



تطوير التطبيقات باستخدام Flutter

Eng Mouaz M. Al-Shahmeh
BSc / ITC - Software Engineer & Developer





أهداف اللقاء

٢

- مقدمة في Flutter
- تهيئة بيئة التطوير
- تشغيل أول تطبيق Flutter
- Declarative UI
- التصميم المتجاوب



أهداف اللقاء

3

- Null Safety •
- State Management •
- المفاتيح •
- التنقل •
- خيارات قواعد البيانات •



أهداف اللقاء

4

- اختبارات الأداء
- أنماط بناء التطبيقات
- ملخص ومراجعة



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

6

- Hot Reload - Hot Restart
- One Code Base (Dart)

بيئات تطوير منافسة لتقنية

Cross-Platform
Kotlin Multi Mobile - Xamarin - React Native - ionic

تطوير التطبيقات الاصلية: اندرويد (java - kotlin) - ابل (objective-c - swift)

تطبيقات ثلاثية الابعاد (Unity)

مقدمة في Flutter



تطبيقات الجوال (اندرويد - ابل)

تطبيقات سطح المكتب (ويندوز - ماك - لينكس)

تطبيقات الويب

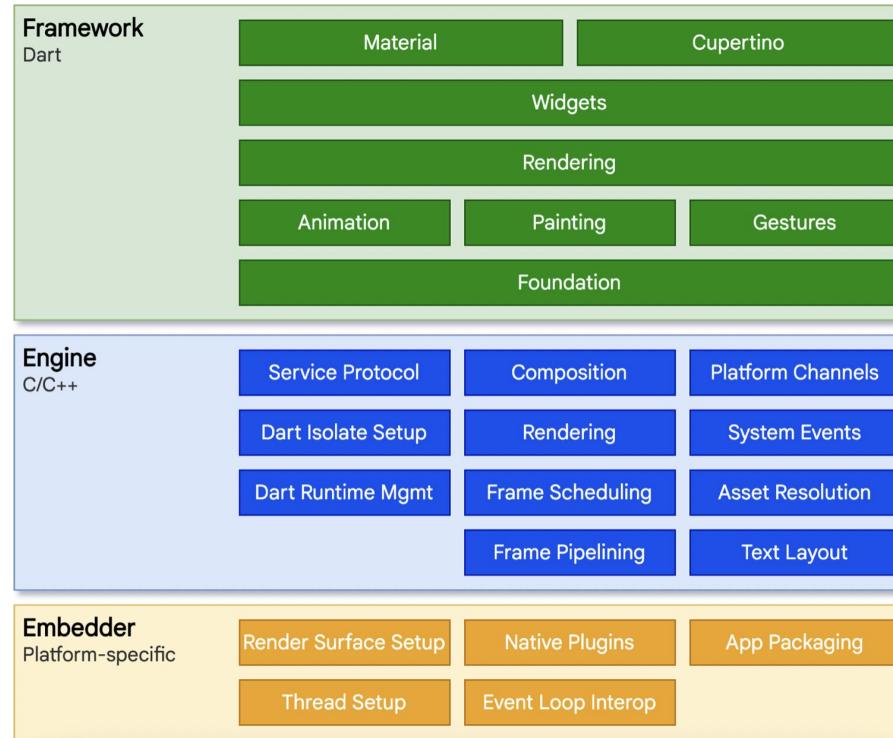
-
-
-





8

مقدمة في Flutter

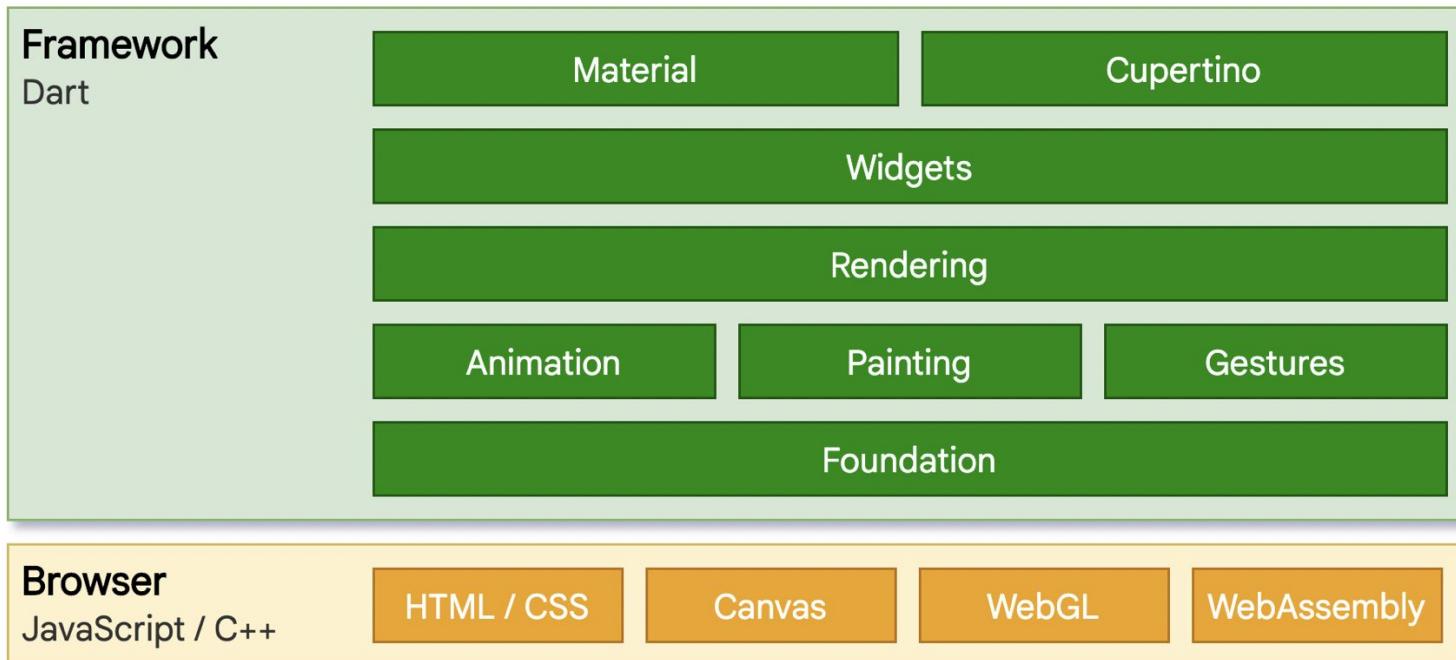




٩

مقدمة في Flutter

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



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 |
| ⑤ | Paint | Converting elements into a visual representation |
| ⑥ | Composition | Overlaying visual elements in draw order |
| ⑦ | Rasterize | Translating output into GPU render instructions |

