



A fresh perspective on multi-platform UI




**Michael S. Thomsen**

Product Manager

Google / Flutter.io



A young man with glasses and a grey hoodie is looking at his smartphone. Next to him, a young woman with long brown hair and a grey cardigan is also looking at her smartphone. They are standing on a city street with blurred buildings and pedestrians in the background. A blue triangle graphic is in the top left corner.

Building a better way for developing  
mobile apps for iOS and Android



# Challenges of mobile development today...

## Platform SDK “to the metal” approaches

---

- ✓ **High-quality apps**  
Platform and system integrations
  - ✓ **High-performance UIs**  
Native code, GPU accelerated
  - ✗ **Must fund two apps**  
Two teams, codebases, & investments
  - ✗ **Inconsistent brand, features**  
Different across devices & OEMs
- 

## “Cross platform” approaches

---

- ✓ **Fast development**  
Quick iterations, hot reload
  - ✓ **Portability, reach**  
Single codebase
  - ✗ **Poor Performance**  
Slow, jerky, unpredictable
  - ✗ **Non-Native Look/Feel**  
Users can tell the difference
-



A young man with dark hair and glasses, wearing a grey patterned hoodie, and a young woman with long brown hair, wearing a light grey cardigan, are standing on a city street. Both are looking down at their smartphones. The background is a blurred city street with other pedestrians and buildings. A blue triangle graphic is in the top left corner.

What if there was a better way?





High-Velocity  
Development



Expressive and  
Flexible Toolkit



Native iOS and  
Android apps



# High-Velocity Development

# Unique UI model 'UI-as-code'

No special markup language to learn for describing UI; it's just code

Reuse traditional programming features for conditionals, lists operations, etc.

Re-use programming skills and technique for refactoring and structuring code

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Friendlychat"),  
        ),  
      ),  
    );  
  }  
}
```

# Unique UI model 'UI-as-code'

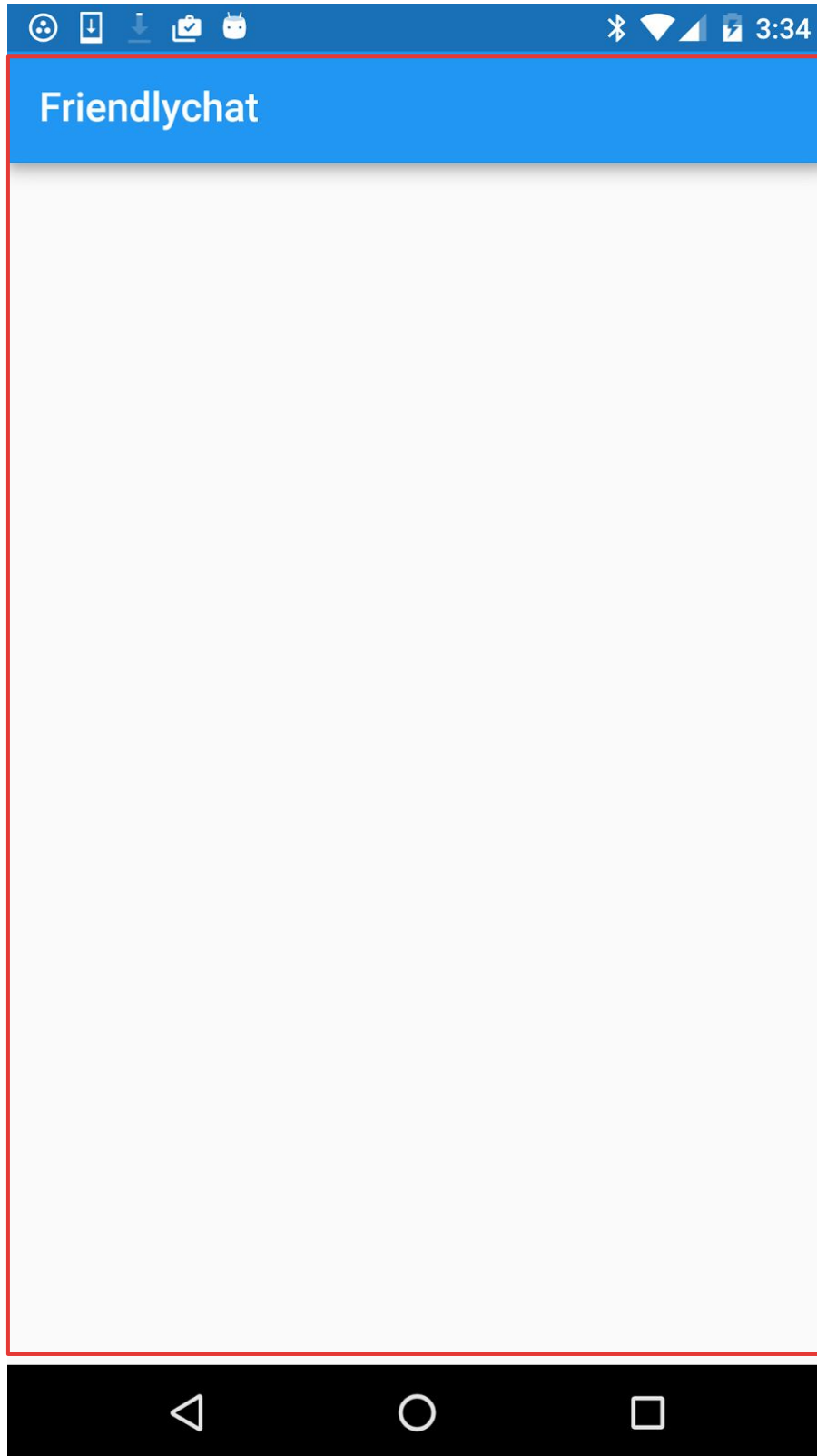
Everything is a **Widget**, even the app itself

**build()** methods render UI

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Friendlychat"),  
        ),  
      ),  
    );  
  }  
}
```

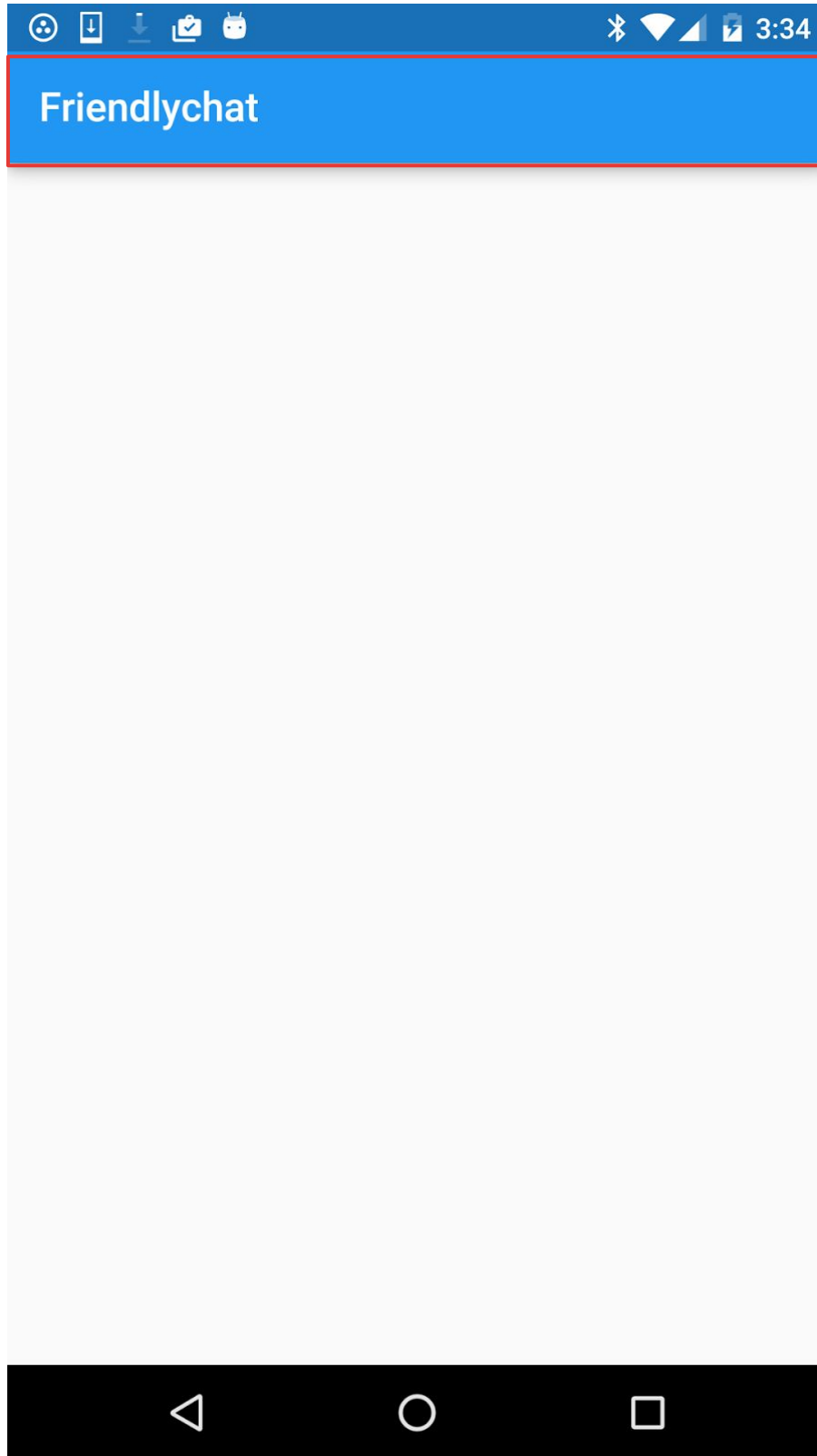


# Unique UI model 'UI-as-code'



```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Friendlychat"),  
        ),  
      ),  
    );  
  }  
}
```

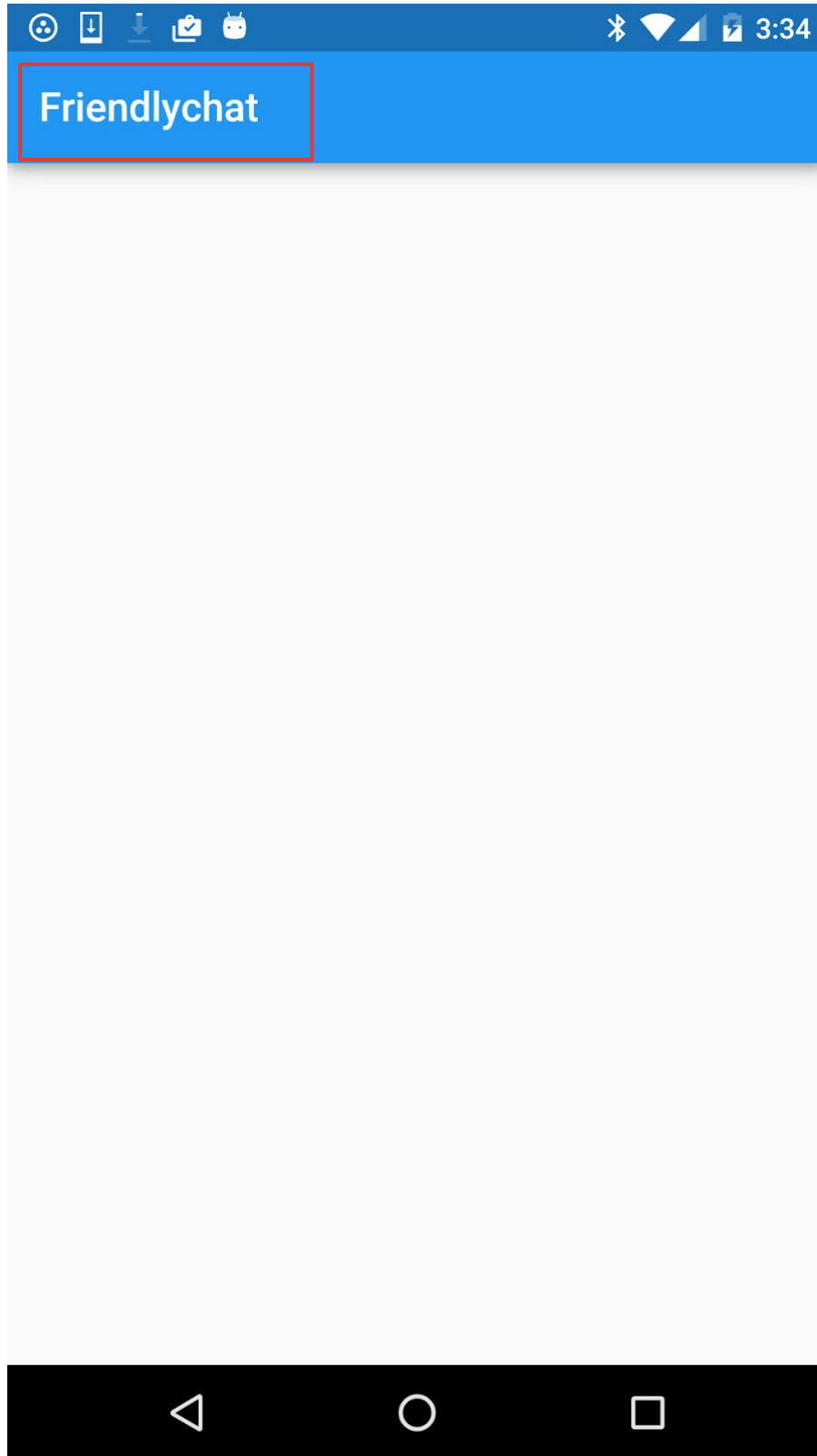
# Unique UI model 'UI-as-code'



```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Friendlychat"),  
        ),  
      ),  
    );  
  }  
}
```



# Unique UI model 'UI-as-code'



```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Friendlychat"),  
        ),  
      ),  
    );  
  }  
}
```

THE #1 PROGRAMMER EXCUSE  
FOR LEGITIMATELY SLACKING OFF:

"MY CODE'S COMPILING."

HEY! GET BACK  
TO WORK!

COMPILING!

OH. CARRY ON.





