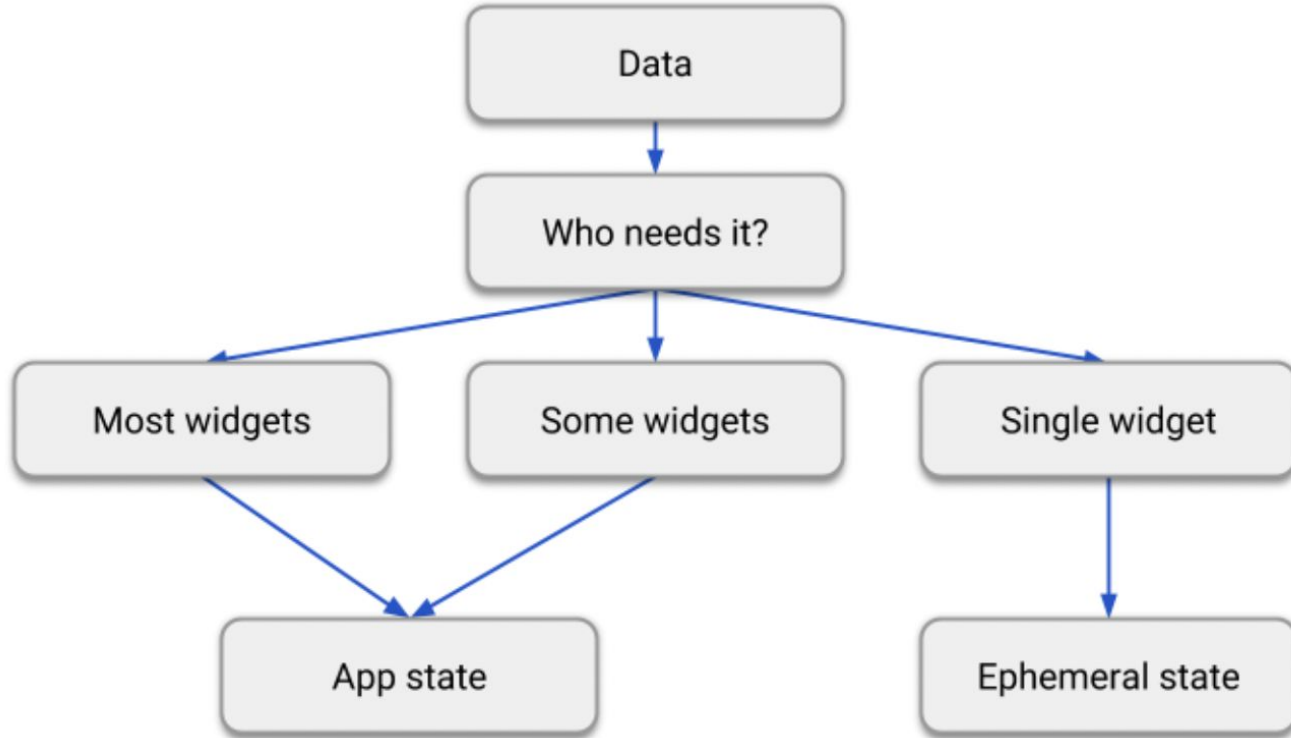


# State Management

## State Object Lifecycle



# State Management





# State Management

## Ephemeral state

Ephemeral state (sometimes called *UI state* or *local state*) is the state you can neatly contain in a single widget.

This is, intentionally, a vague definition, so here are a few examples.

- current page in a `PageView`
- current progress of a complex animation
- current selected tab in a `BottomNavigationBar`



# State Management

## App state

State that is not ephemeral, that you want to share across many parts of your app, and that you want to keep between user sessions, is what we call application state (sometimes also called shared state).

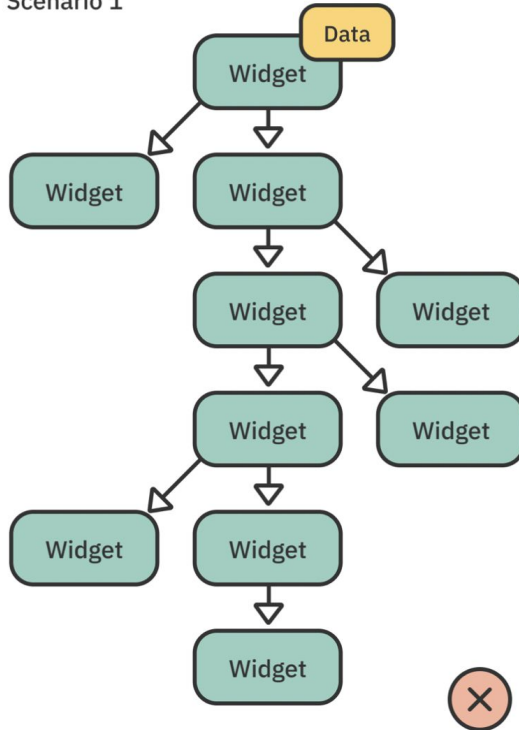
Examples of application state:

- User preferences
- Login info
- Notifications in a social networking app
- The shopping cart in an e-commerce app
- Read/unread state of articles in a news app