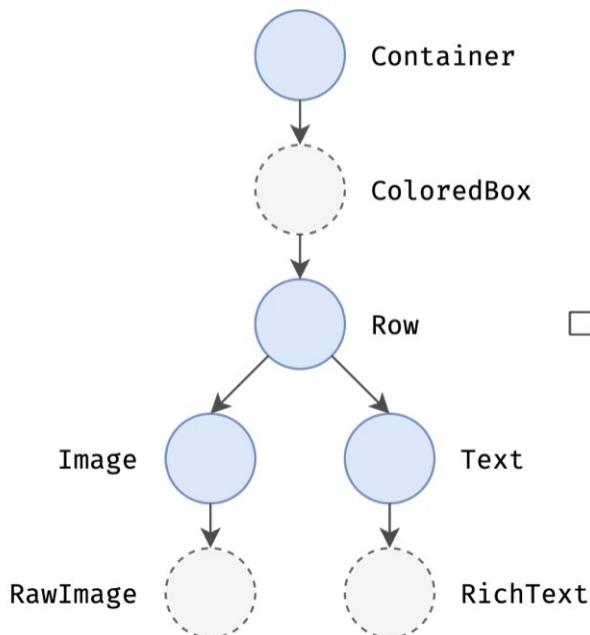
RENDERING



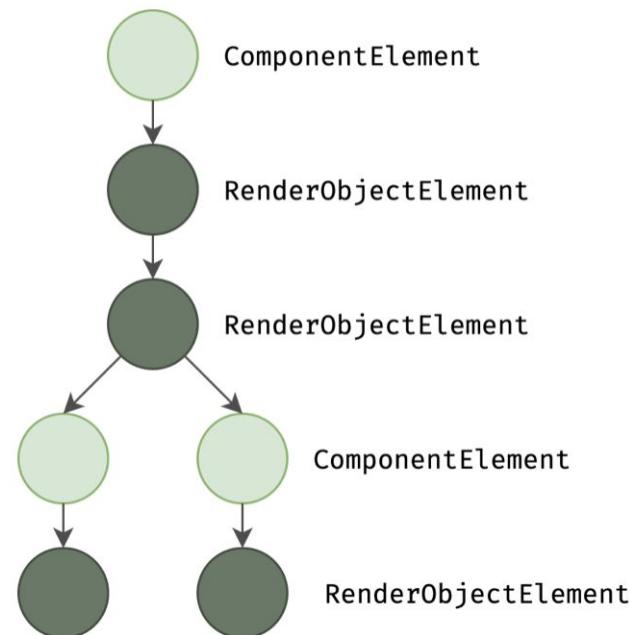
مقدمة في Flutter

11

Widgets



Element Tree

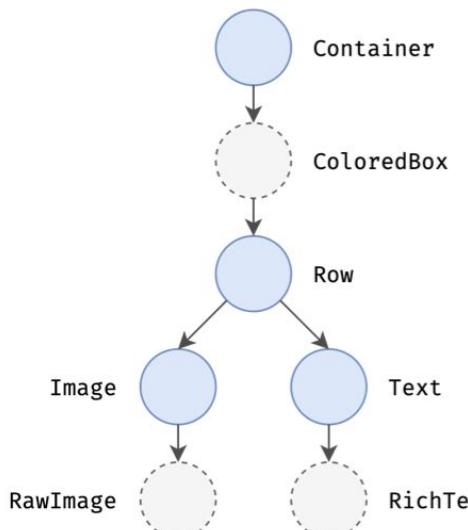




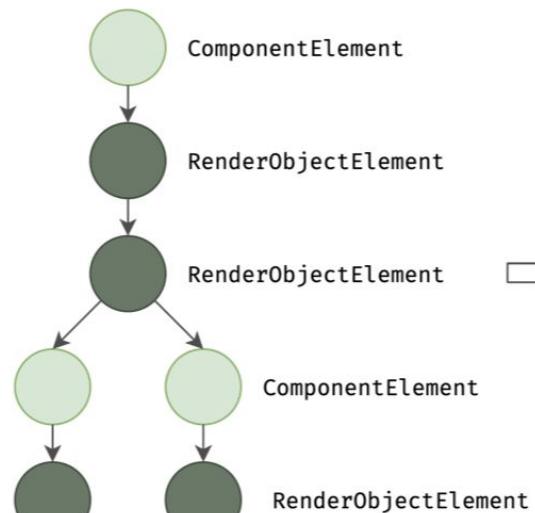
مقدمة في Flutter

12