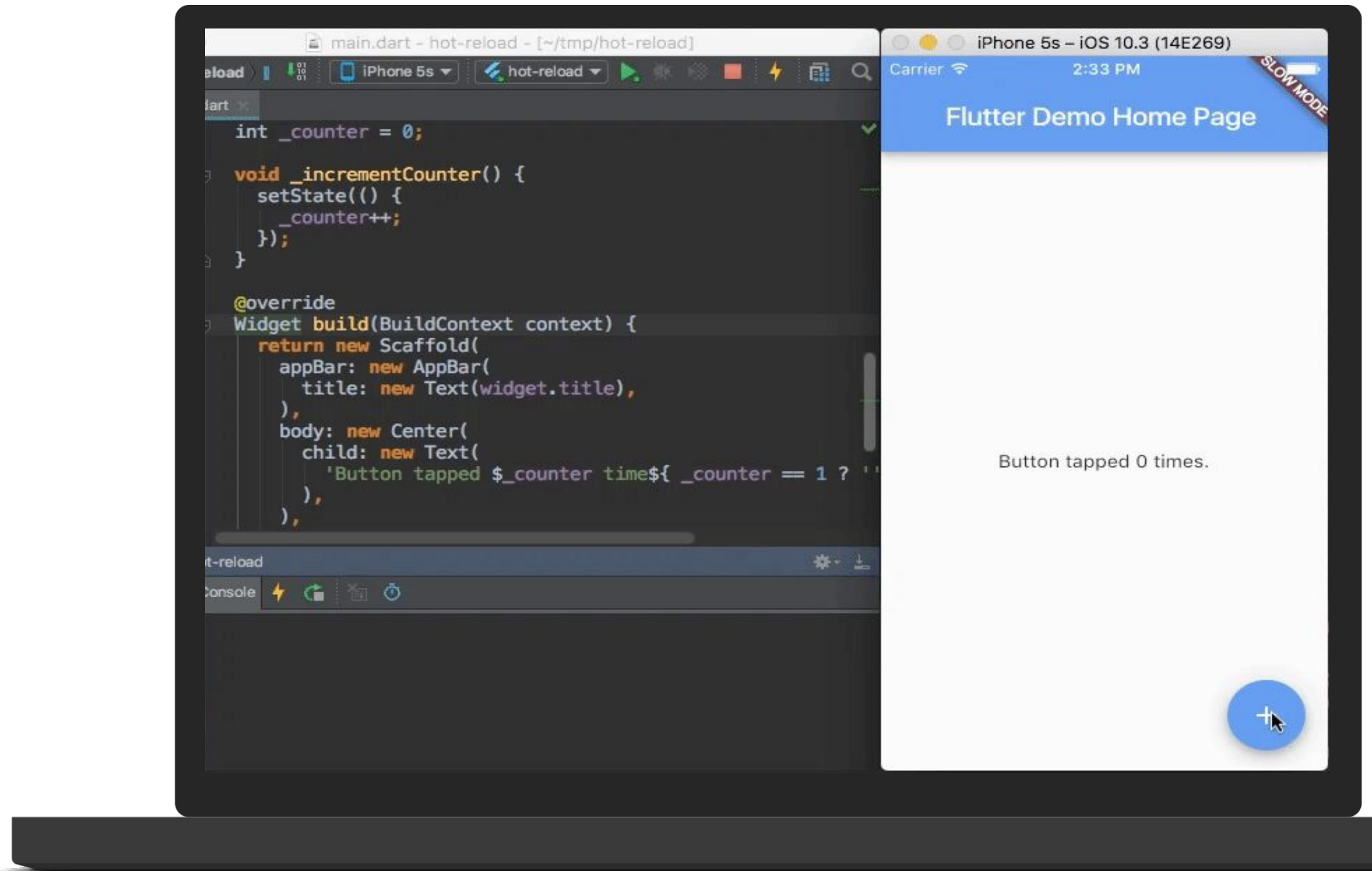
# High-velocity development

Sub-second reload times

Paint your app to life

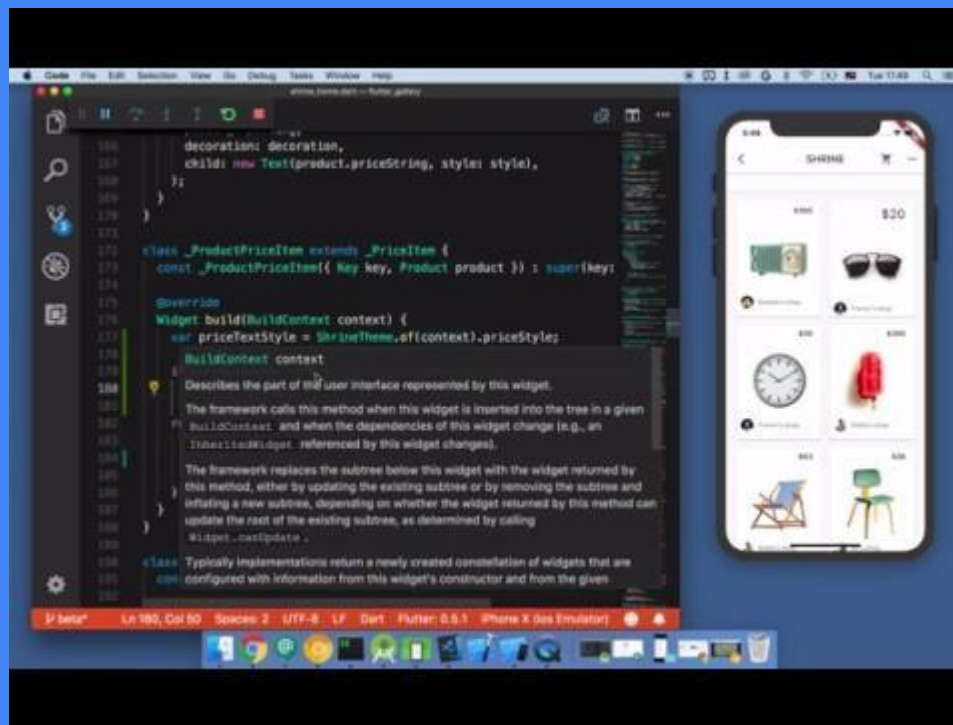
Inspect running UI

3X Productivity Gains



Demo:

**Flutter's super fast,  
stateful hot reload**

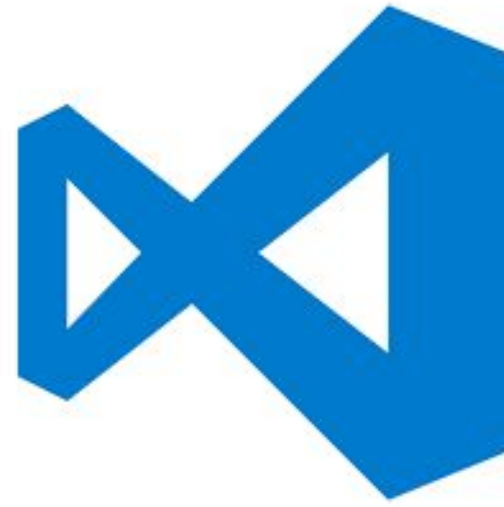




# Works with popular IDEs and editors



Android Studio



VS Code

```
$ flutter create lakesApp
```

```
$ flutter devices  
2 connected devices:  
Emulator - Pixel 2  
iPhone 6
```

```
$ flutter run -d iPhone
```

Terminal tool +  
bring-your-own editor

# Works with popular libraries, APIs, and SDKs



Android APIs



Android



iOS APIs



iOS



Firebase



Android  
Studio

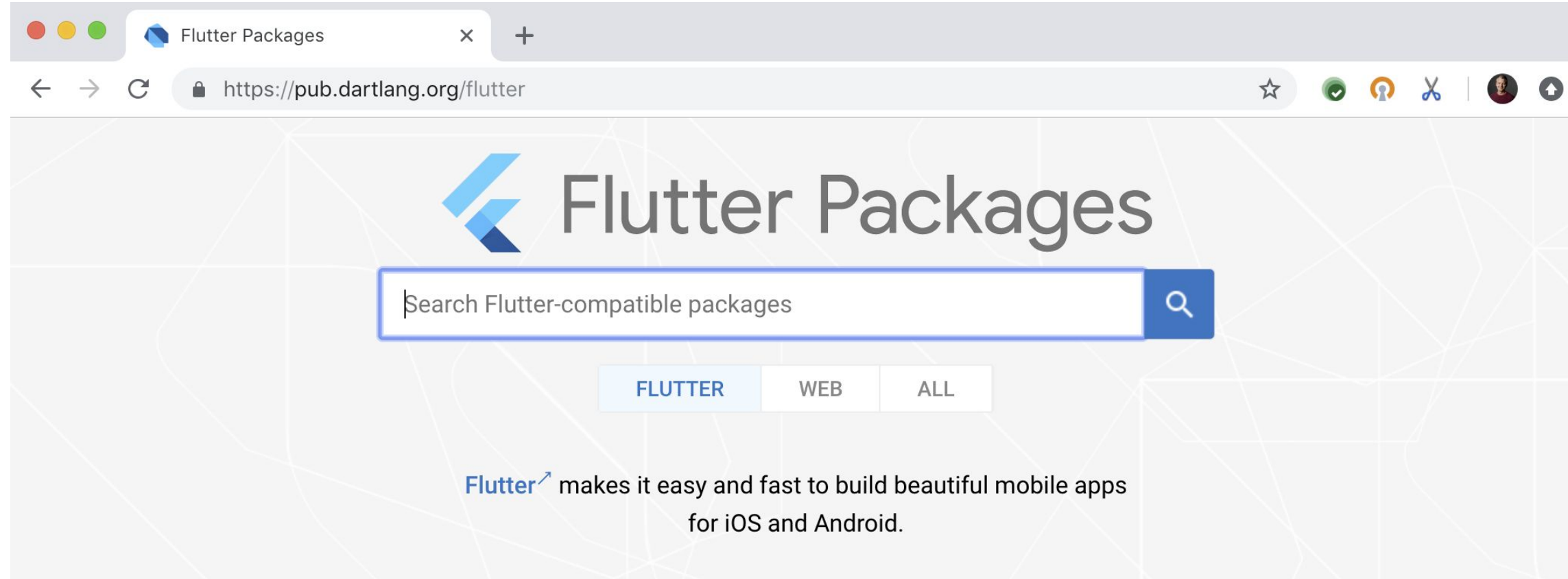


Xcode



# Package repository

- Download packages written by others
- Upload package to share
- 800+ Flutter packages published in the past year



## Top Flutter-compatible packages

### shared\_preferences

FLUTTER

Flutter plugin for reading and writing simple key-value pairs. Wraps NSUserDefaults on iOS and SharedPreferences on Android.

### path\_provider

FLUTTER

Flutter plugin for getting commonly used locations on the Android & iOS file systems, such as the temp and app data directories.

### url\_launcher

FLUTTER

Flutter plugin for launching a URL on Android and iOS. Supports web, phone, SMS, and email schemes.

### firebase\_auth

FLUTTER

Flutter plugin for Firebase Auth, enabling Android and iOS authentication using passwords, phone numbers and identity providers like Google, Facebook and Twitter.

### cloud\_firestore

FLUTTER

Flutter plugin for Cloud Firestore, a cloud-hosted, noSQL database with live synchronization and offline support on Android and iOS.