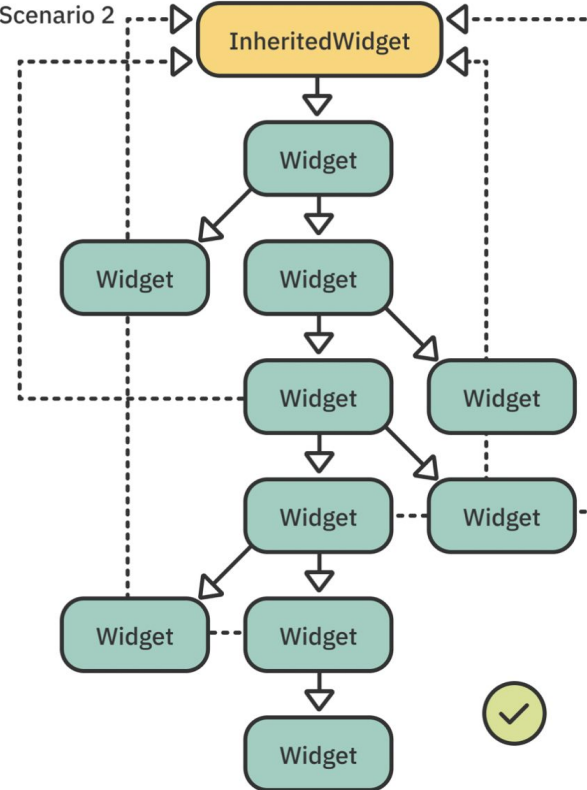


# State Management

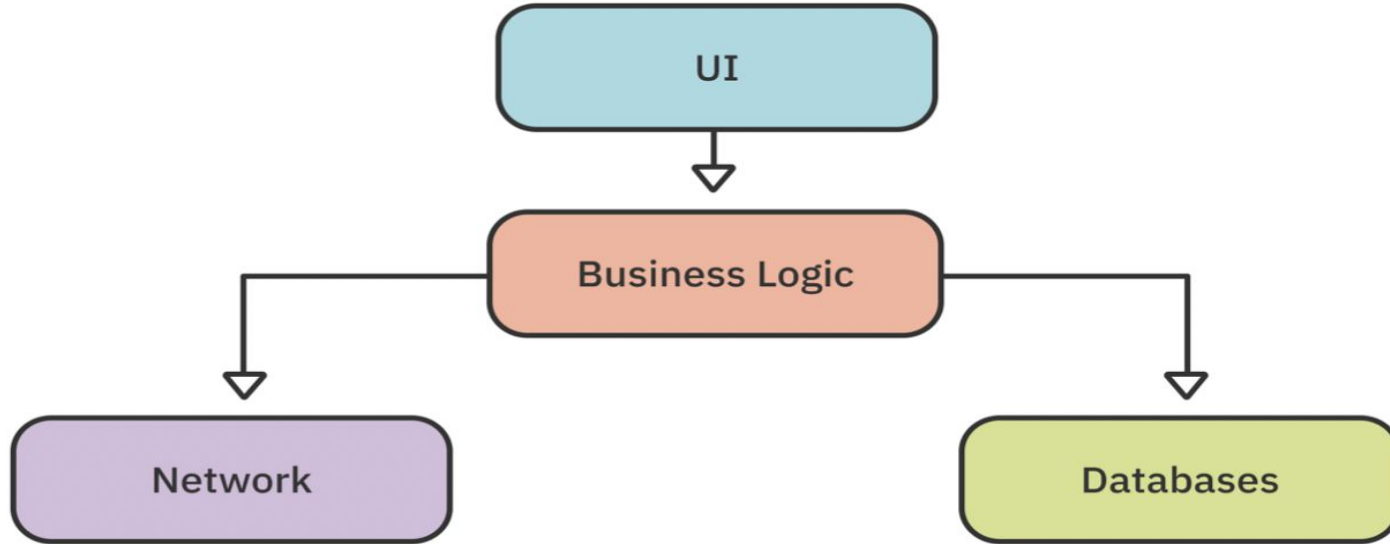
Scenario 1



Scenario 2

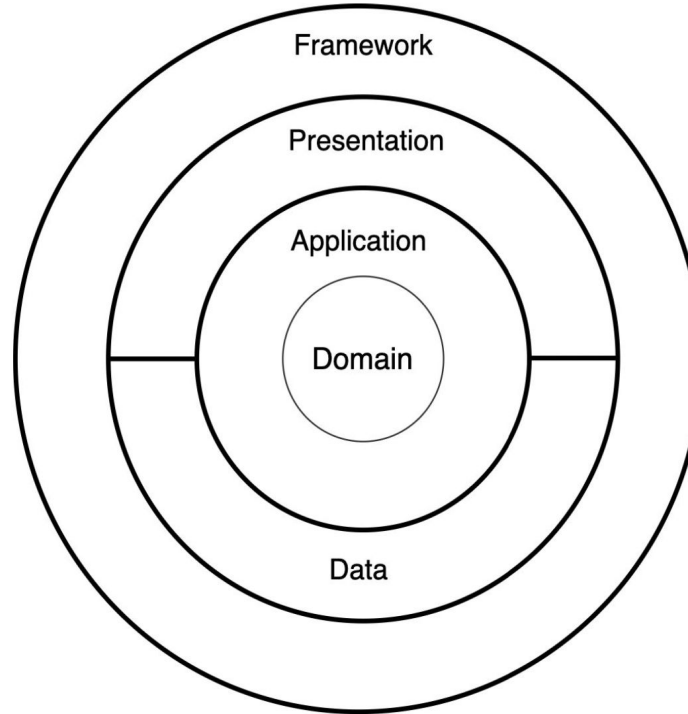


# State Management



# State Management

## Clean Architecture

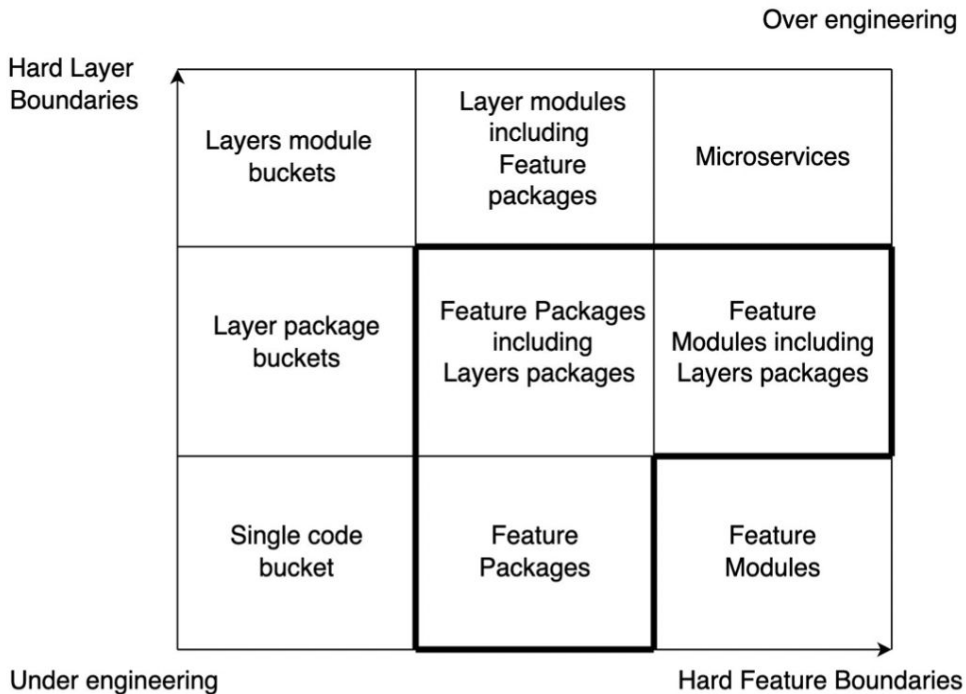


Enterprise Mobile Layering



# State Management

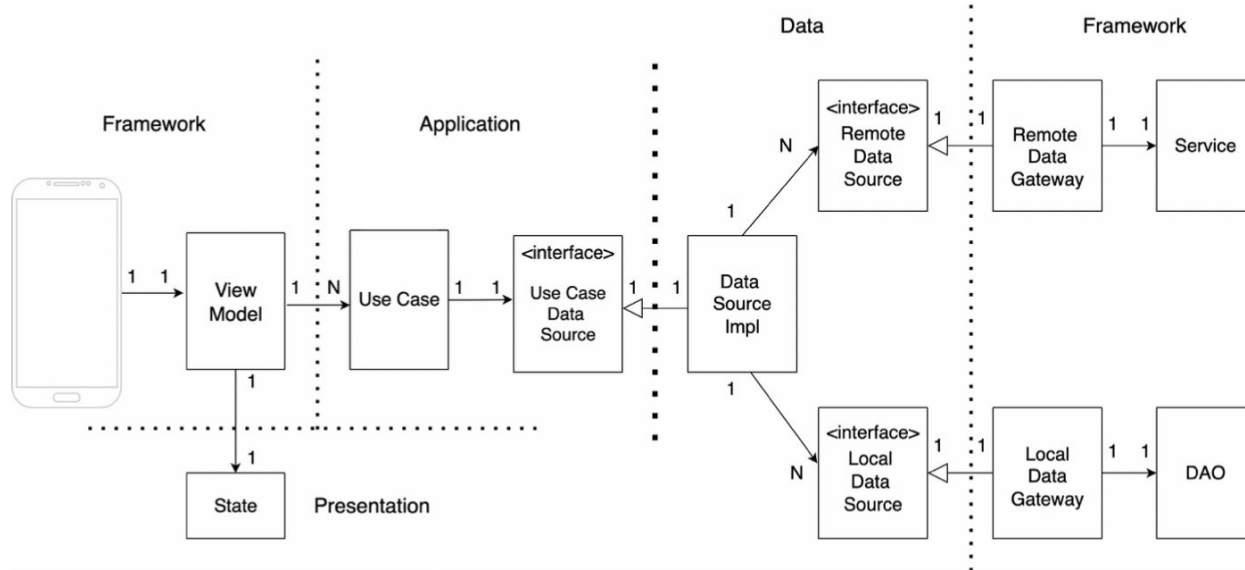
## Clean Architecture



Architecture categorization based on boundaries.