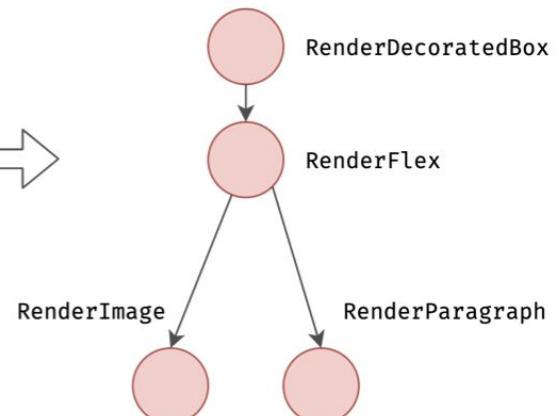
Widgets



Element Tree



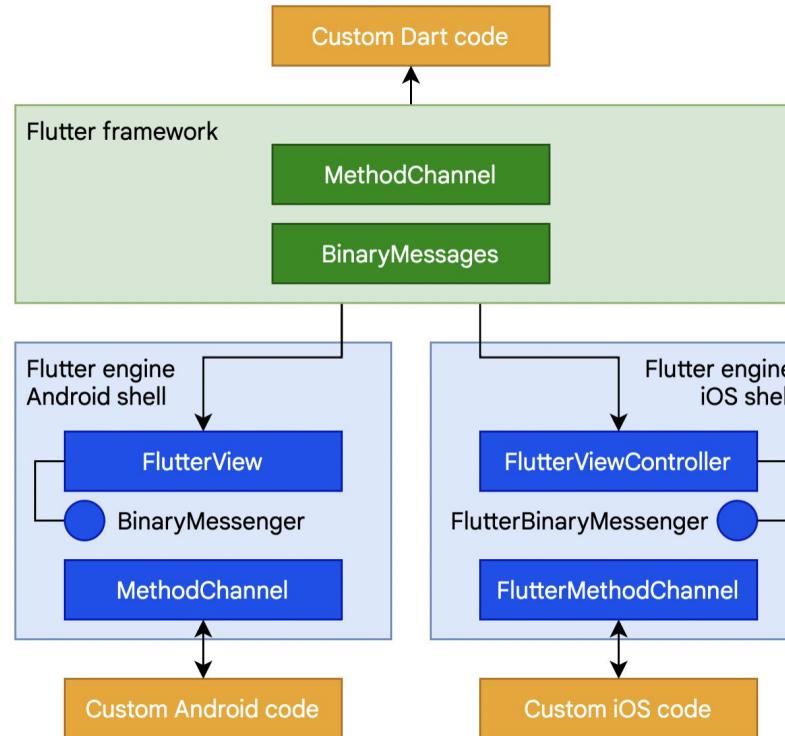
Render Tree





13

مقدمة في Flutter





تهيئة بيئة التطوير

14

- مواصفات جهاز جيدة
- تحميل Flutter SDK
- تحميل برنامج أ��واد
- ابل (JDK java) - اندرويد (cocoapods)