### image\_picker

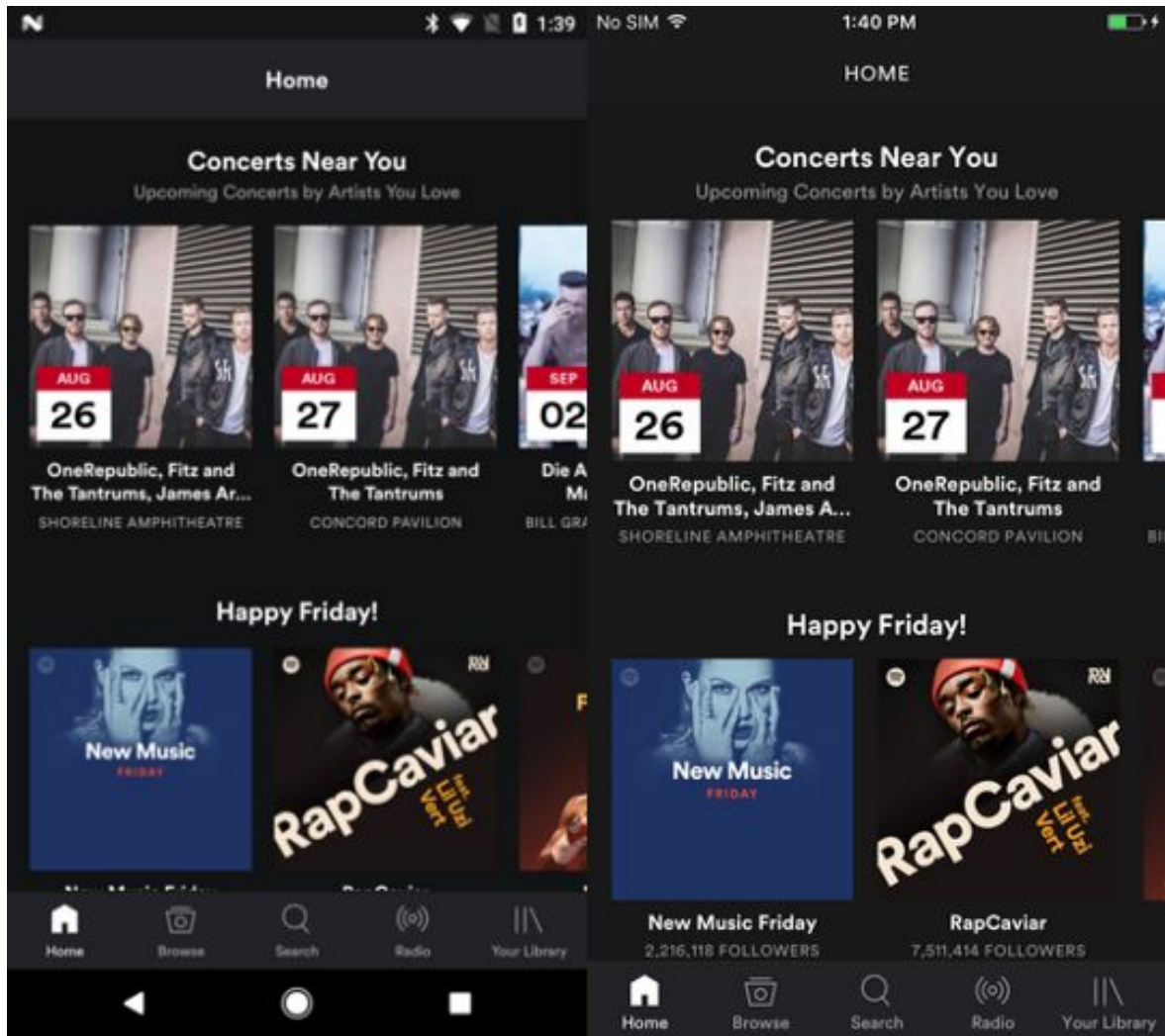
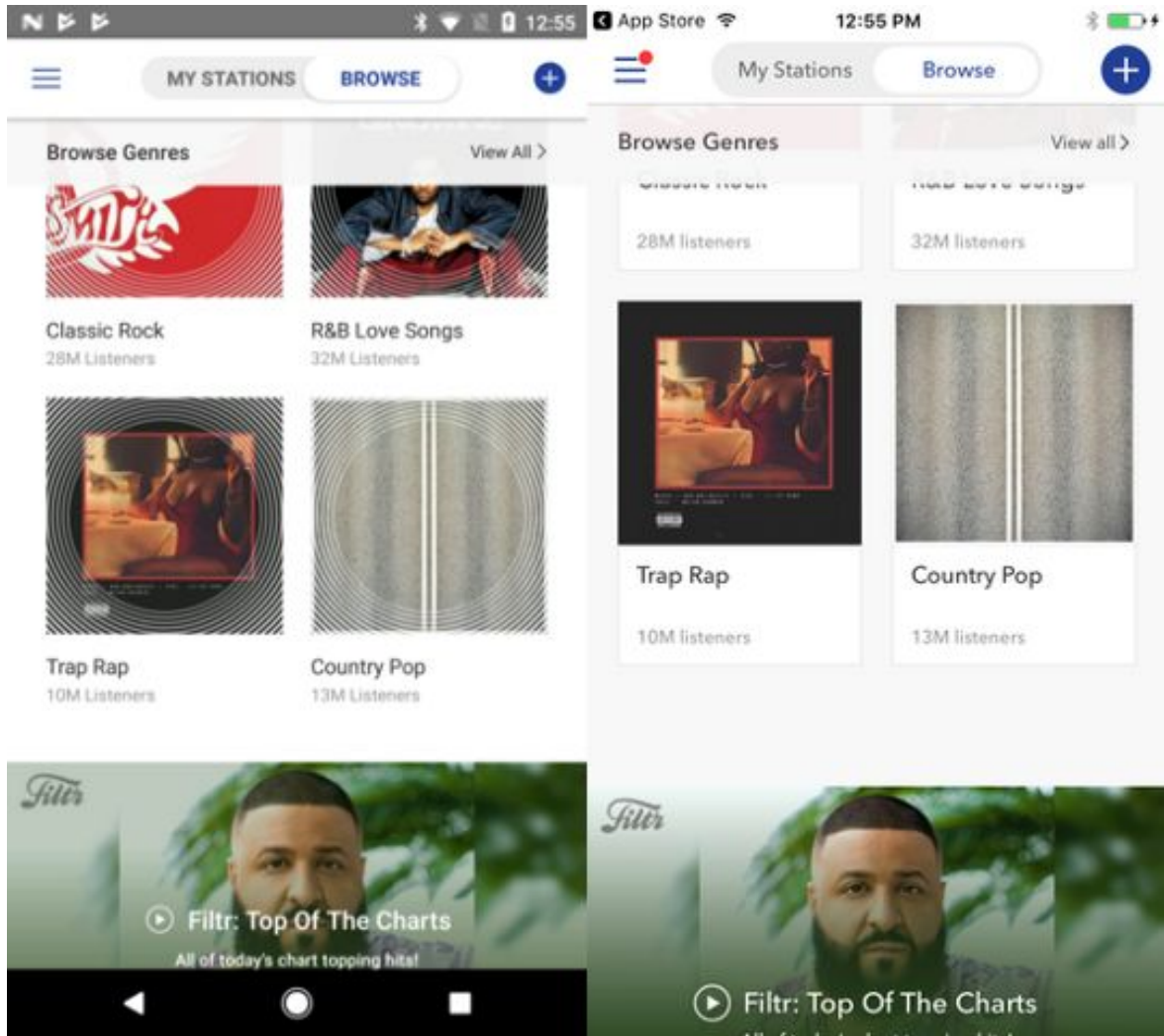
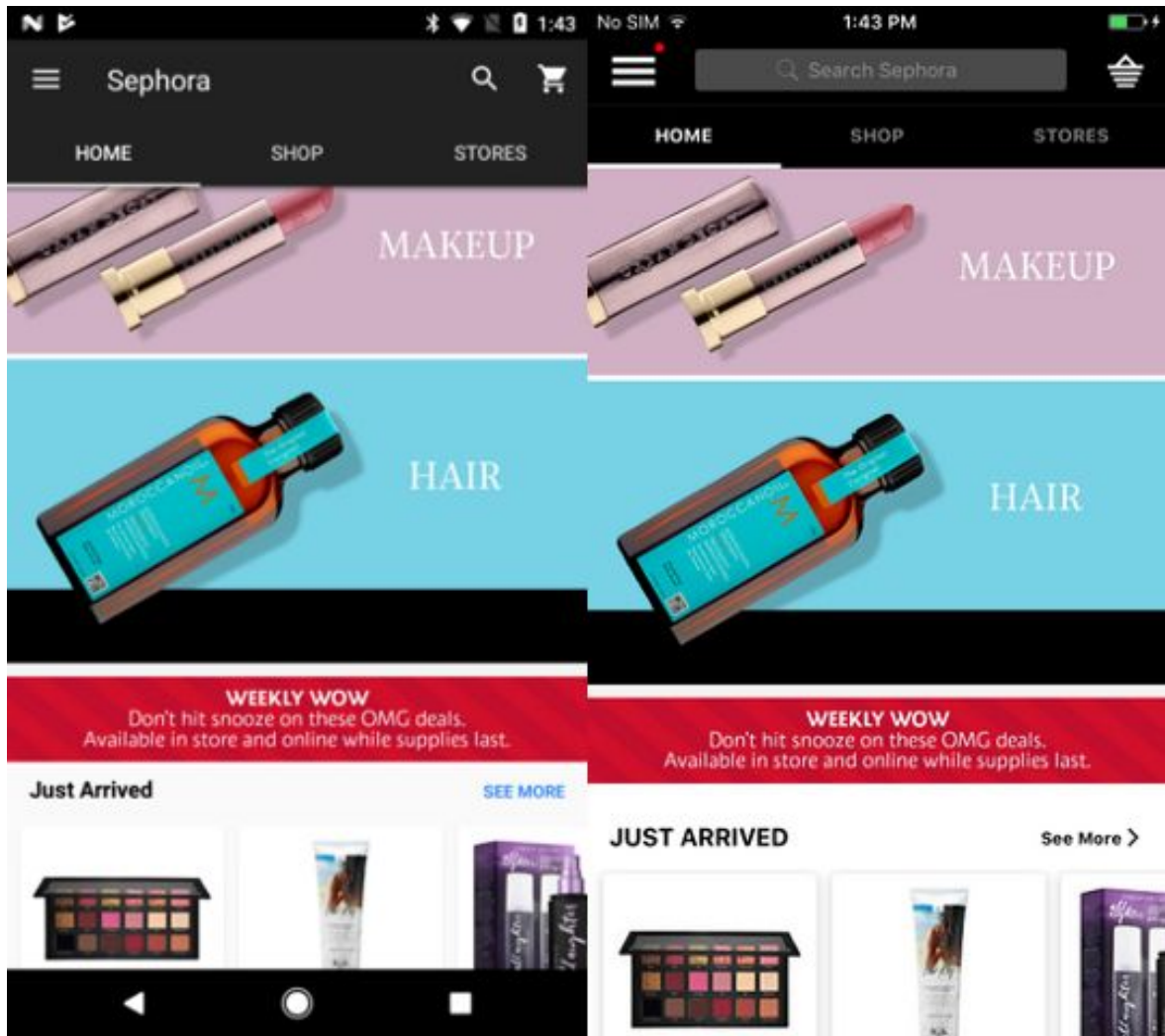
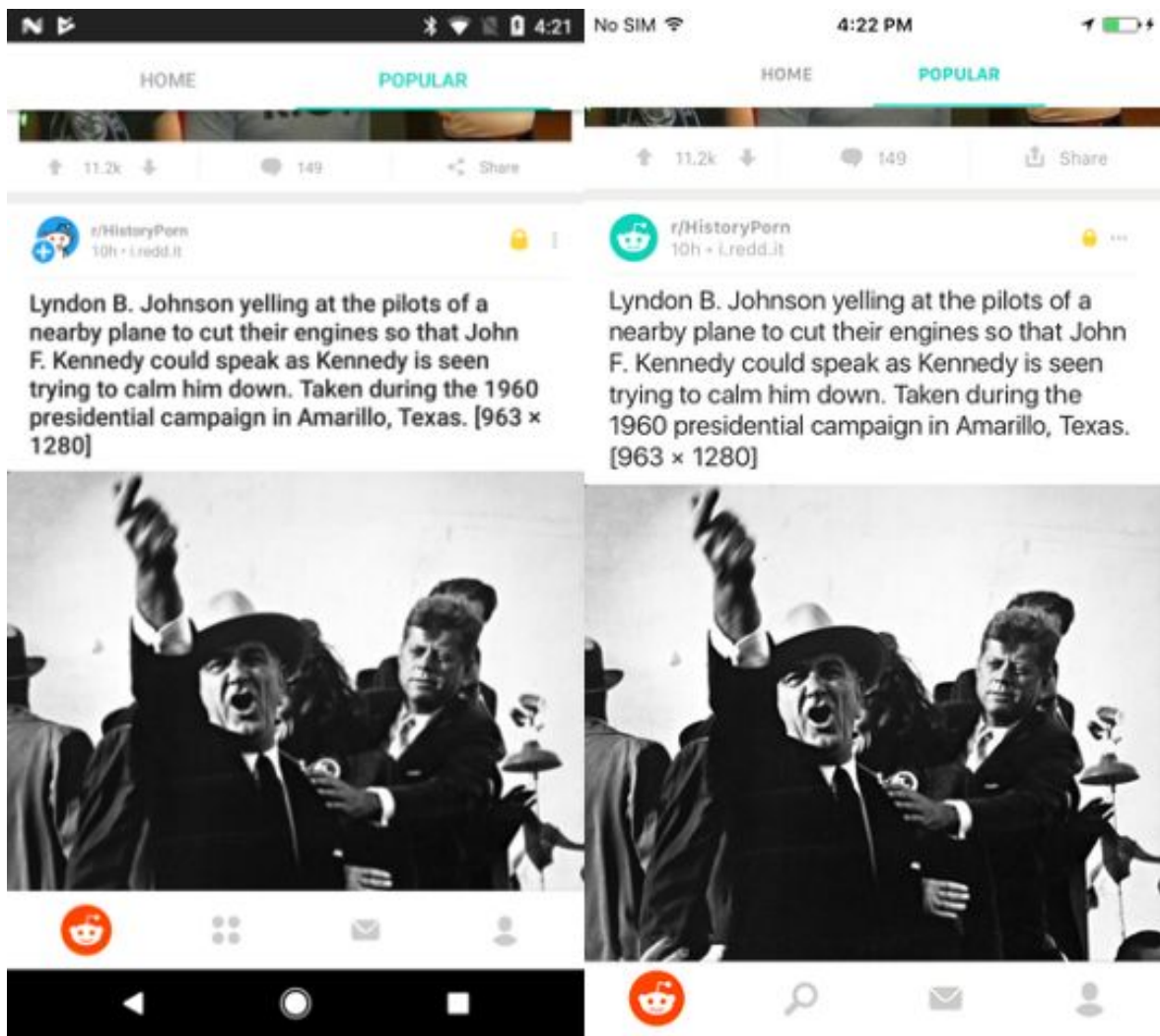
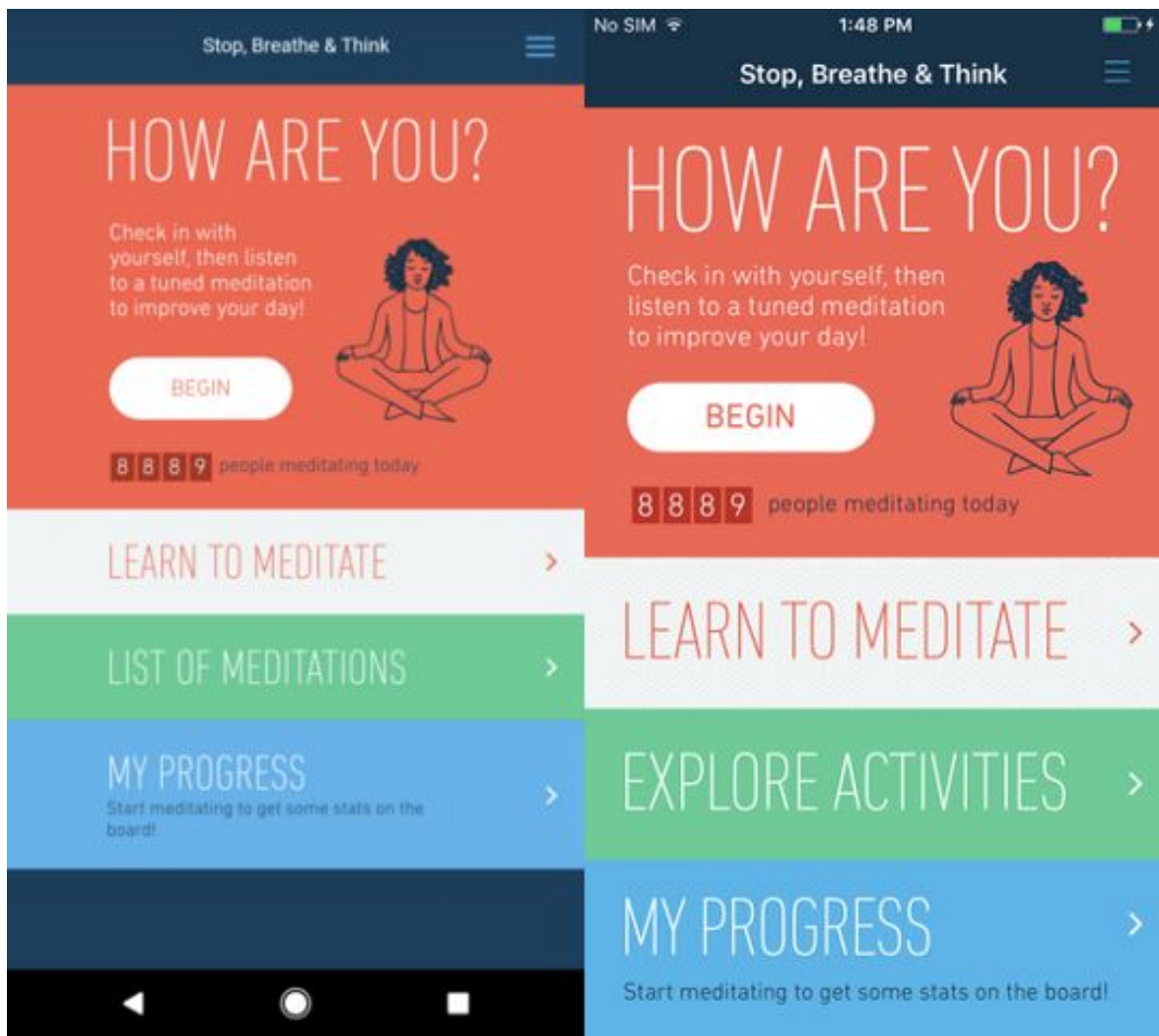
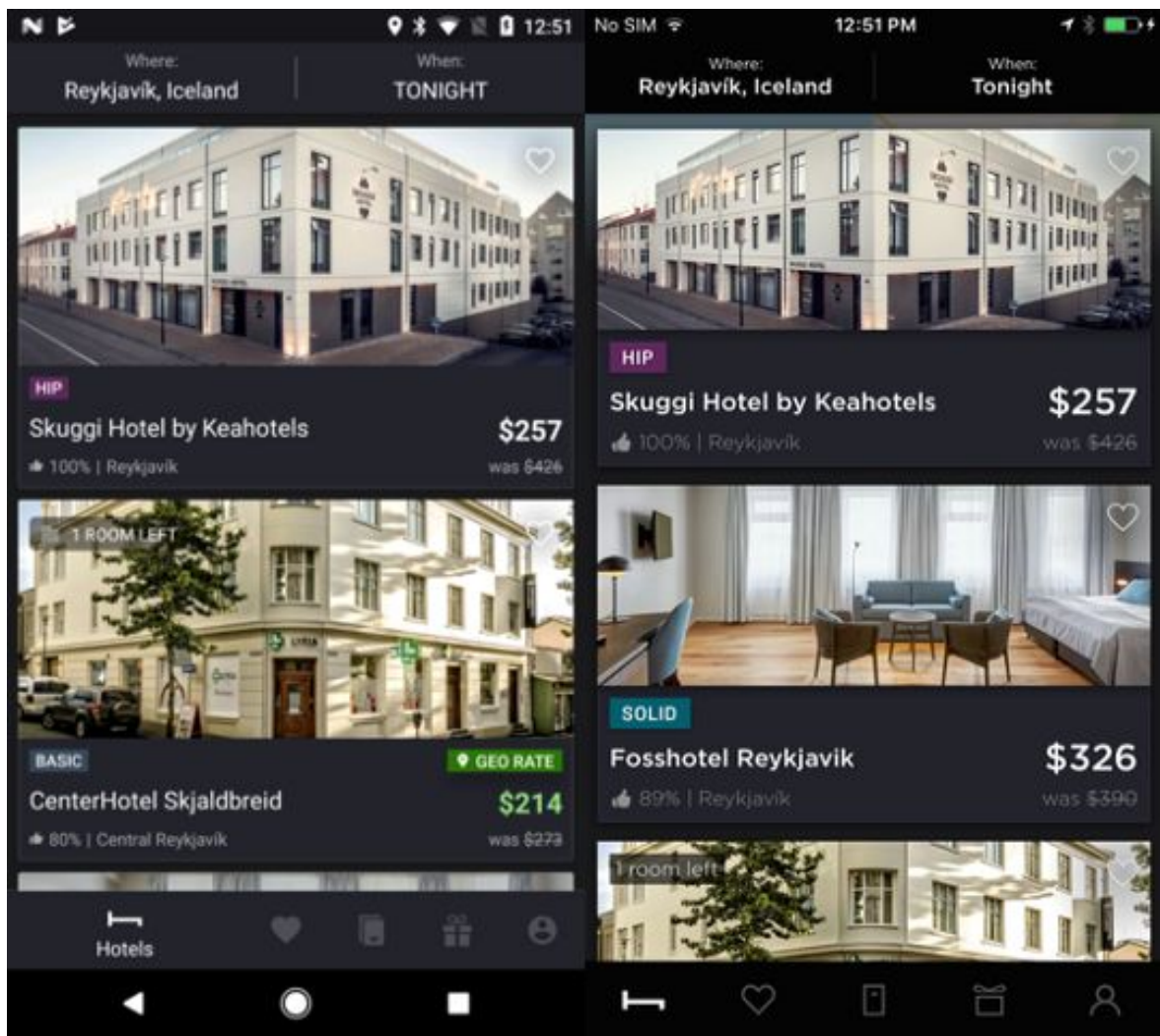
FLUTTER

Flutter plugin for selecting images from the Android and iOS image library, and taking new pictures with the camera.