# State Management

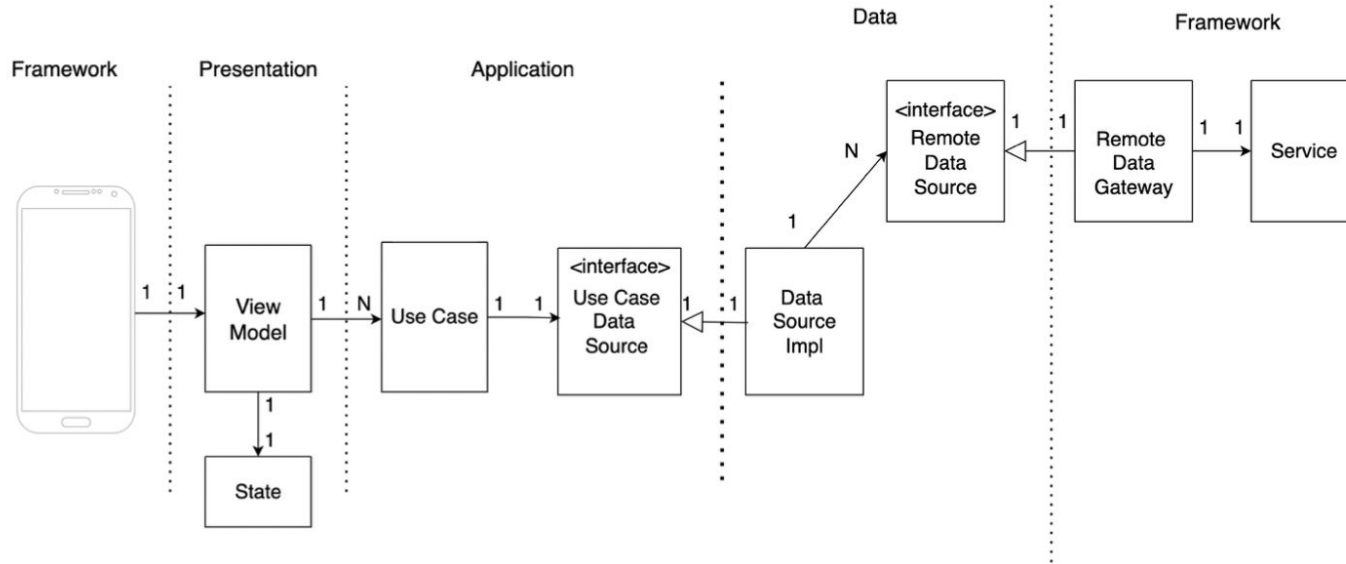
## Clean Architecture



Microservices Architecture.

# State Management

## Clean Architecture

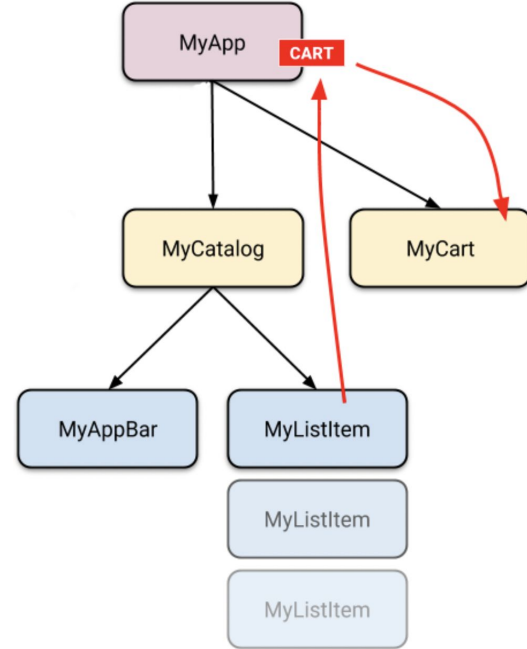
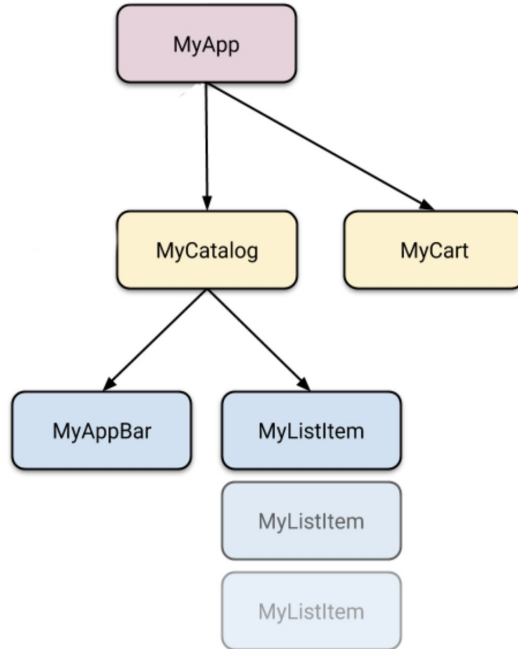


Pragmatic Clean Architecture.



# State Management

## Provider



# State Management

## Provider

```
class CartModel extends ChangeNotifier {  
  /// Internal, private state of the cart.  
  final List<Item> _items = [];  
  
  /// An unmodifiable view of the items in the cart.  
  UnmodifiableListView<Item> get items => UnmodifiableListView(_items);  
  
  /// The current total price of all items (assuming all items cost $42).  
  int get totalPrice => _items.length * 42;  
  
  /// Adds [item] to cart. This and [removeAll] are the only ways to modify the  
  /// cart from the outside.  
  void add(Item item) {  
    _items.add(item);  
    // This call tells the widgets that are listening to this model to rebuild.  
    notifyListeners();  
  }  
  
  /// Removes all items from the cart.  
  void removeAll() {  
    _items.clear();  
    // This call tells the widgets that are listening to this model to rebuild.  
    notifyListeners();  
  }  
}
```

# State Management

## Provider

```
void main() {  
  runApp(  
    ChangeNotifierProvider(  
      create: (context) => CartModel(),  
      child: const MyApp(),  
    ),  
  );  
}
```

```
void main() {  
  runApp(  
    MultiProvider(  
      providers: [  
        ChangeNotifierProvider(create: (context) => CartModel()),  
        Provider(create: (context) => SomeOtherClass()),  
      ],  
      child: const MyApp(),  
    ),  
  );  
}
```



# State Management

## Provider

```
// DON'T DO THIS
return Consumer<CartModel>(
  builder: (context, cart, child) {
    return HumongousWidget(
      // ...
      child: AnotherMonstrousWidget(
        // ...
        child: Text('Total price: ${cart.totalPrice}'),
      ),
    );
  },
);
```



```
// DO THIS
return HumongousWidget(
  // ...
  child: AnotherMonstrousWidget(
    // ...
    child: Consumer<CartModel>(
      builder: (context, cart, child) {
        return Text('Total price: ${cart.totalPrice}');
      },
    ),
  ),
);
```



