

Flutter

How to make a
beautiful app in an hour?

Paulina Szklarska

Android Developer

Flutter Developer

Toast Wrocław (Android) meetups co-org

GDG Wrocław co-org

Women Techmakers Wrocław lead

cat owner

travel enthusiast



@pszklarska



@pszklarska



@p_szklarska

Agenda

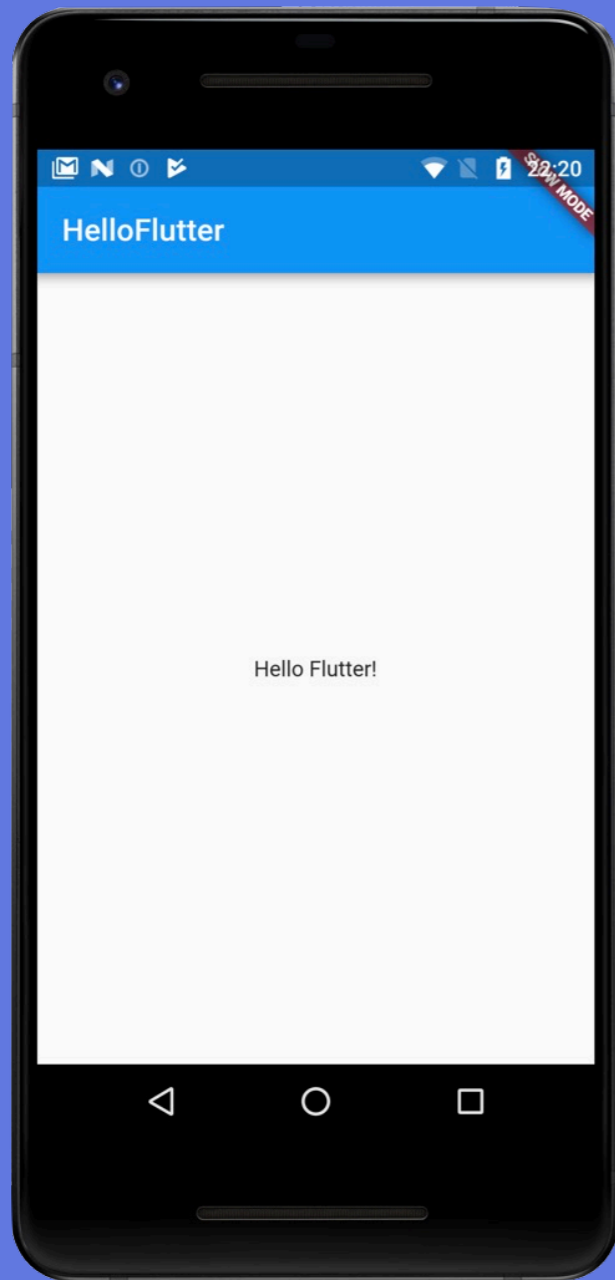
1. What is Flutter?
2. How is Flutter different from other solutions?
3. How to make an app?
4. How does it work?
5. Cool, but... What about X/Y/Z?

What is Flutter?

What is Flutter?

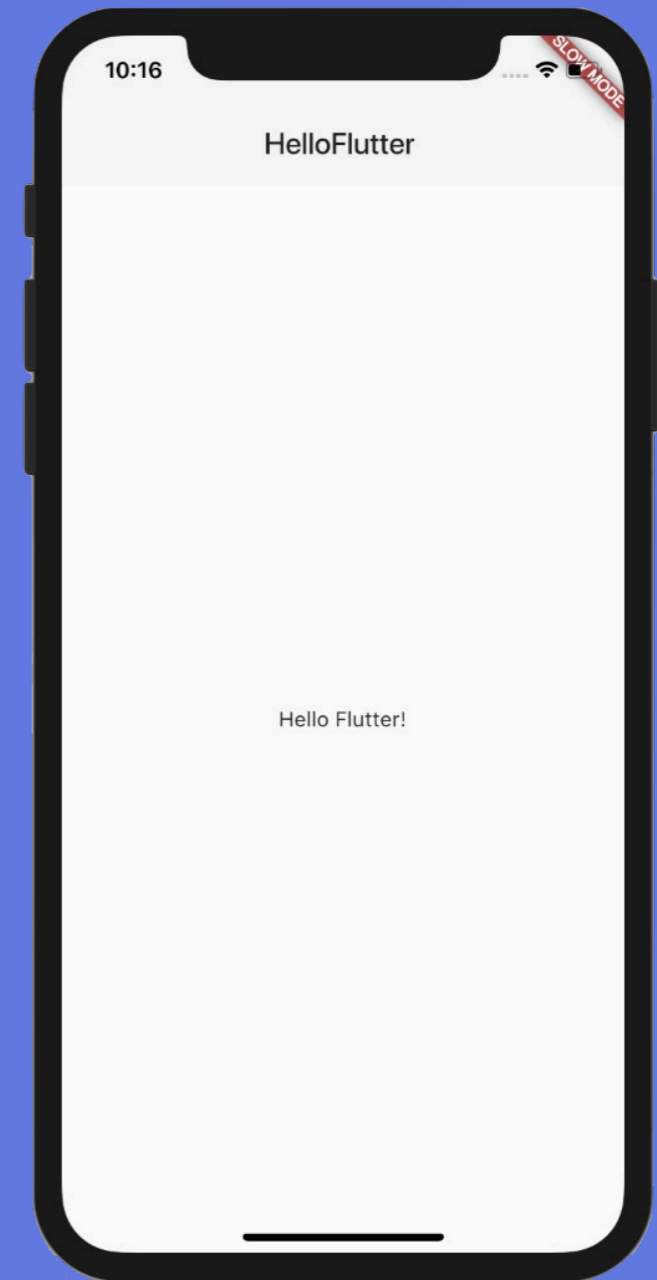
Flutter is Google's mobile UI framework for **crafting** high-quality native interfaces on iOS and Android in record time.

What is Flutter?



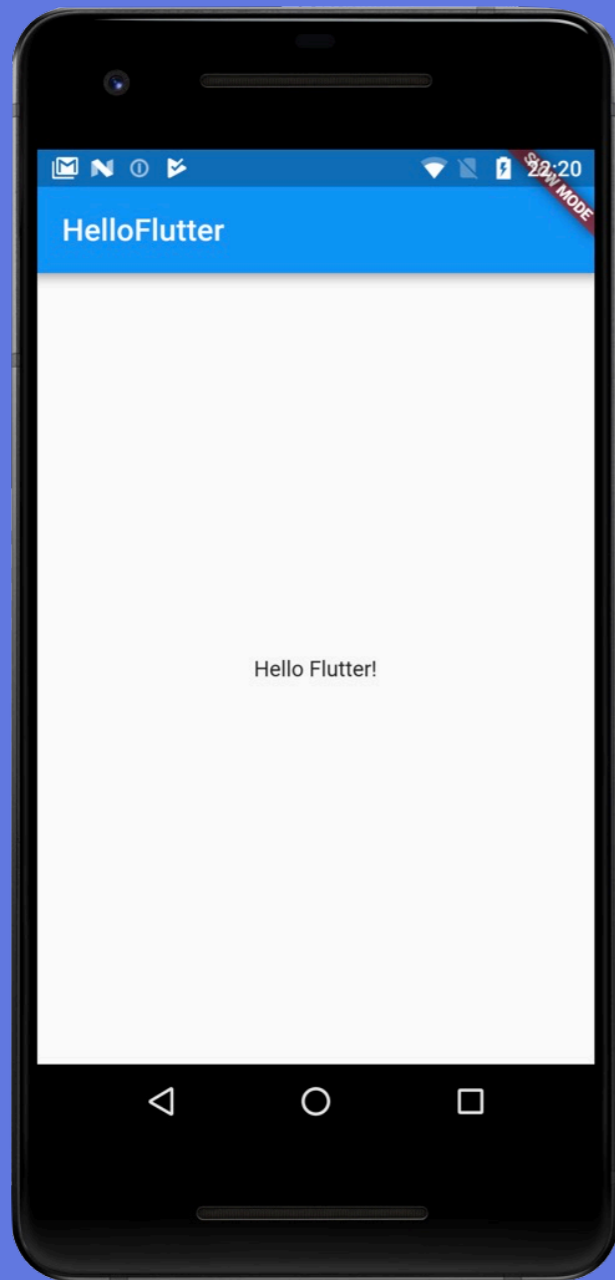
Android

Flutter



iOS

What is Flutter?

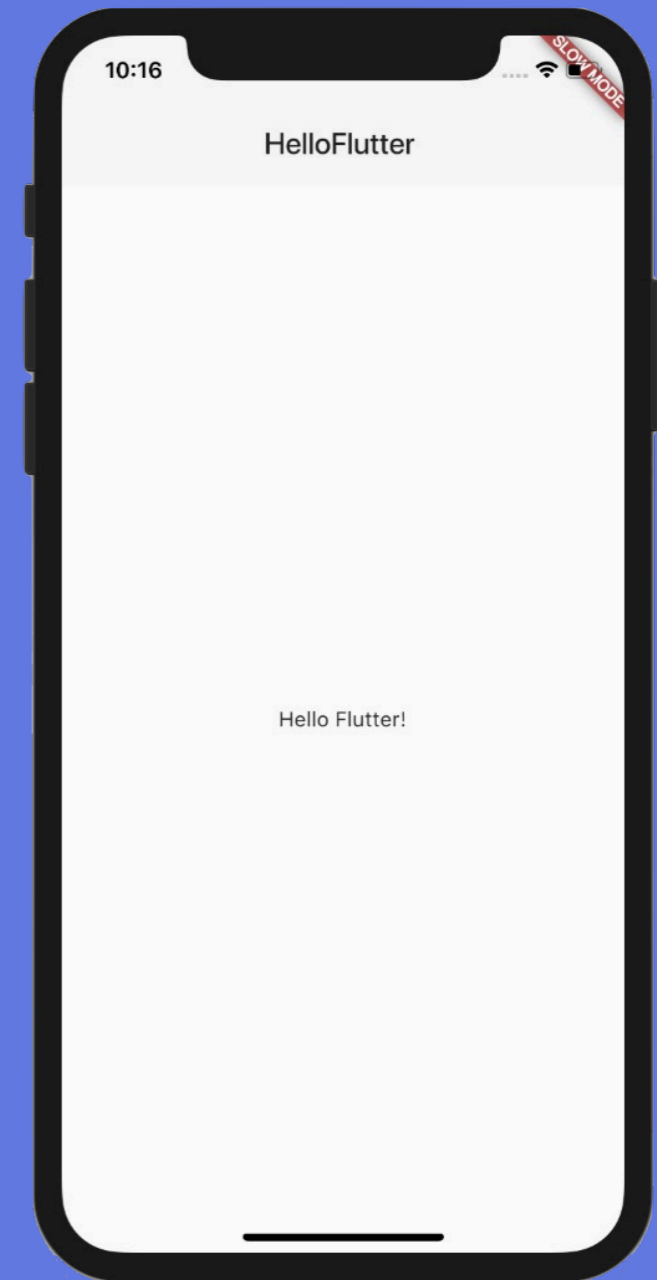


Android

Flutter

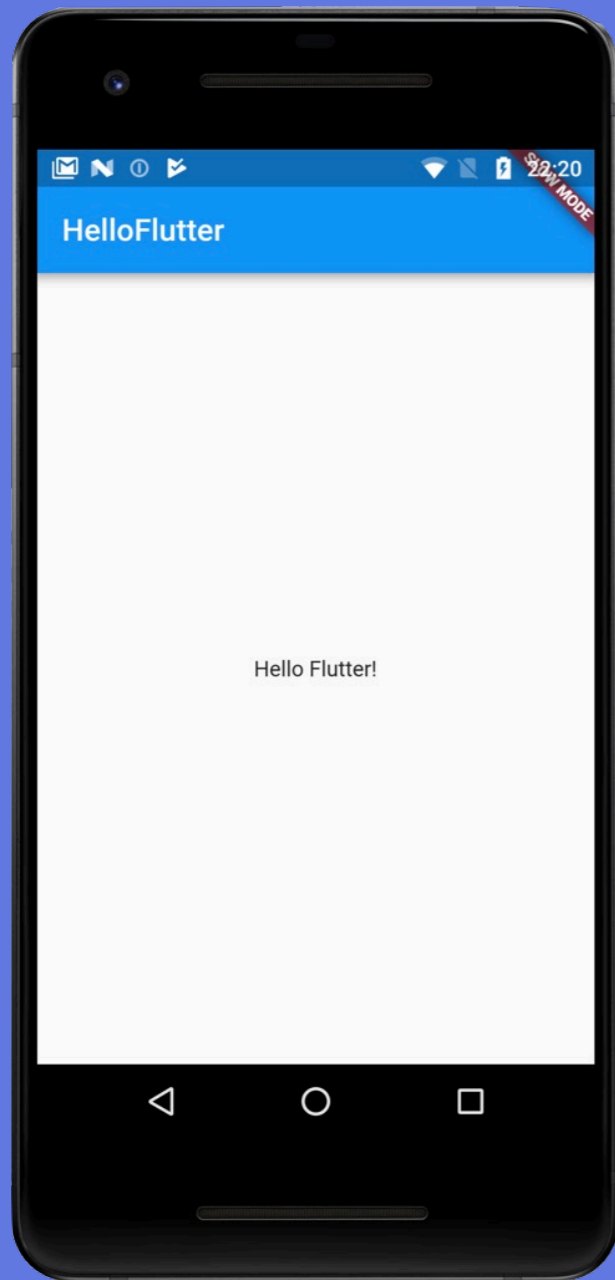


Dart



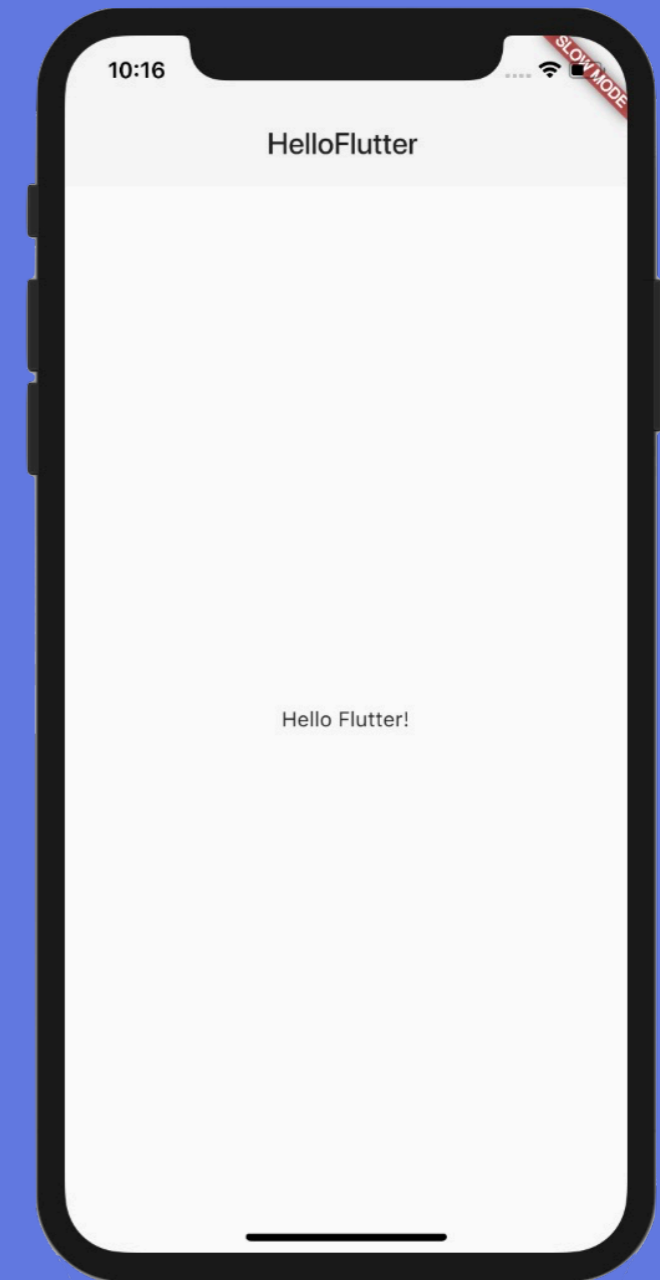
iOS

What is Flutter?