Expressive, brand-centric designs







# Custom UI is not new



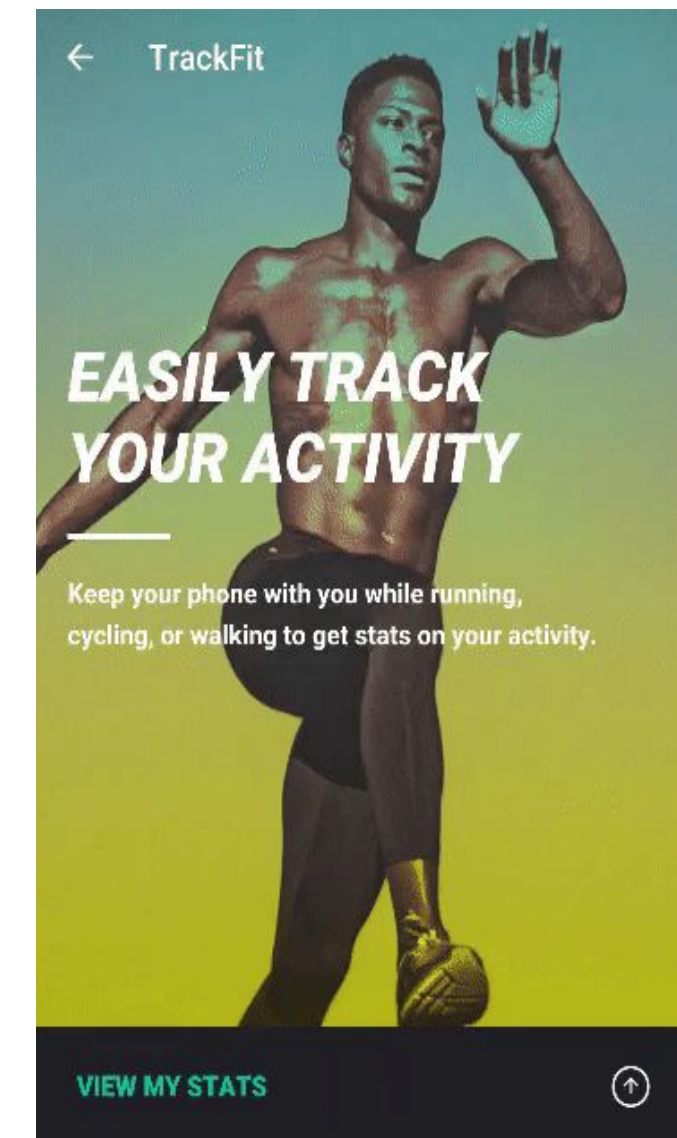
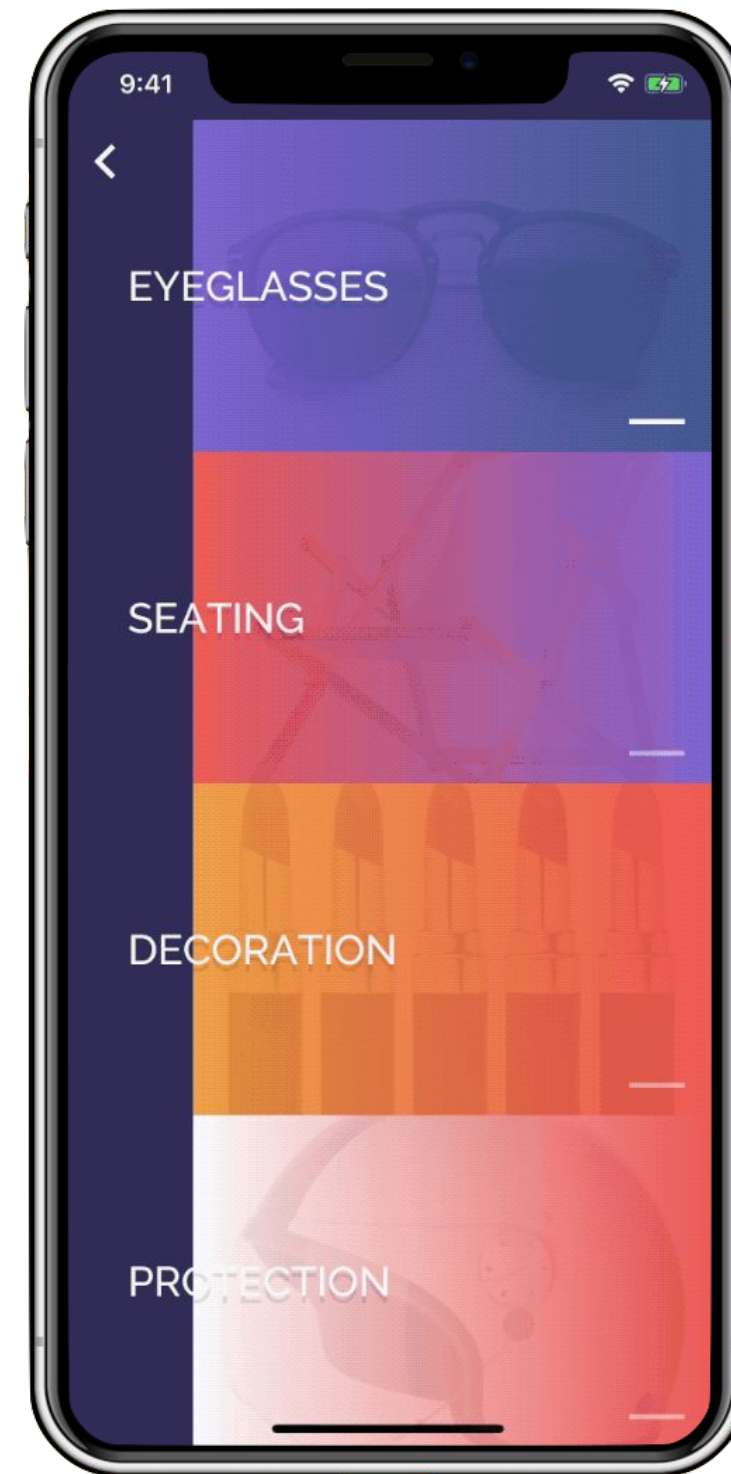
# Flexibility and Control for Beautiful UIs

Stand out in the marketplace

Make your brand come to life

Never say "no" to your designer

Win awards with beautiful UI



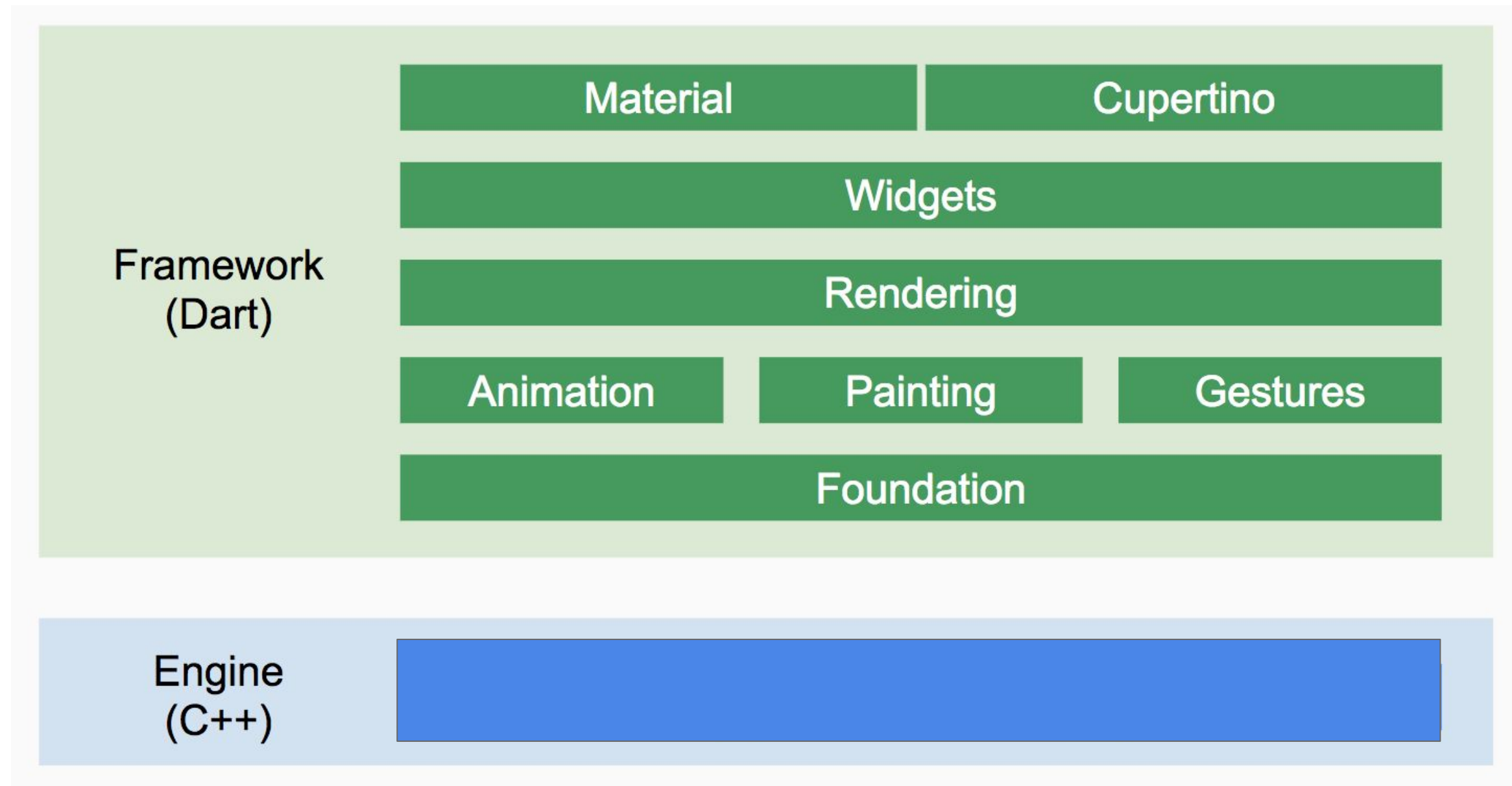


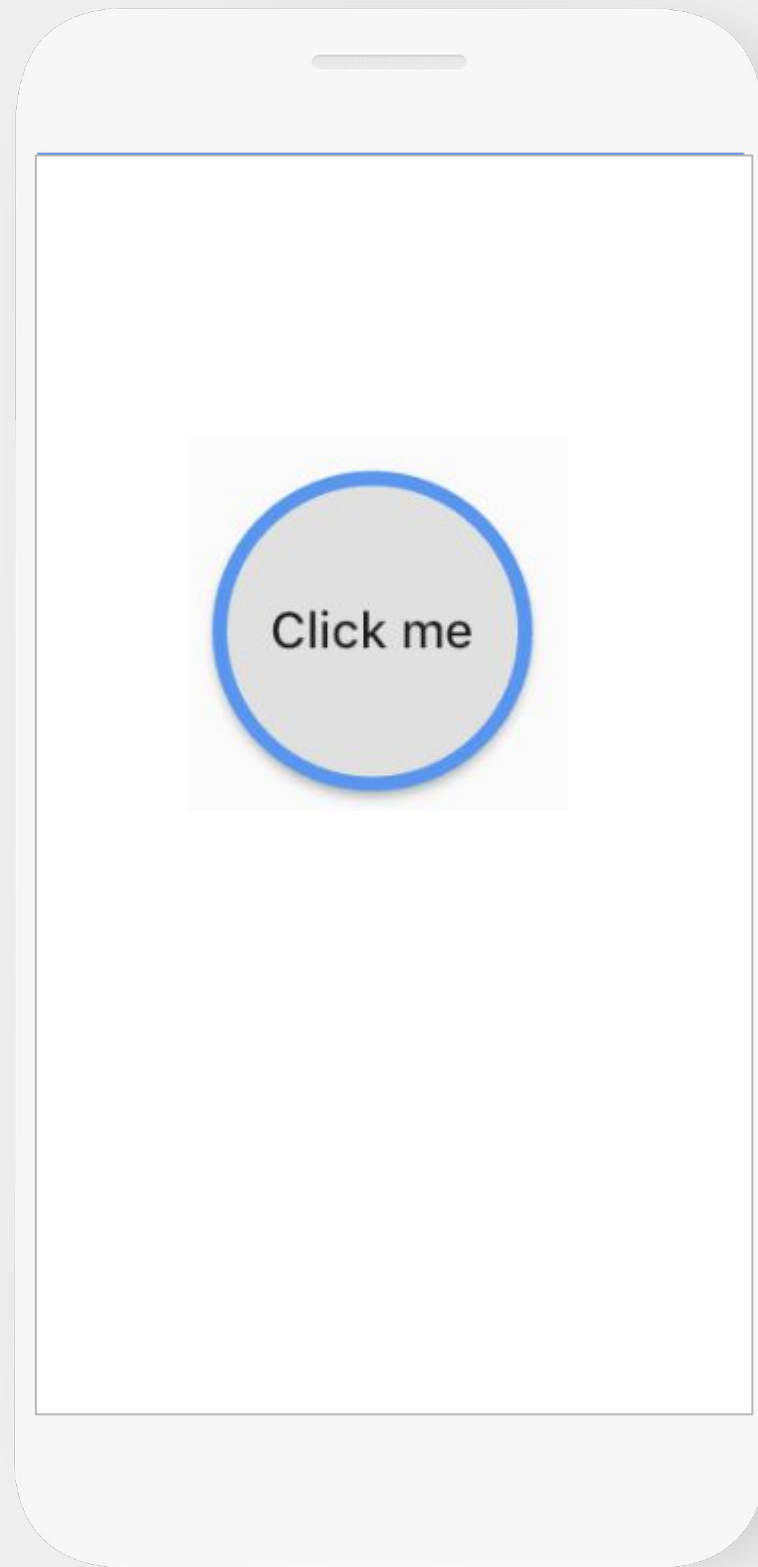
# Framework support for brand-specific design

Flexible widgets for traditional (Material & iOS) and brand-specific design

Compose your UI from flexible building blocks

Ship a consistent feature set to customers on both iOS and Android





DEMO: Customer wanted a custom 'CircularButton' widget

- Fixed size (120x120 dp).
- Text in middle.
- Blue border.
- Able to receive 'pressed' events

General approach:

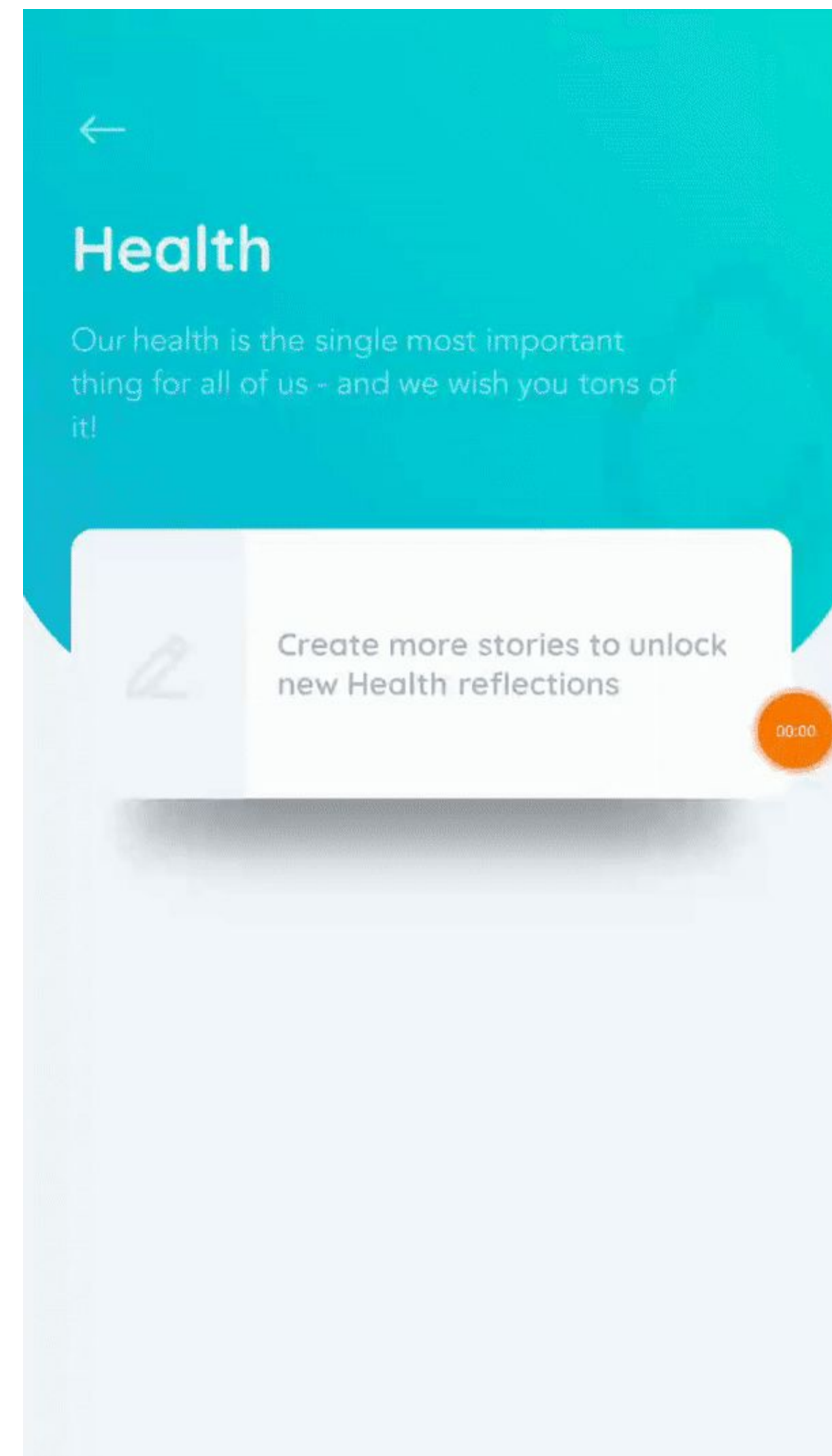
- Compose from existing widgets.
- Look at source code of existing widgets for inspiration.