# State Management

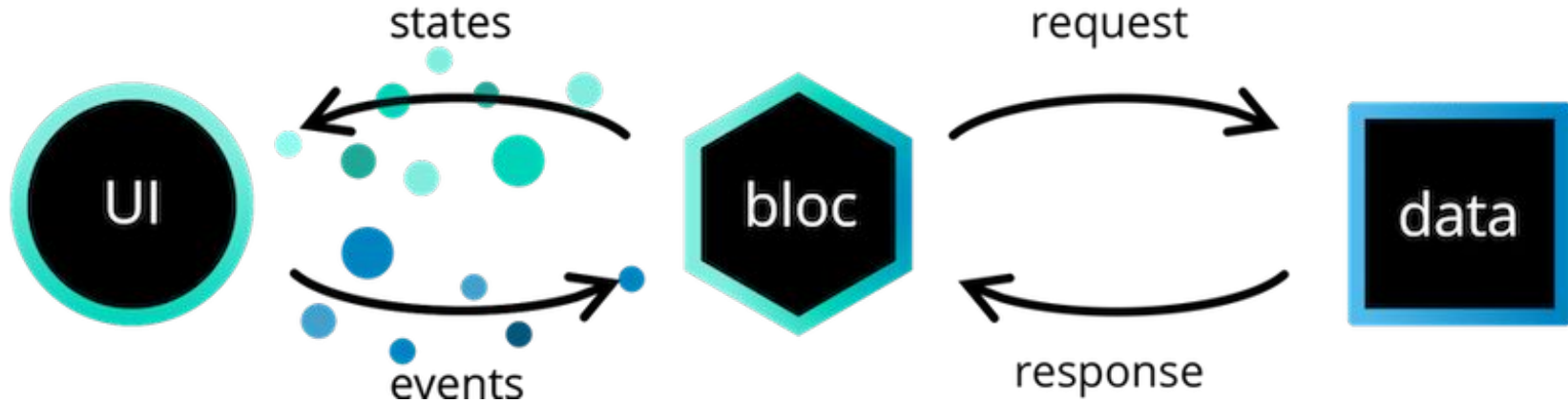
Bloc

**Business logic component**



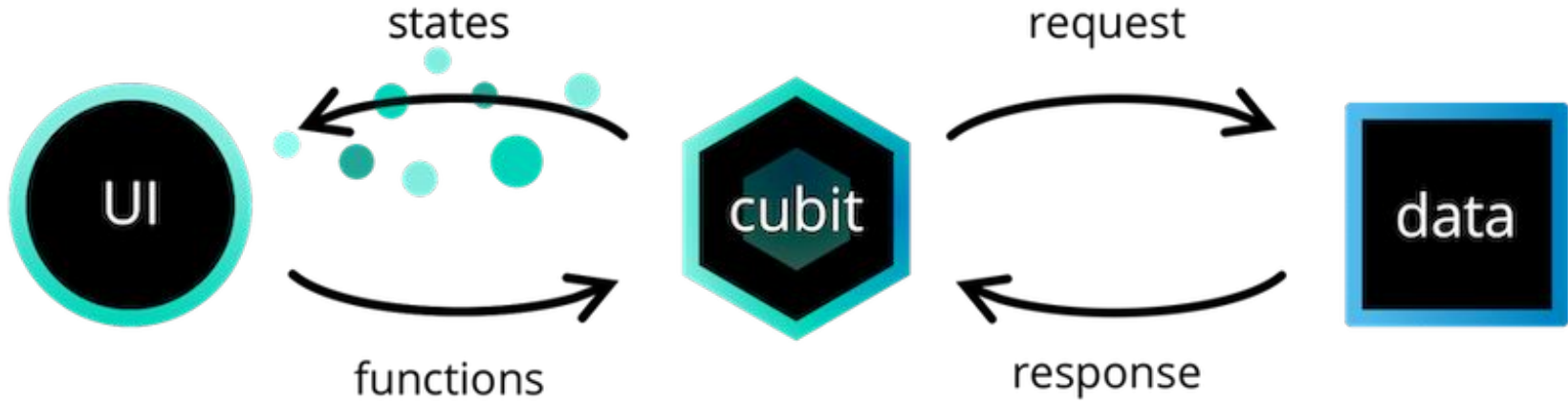
# State Management

## Bloc



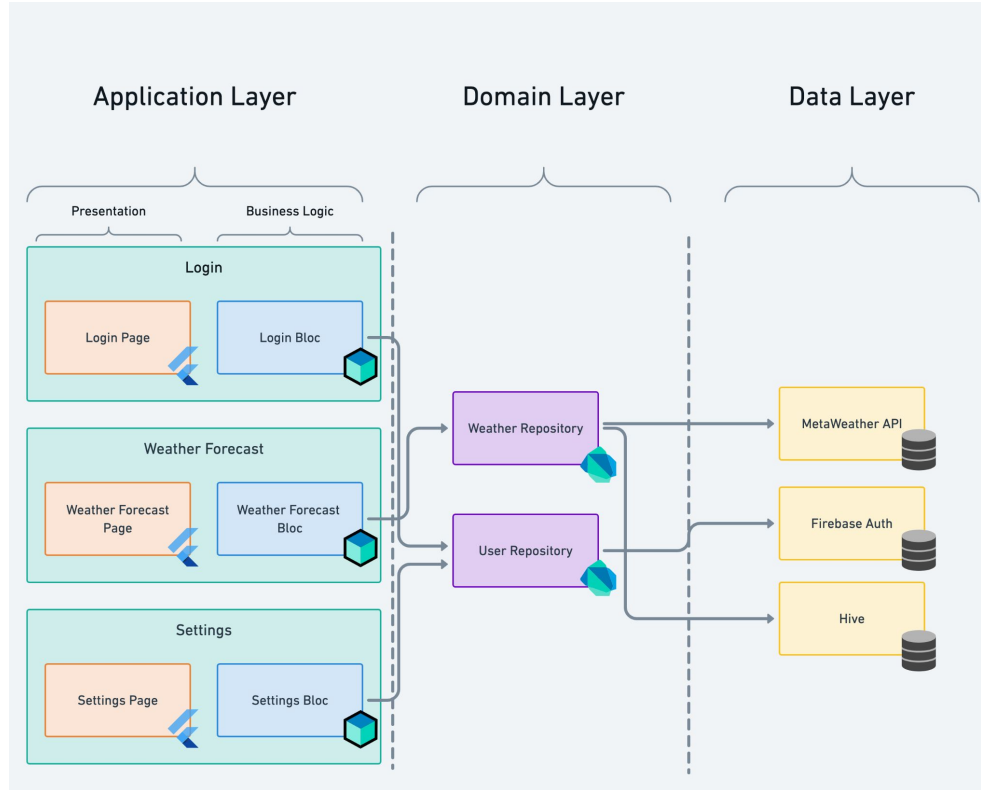
# State Management

Bloc



# State Management

## Bloc





# State Management

## Riverpod

Provider Type	Provider Create Function	Example Use Case
Provider	Returns any type	A service class / computed property (filtered list)
StateProvider	Returns any type	A filter condition / simple state object
FutureProvider	Returns a Future of any type	A result from an API call
StreamProvider	Returns a Stream of any type	A stream of results from an API
StateNotifierProvider	Returns a subclass of StateNotifier	A complex state object that is immutable except through an interface
ChangeNotifierProvider	Returns a subclass of ChangeNotifier	A complex state object that requires mutability



# State Management

## Riverpod

The `.family` modifier has one purpose: Getting a unique provider based on external parameters.

Some common use-cases for `family` would be:

- Combining `FutureProvider` with `.family` to fetch a `Message` from its ID
- Passing the current `Locale` to a provider, so that we can handle translations



# State Management

## Riverpod

### **.autoDispose**

A common use case is to destroy the state of a provider when it is no-longer used.

There are multiple reasons for doing so, such as:

- When using Firebase, to close the connection and avoid unnecessary cost.
- To reset the state when the user leaves a screen and re-enters it.

Providers come with built-in support for this use case, through the `.autoDispose` modifier.

# State Management

## Riverpod

- **`ref.read()`**

As the name implies, we read a notifier and or its value and not trigger for changes. This is exactly the same thing we do when we use the Provider counterpart of `context.read()` or `Provider.of(context, listen: false)`. This is very important as it gives us the possibility of accessing the provider and make changes outside of the build context that is, in the `onTap()` etc. Never call the `ref.read()` inside build context as much as it does not trigger rebuild of the widget when changes occur and could be seen as a way of preserving and reducing the number of useless UI rebuilds, it is highly contested against as the official docs for riverpod will say



# State Management

## Riverpod

- `ref.watch()`

This is probably the most versatile of them all. It is used to get the value of a provider and also watch for changes. It is typically used only inside the build method or inside another provider. Never use it inside any of the lifecycle methods like `initstate` or `setstate` as there will be deadlock and conflict since both may be triggering a rebuild of the widget and still depending on each other



# State Management

## Riverpod

- `ref.listen()`

This is a very useful one as it is similar in every way to `ref.watch`, but that it does not trigger a rebuild of the widget but instead it watches the provider and when a change occurs you can call a function. This is useful perhaps if for navigation etc.



# State Management

## Riverpod

- `provider.select()`

The other day I was having a conversation with a friend who brought to my attention from logs that riverpod with `ChangeNotifier` rebuilds a few more times than provider and bloc and more rebuilds when used an immutable state like with `statenotifier`. This is understandable as few numbers of rebuilds is not a stronghold that riverpod is proud of, but the solution it provides is geared at different difficulties. But if you are concerned about the number of rebuilds you can mitigate this with the select method. This works in that, in a state or class with more than one property, you can decide to listen to changes only from one or a few properties (the provider listens to the whole object by default).