تهيئة بيئة التطوير

15



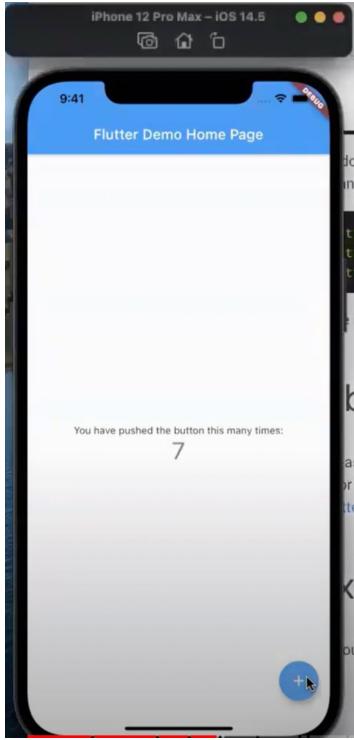
Vim





16

تشغيل أول تطبيق



The image shows a screenshot of an iPhone 12 Pro Max displaying the Flutter Demo Home Page. The screen shows the title "Flutter Demo Home Page" at the top, followed by a message "You have pushed the button this many times:" and the number "7". Below this, there is a large blue button with a plus sign. To the right of the phone screenshot is a screenshot of a code editor (VS Code) showing the main.dart file of a Flutter project named "FLUTTER_EXAMPLE". The code defines a MyApp widget that extends StatelessWidget. It sets the theme to ThemeData with a primarySwatch of Colors.blue and runs a MyHomePage with the title "Flutter Demo". The code editor also shows the output of the build process in the DEBUG CONSOLE, indicating that the app is launching on an iPhone 12 Pro Max in debug mode.

```
main.dart — flutter_example
lib/main.dart
1 import 'package:flutter/material.dart';
2
3 Run | Debug | Profile
4 void main() {
5   runApp(MyApp());
6
7 class MyApp extends StatelessWidget {
8   // This widget is the root of your application.
9   @override
10  Widget build(BuildContext context) {
11    return MaterialApp(
12      title: 'Flutter Demo',
13      theme: ThemeData(
14        // This is the theme of your application.
15        // Try running your application with "flutter run -d
16        // application has a blue toolbar. Then,
17        // changing the primarySwatch below to
18        // "hot reload" (press "r" in the console)
19        // or simply save your changes to "hot
20        // Notice that the counter didn't reset
21        // when the theme was changed.
22        primarySwatch: Colors.blue,
23      ), // ThemeData
24      home: MyHomePage(title: 'Flutter Demo Home Page'),
25    ); // MaterialApp
26 }

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE Filter
Launching lib/main.dart on iPhone 12 Pro Max in debug mode.
Xcode build done.
Connecting to VM Service at ws://127.0.0.1:41983
Lost connection to device.
Dart DevTools profiling file
Exited (sigterm)

Source: Dart (E)
```



UI = f(state)

The layout
on the screen

Your
build
methods

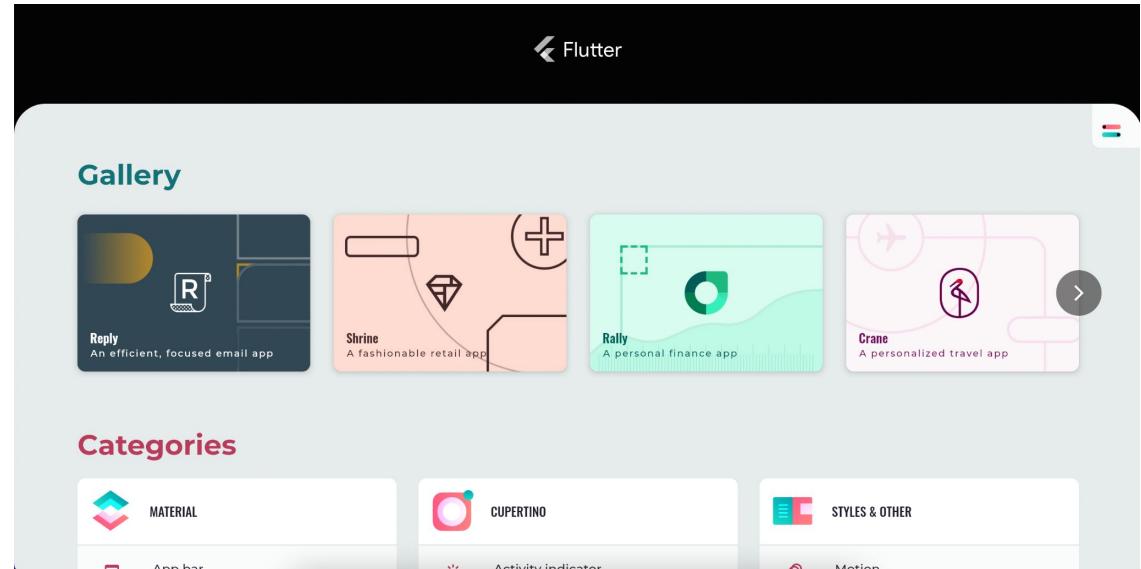
The application state



18

التصميم المتجاوب

- MediaQuery.of(context)
- Cupertino (Apple)
- Material (Android)
- gallery.flutter.dev