# Case Study: Reflectly

100% custom, brand-specific design

With shared UI, and Flutter's shared widget set, able to have create this UI from a single, shared codebase across iOS & Android.

Implemented by a very small team in just a few months





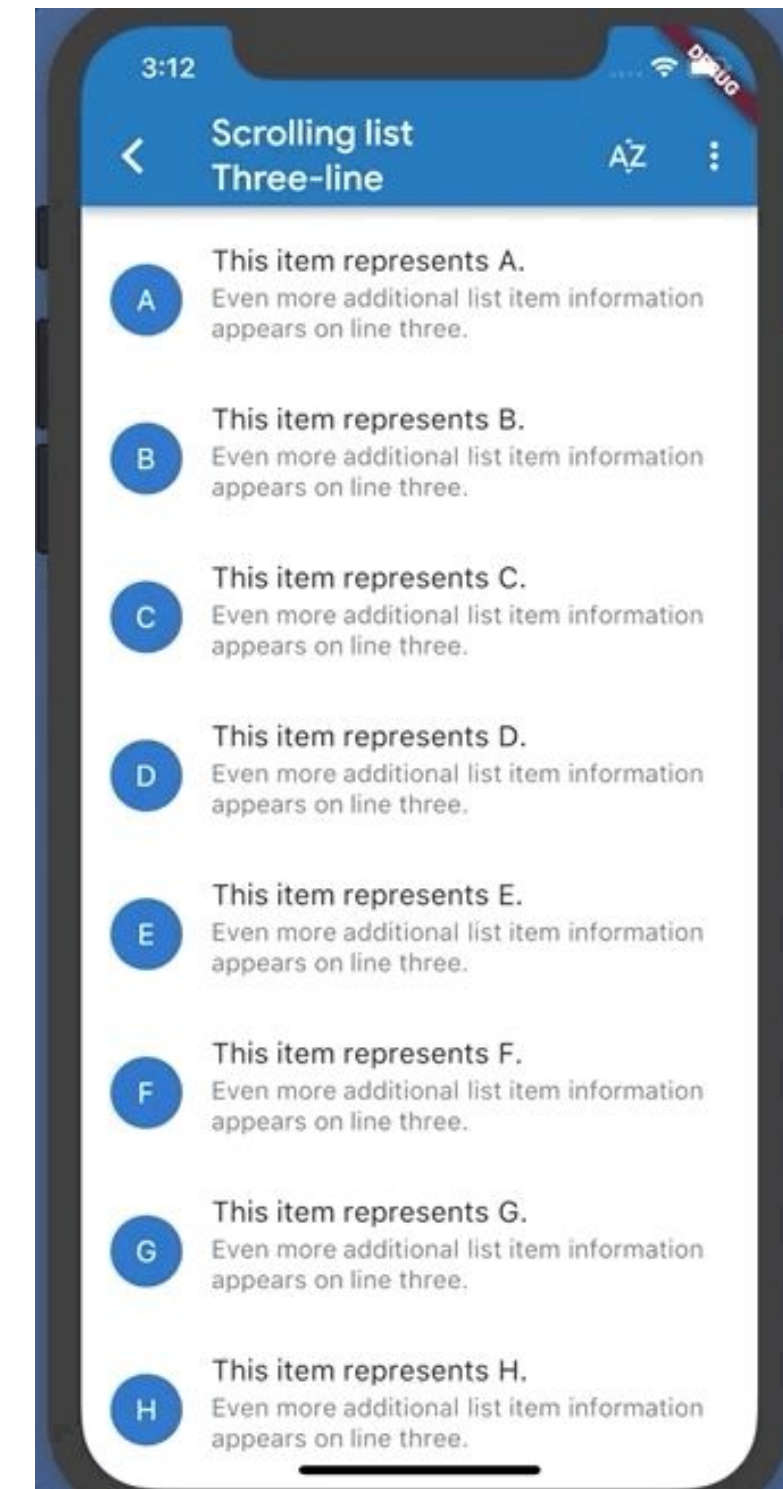
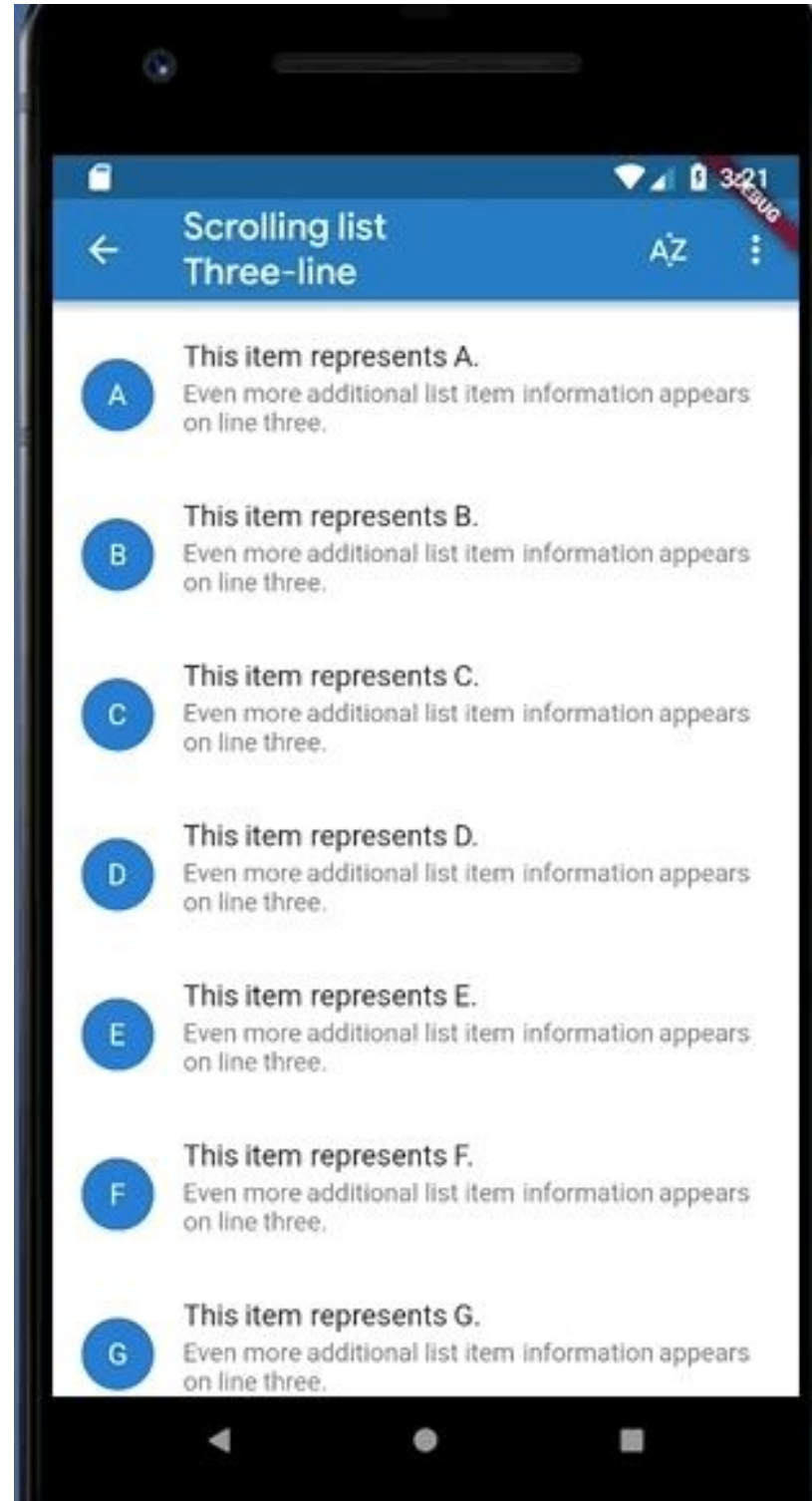


# Native iOS and Android App

# Natural platform affordances

Automatically supports  
critical platform  
differences

- Scrolling
- Navigation
- Fonts



# Natural platform affordances

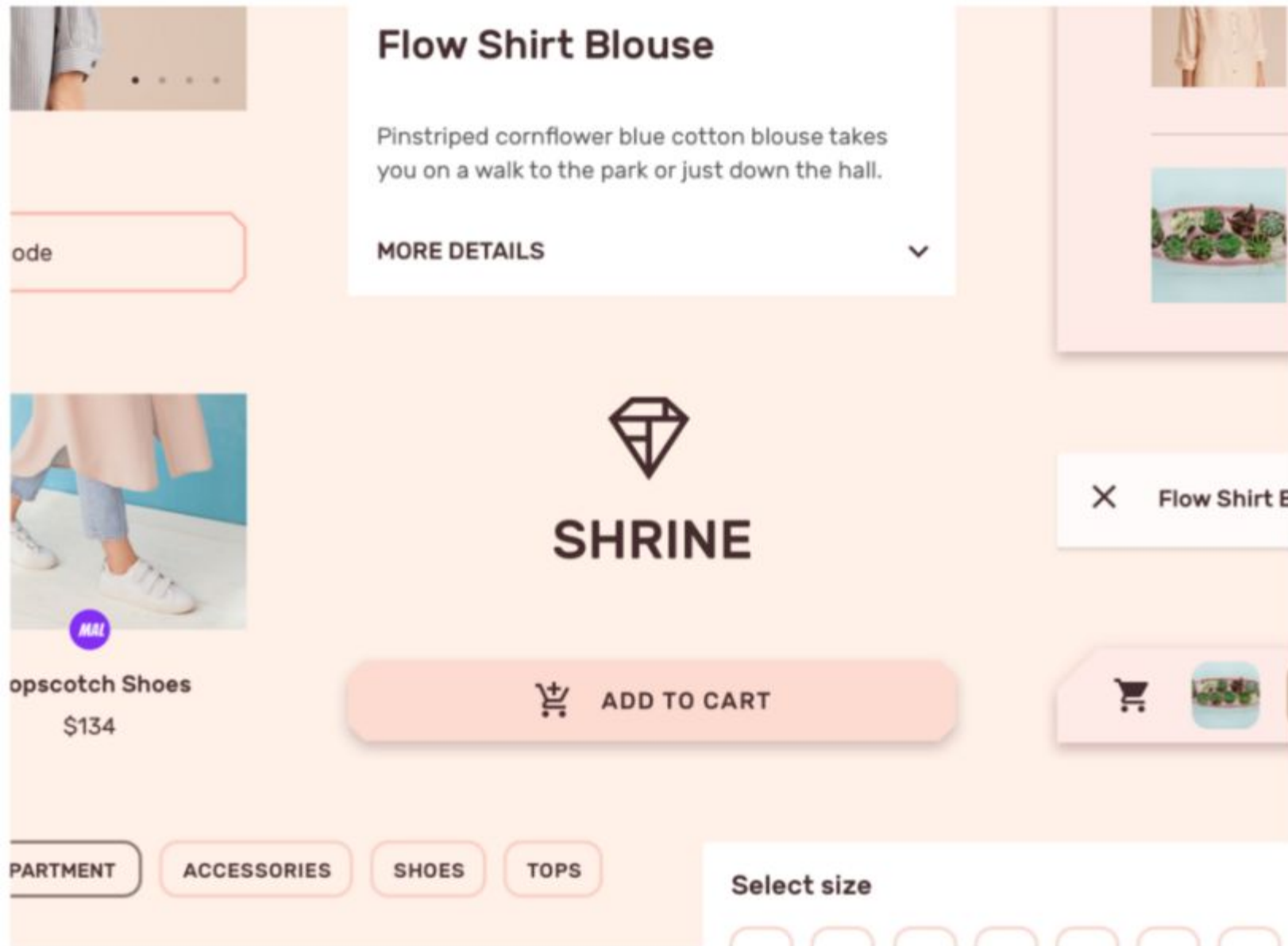
Automatically supports  
critical platform  
differences

- Scrolling
- Navigation
- Fonts





# First-class Material Design



*“I’m incredibly excited to welcome Flutter into the **official set of Material Design Components** as a full fledged peer to our Android, iOS and Web offerings.*

*Flutter’s philosophy of flexible and adaptable widgets is a great fit for Material Theming, and **Flutter’s ability for real time UI iteration is a game changer in the way we polish and refine designs.**”*

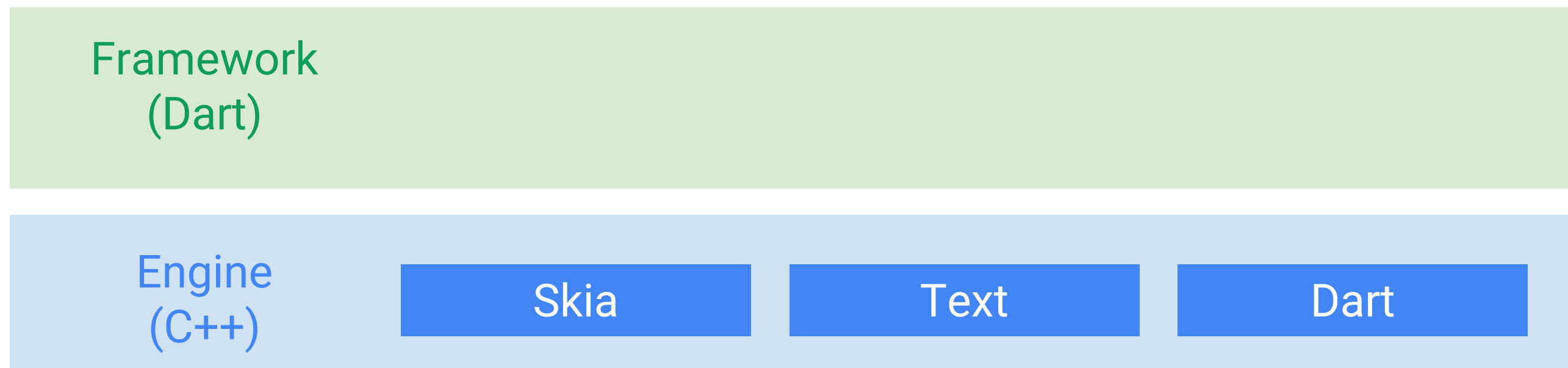
Matías Duarte  
VP, Google Material Design

# Layered stack, very fast primitives at the bottom

Graphics with Skia: Android graphics engine

Text: Custom text rendering

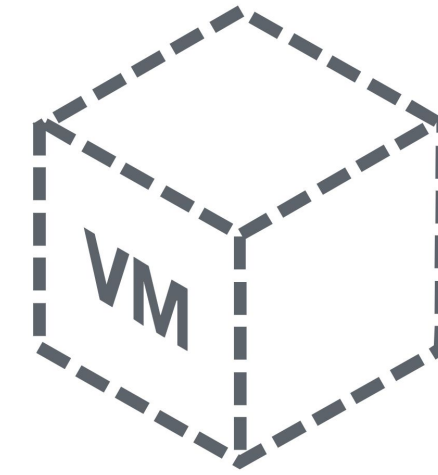
Dart: Google open-source app language



# Speed through many custom language technology

## Development builds

Custom **VM** offers super fast **hot reload** change cycle



## Release builds (when shipping)

Full AOT-compilation to native **machine code**  
offers **super fast startup and execution**

```
...  
ldr r1, [sp]  
ldr d0, [r1, #3]  
vcmpd d0, d0  
vmstat  
bvs L0  
vcvtid s2, d0  
vmovrs r0, s2  
adds r0, r0, r0  
bvs L1
```