# State Management

## Riverpod

```
class BlueSquare extends ConsumerWidget {  
  const BlueSquare({  
    Key? key,  
  }) : super(key: key);  
  
  @override  
  Widget build(BuildContext context, WidgetRef ref) {  
    // But if you want only the size and dont want unnecessary rebuild when  
    // color changes  
    final size =  
      ref.watch(blueSquareProvider.select((blueSquare) => blueSquare.size));  
  
    return Container(  
      width: size,  
      height: size,  
      color: Colors.blue,  
    );  
  }  
}
```





8

# المفاتيح

# المفاتيح

When? Adding - Removing - Reordering (collection of widgets of the same type of some state)

GlobalKey

LocalKey (

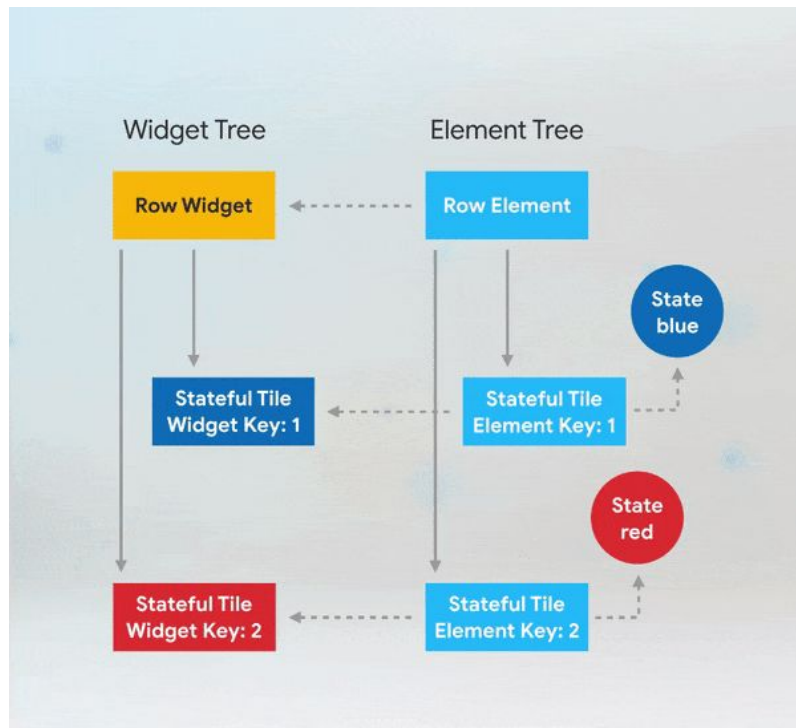
ValueKey('Mouaz Shahmeh') , ObjectKey(1, 2, 3, 4), UniqueKey(), PageStorageKey(scrollLocation )

Examples: ScaffoldKey and FormKey

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_keys\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_keys_example)



## المفاتيح





9

التنقل

GoRouter (push - go) (Declarative navigation - no context)

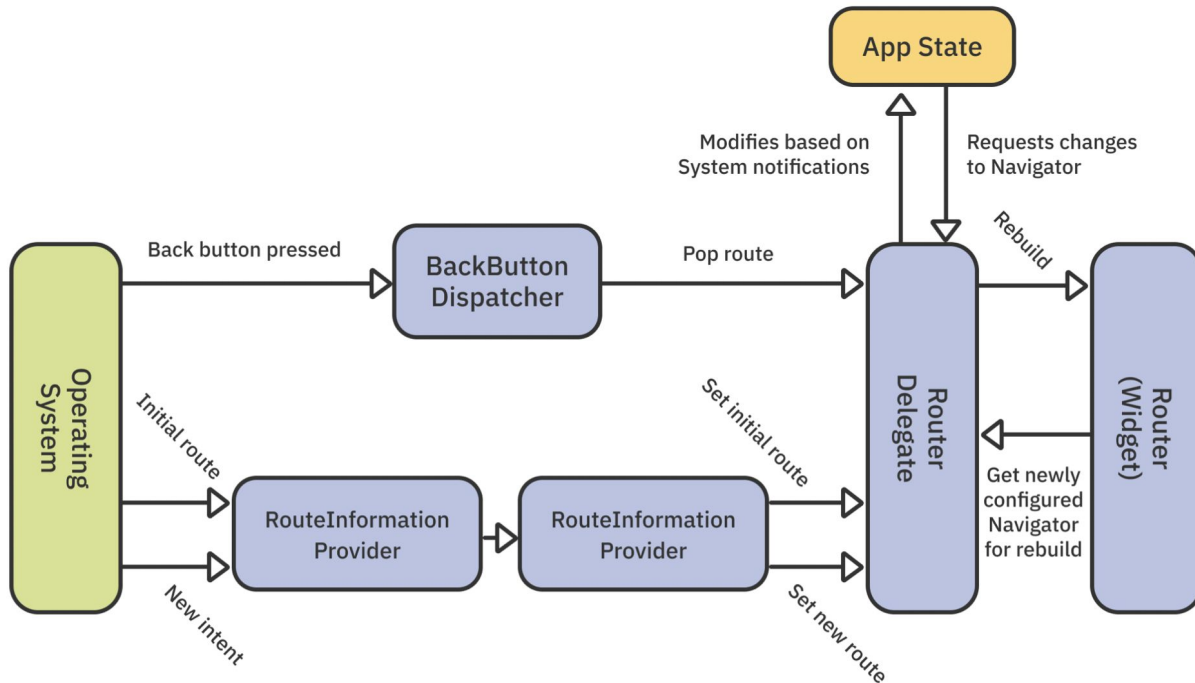
[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_gorouter\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_gorouter_example)

Navigator 2 (push - pop - pushReplacement - pushNamed - popUntil ...)

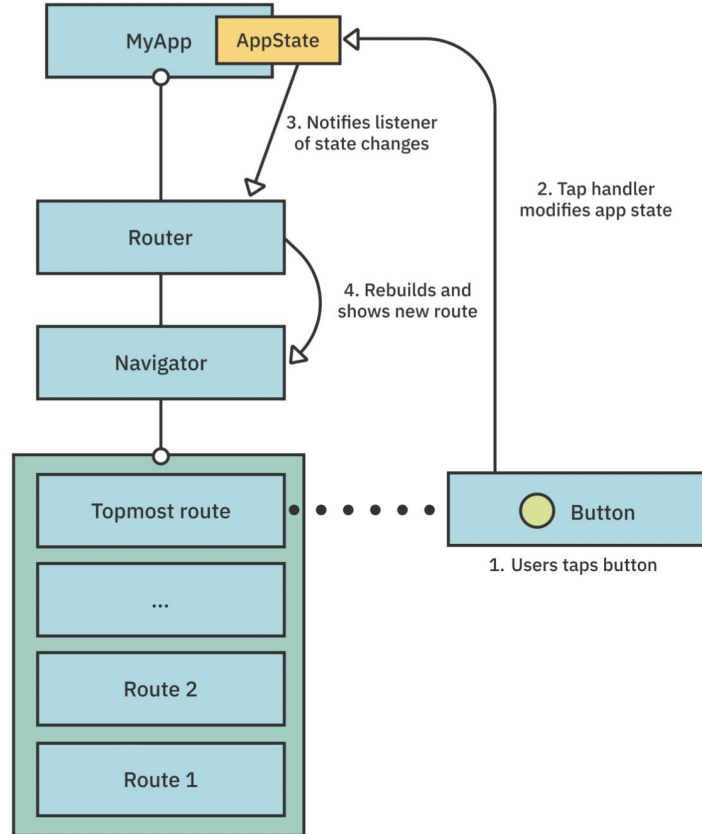
- <https://pub.dev/packages/beamer>
- [https://pub.dev/packages/flow\\_builder](https://pub.dev/packages/flow_builder)
- <https://pub.dev/packages/fluro>
- <https://pub.dev/packages/vrouter>
- [https://pub.dev/packages/auto\\_route](https://pub.dev/packages/auto_route)



Here are the new abstractions that make up Navigator 2.0's declarative API:



## التنقل



10

# خيارات قواعد البيانات



# خيارات قواعد البيانات



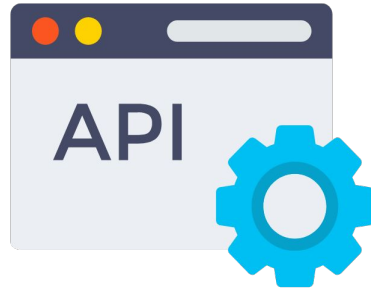
Firestore



supabase



SQLite



Shared  
Preferences



# خيارات قواعد البيانات

Firestore (FlutterFire) - BAAS (Backend As A Service)

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_firestore\\_crud\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_firestore_crud_example)

Supabase (PostgreSQL) - BAAS (Backend As A Service)

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_supabase\\_crud\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_supabase_crud_example)

Sqflite - SqfEntity (local)

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_sqflite\\_crud\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_sqflite_crud_example)

Shared Preferences (path)

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_shared\\_preferences\\_crud\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_shared_preferences_crud_example)

API http (json)

[https://github.com/Eng-Mouaz-M-AlShahmeh/flutter\\_development\\_api\\_http\\_crud\\_example](https://github.com/Eng-Mouaz-M-AlShahmeh/flutter_development_api_http_crud_example)

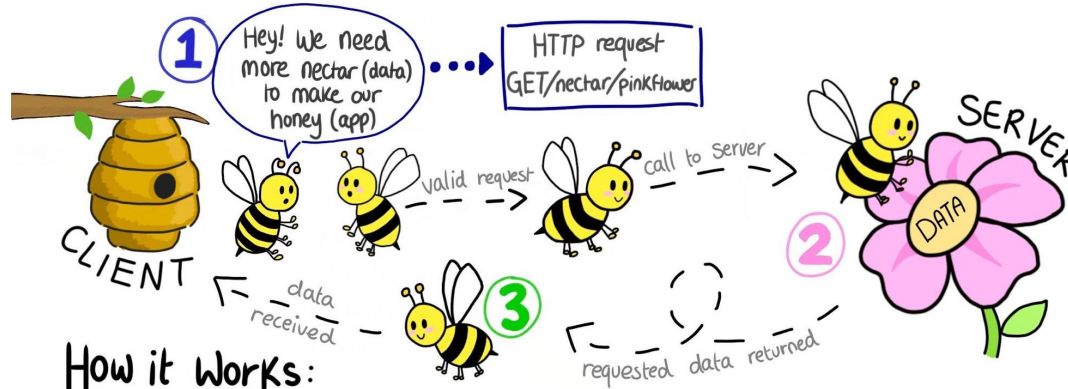


# خيارات قواعد البيانات

## What is an API?

@Rapid\_API 

An application programming interface allows two programs to communicate. On the web, APIs sit between an application and a Web Server, and facilitate the transfer of data.



How it Works:

**1 Request**

API call is initiated by the Client application via a HTTP request

**2 Receive**

Our Worker bee acts as an API, going to a Flower (server) to collect nectar (data)

**3 Response**

The API transfers the requested data back to the requesting application, usually in JSON format

# خيارات قواعد البيانات

DIFFERENT TYPES OF API				
	REST	GRAPHQL	SOAP	RPC
STRUCTURE	FOLLOWS SIX ARCHITECTURAL CONSTRAINTS	SCHEMA AND TYPE	MESSAGE STRUCTURE	LOCAL PROCEDURAL CALLS
FORMAT	JSON, XML, HTML, PLAIN TEXT	JSON	XML	JSON, XML, FLATBUFFERS, ETC
ADVANTAGES	FLEXIBLE IN TERMS OF DATA FORMAT AND STRUCTURE	SOLVES OVER-FETCHING AND UNDER-FETCHING	HIGHLY SECURE AND EXTENSIBLE	LIGHTWEIGHT PAYLOADS MAKE IT HIGH PERFORMING
USE CASES	RESOURCES BASED APPS	MOBILE APIS	PAYMENT GATEWAYS	COMMAND-FOCUSED SYSTEMS

RapidAPI.com/hub  
@Rapid\_API



# خيارات قواعد البيانات



Swagger™

Supported by SMARTBEAR



POSTMAN





11

# اختبارات الأداء

# اختبارات الأداء

Dev Tools

Flutter Doctor

Flutter Debug Mode (Code Trace)