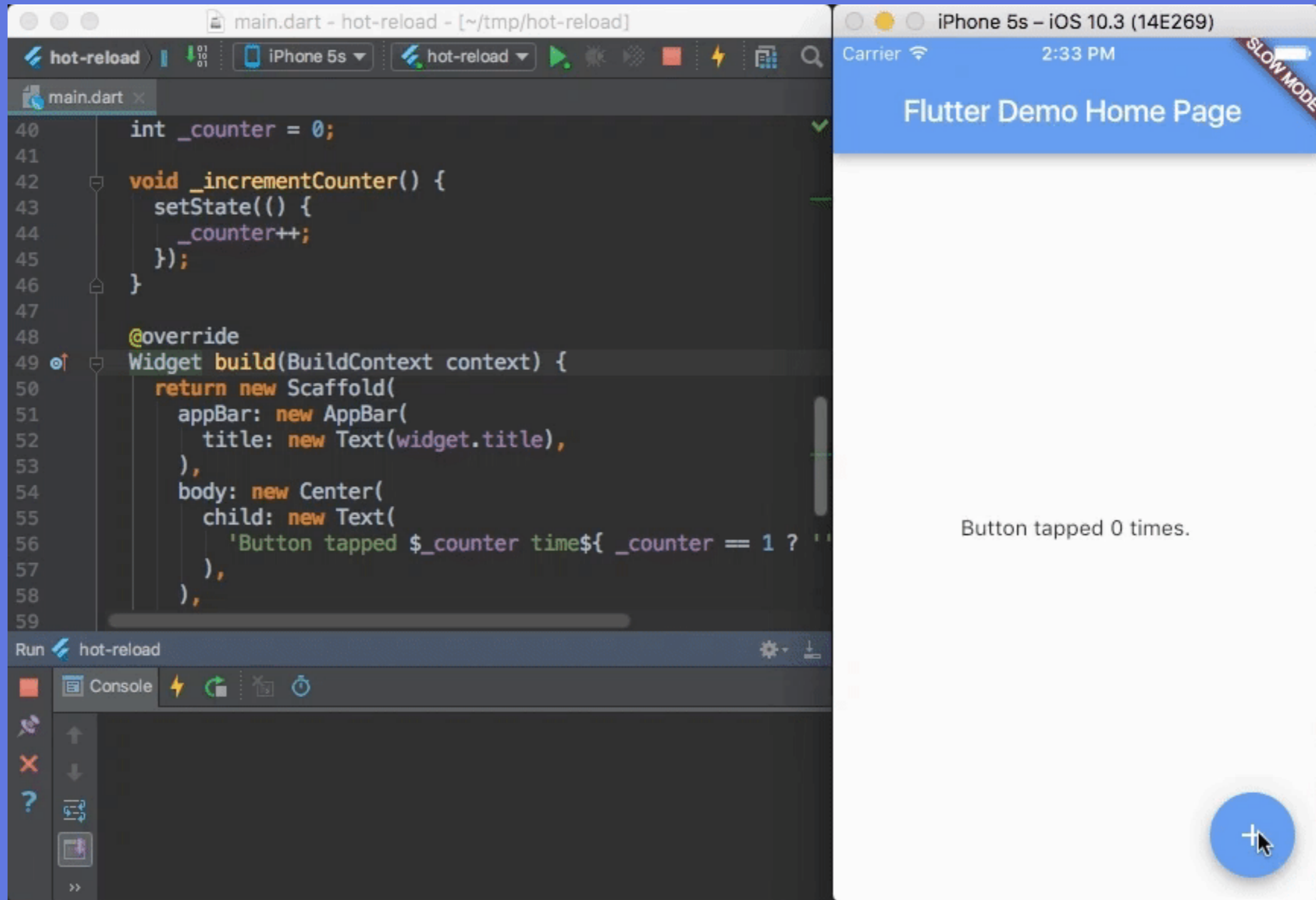
Android

Flutter
1.0

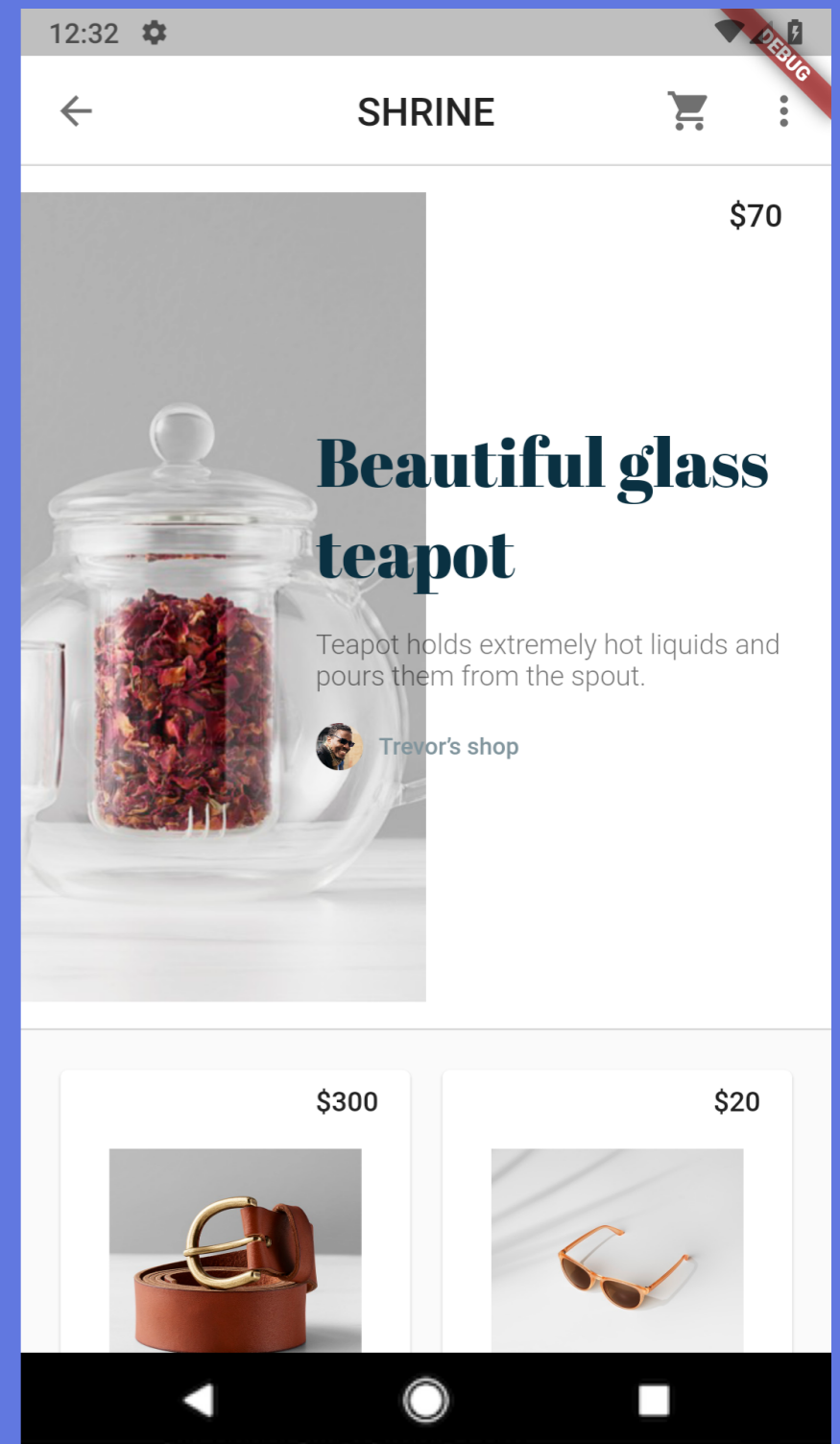
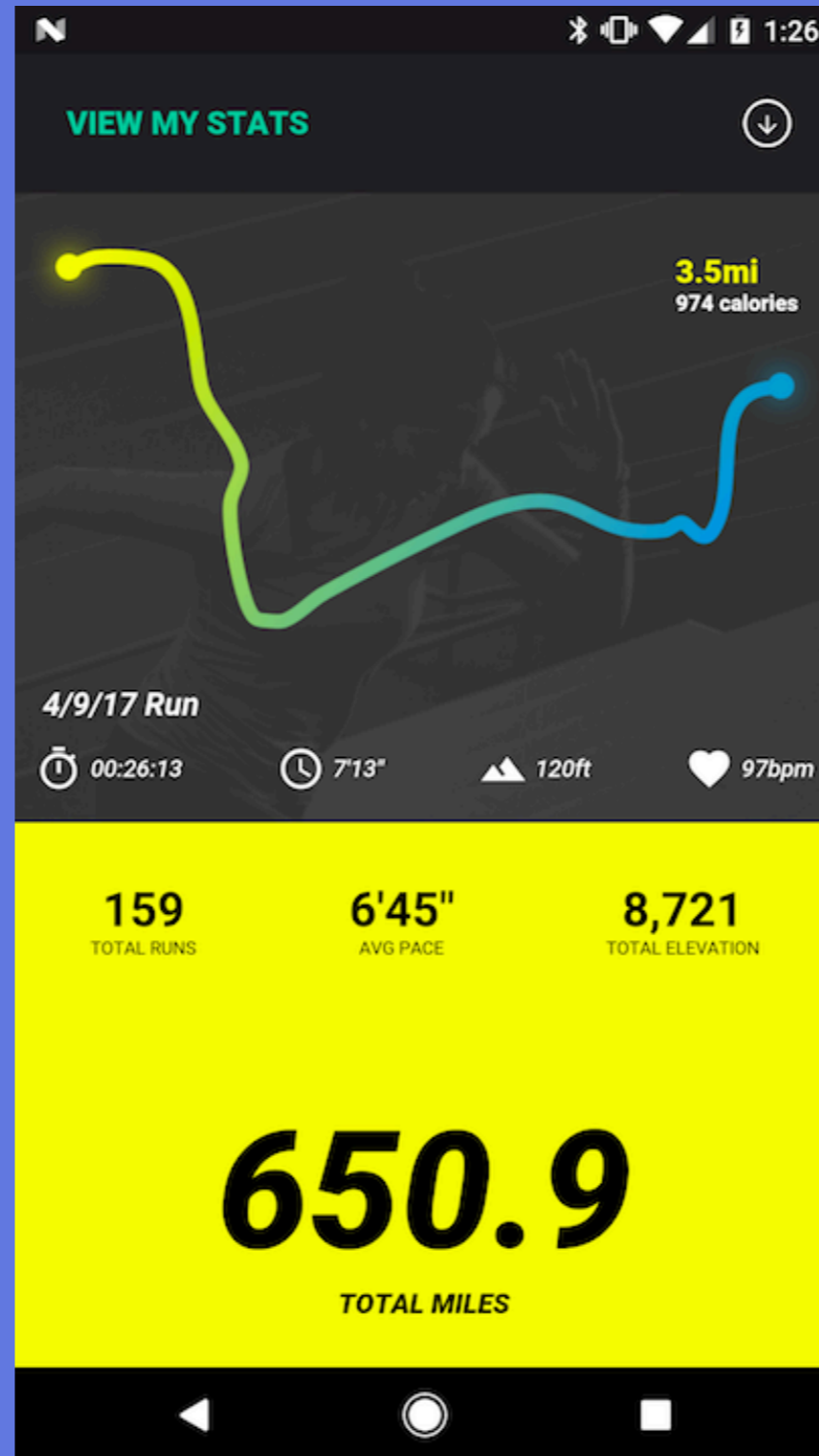
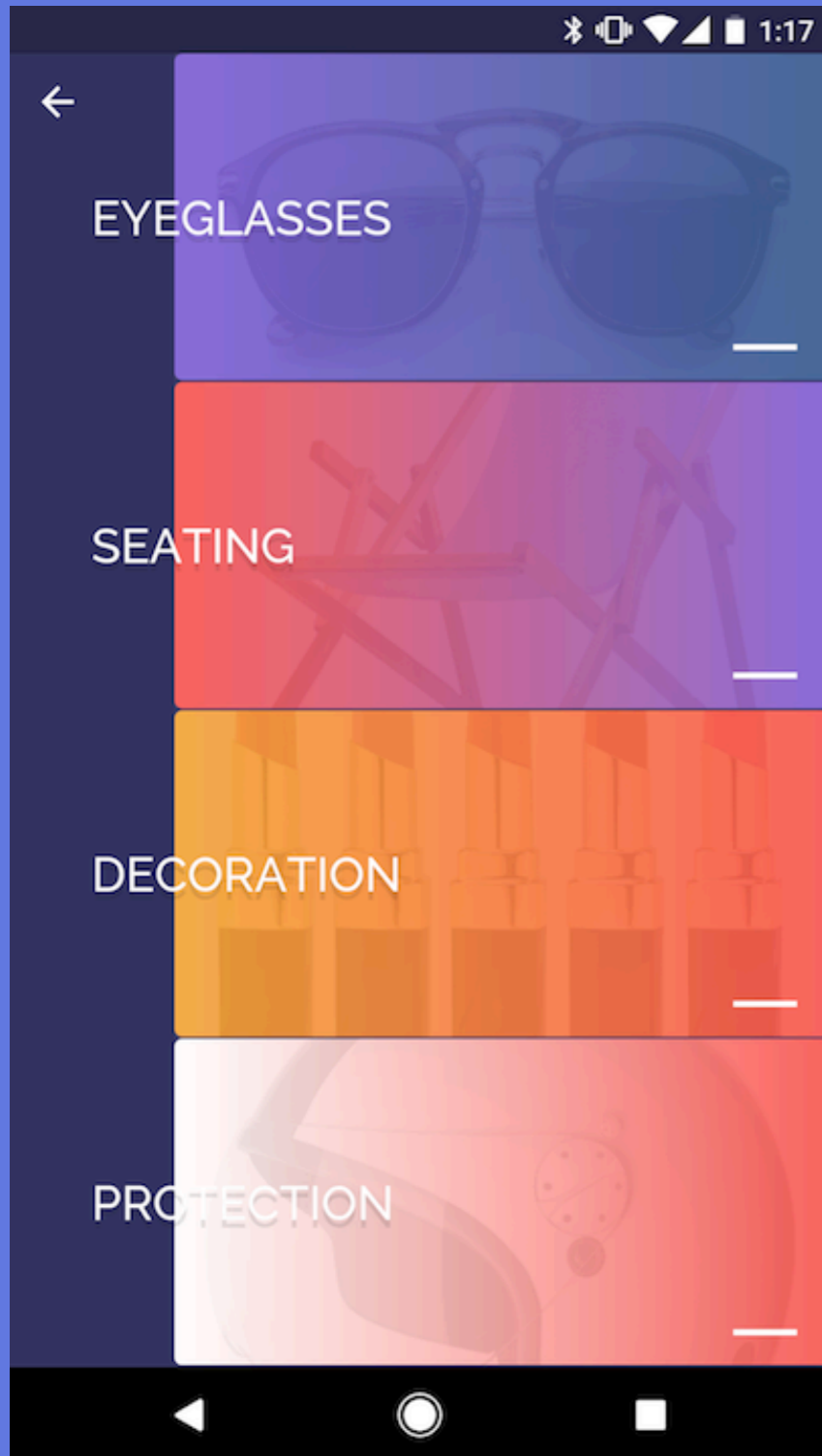


iOS

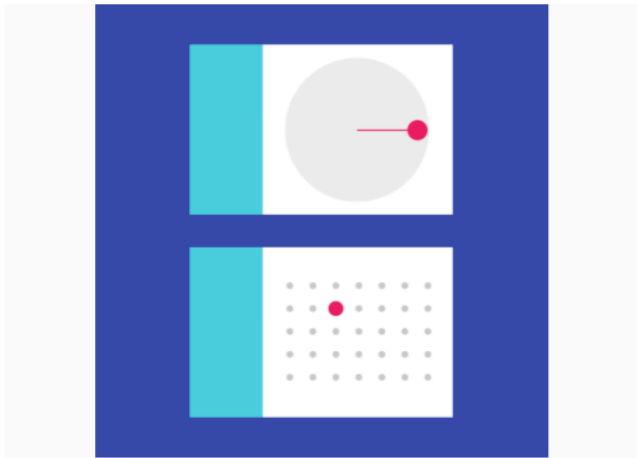
Fast development



Beautiful UI



Native Performance



Date & Time Pickers

Date pickers use a dialog window to select a single date on mobile. Time pickers use a dialog to select a single time (in the hours:minutes format) on mobile.

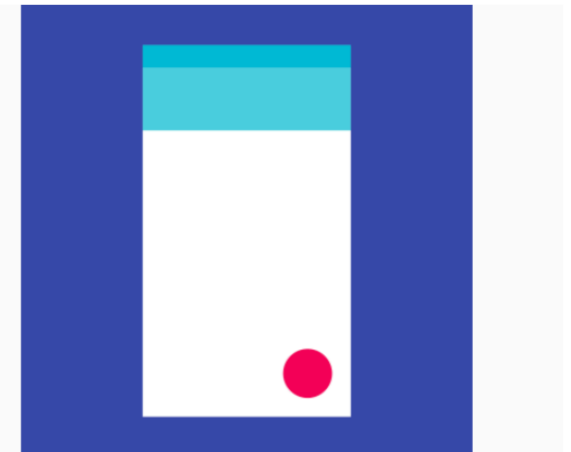
[Documentation](#)



TabBar

A Material Design widget that displays a horizontal row of tabs.

[Documentation](#) , [Samples](#)

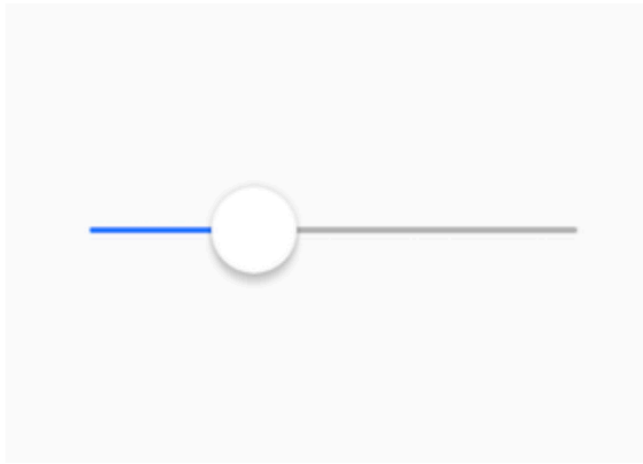


FloatingActionButton

A floating action button is a circular icon button that hovers over content to promote a primary action in the application. Floating action buttons are...

[Documentation](#)

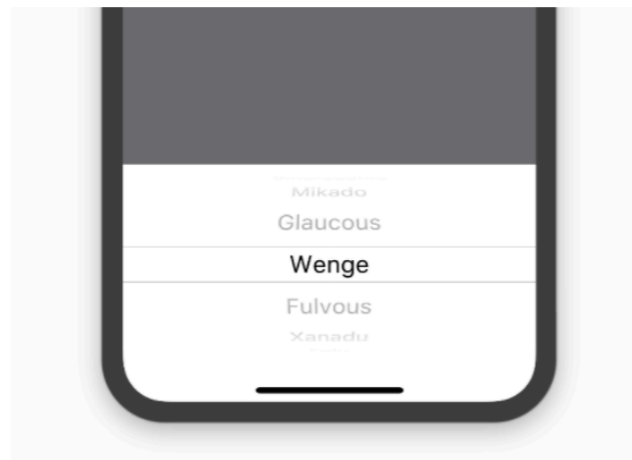
Native Performance



CupertinoSlider

Used to select from a range of values.

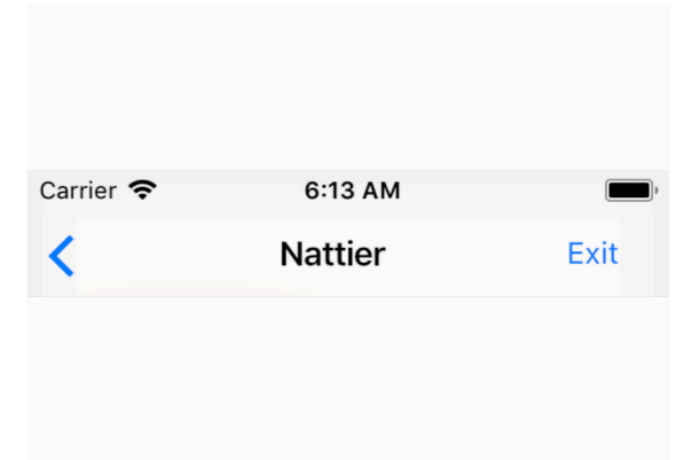
[Documentation](#)



CupertinoPicker

An iOS-style picker control. Used to select an item in a short list.

[Documentation](#)



CupertinoNavigationBar

An iOS-style top navigation bar. Typically used with CupertinoPageScaffold.

[Documentation](#)

How is Flutter different?

How is Flutter different?

Application Code (JS)

React Native

How is Flutter different?