## In both

Language **run-time optimized for UI**

- Many new objects: Lock-free, fast allocation
- Short-lived objects: Precise, generational garbage collection

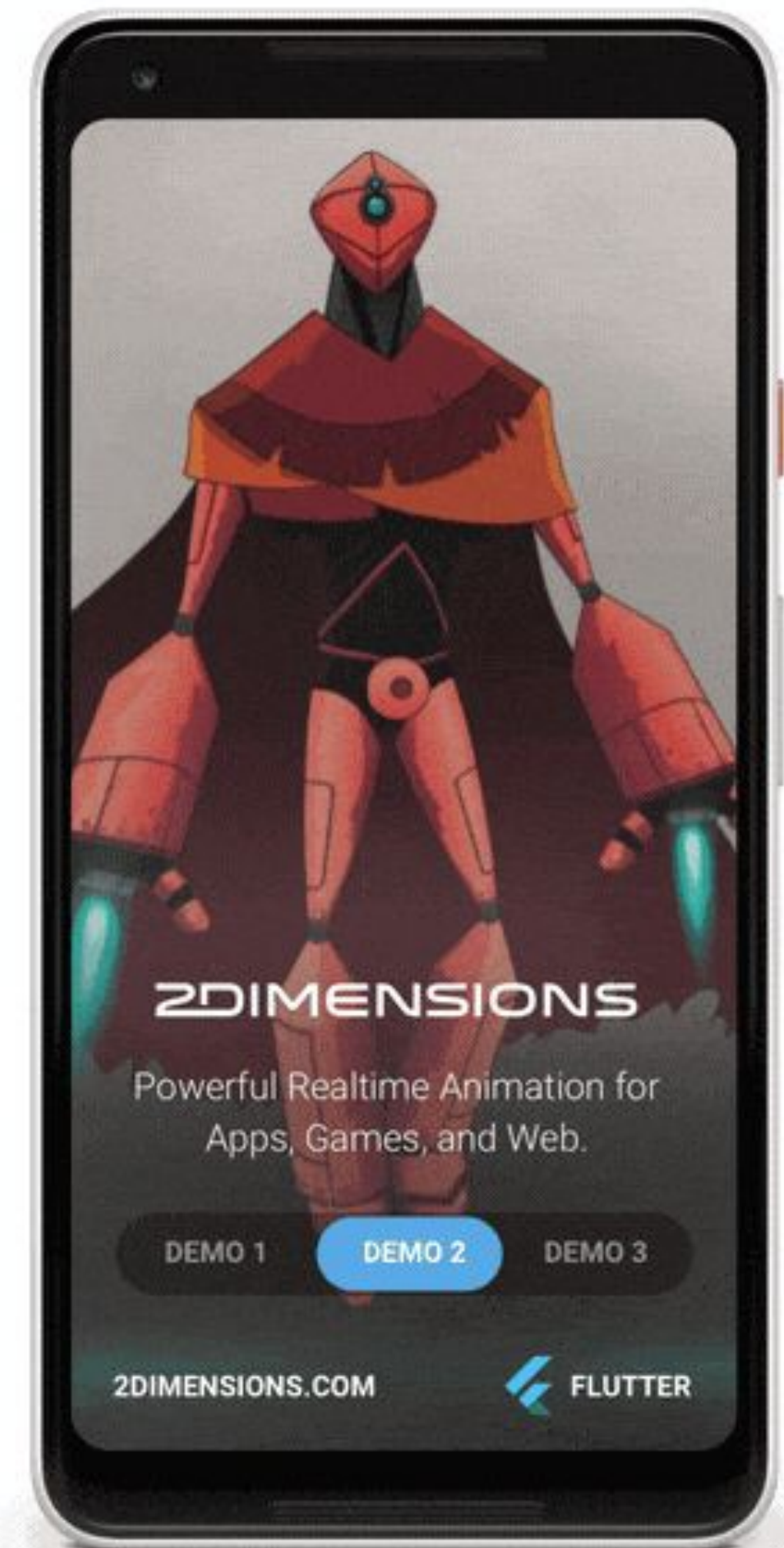


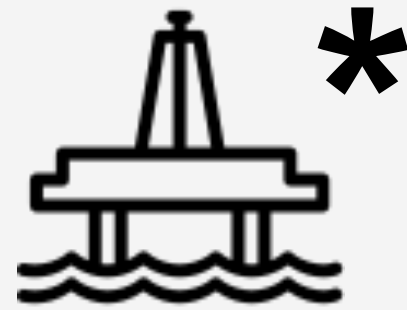


# Case Study: 2Dimensions

2D skeletal mesh animation rendered on-device in real-time

Seamlessly integrates with the Flutter framework widgets  
(seen here stacked on top of the animation)

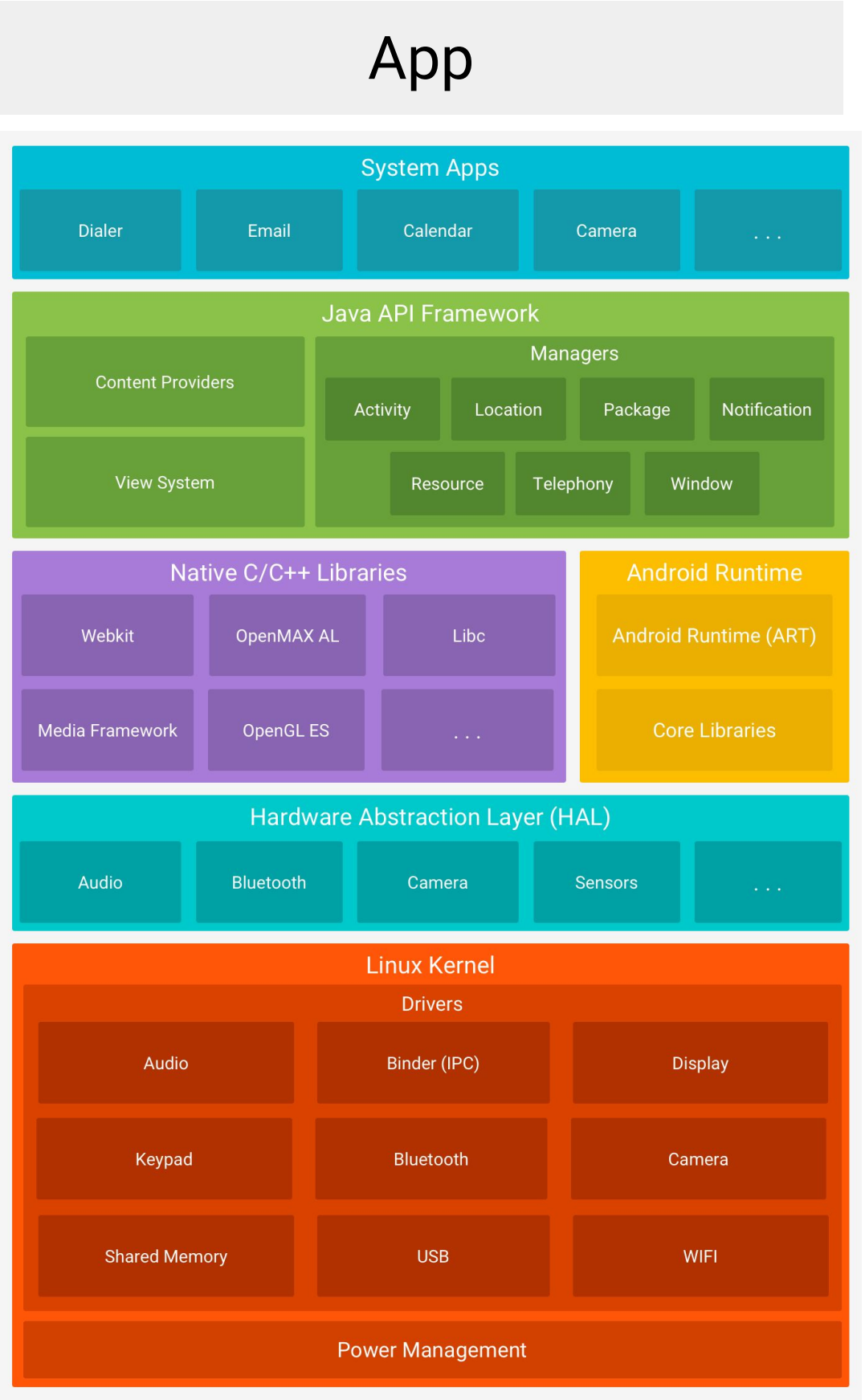




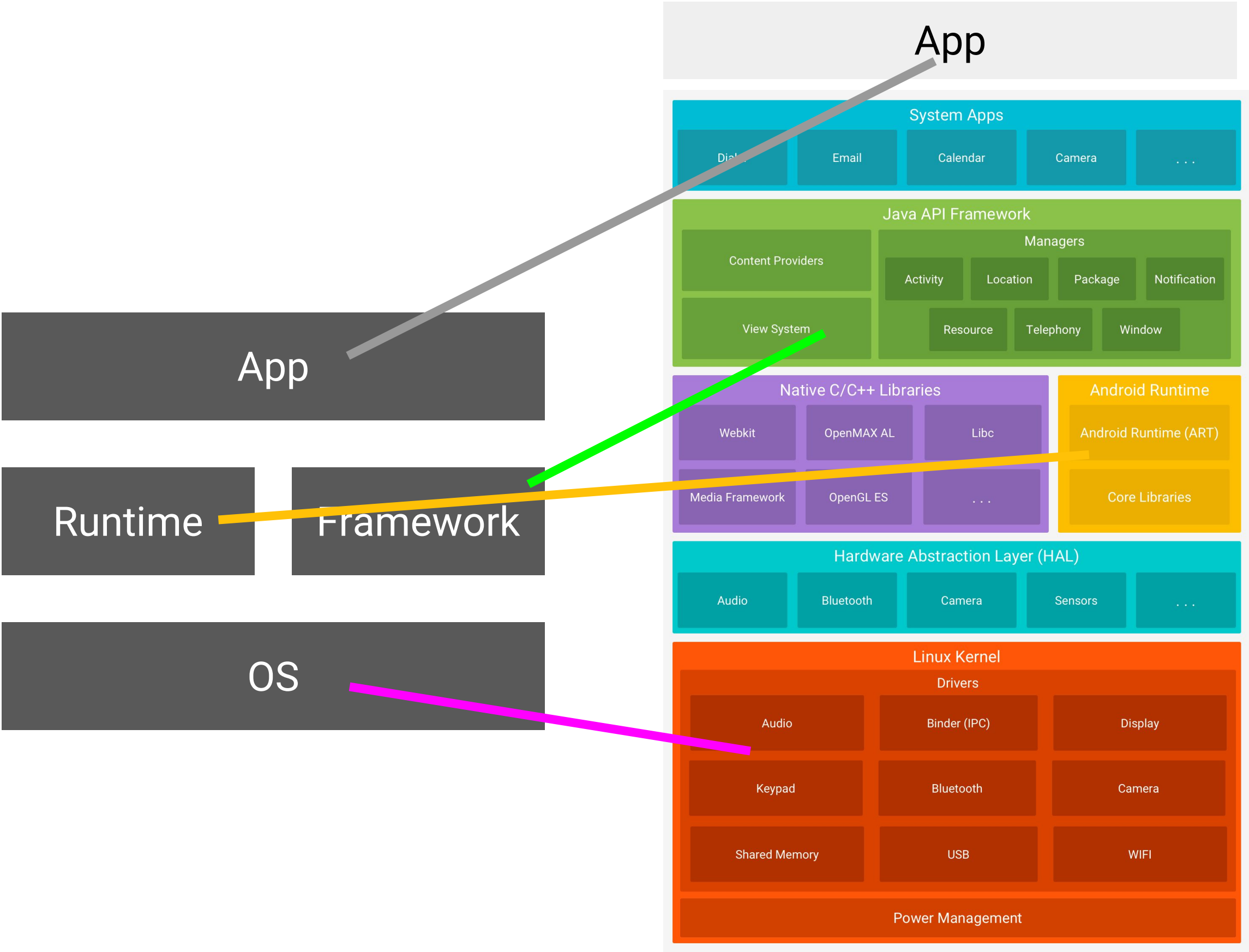
Let's step back a bit...