Null Safety

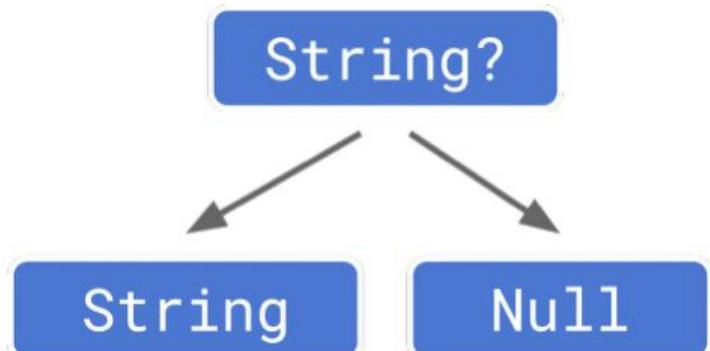
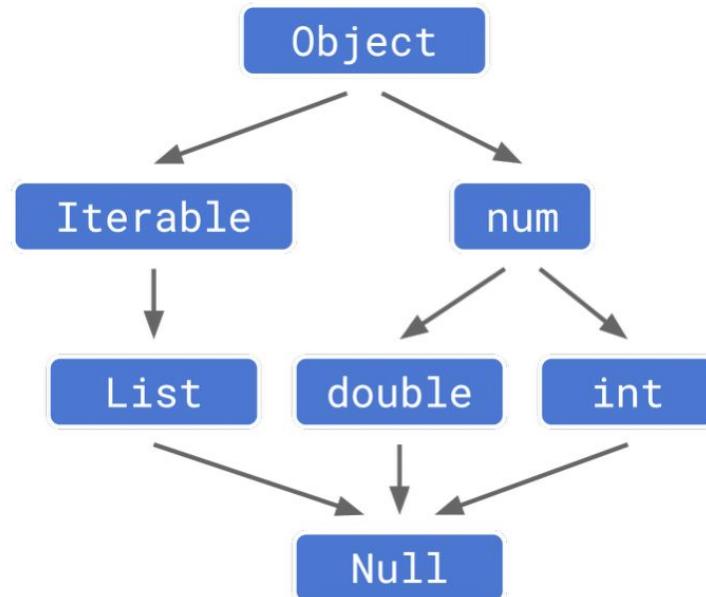


Sound null safety is available in Dart 2.12 and Flutter 2.

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?

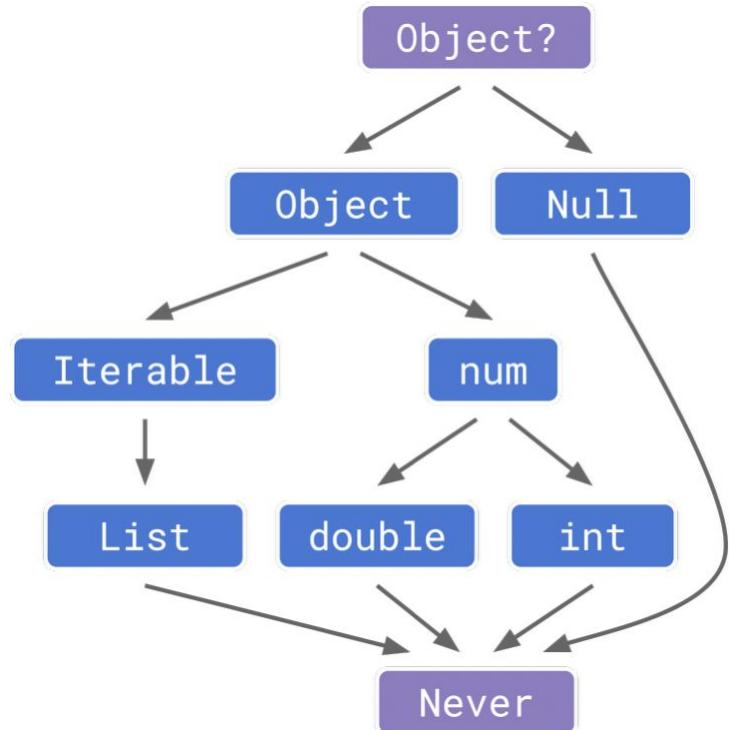
Iterable?

bool?