Application Code (JS)

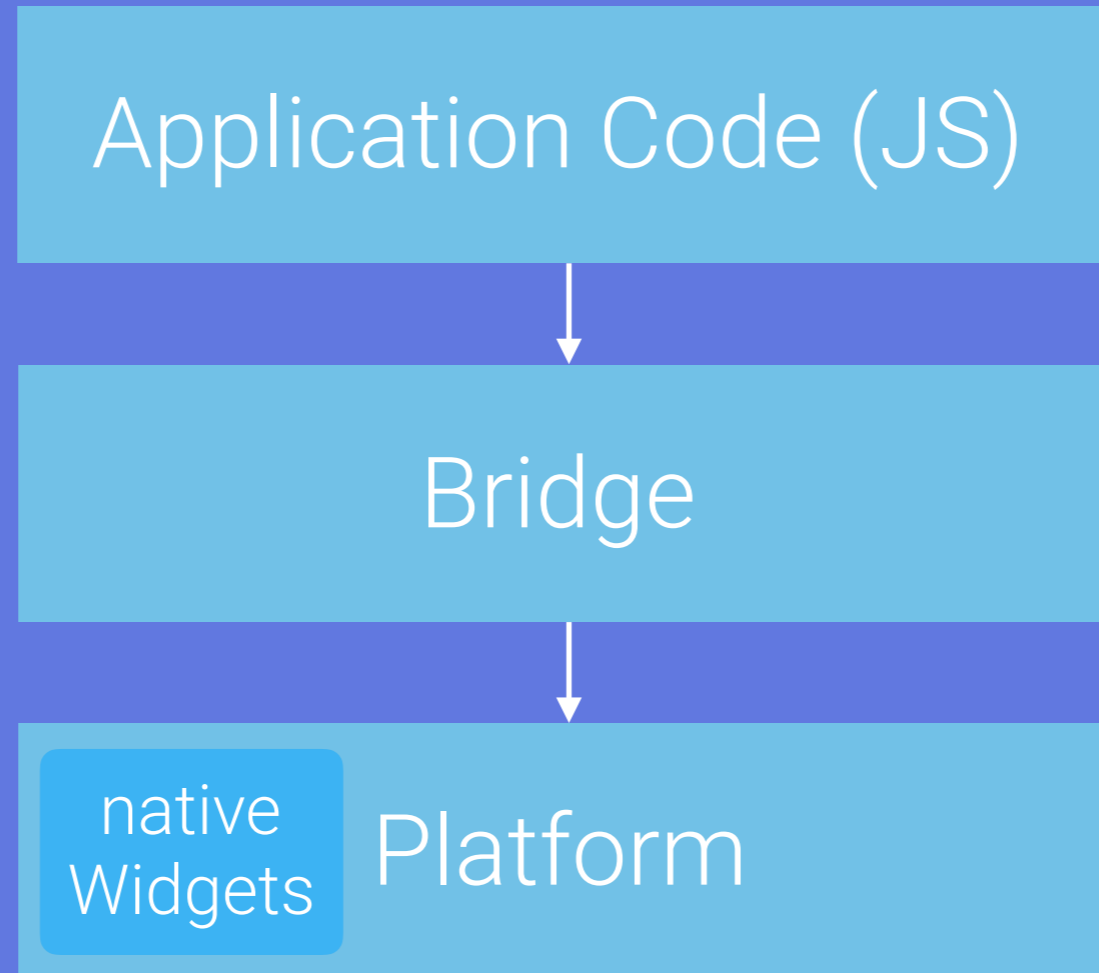
rendering

native
Widgets

Platform

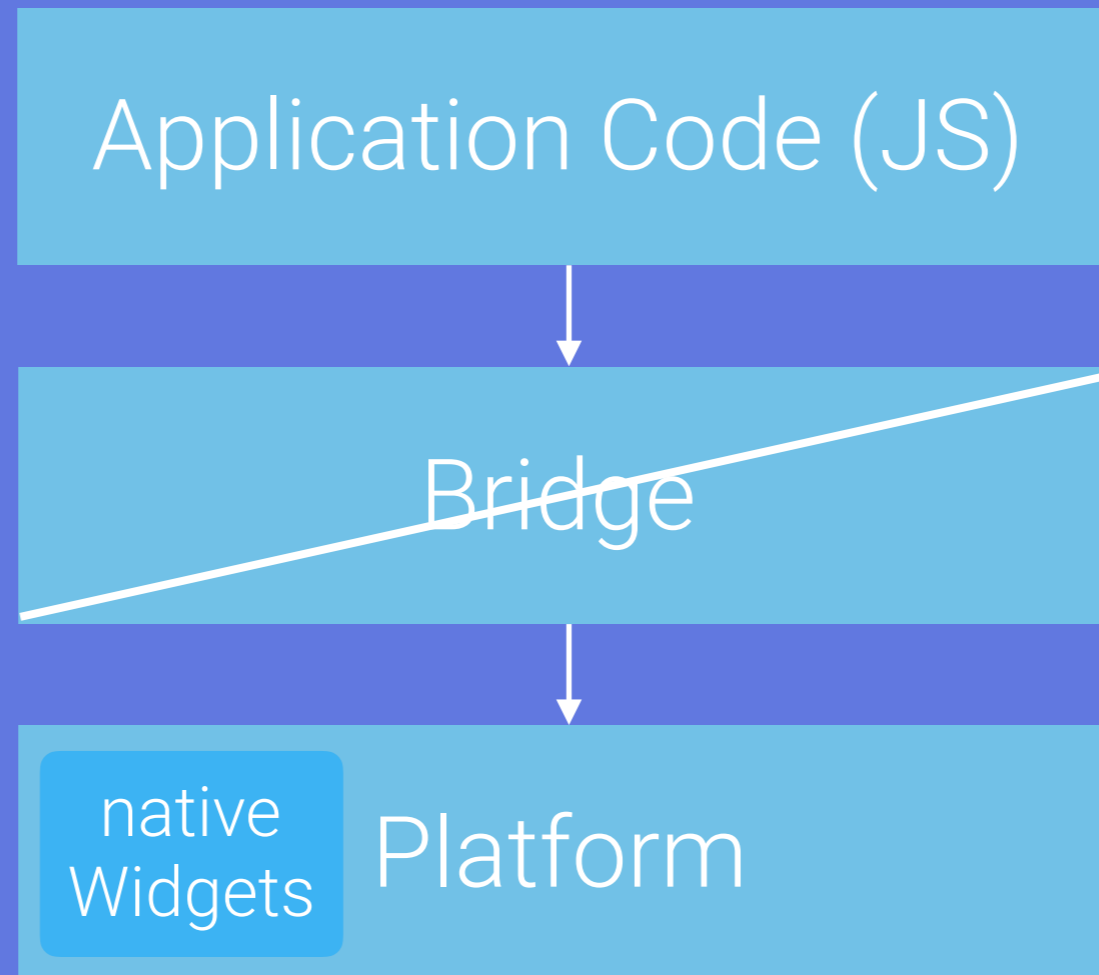
React Native

How is Flutter different?



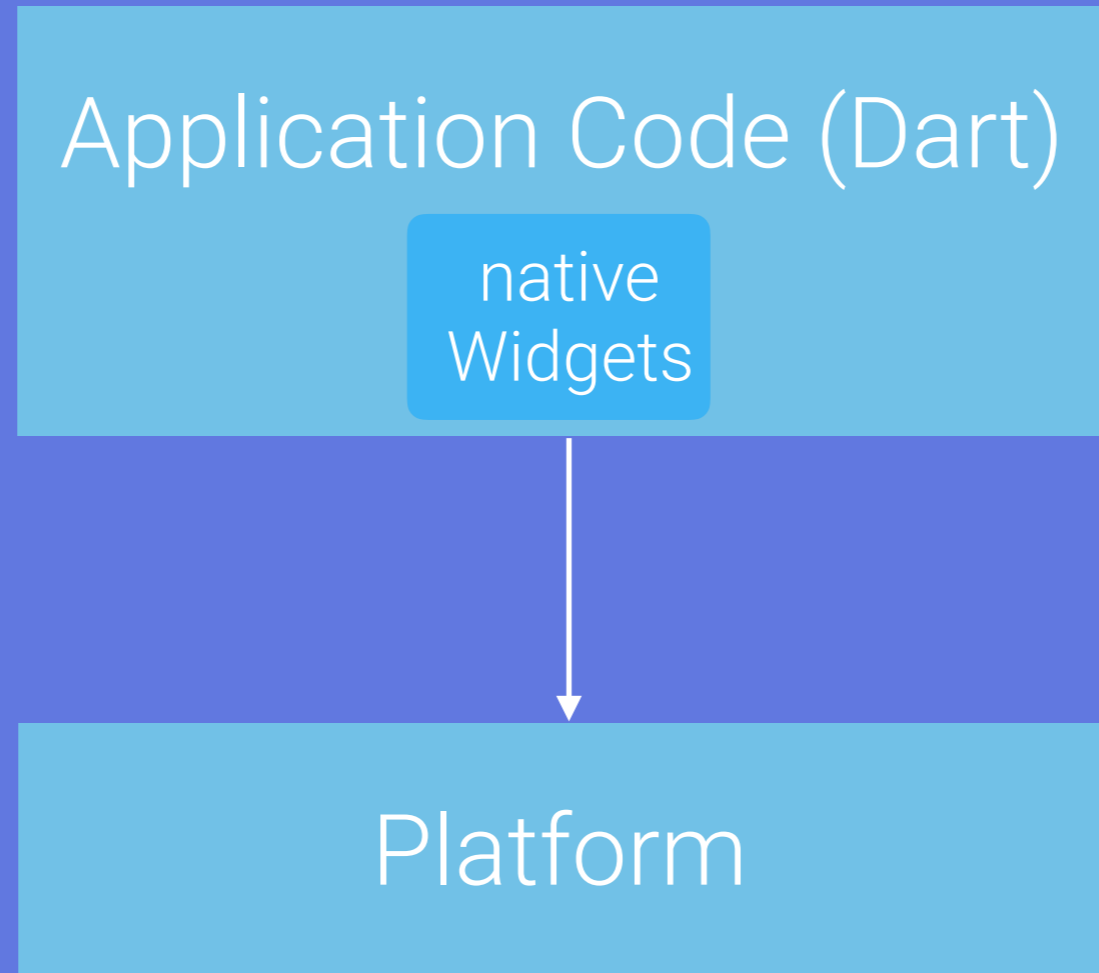
React Native

How is Flutter different?



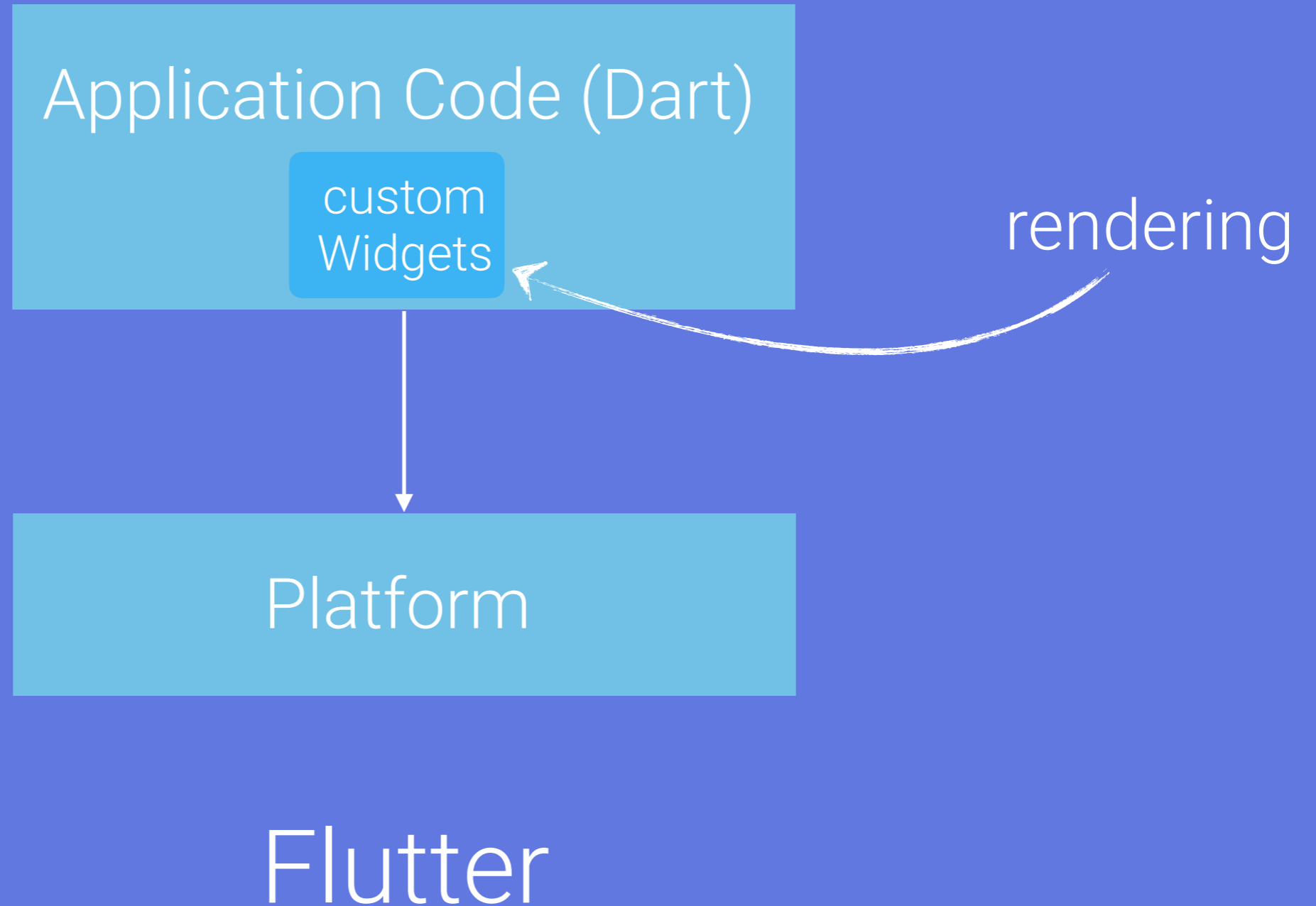
React Native

How is Flutter different?

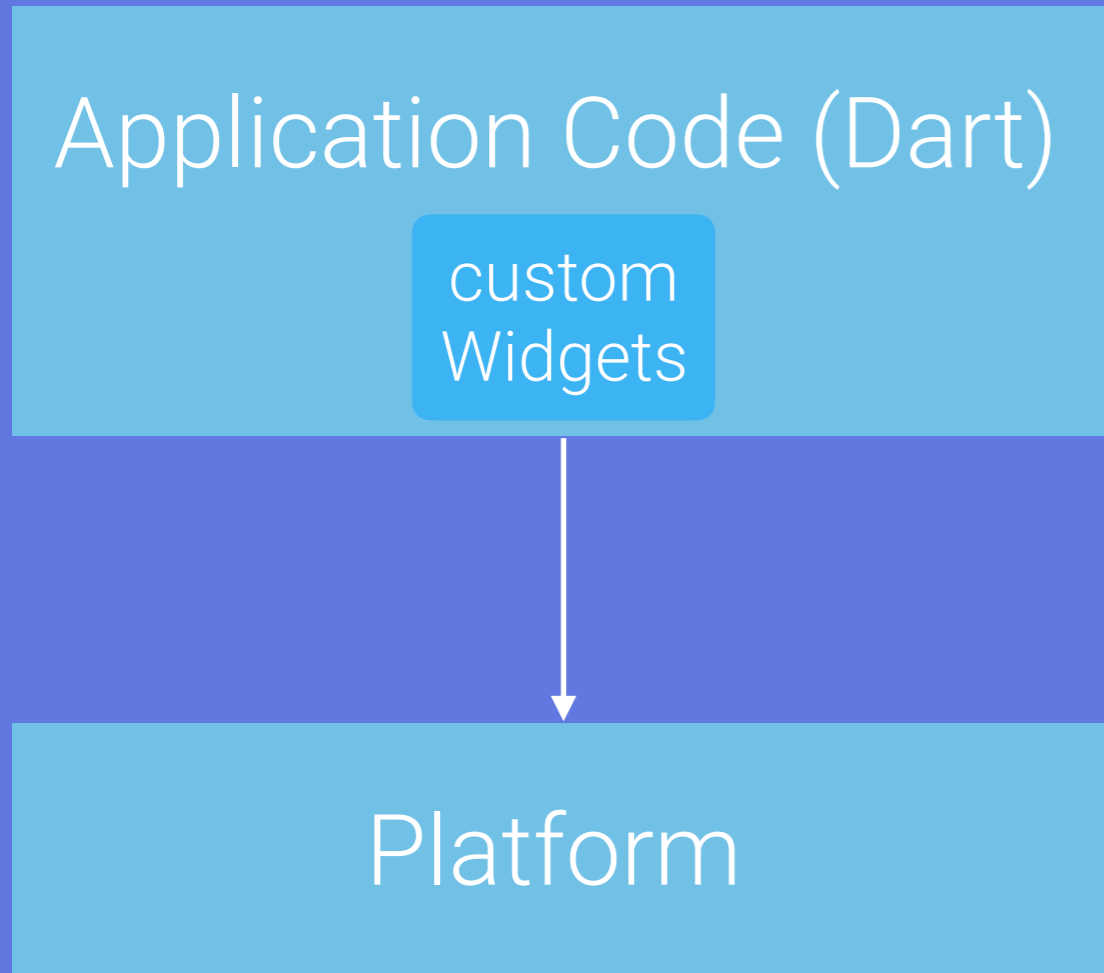


Flutter

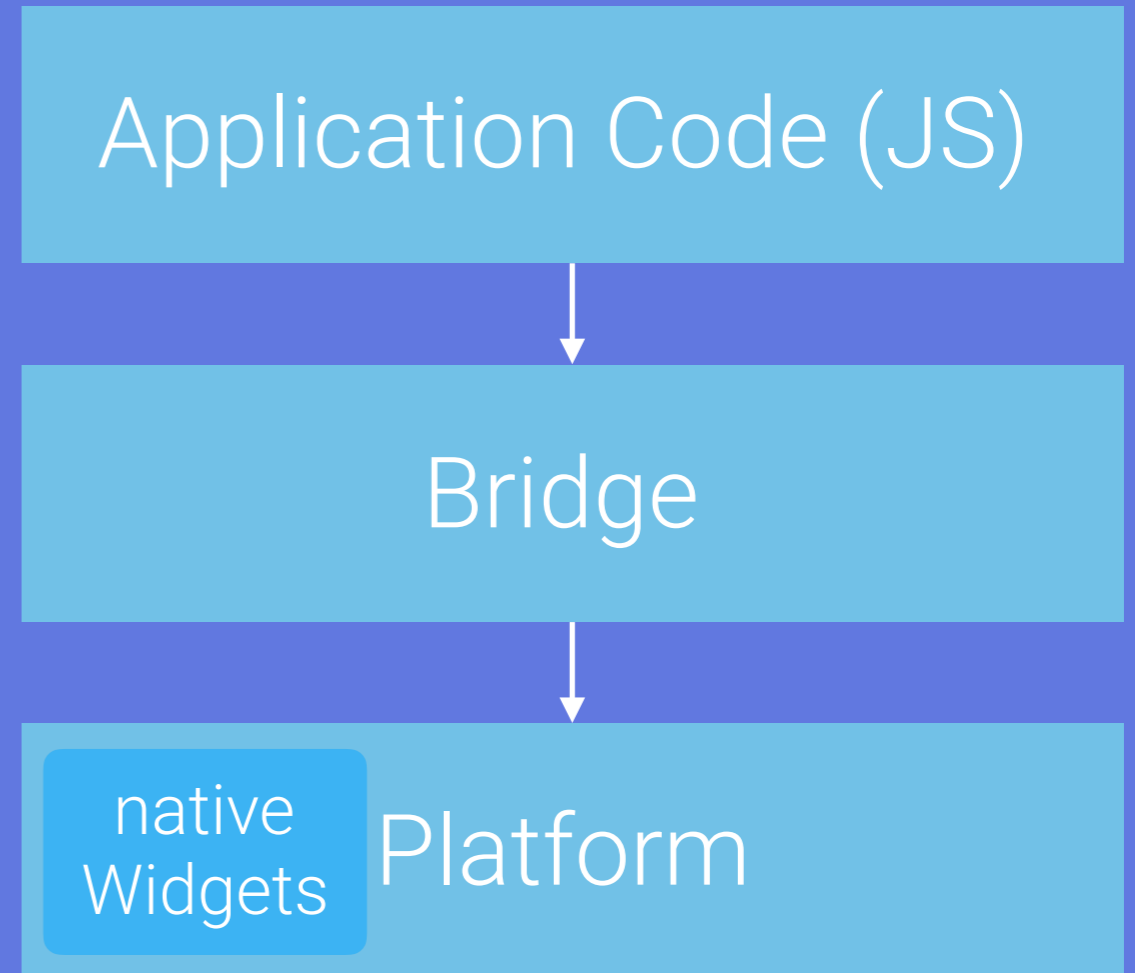
How is Flutter different?



How is Flutter different?