*Experimental*

# Native SDKs

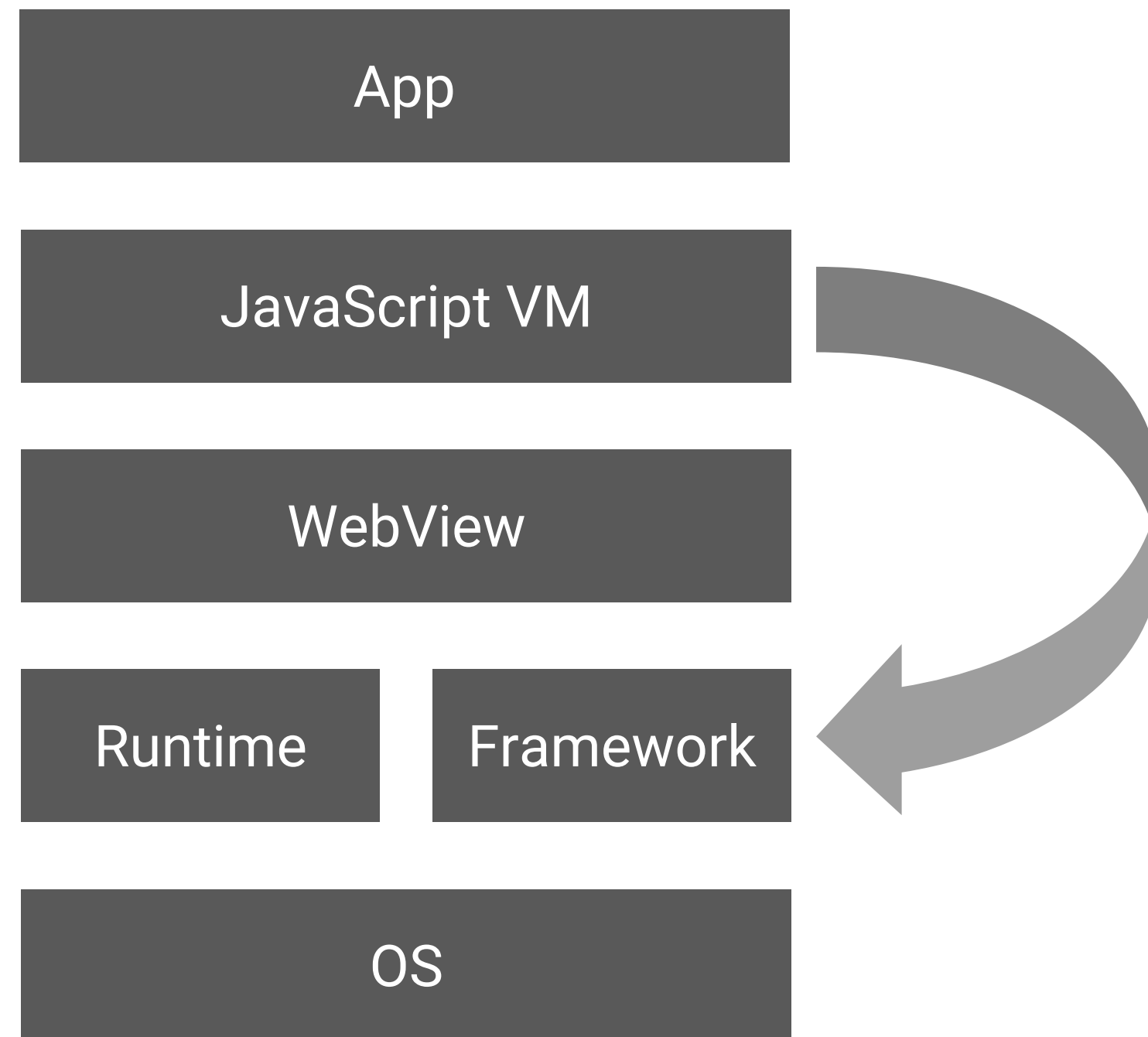


# Native SDKs

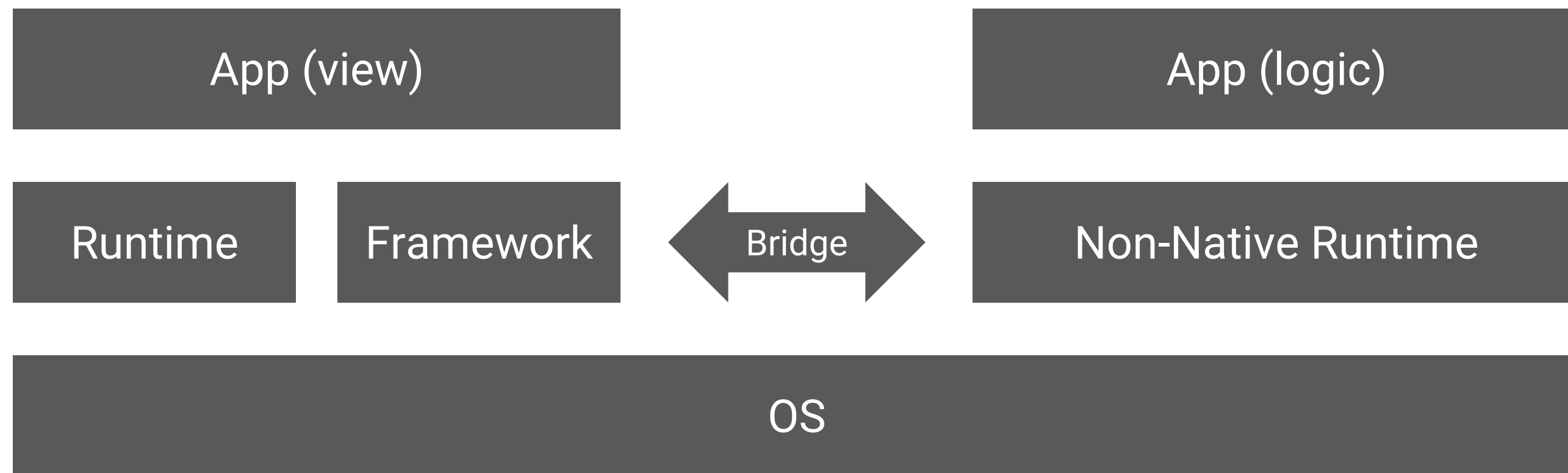




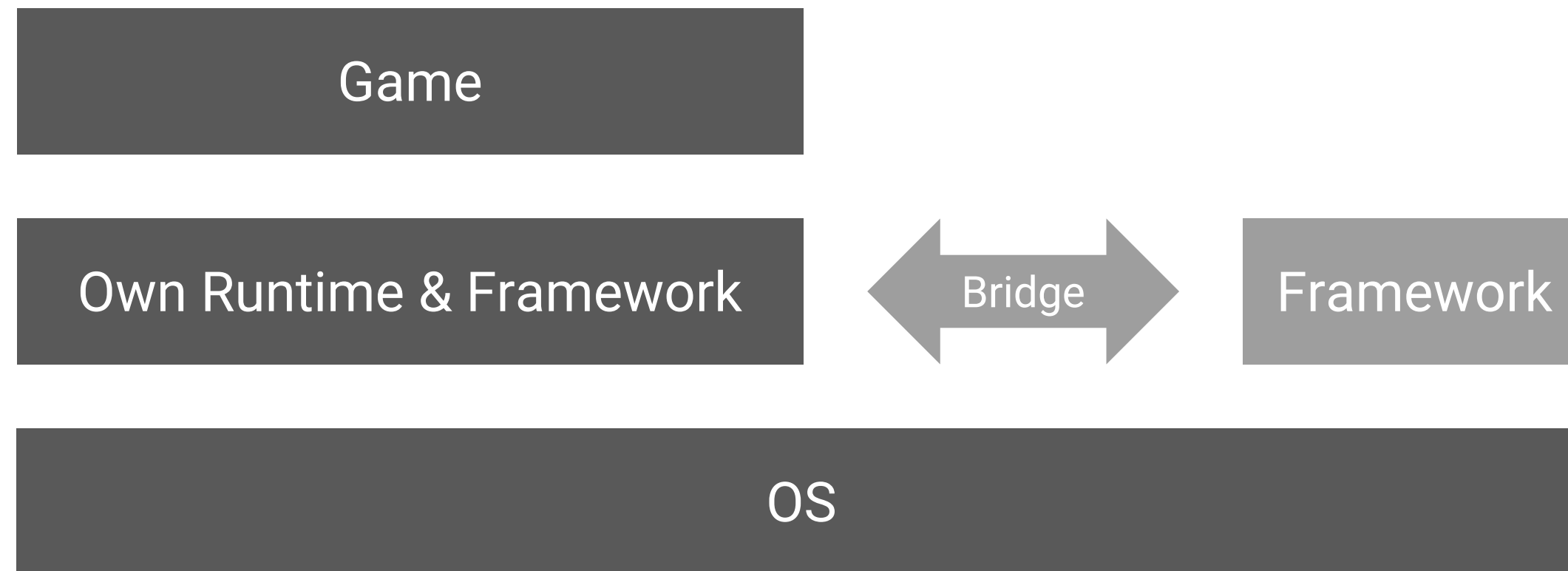
# WebView-based SDKs



# Other runtimes + native SDKs

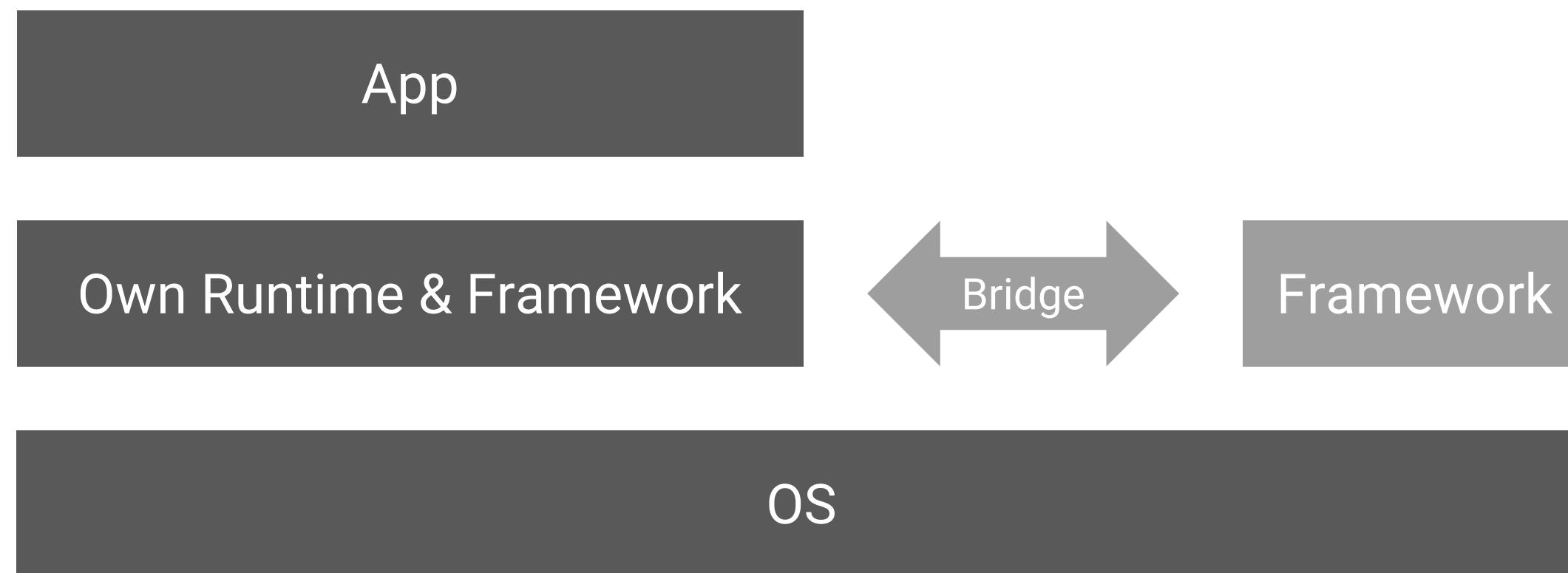


# The fourth option

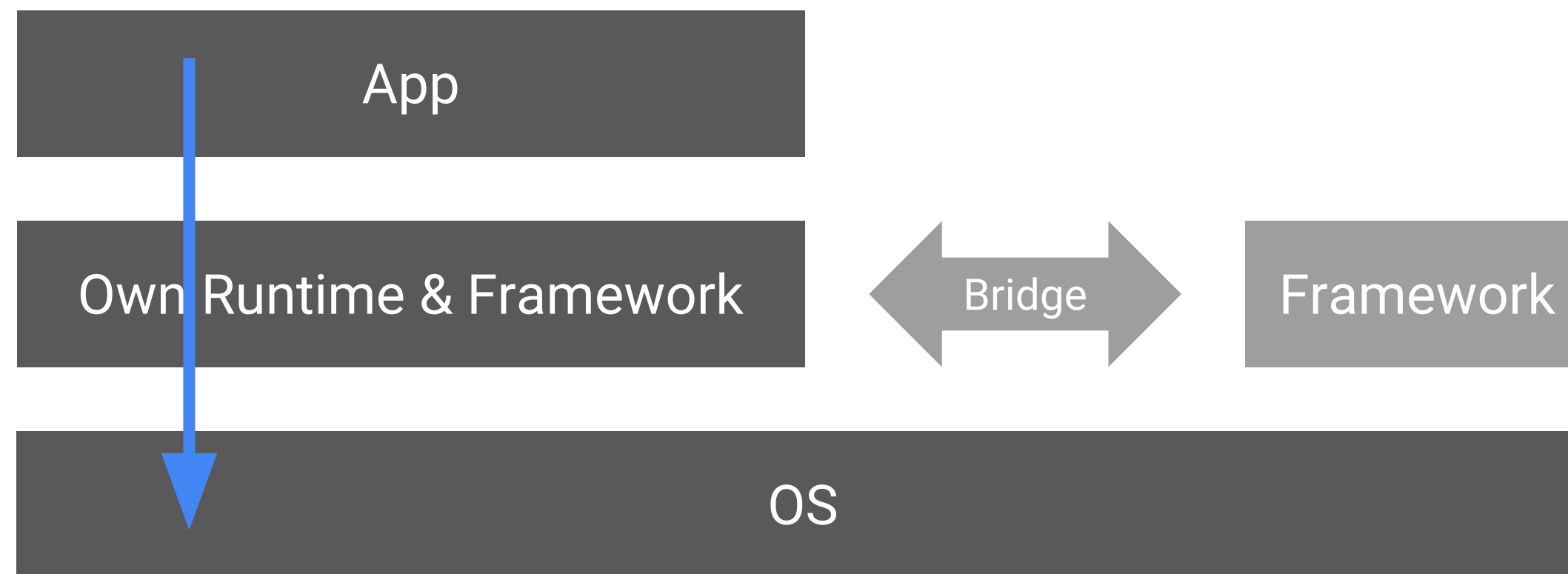




# The fourth option



# The fourth option

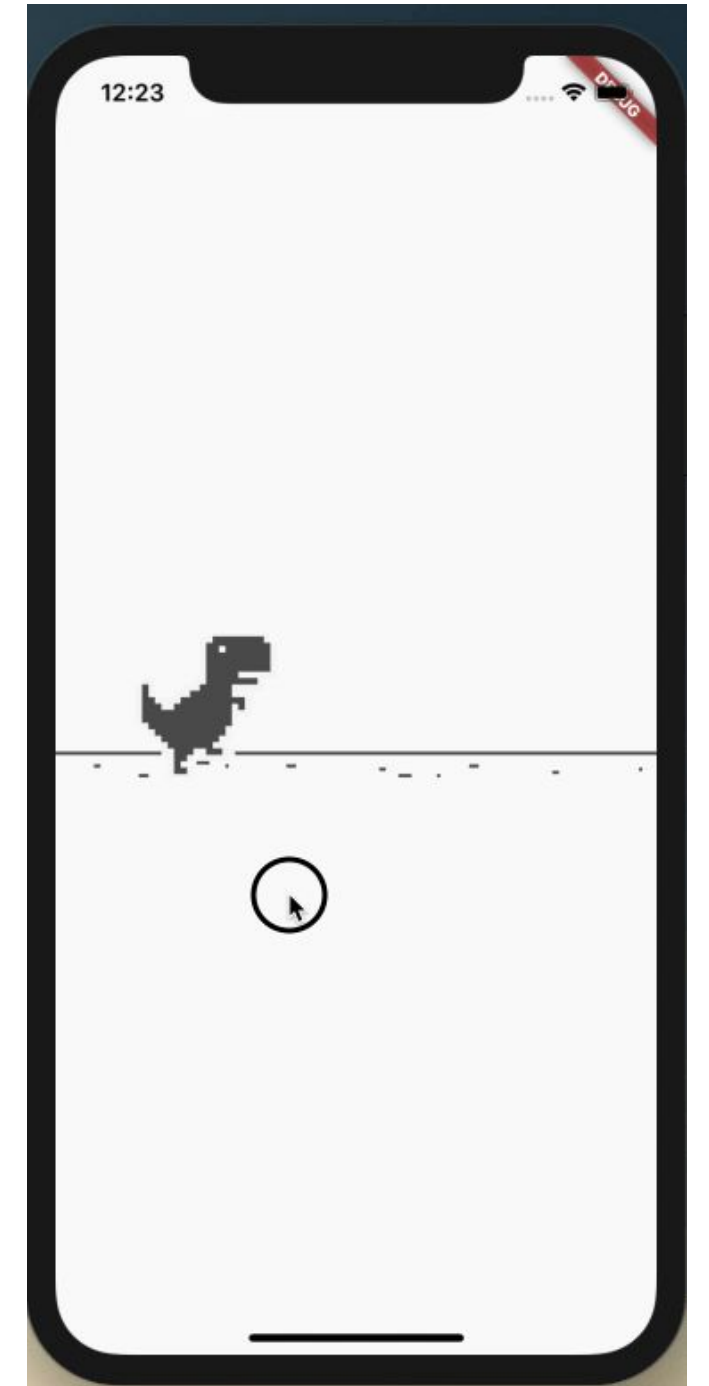


Direct path between UI and OS: Good **performance**

Very few dependencies between app and OS: Good **portability**

# Gaming engine & games

Flutter is an apps framework, but that doesn't seem to stop developers from building gaming engines and games with it.



<https://github.com/luanpotter/flame>

<https://medium.com/flutter-community/flutter-crush-debee5f389c3>

<https://medium.com/dextra-digital/creating-the-t-rex-game-with-flutter-and-flame-6d01add1ad5b>



# Wear OS

Variant of Android, so core framework just worked.