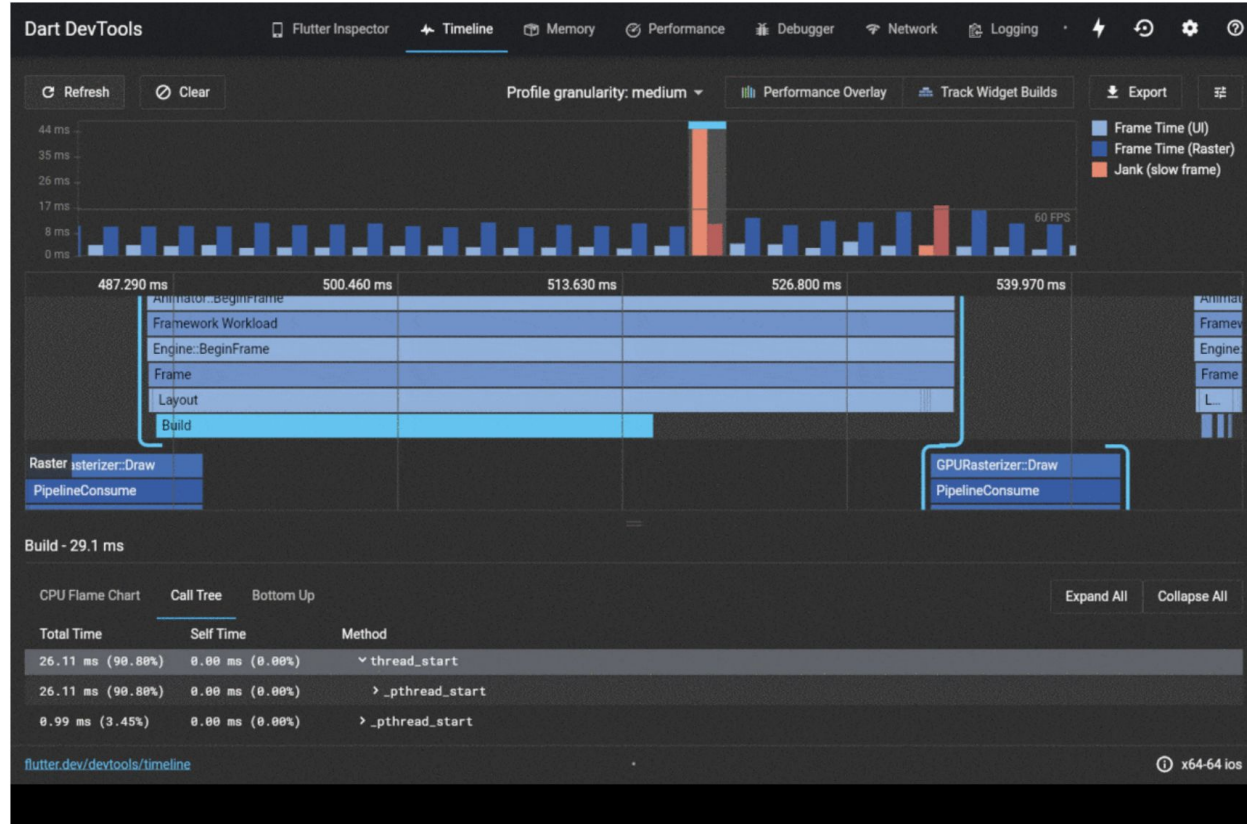


# اختبارات الأداء

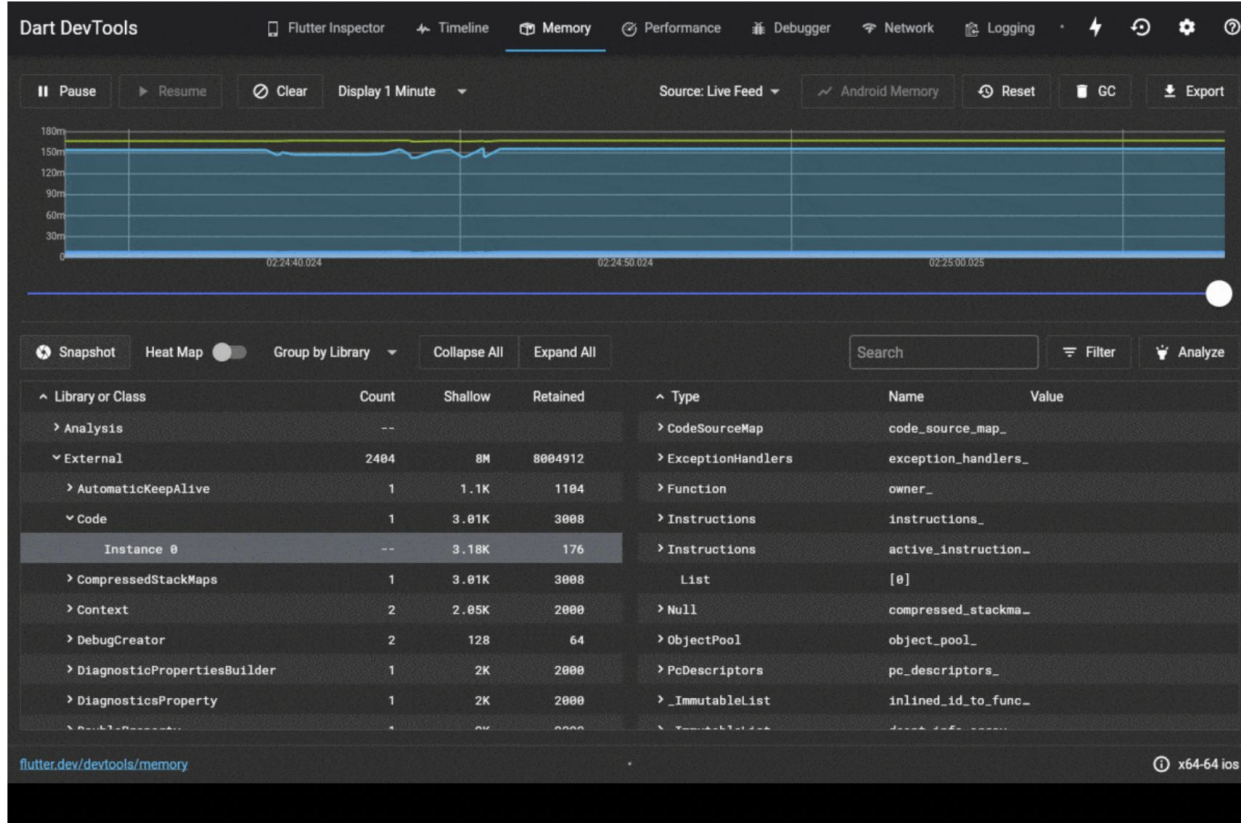




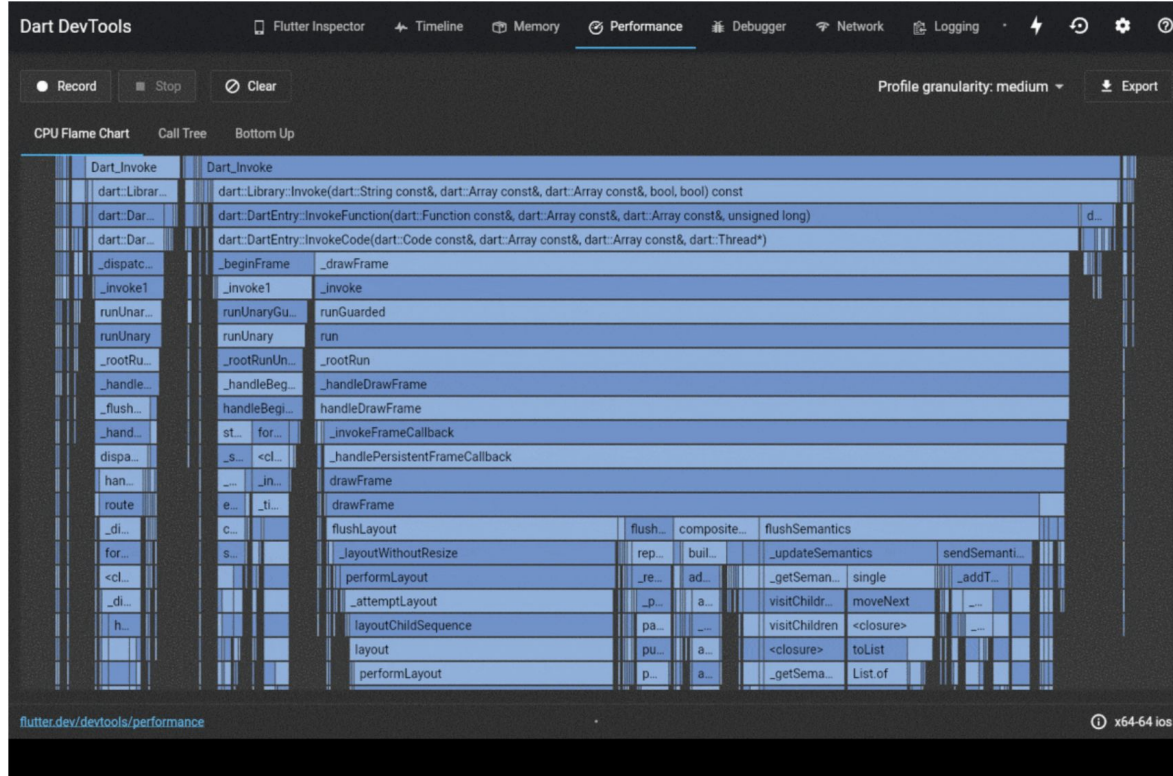
## اختبارات الأداء



## اختبارات الأداء



## اختبارات الأداء



# اختبارات الأداء

Dart DevTools

Flutter Inspector Timeline Memory Performance Debugger Network Logging Show structured errors

Clear Filter

When	Kind	Message
14:29:45.098	flutter.frame	#219 37.5ms
14:29:45.098	flutter.frame	#220 31.9ms
14:29:45.098	flutter.frame	#221 24.8ms
14:29:45.098	flutter.frame	#222 18.8ms
14:29:45.098	flutter.frame	#223 29.5ms
14:29:45.098	flutter.frame	#224 23.9ms
14:29:45.966	stdout	flutter: Regular print() message.
14:29:48.965	my_network_log	Sending request to example.com.
14:29:49.023	my_network_log	Response from example.com received (HTTP 200).
14:29:54.965	stdout	flutter: Regular print() message.
14:29:56.920	flutter.frame	#225 8.1ms
14:29:57.097	flutter.frame	#226 12.5ms
14:29:57.097	flutter.frame	#227 20.9ms
14:29:57.097	flutter.frame	#228 29.5ms
14:29:57.097	flutter.frame	#229 21.7ms

Details

```
{
  "number": 226,
  "startTime": 241725659429,
  "elapsed": 12521,
  "build": 4071,
  "raster": 8351
}
```

flutter.dev/devtools/logging 96 events x64-64 ios



Dart DevTools

Flutter Inspector Timeline Memory Performance Debugger Network Logging

Record HTTP traffic Stop Clear

Method	Request URI	Status	Duration	Timestamp
GET	http://www.example.com/	200	50 ms	2:30:09.005 PM
GET	http://www.example.com/	200	56 ms	2:30:19.006 PM
GET	http://www.example.com/	200	65 ms	2:30:29.006 PM
GET	http://www.example.com/	200	54 ms	2:30:39.003 PM
GET	http://www.example.com/	200	47 ms	2:30:49.006 PM
GET	http://www.example.com/	200	53 ms	2:30:59.007 PM
GET	http://www.example.com/	200	57 ms	2:31:09.007 PM
GET	http://www.example.com/	200	49 ms	2:31:19.004 PM
GET	http://www.example.com/	200	60 ms	2:31:29.003 PM
GET	http://www.example.com/	200	47 ms	2:31:39.004 PM
GET	http://www.example.com/	200	52 ms	2:31:49.008 PM
GET	http://www.example.com/	200	52 ms	2:31:59.005 PM
GET	http://www.example.com/	200	52 ms	2:32:09.006 PM
GET	http://www.example.com/	200	54 ms	2:32:19.006 PM
GET	http://www.example.com/	200	52 ms	2:32:29.003 PM
GET	http://www.example.com/	200	48 ms	2:32:39.005 PM
GET	http://www.example.com/	200	52 ms	2:32:49.005 PM
GET	http://www.example.com/	200	57 ms	2:32:59.006 PM
GET	http://www.example.com/	200	60 ms	2:33:09.005 PM
GET	http://www.example.com/	200	55 ms	2:33:19.005 PM
GET	http://www.example.com/	200	78 ms	2:33:29.006 PM