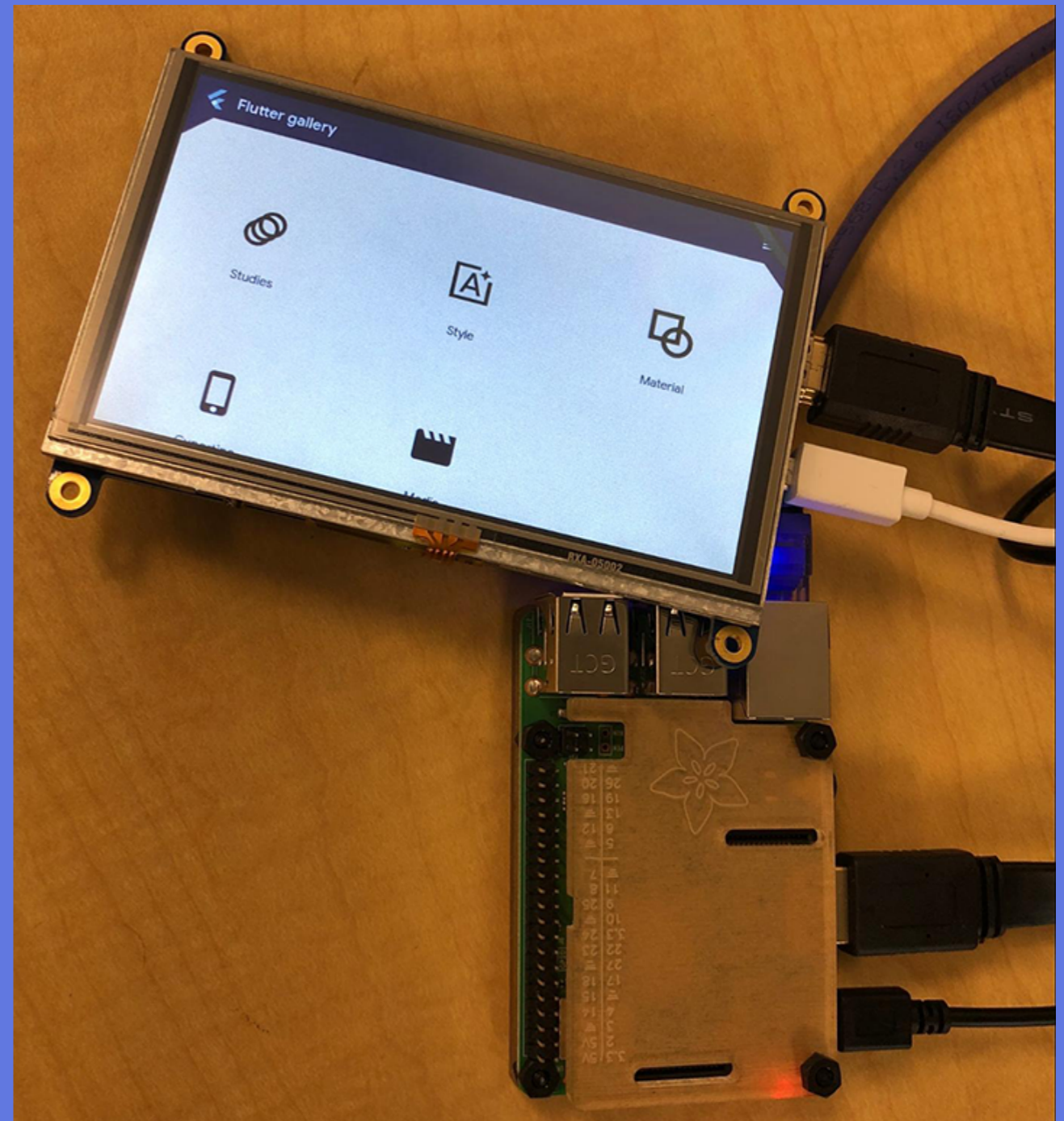
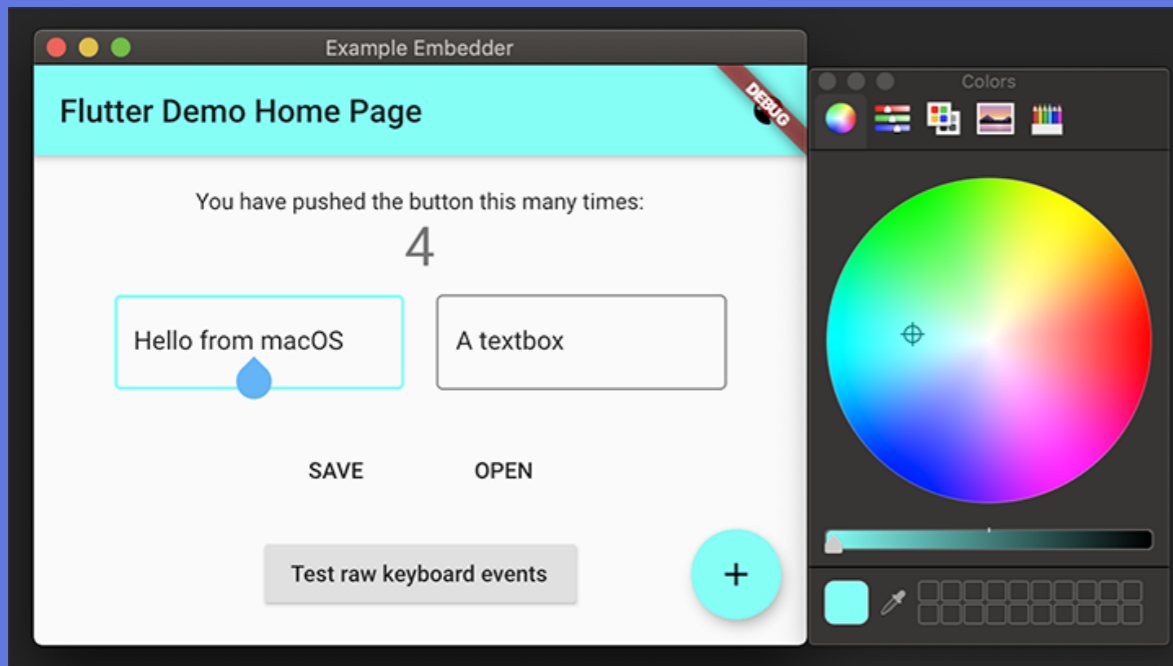


Flutter



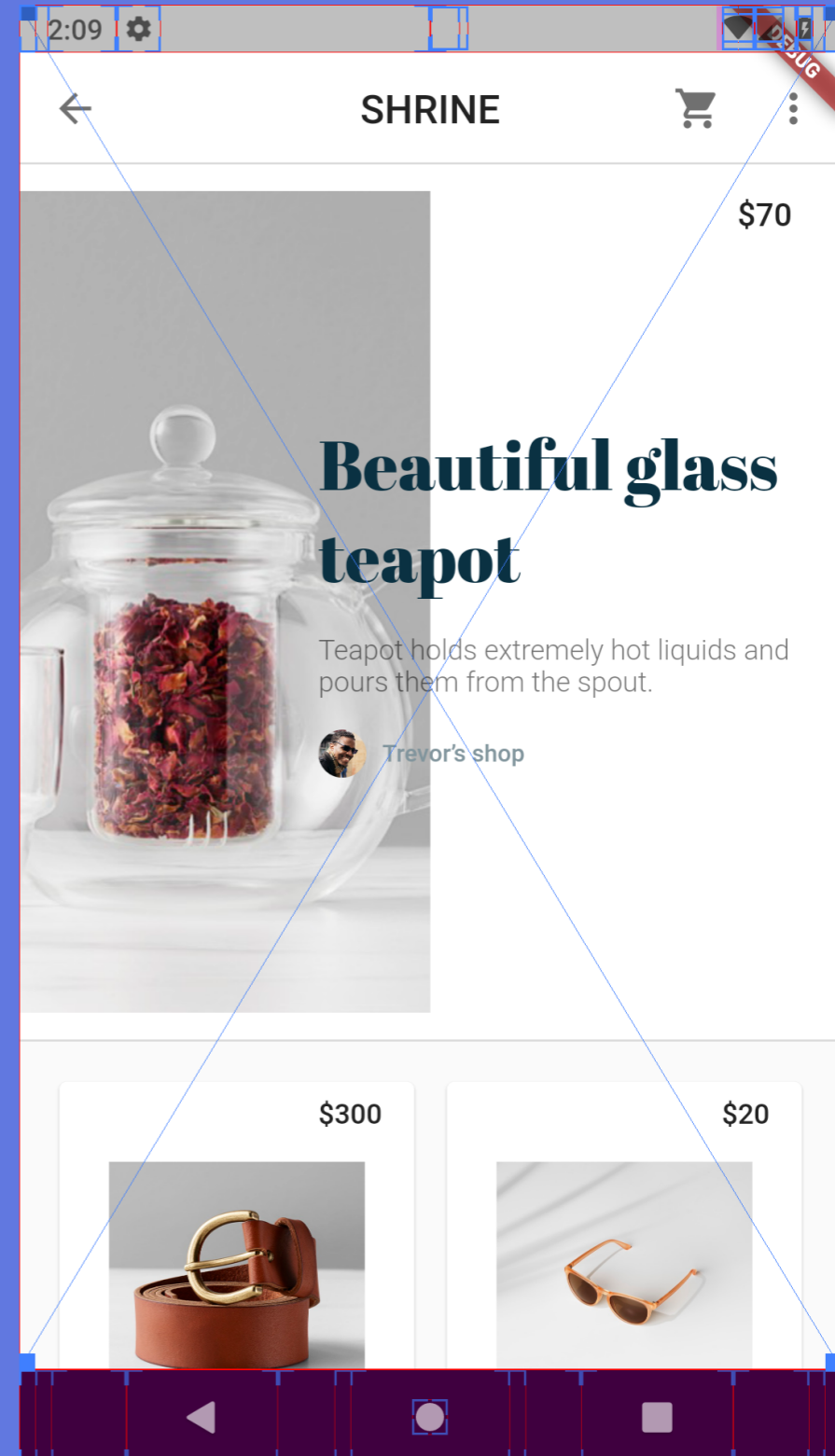
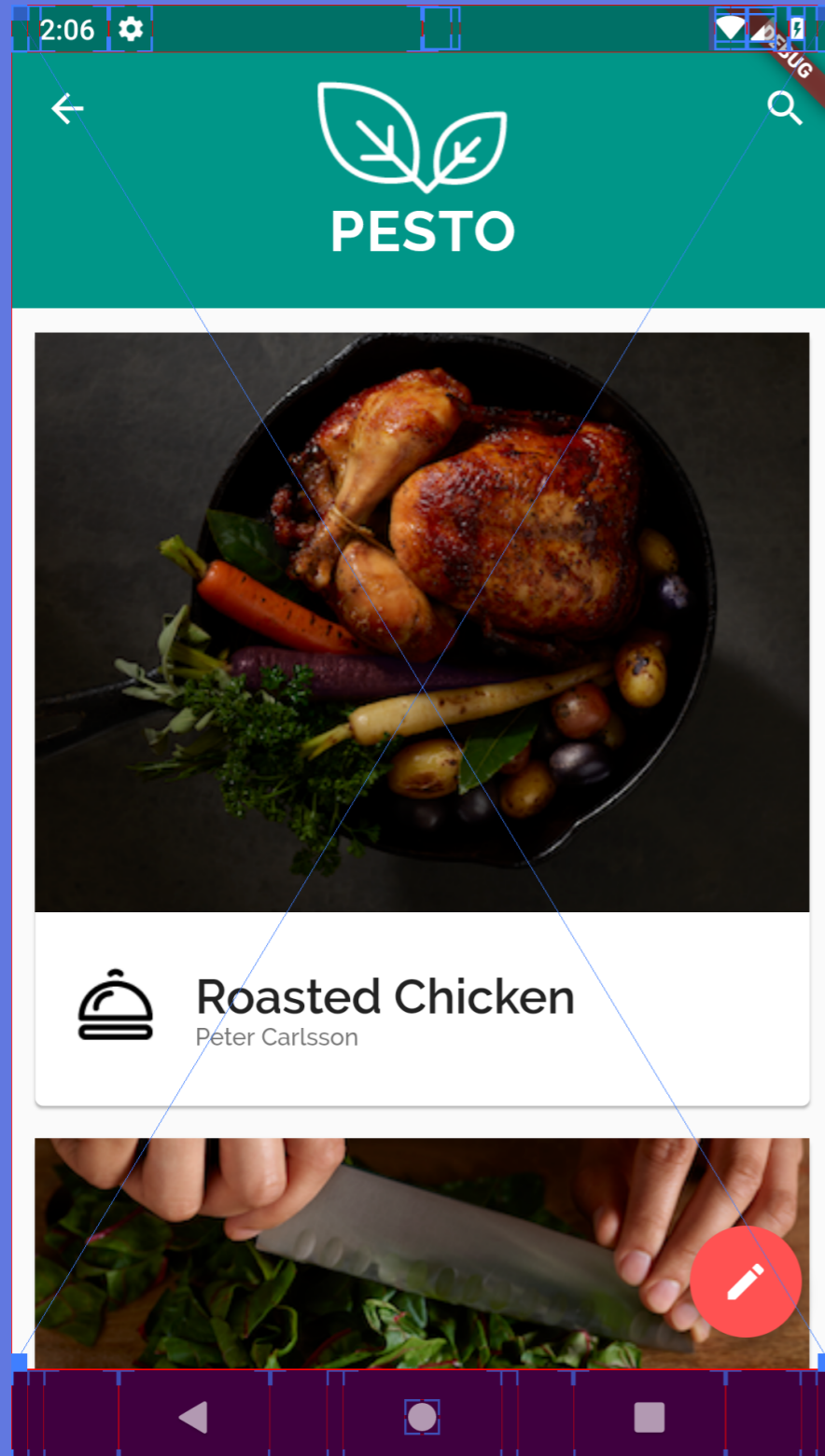
React Native

How is Flutter different?



Photos from: <https://medium.com/flutter-io/https-medium-com-flutter-io-pitching-flutter-2d4f494e47d1>

How is Flutter different?



How is Flutter different?

FAST
60 fps

How is Flutter different?

FAST
60 fps

FLEXIBLE
WIDGETS

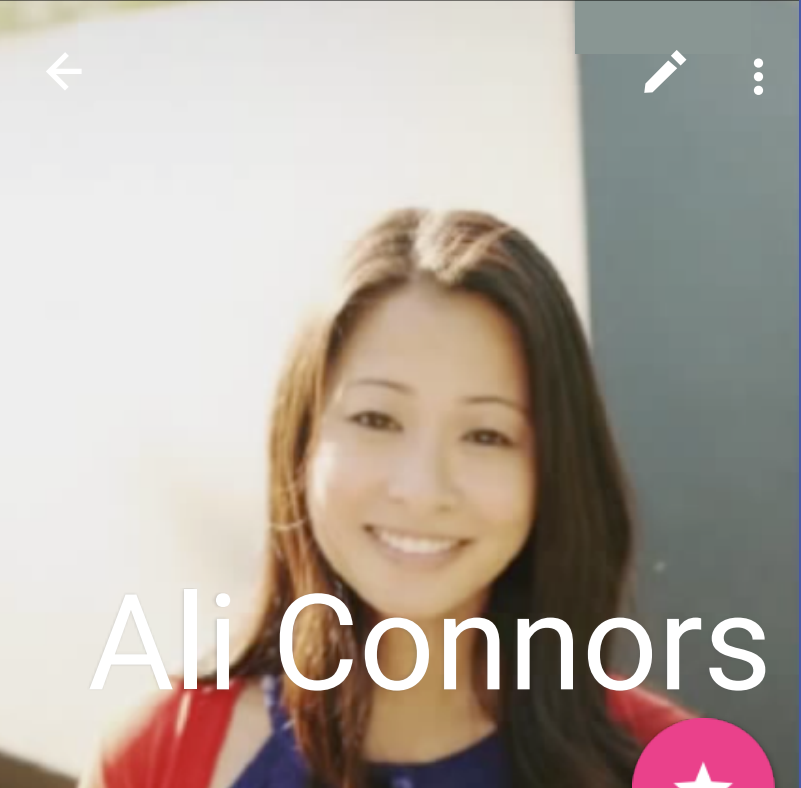
How is Flutter different?

EVERYTHING IS A
WIDGET

4:10

←

✎ ⋮



Ali Connors

★

📞 (650) 555-1234
Mobile

🗨️

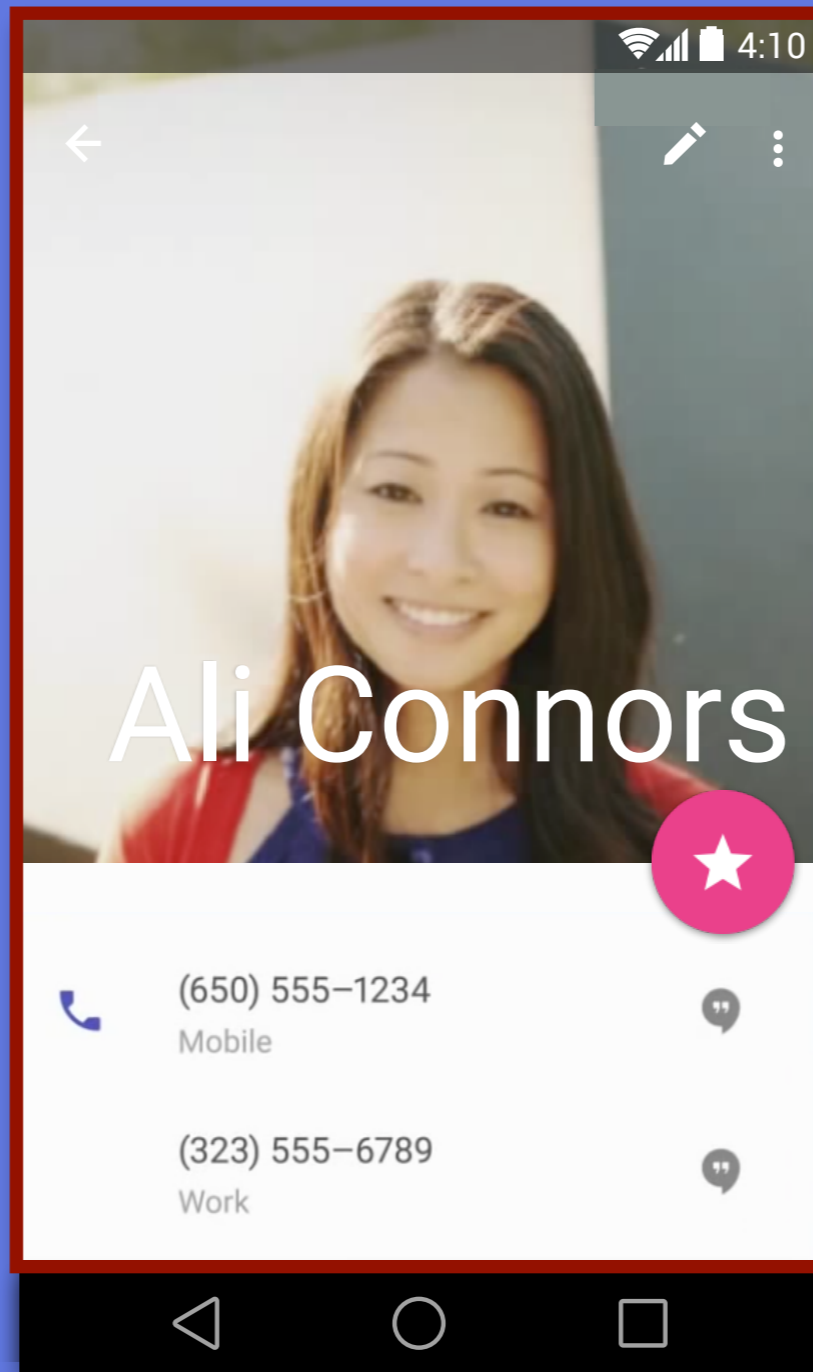
(323) 555-6789
Work

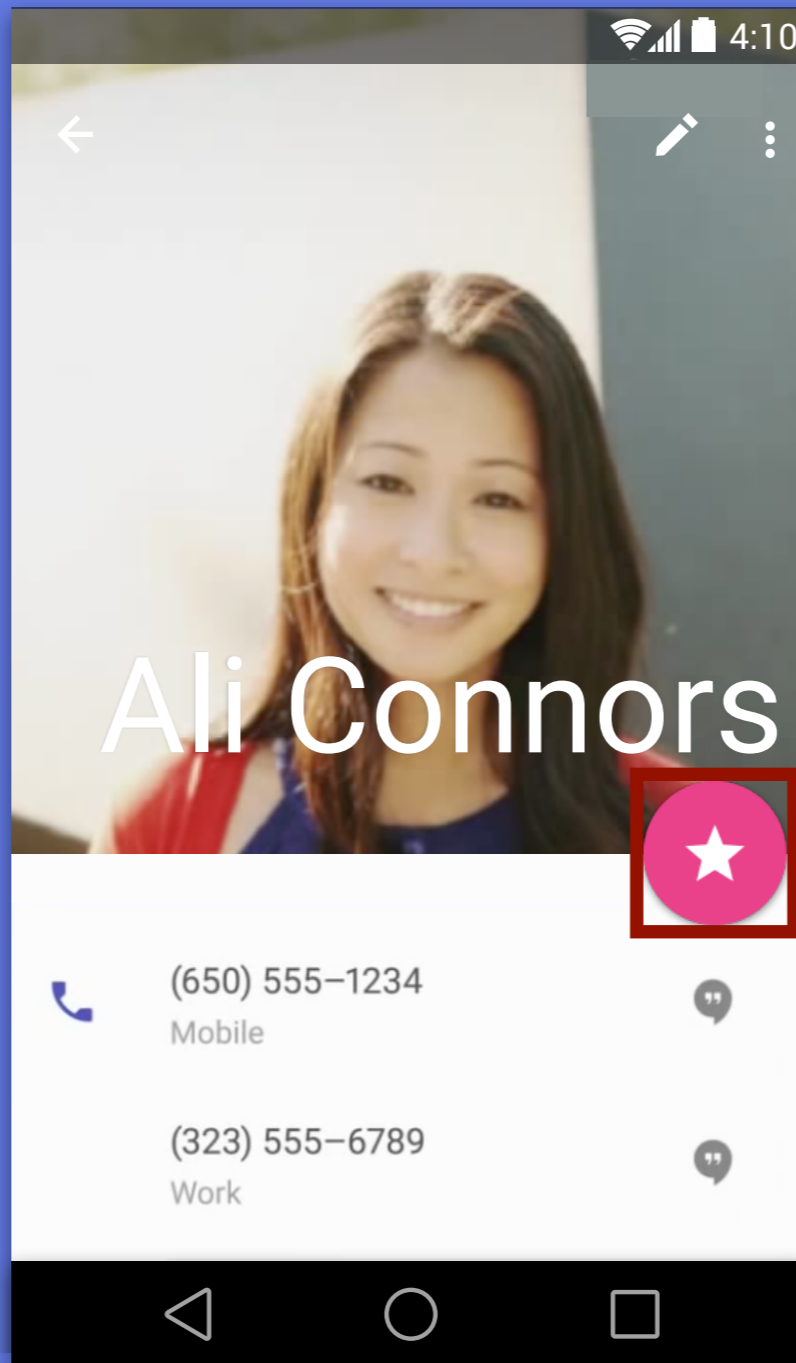
🗨️

◀ ○ ◻

Detailed description: This is a screenshot of a mobile contact card for Ali Connors. The top section features a portrait of a woman with long dark hair, smiling, wearing a red top. The name 'Ali Connors' is overlaid in large white text. To the right of the name is a pink circular button with a white star. Below the photo, there are two phone numbers: '(650) 555-1234' labeled 'Mobile' and '(323) 555-6789' labeled 'Work'. Each number has a small speech bubble icon to its right. The interface includes a back arrow in the top left, edit and menu icons in the top right, and standard Android navigation icons at the bottom. The status bar at the top right shows the time as 4:10, along with signal and battery icons.