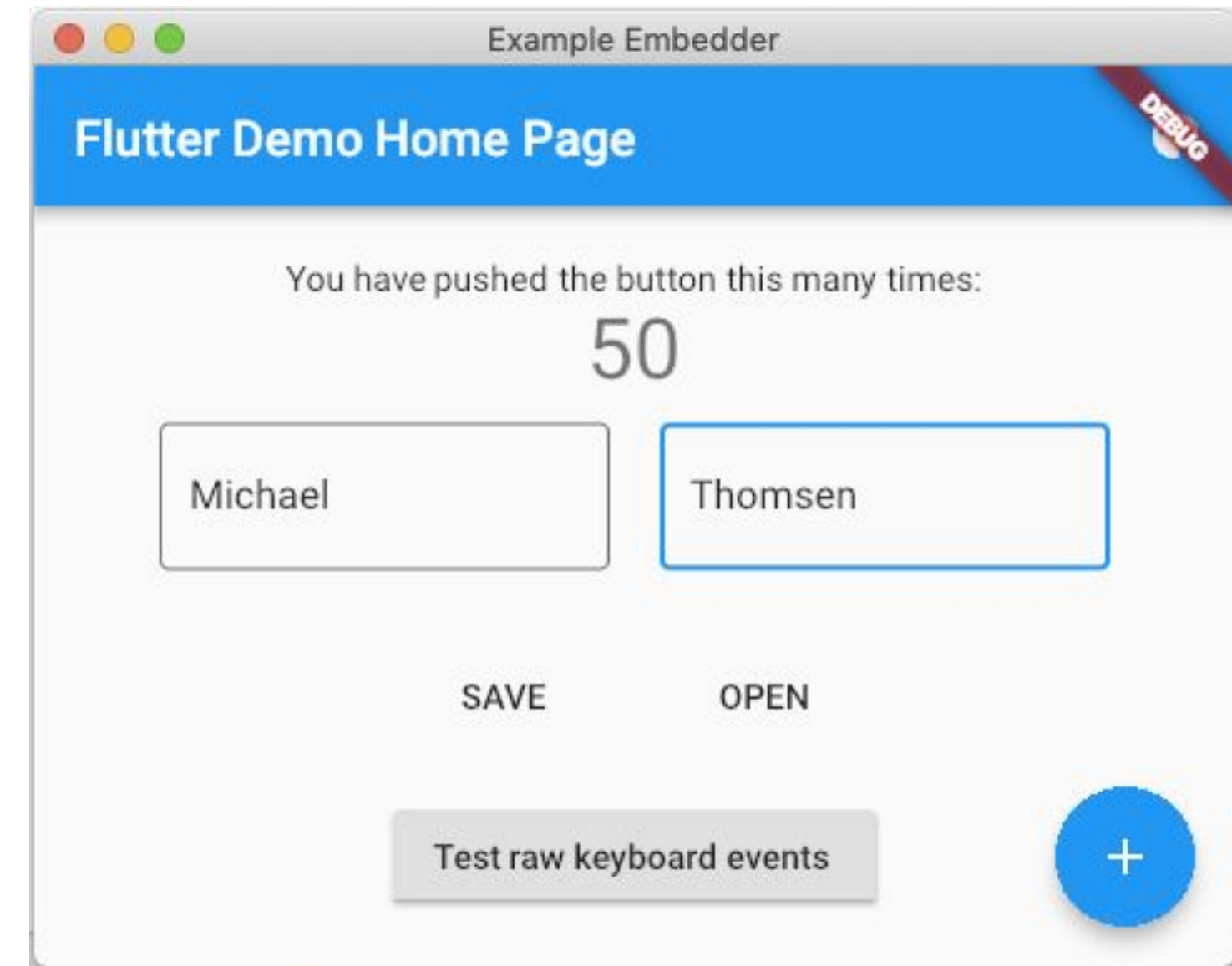
Some work to handle round screen,  
and integrate with Ambient mode.



# Desktop Embedding

Experimental implementation of the Embedding API for Windows, macOS, and Linux.

Base support for rendering, mouse/keyboard Input, and a few system APIs (e.g. Files).



# Flutter the technology vs Flutter the product

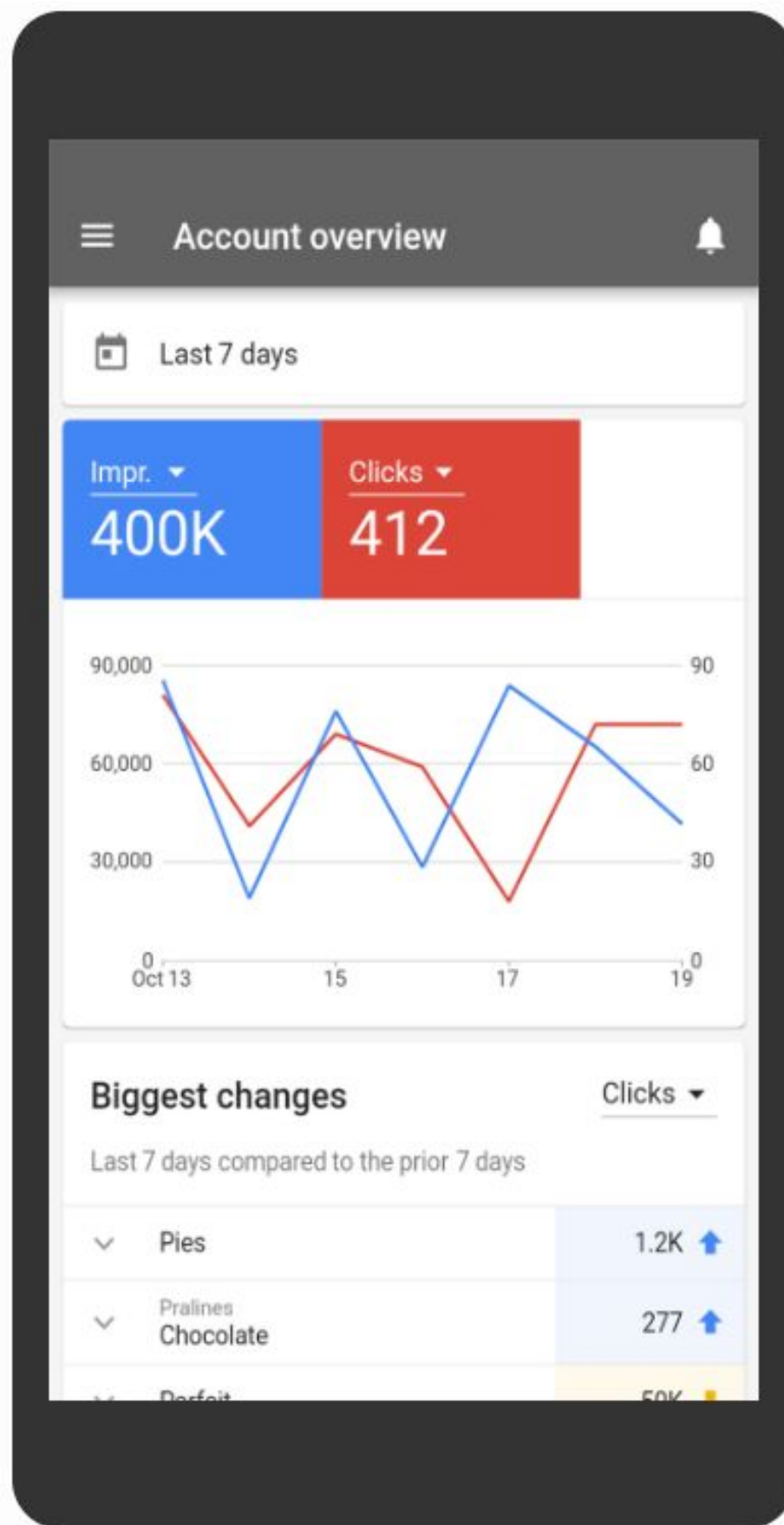
Flutter is currently focused on creating **a complete product** for developing great apps for iOS and Android.

However, Flutter **the technology is extremely flexible**, and there is nothing inherently mobile about the rendering...



# Customers & Community

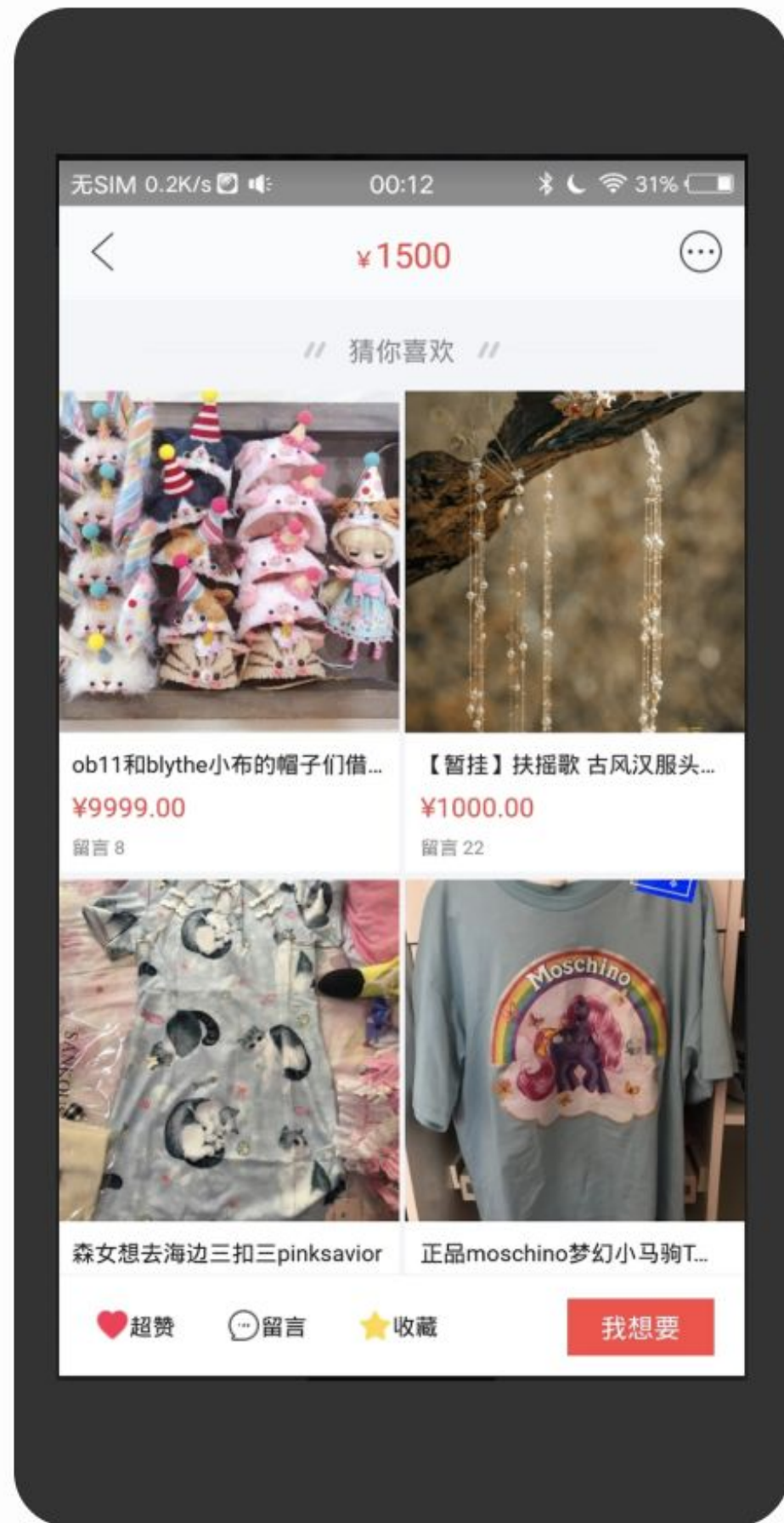




“Flutter provides a modern reactive framework that enabled us to *unify the codebase and teams* for our Android and iOS applications.

It's allowed the team to be much more productive, while still *delivering a native application experience* to both platforms. *Stateful hot reload has been a game changer* for productivity”

Sridhar Ramaswamy  
SVP, Google Ads and Commerce



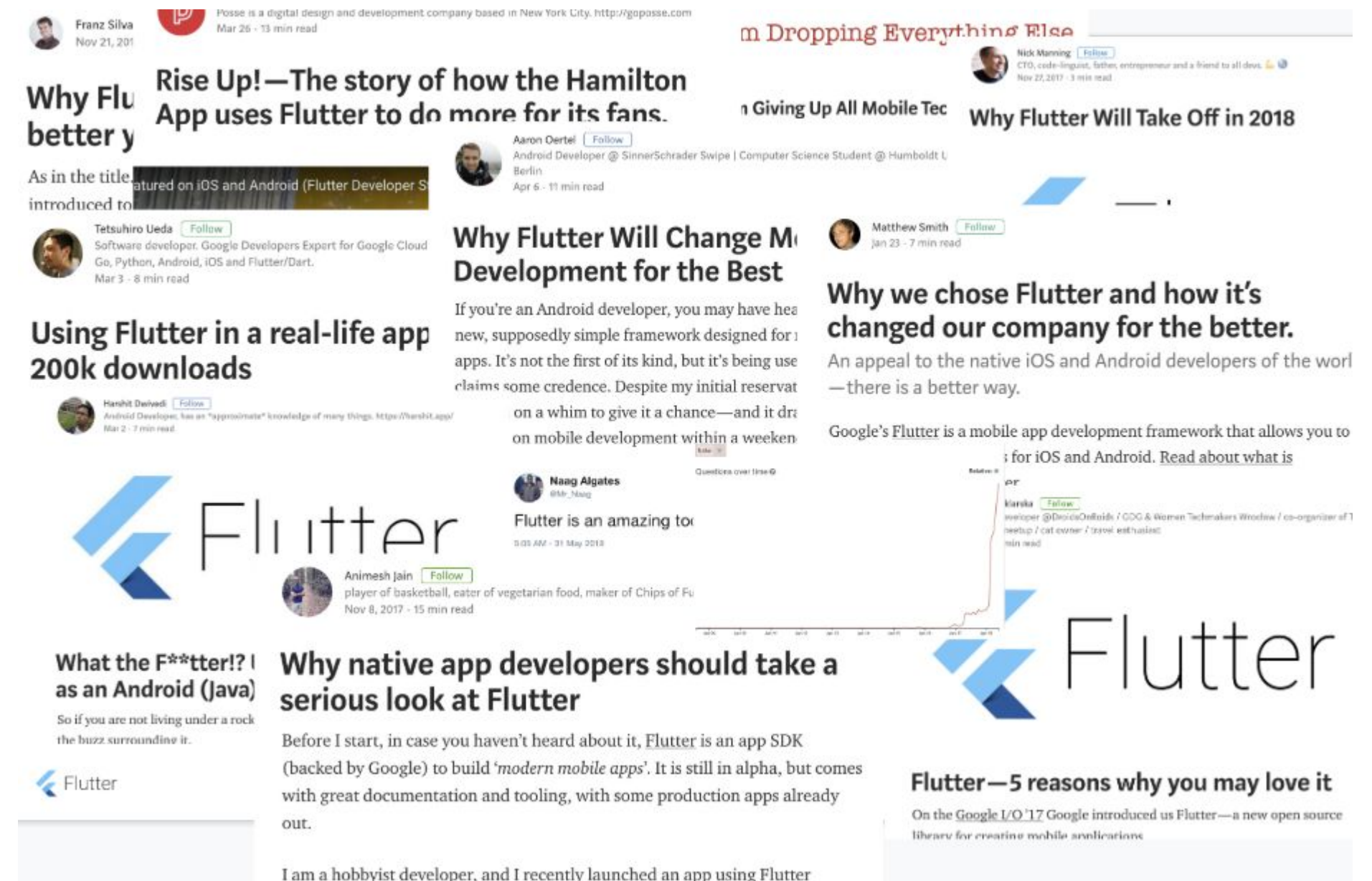
Alibaba's app incorporates Flutter to power parts of their app.

Deployed to **millions of devices**.

Alibaba praises Flutter for its consistency across platforms, the ease of generating UI code from designer redlines, and the **ease with which their native developers have learned Flutter**.

# Flutter Momentum

- 17x increase in active developers since beta 1 @ MWC'18
- Open source (250+ contributors), BSD license
- 41,016 Github stars (top-50!)
- 2500+ apps in Play Store
- [startflutter.com](https://startflutter.com), [flutter.rocks](https://flutter.rocks), [flutter.institute](https://flutter.institute), and more





# Four ways to use Flutter today

Start a new app  
from scratch

Build your new idea in  
Flutter, and reach  
both iOS and Android  
at the same time.

Prototype a new  
app idea

Use Flutter to test  
out an app concept  
or idea in record  
time.

Bring your app  
to the the *other*  
platform

You already have an  
iOS or Android app?  
Use Flutter to build  
for the other  
platform. Combine  
codebases when  
you've proven your  
Flutter app.

Use Flutter for a  
part of your app

Test Flutter in  
production with one  
or two screens in  
your existing app.



# Summary

- Three core pillars of Flutter
  - High-Velocity Development
  - Expressive and Flexible Toolkit
  - Native iOS and Android apps
- Very different architecture
  - Code compiles to native ARM code
  - UI is rendered with custom rendering framework
- Very portable, very fast, very extensible



# Flutter

# Thank you!

More information about Flutter:

<http://flutter.io>

<http://medium.com/flutter-io>