Headers Timing

General

method: GET

uri: http://www.example.com/

isolateId: isolates/76796289279471

isolateGroupId: isolateGroups/16860335164400473611

compressionState: HttpClientResponseCompressionState.decompressed

connectionInfo: (localPort: 53877, remoteAddress: 2606:2800:220:1:248:1893:25c8:1946, remotePort: 80)

contentLength: 648

cookies: []

isRedirect: false

persistentConnection: true

reasonPhrase: OK

redirects: []

statusCode: 200

Response Headers

last-modified: [Thu, 17 Oct 2019 07:18:26 GMT]

cache-control: [max-age=604800]

date: [Wed, 24 Jun 2020 21:30:18 GMT]

vary: [Accept-Encoding]

DevTools 0.8.0-dev.1 25 requests x64-64 ios

Dart DevTools

Flutter Inspector Timeline Memory Performance Debugger Network Logging

Call Stack

- main.<closure>() main.dart 25
- \_rootRunUnary() zone.dart 1198
- \_rootRunUnary() zone.dart
- \_CustomZone.runUnary() zone.dart 1100
- \_CustomZone.runUnaryGuarded() zone.dart 1005
- \_CustomZone.bindUnaryCallbackGuarded.<clos...
- \_rootRunUnary() zone.dart 1206
- \_rootRunUnary() zone.dart
- CustomZone.runUnary() zone.dart 1100

Variables

- client: \_HttpClient
  - \_closing: false
  - \_closingForcefully: false
  - \_connectionTargets: \_HashMap
    - \_elementCount: 1
    - \_buckets: \_List (8 items)
    - \_modificationCount: 1
  - \_credentials: \_GrowableList (0 items)
  - \_proxyCredentials: GrowableList (0 items)

Breakpoints 3

- main.dart:25 (package:performance\_test\_app/...
- main.dart:30 (package:performance\_test\_app/...
- main.dart:38 (package:performance\_test\_app/...

package:performance\_test\_app/main.dart

```
14 import 'package:performance_test_app/src/lime.dart';
15 import 'package:performance_test_app/src/orange.dart';
16 import 'package:performance_test_app/src/pink.dart';
17 import 'package:performance_test_app/src/welcome.dart';
18 import 'package:performance_test_app/src/yellow.dart';
19
20 void main() {
21   // Just something to have in the networking tab and logging tab.
22   HttpClient client = HttpClient();
23   Timer.periodic(const Duration(seconds: 10), (_) async {
24     const networkLoggerName = 'my_network_log';
25     developer.log('Sending request to example.com.', name: _
26     var request = await client.getUrl(
27       Uri.parse('http://www.example.com/'),
28     );
29     var response = await request.close();
30     developer.log(
31       'Response from example.com received (HTTP $_
32       name: networkLoggerName,
33       error: '{"contentLength": ${response.contentLength}}',
34     );
35   });
36 }
```

Libraries 802

Filter (IP)

- dart:async
- dart:async-patch/async\_patch.dart
- dart:async-patch/\_
- dart:async-patch/\_
- dart:async-patch/timer\_patch.dart
- dart:async/async\_error.dart
- dart:async/\_
- dart:async/deferred\_load.dart
- dart:async/future.dart
- dart:async/future\_impl.dart
- dart:async/schedule\_microtask.dart
- dart:async/stream.dart
- dart:async/stream\_controller.dart
- dart:async/stream\_impl.dart
- dart:async/stream\_nine.dart

Console

```
flutter: Regular print() message.
flutter: Regular print() message.
flutter: Regular print() message.
flutter: Regular print() message.
flutter: Regular print() message.
flutter: Regular print() message.
flutter: Regular print() message.
```

flutter.dev/devtools/debugger isolate main #1 paused at main.dart 25:5 x64-64 ios

# أنماط بناء التطبيقات

# أنماط بناء التطبيقات

**Debug** (print - debugShowCheckedModeBanner - run - hot reload & restart - debug mode)

**Profile** (run - real performance - real device - hot reload & restart)

**Release** (archive in xcode - apk or bundle in android - app signing - unique id - play store - app store)

تطبيق عملي





# المشروع

# تفاصيل المشروع

- عمل مشروع Flutter جديد باستخدام برنامج الأكواد واخر اصدار لـ Flutter SDK بدعم Null Safety.
- استخدام MediaQuery.of(context) في شاشات المشروع لدعم مقاسات الشاشات.
- استخدام واحدة من ادارة الحالة في التطبيق (provider - riverpod - bloc)
- استخدام GoRouter للتنقل بين الشاشات في التطبيق
- استخدام واحدة من خيارات قواعد البيانات لعمل عملية أو أكثر من عمليات CRUD (Firebase - Supabase - Sqfite or Sqfentity - api http - shared preferences)
- رفع المشروع في ريبو Github جديد.
- تصوير 3 شاشات من التطبيق على الأقل
- إرسال إيميل بعنوان "دورة تطوير التطبيقات باستخدام Flutter - الاسم الثلاثي" الى [m.m.shahmeh@gmail.com](mailto:m.m.shahmeh@gmail.com)
- إرفاق رابط الـ github ضمن الإيميل
- إرفاق صور التطبيق بشكل ملف مضغوط (rar - zip)



13

# ملخص ومراجع

# ملخص ومراجع

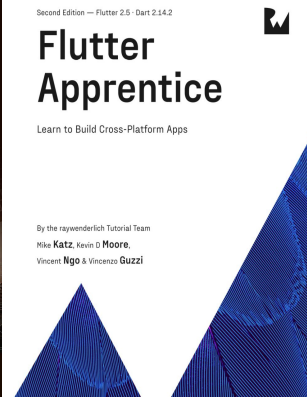
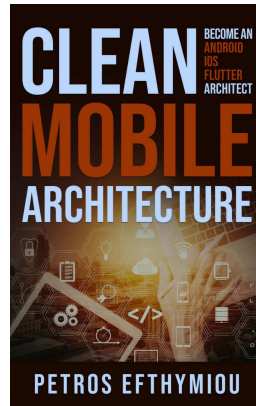
[flutter.dev](https://flutter.dev)

[The Boring Flutter Development Show \(Youtube\)](#)

[Widget of the Week \(Youtube\)](#)

[pub.dev](https://pub.dev)

[fuchsia.dev](#) (Zircon is Dart package)



## Ask Me Anything

m.m.shahmeh@gmail.com

g.dev/mouaz\_m\_shahmeh



### LINKS


 [github.com/Eng-Mouaz-M-AlShahmeh](https://github.com/Eng-Mouaz-M-AlShahmeh)


 [linkedin.com/in/mouaz-shahmeh](https://linkedin.com/in/mouaz-shahmeh)


 [stackoverflow.com/users/18449528](https://stackoverflow.com/users/18449528)

 [twitter.com/mouaz\\_m\\_shahmeh](https://twitter.com/mouaz_m_shahmeh)





**Mouaz M. Shahmeh**  
Developer at AlRaedah  
He/Him  
[g.dev/mouaz\\_m\\_shahmeh](https://g.dev/mouaz_m_shahmeh) 

 Public

**CITY/TOWN**  
 Riyadh Saudi Arabia

**BIO**  
Software Engineer and Developer.  
General Director Developer of real-estate  
Tadawl App. Founder of TQDR Pay system.

**STATS**  
 **62** • Badges earned  
 **9** • Pages saved



شكراً لكم