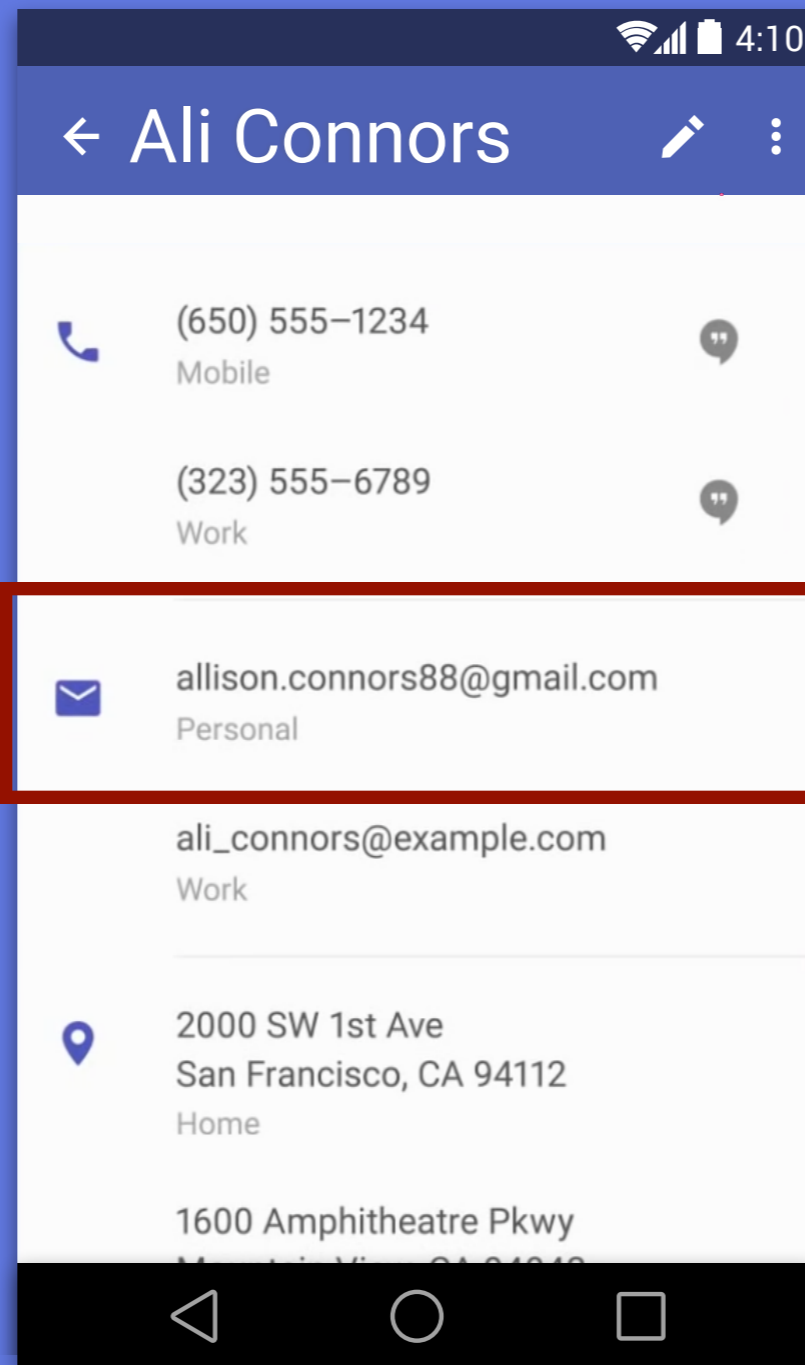
Scaffold



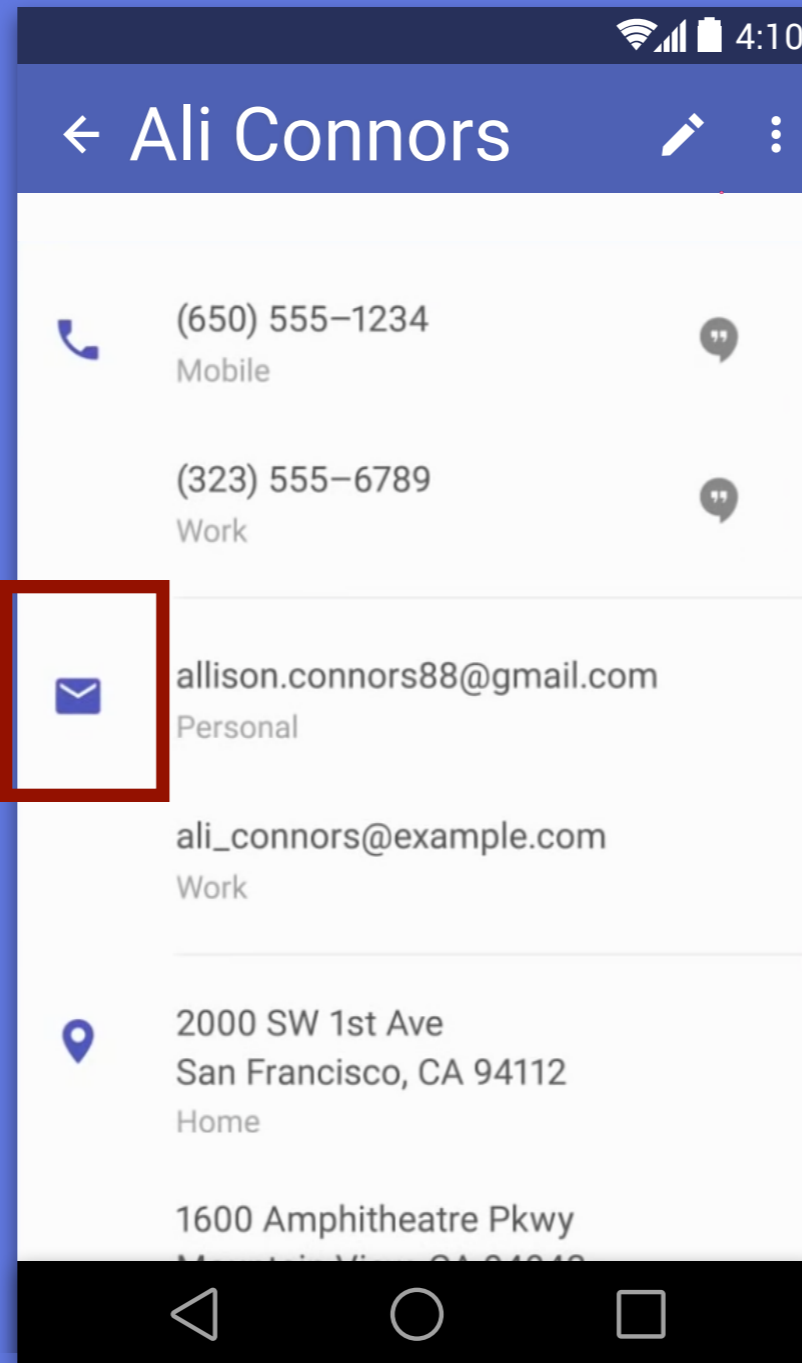


Floating
ActionButton

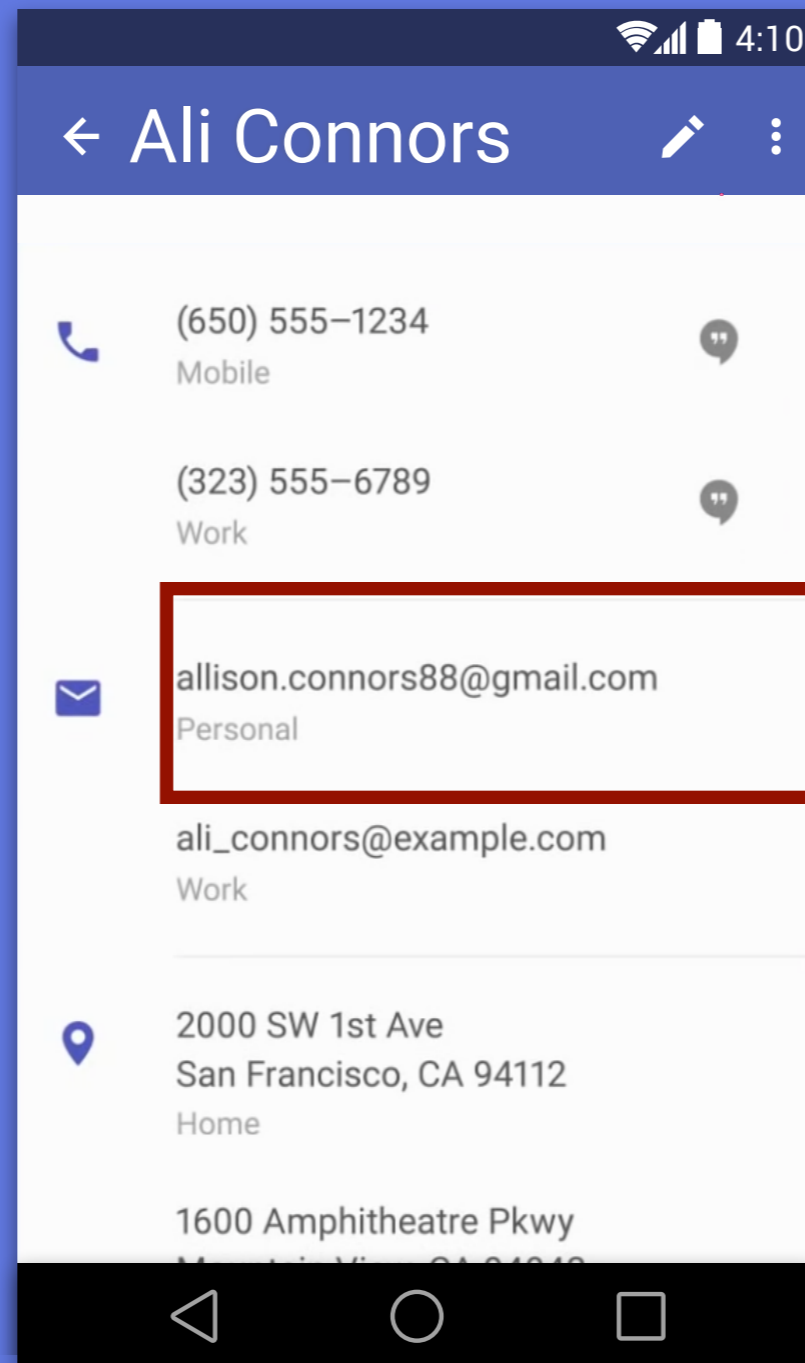
Row



Center



Padding



Android Equivalents

Android	Flutter
LinearLayout	Column
RelativeLayout	Column, Row, Stack
RecyclerView	ListView
gestures	GestureDetector
padding	Padding

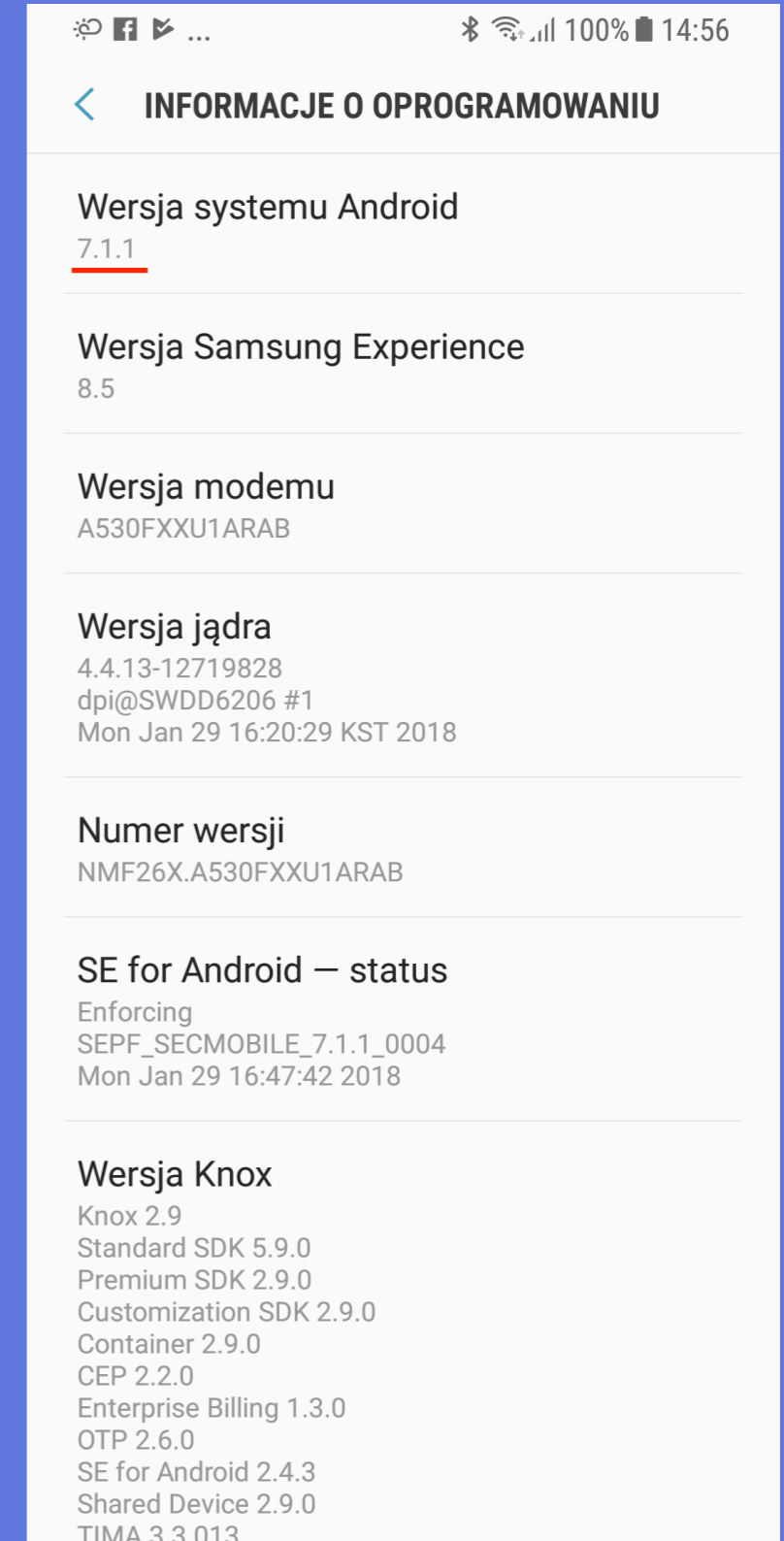
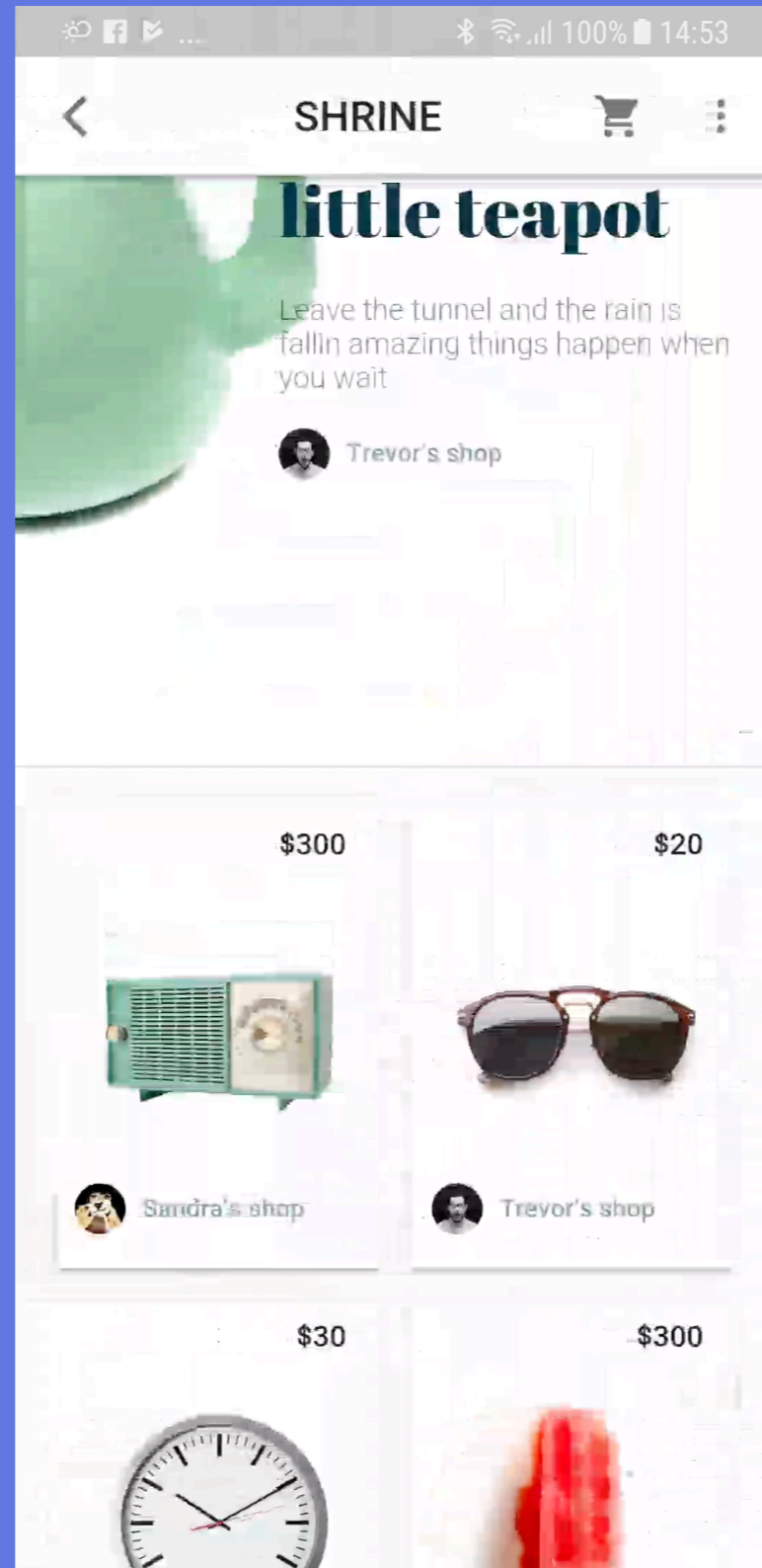
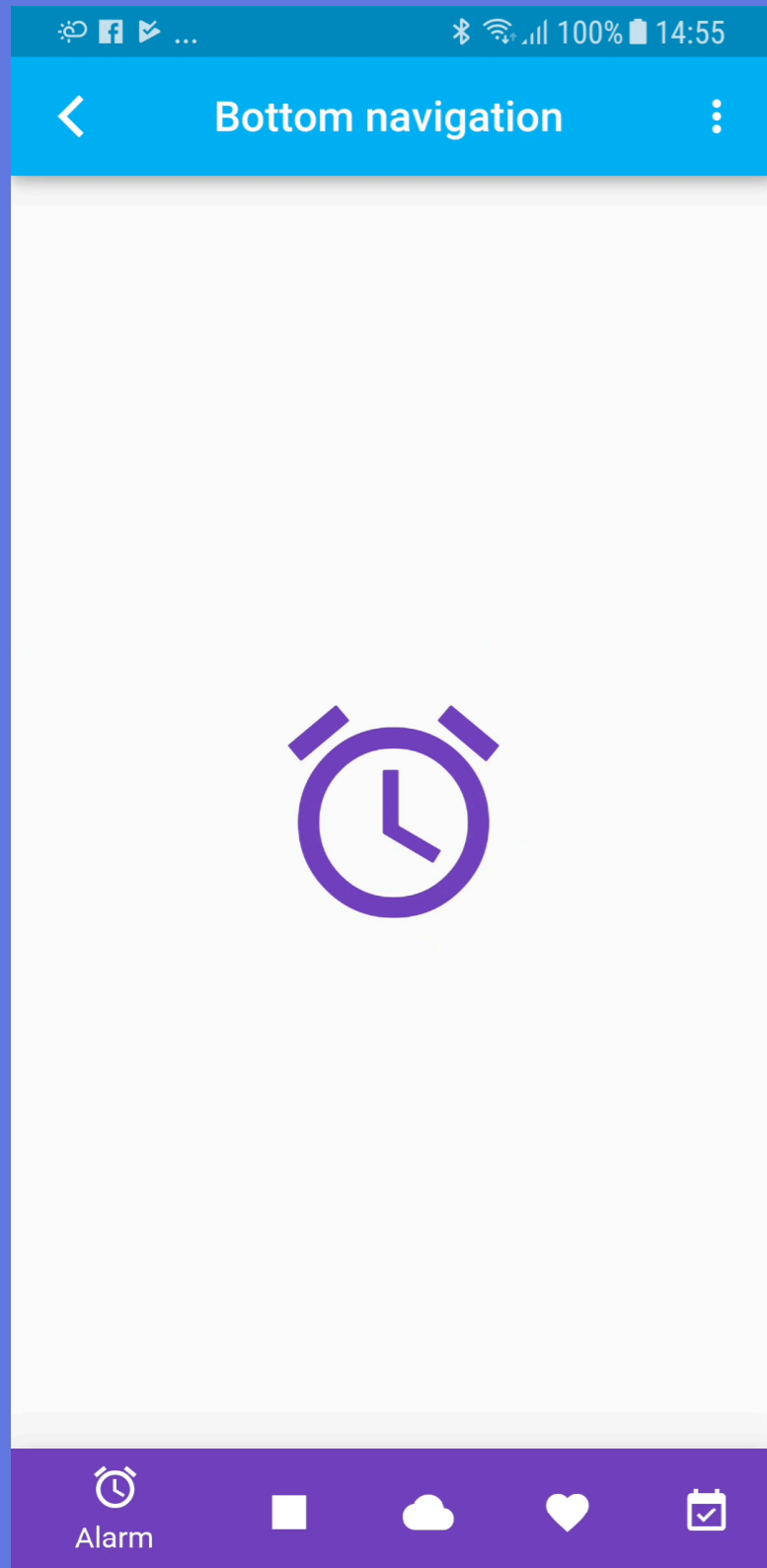
How is Flutter different?