List?



Null Safety



Null Safety



- Types are non-null 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.

State Management



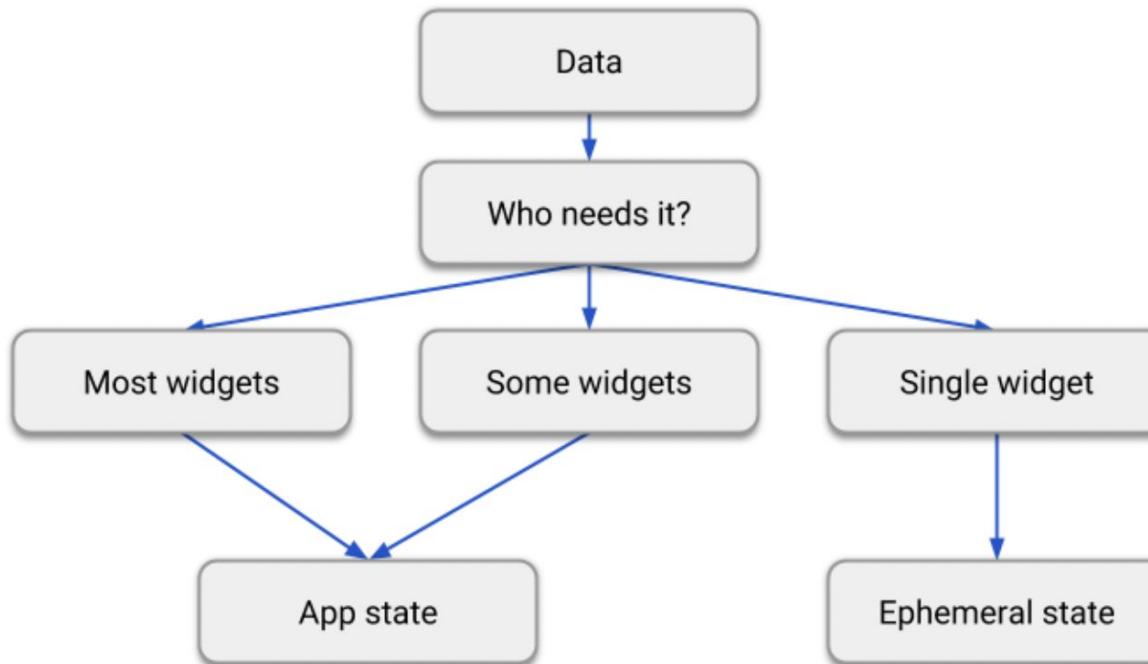
- Provider
- Riverpod
- setState (StatefulWidget - StatelessWidget - InheritedWidget)
- GetX
- Redux

State Management



- Fish-Redux
- BLoC / Rx
- GetIt
- MobX
- Binder

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](#)



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).