FAST
60 fps

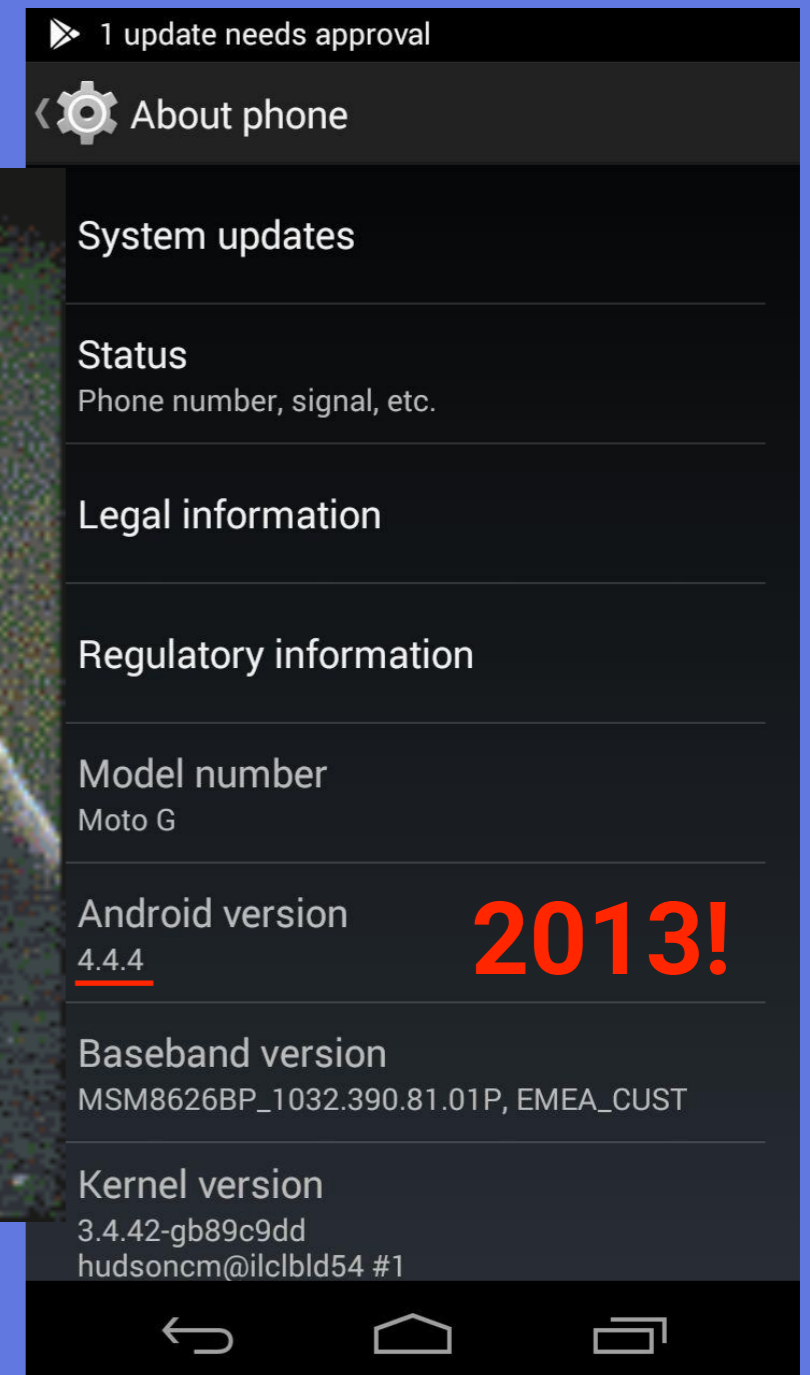
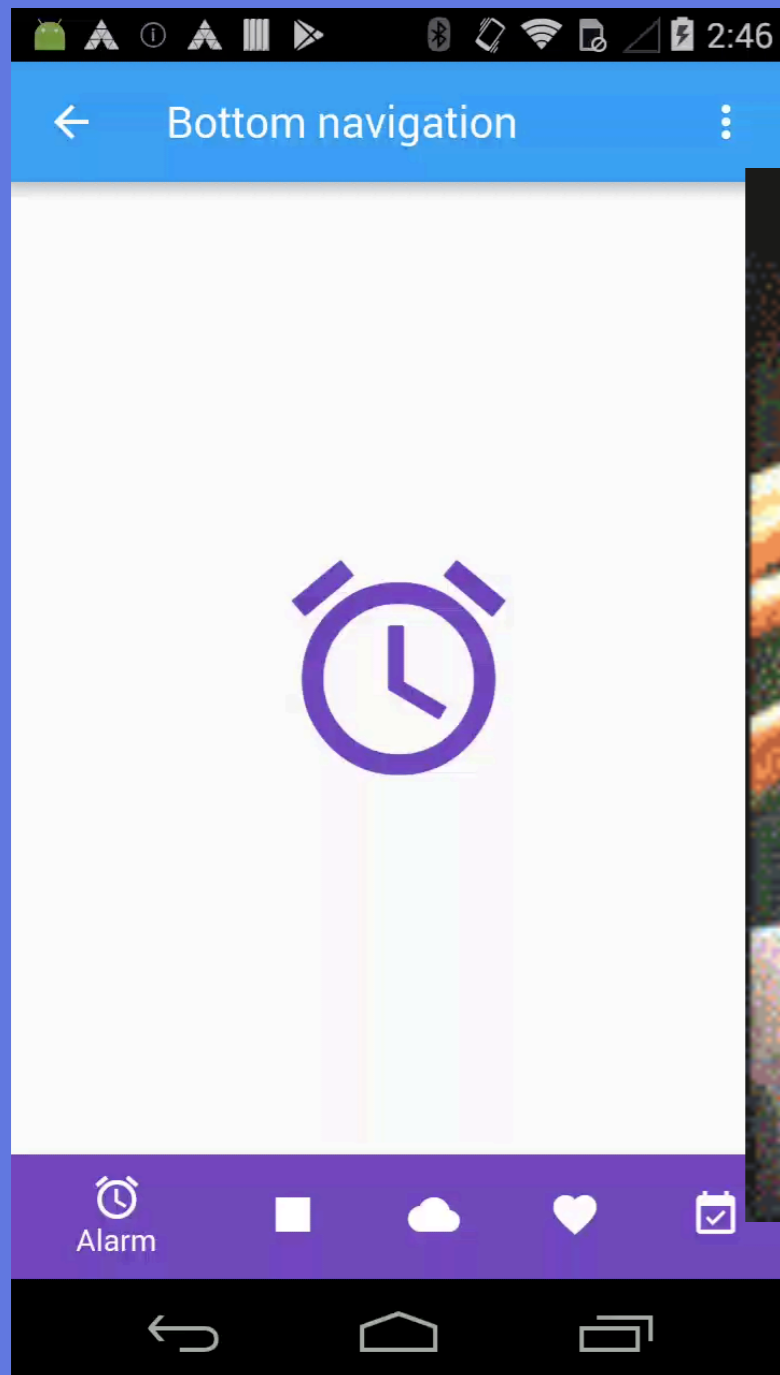
FLEXIBLE
WIDGETS

NATIVE
LOOK&FEEL

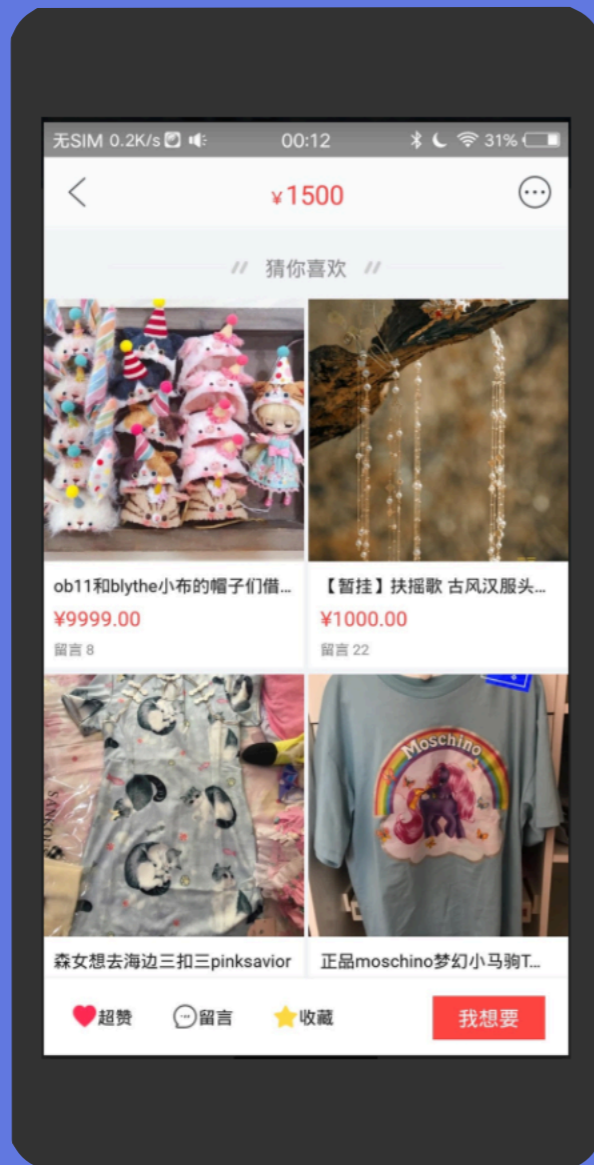
How is Flutter different?



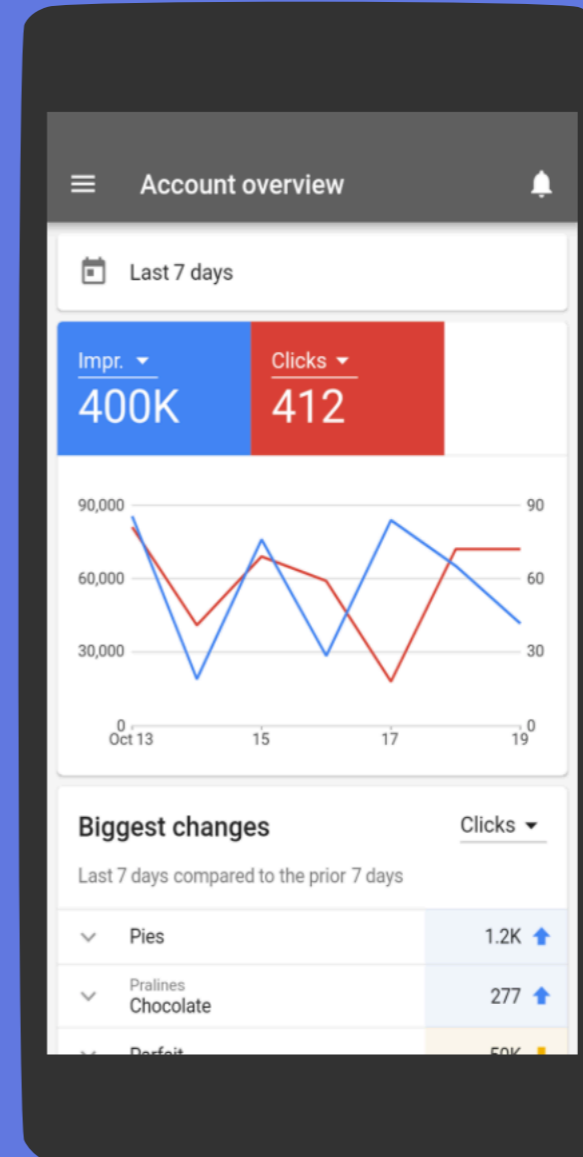
How is Flutter different?



Who uses Flutter?



Alibaba
50+ million users



Google Ads,
5+ million users

Who uses Flutter?



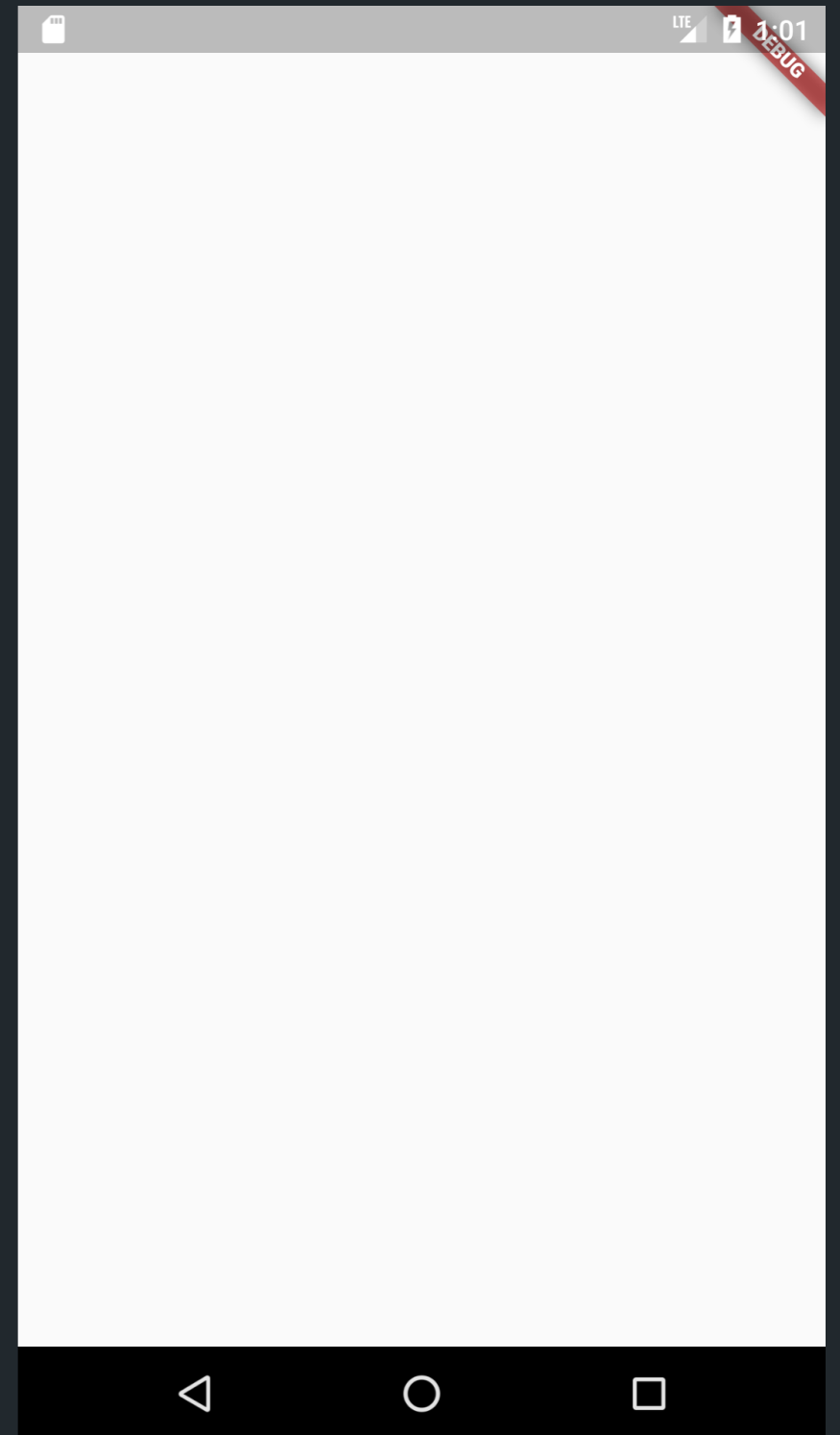
Reflectly, 500k+ users

How to make an app?

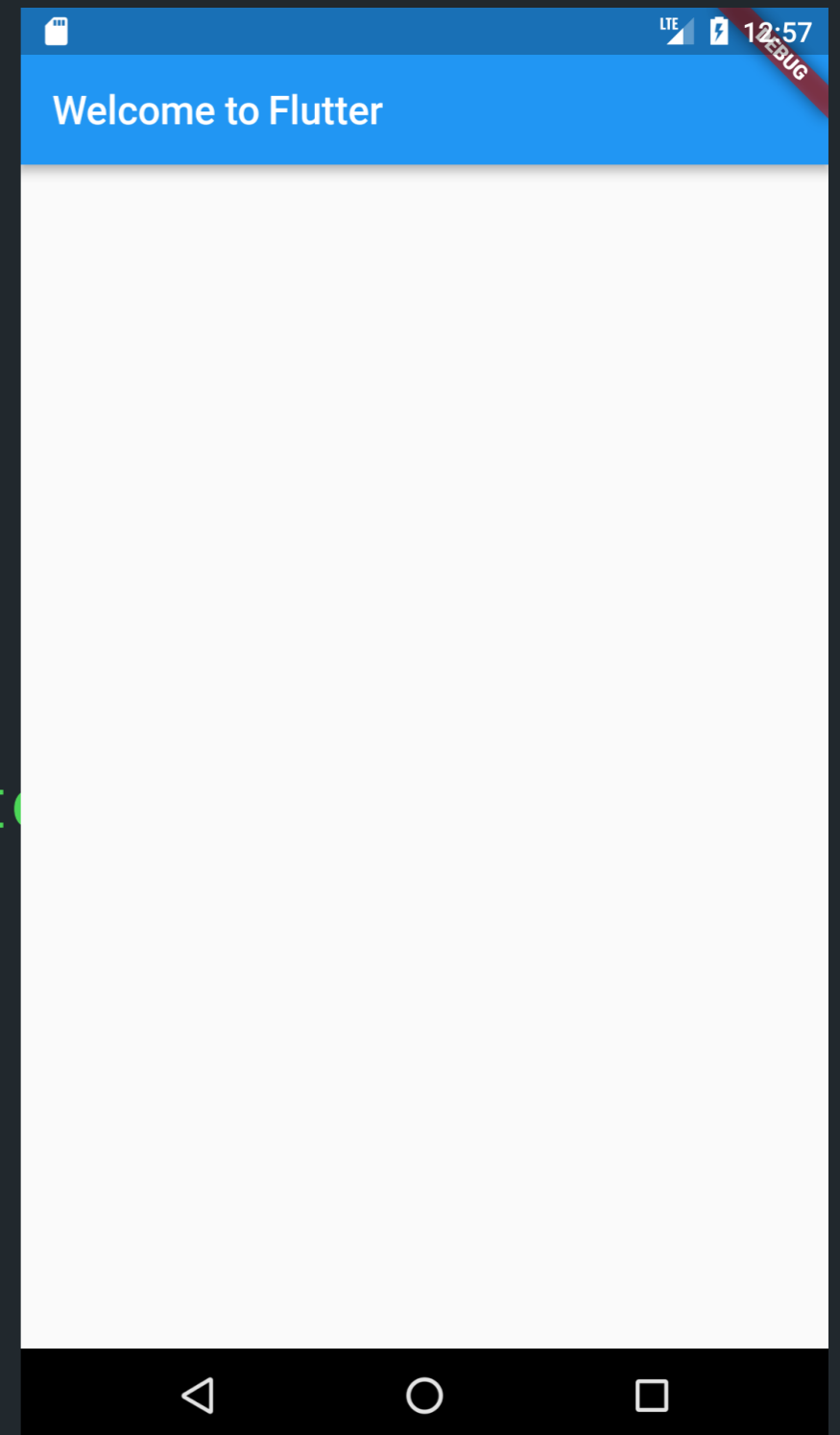
```
import 'package:flutter/material.dart';
```

```
void main() => runApp(MyApp());
```

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'FlutterApp',  
      home: Scaffold(  
        ),  
    );  
  }  
}
```



```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'FlutterApp',  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text('Welcome to Flutter'),  
        ),  
      ),  
    );  
  }  
}
```



```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'FlutterApp',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Welcome to Flutter'),
        ),
        body: MyAppBody(),
      ),
    );
  }
}
```

```
class MyAppBody extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
  
  }  
}
```

```
class MyAppBody extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return Column(  
      children: <Widget>[  
  
        ],  
    );  
  }  
}
```



```
class MyAppBody extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return Column(  
      children: <Widget>[  
        Flexible(  
          ),  
        ],  
      );  
  }  
}
```

```
class MyAppBody extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
        Flexible(
          child: ListView.builder(
            ),
        ),
      ],
    );
  }
}
```

```
class MyAppBody extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return Column(  
      children: <Widget>[  
        Flexible(  
          child: ListView.builder(  
            itemCount: 10,  
          ),  
        ),  
      ],  
    );  
  }  
}
```

```
class MyAppBody extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
        Flexible(
          child: ListView.builder(
            itemCount: 10,
            itemBuilder: (_, position) =>
              MyAppListItem(position),
          ),
        ),
      ],
    );
  }
}
```

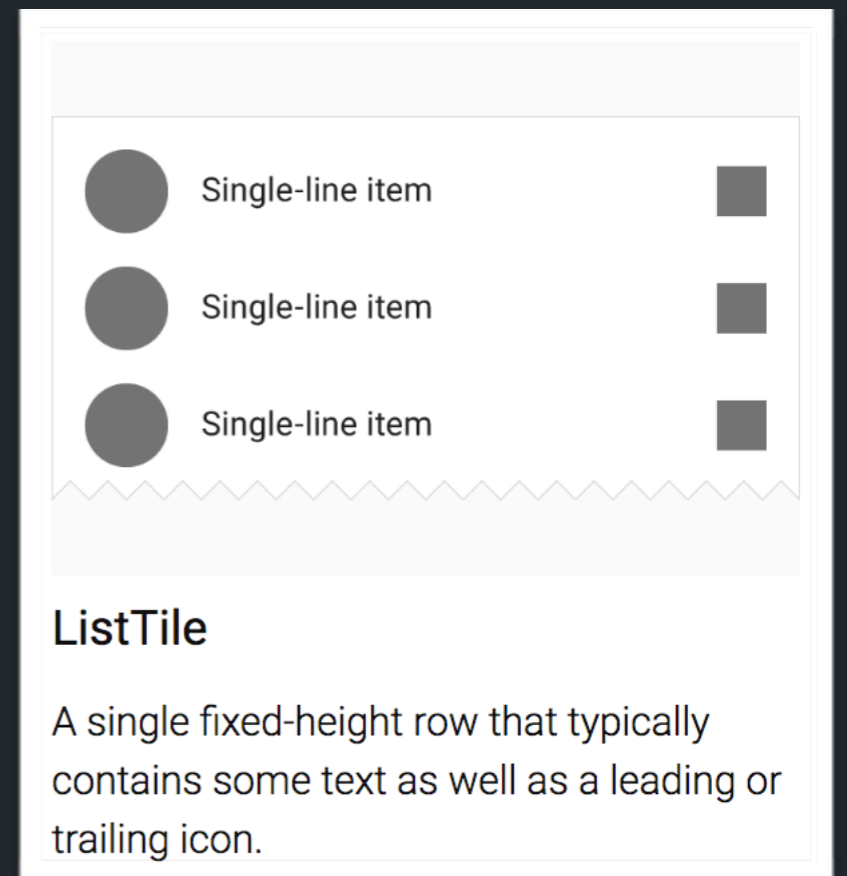
```

class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return ListTile(
      );
  }
}

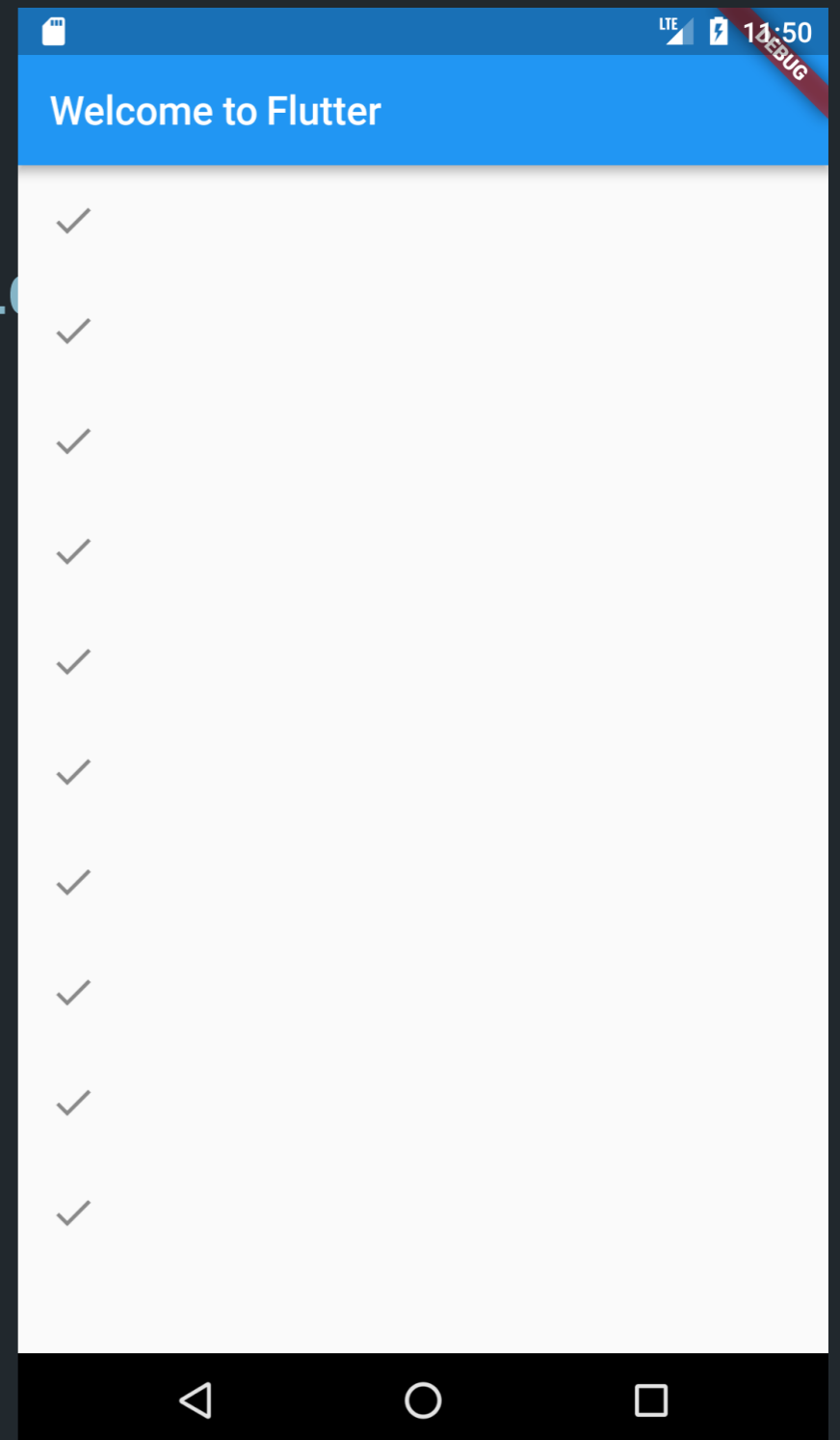
```



```
class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

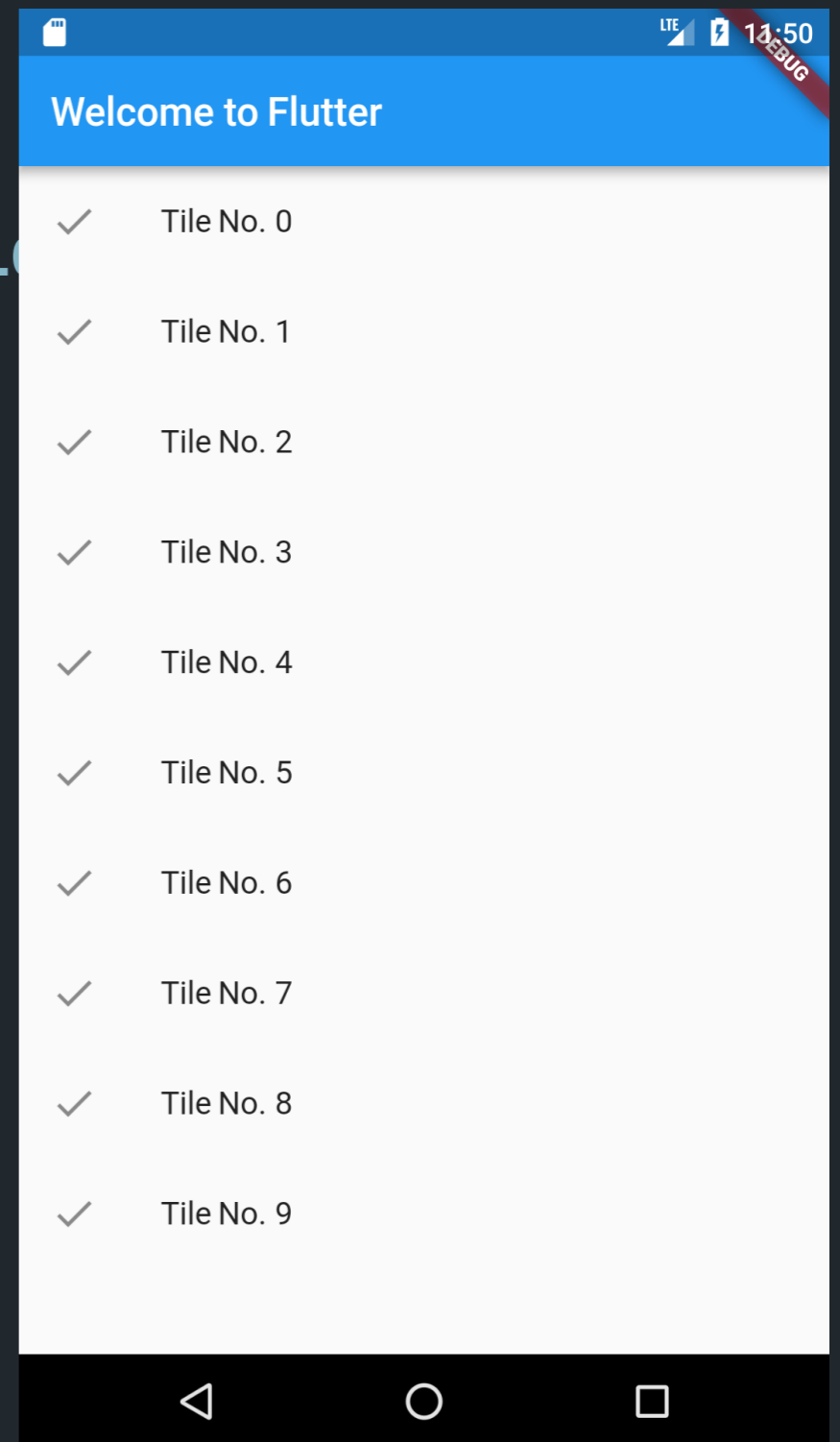
  @override
  Widget build(BuildContext context) {
    return ListTile(
      leading: Icon(Icons.done),
    );
  }
}
```



```
class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return ListTile(
      leading: Icon(Icons.done),
      title: Text("Tile No. $position")
    );
  }
}
```



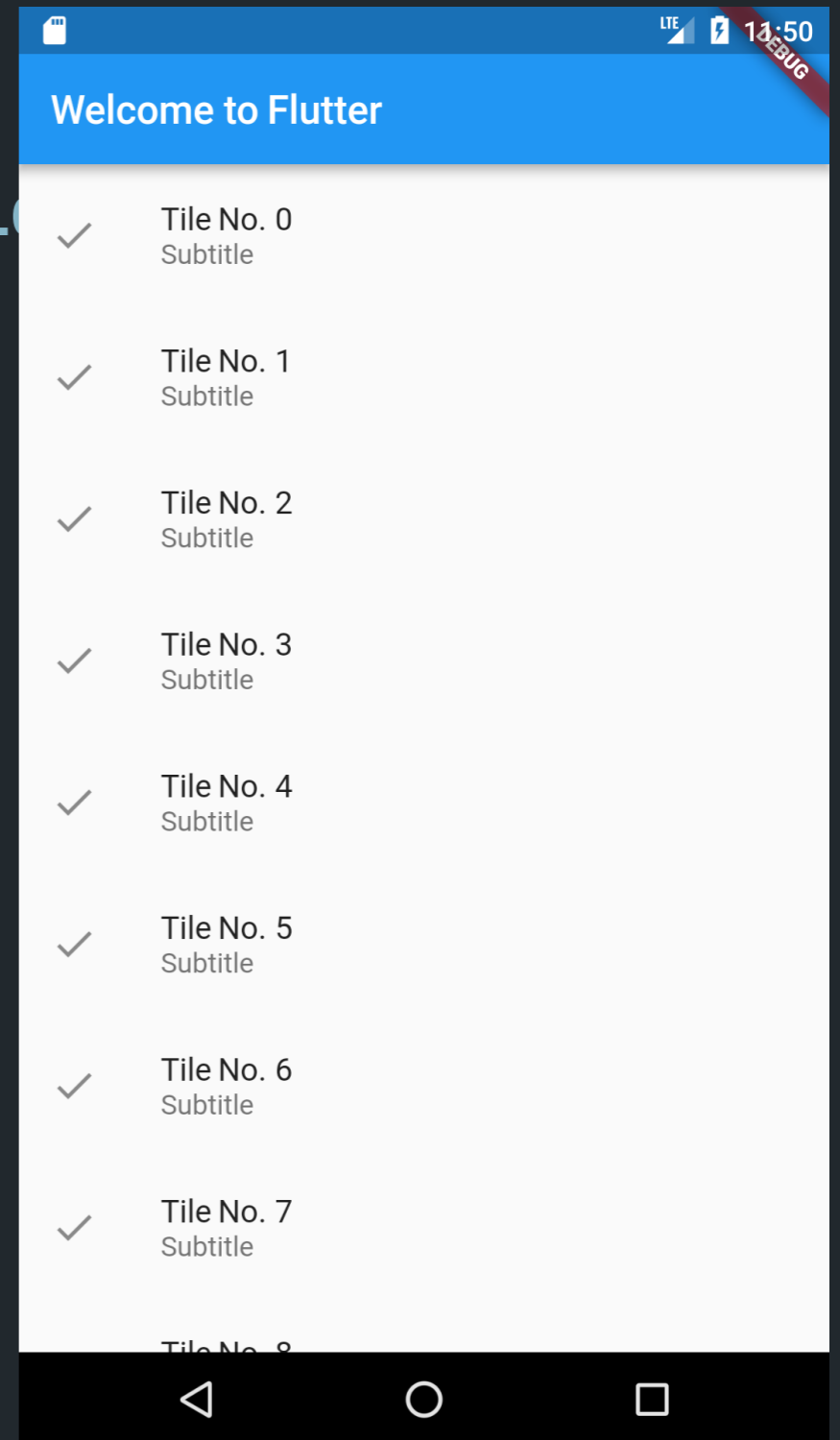
```

class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return ListTile(
      leading: Icon(Icons.done),
      title: Text("Tile No. $position"),
      subtitle: Text("Subtitle"),
    );
  }
}