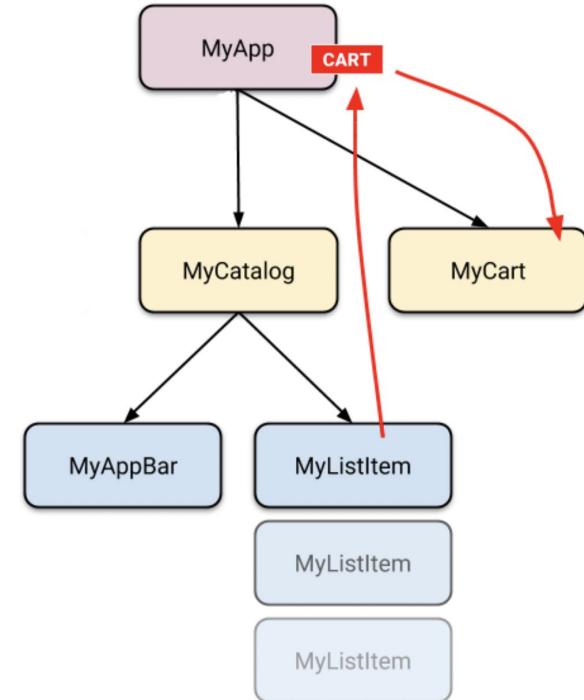
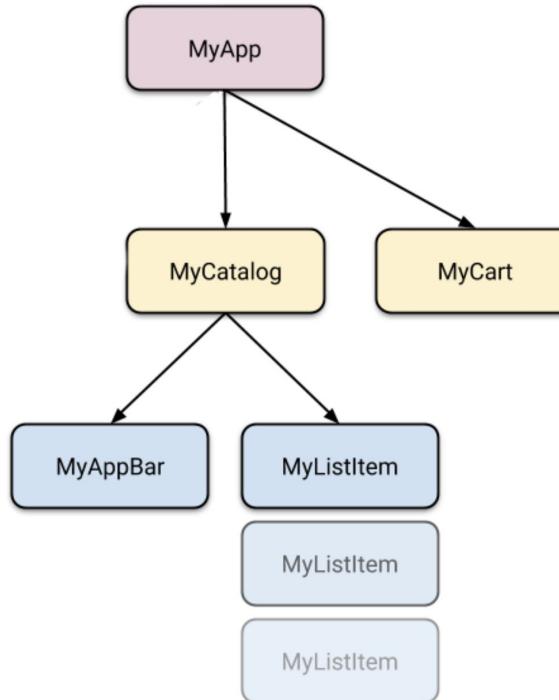
Examples of application state:

- 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



- 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}');
            },
        ),
    ),
);
```



المفاتيح

33

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

- Global Key
- Local Key (
 - ValueKey('Emily Furtuna') -
 - ObjectKey(MutableRectangle(1,2,3,4)) -
 - UniqueKey() -
 - PageStorageKey(scrollLocation))

Example: Scaffold and Form.



التنقل

34

- GoRouter (push - go) - (Declarative Navigation - no context)
- Navigator 2 (push - pop - pushReplacement - pushNamed - popUntil)



خيارات قواعد البيانات

35

- Firebase (FlutterFire) - BAAS (Backend As A Service)
- Sqflite - SqfEntity (local)
- Supabase (PostgreSQL)
- API http (json)
- Shared Preferences (path)



اختبارات الأداء

36

- Dev Tools
- flutter doctor
- flutter debug mode (code trace)



اختبارات الأداء

37

Dart DevTools Flutter Inspector Timeline Memory Performance Debugger Network Logging ⚡ ⚙️ ⚛️ ⚖️ ⚓

Select Widget Mode Refresh Tree Slow Animations Debug Paint Paint Baselines Repaint Rainbow Debug Banner

[root] MyApp MaterialApp MyHomePage Scaffold PageView CyanPage MyRoundedCard Container PhysicalShape ListView MyNormalWidget Row SizedBox Padding Container Expanded Column Padding Text MyNormalWidget Row SizedBox Padding Container Expanded

Main Axis start end

Details Tree Layout Explorer

Row Total Flex Factor: 1

SizedBox Expanded

flex: null unconstrained horizontal
fit: tight

(height is unconstrained) h=70.0

w=320.0; (w=320.0)

flex: 1 fit: tight
h=15.0

(height is unconstrained) h=70.0

w=390.0; (w=390.0)

w=70.0; (width is unconstrained)

h=25.0

Flutter DevTools Inspector x64-64 ios



أنماط بناء التطبيقات

38

- 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)



ملخص ومراجعة

39

- flutter.dev
- The Boring Flutter Development Show (Youtube)
- Widget of the Week (Youtube playlist)
- pub.dev
- fuchsia.dev (zircon is dart package)