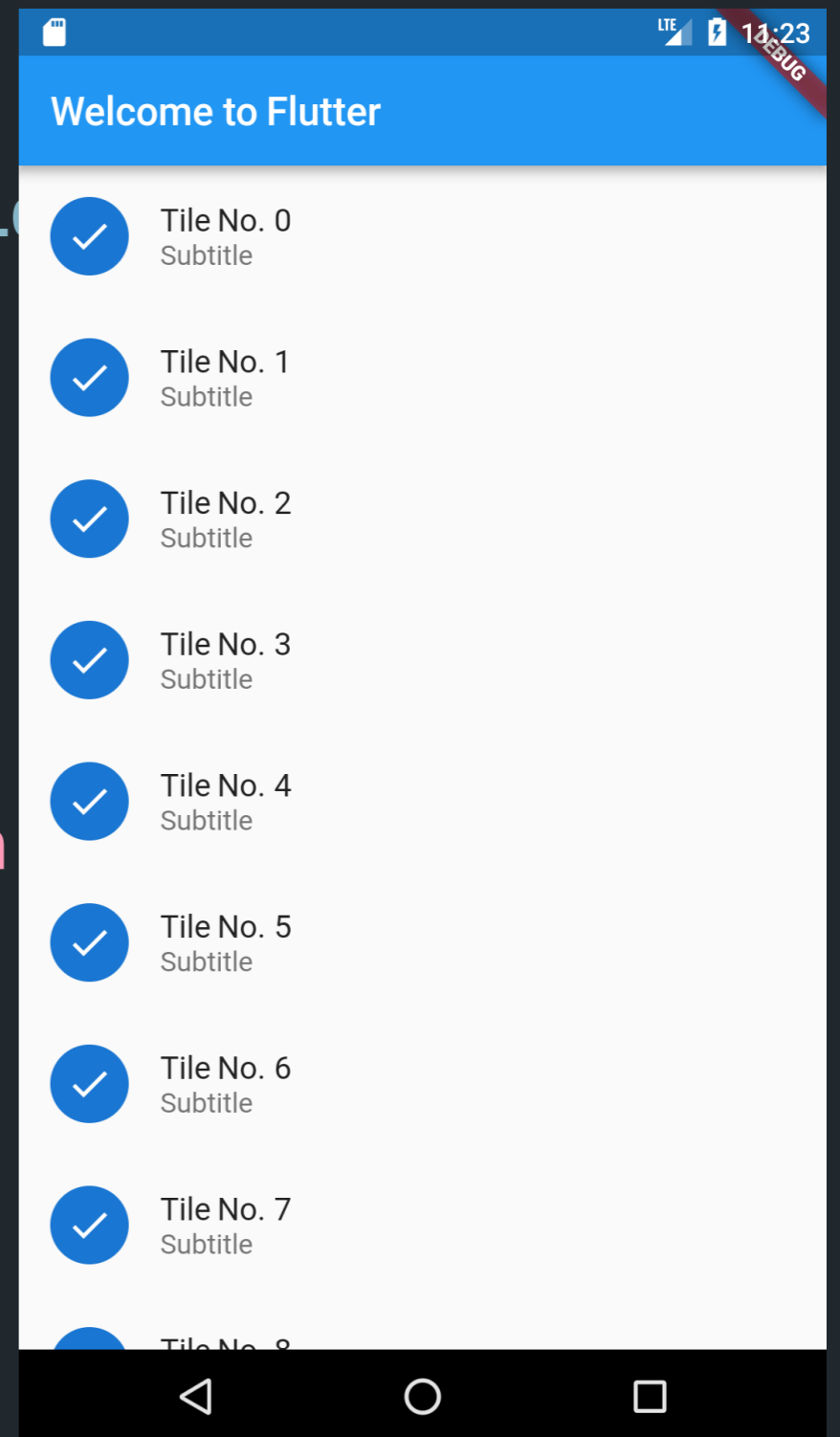
```




```
class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return ListTile(
      leading: CircleAvatar(child: Icon(
        title: Text("Tile No. $position")
        subtitle: Text("Subtitle"),
      );
    );
  }
}
```



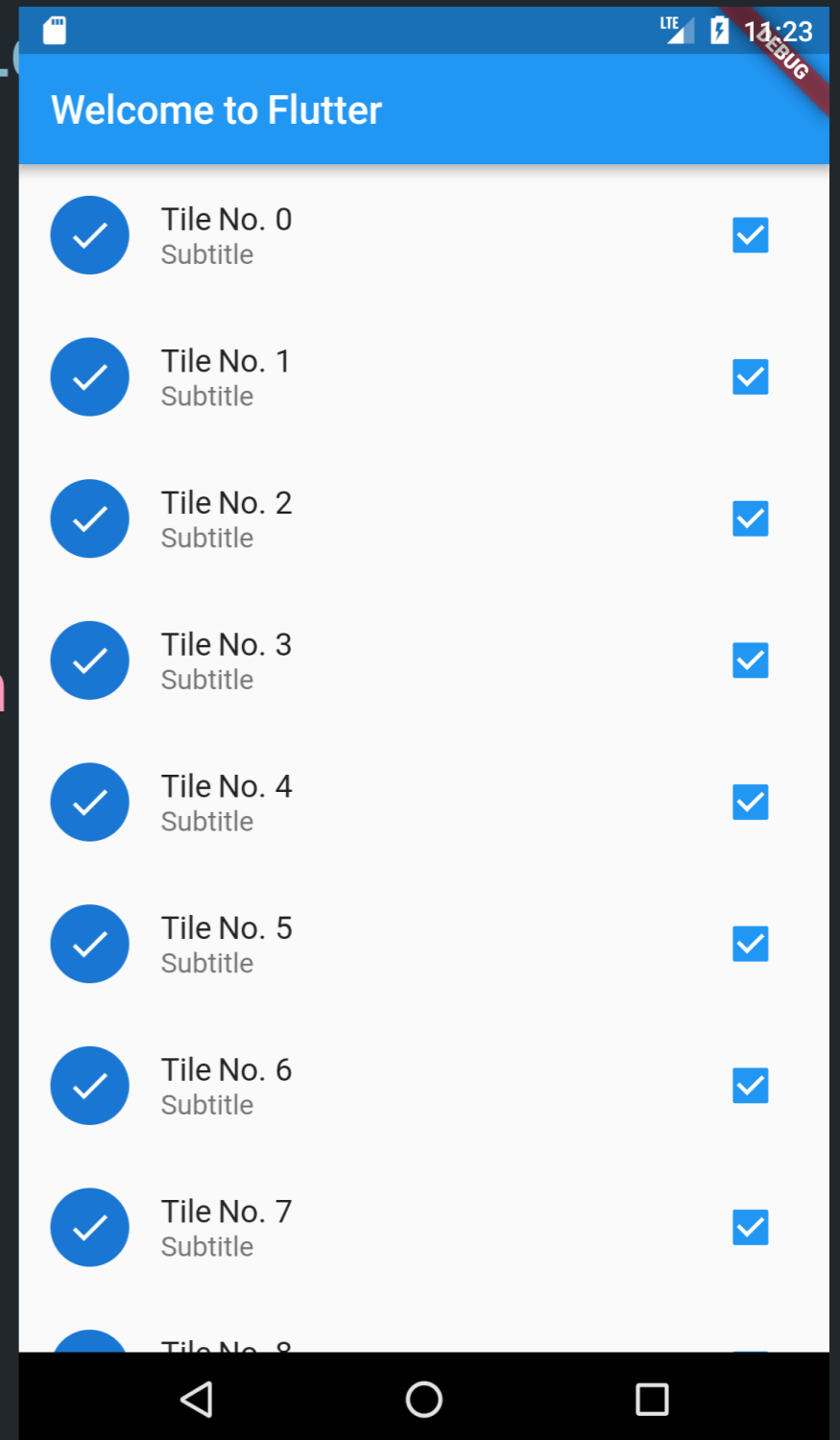
```

class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return ListTile(
      leading: CircleAvatar(child: Icon(
        title: Text("Tile No. $position")
        subtitle: Text("Subtitle"),
        trailing: Checkbox(
          value: true,
          onChanged: handleCheck,
        ),
      ),
    );
  }
}

```



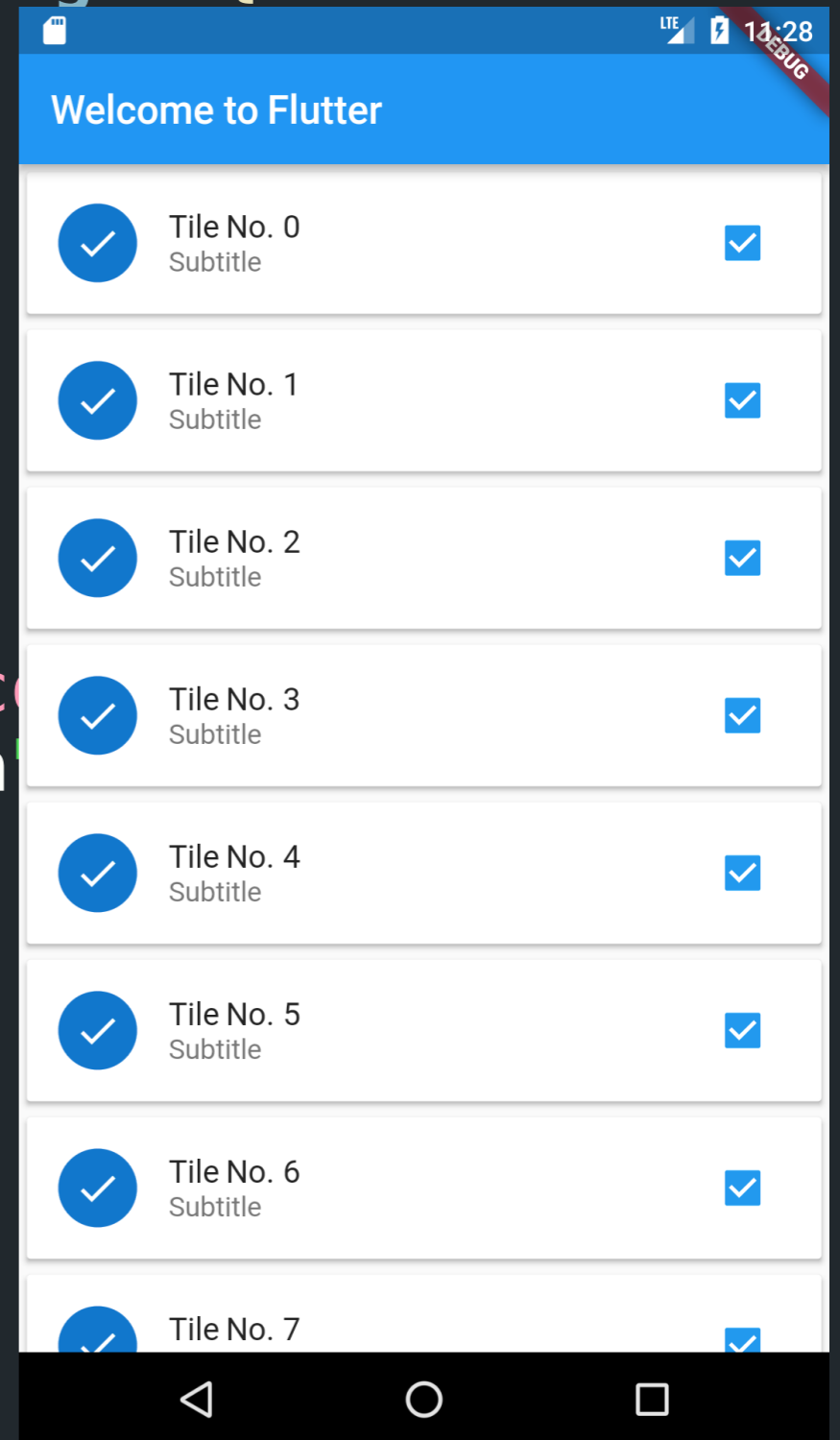
```

class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

  @override
  Widget build(BuildContext context) {
    return Card(
      child: ListTile(
        leading: CircleAvatar(child: Icon(Icons.check)),
        title: Text("Tile No. $position"),
        subtitle: Text("Subtitle"),
        trailing: Checkbox(
          value: true,
          onChanged: handleCheck,
        ),
      ),
    );
  }
}

```



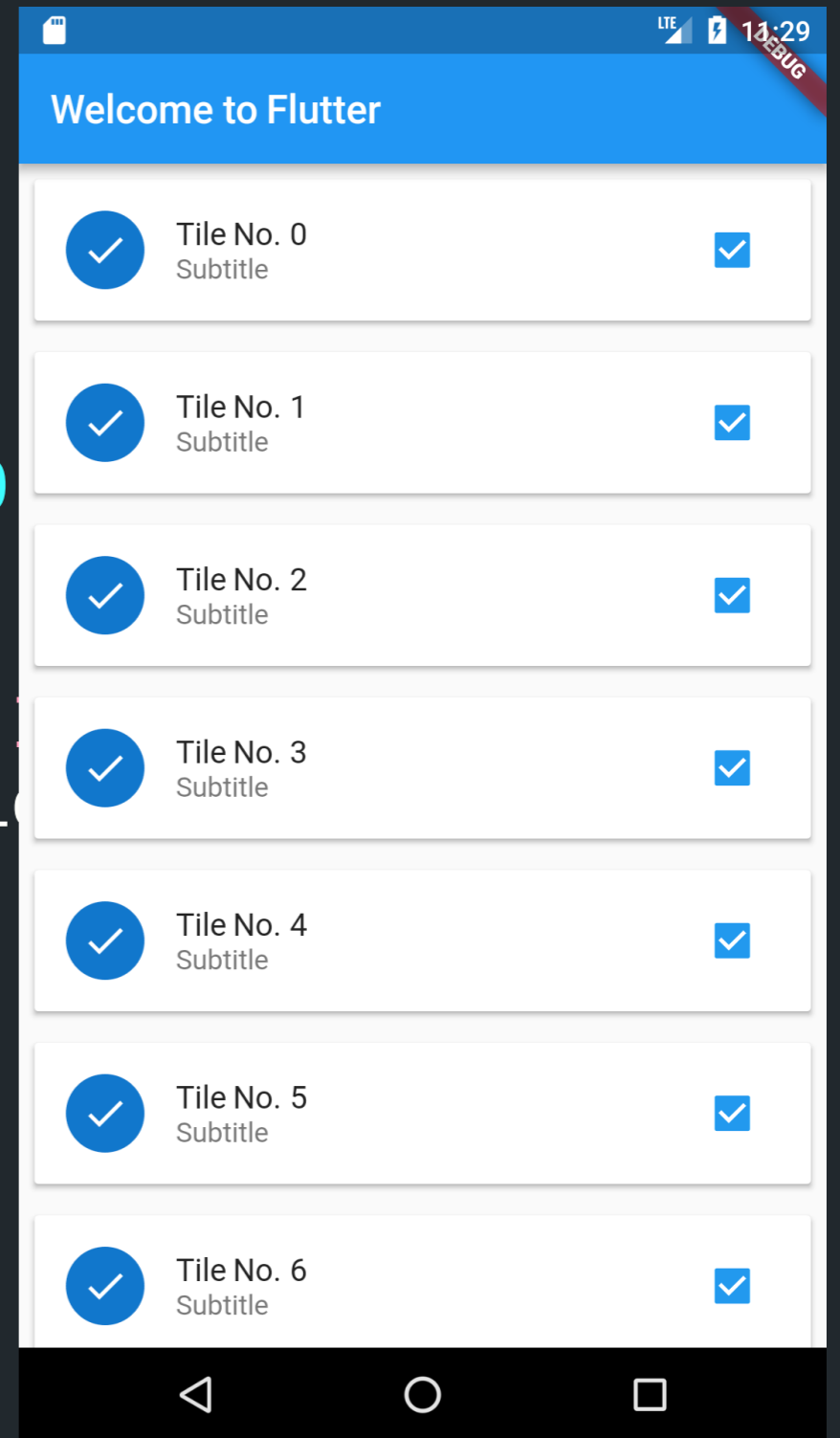
```

class MyAppListItem extends StatelessWidget {
  final int position;

  MyAppListItem(this.position);

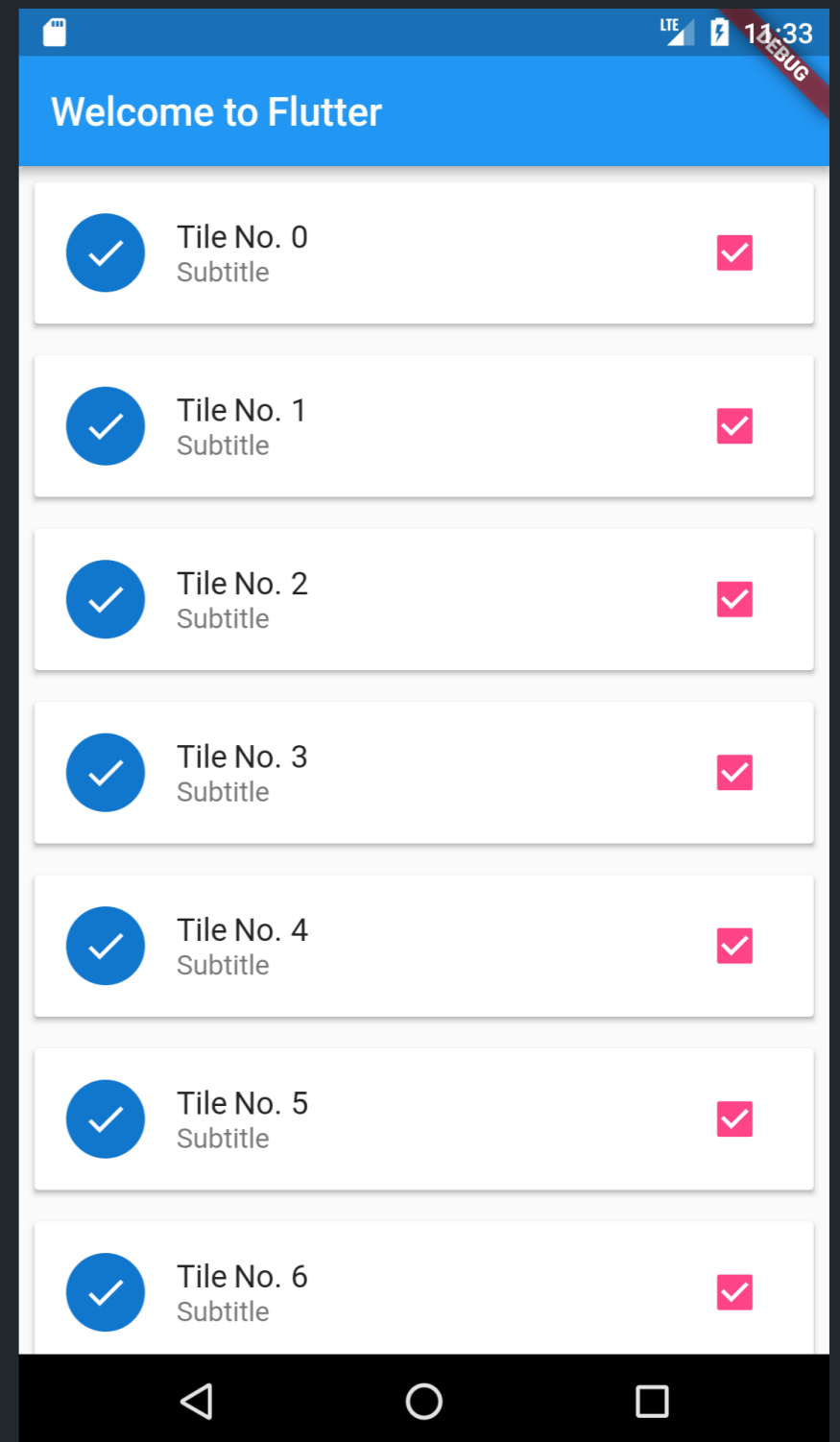
  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(4.0),
      child: Card(
        child: ListTile(
          leading: CircleAvatar(
            child: Text("Tile No. $position"),
            title: Text("Tile No. $position"),
            subtitle: Text("Subtitle"),
            trailing: Checkbox(
              value: true,
              onChanged: handleCheck,
            ),
          ),
        ),
      ),
    );
  }
}

```

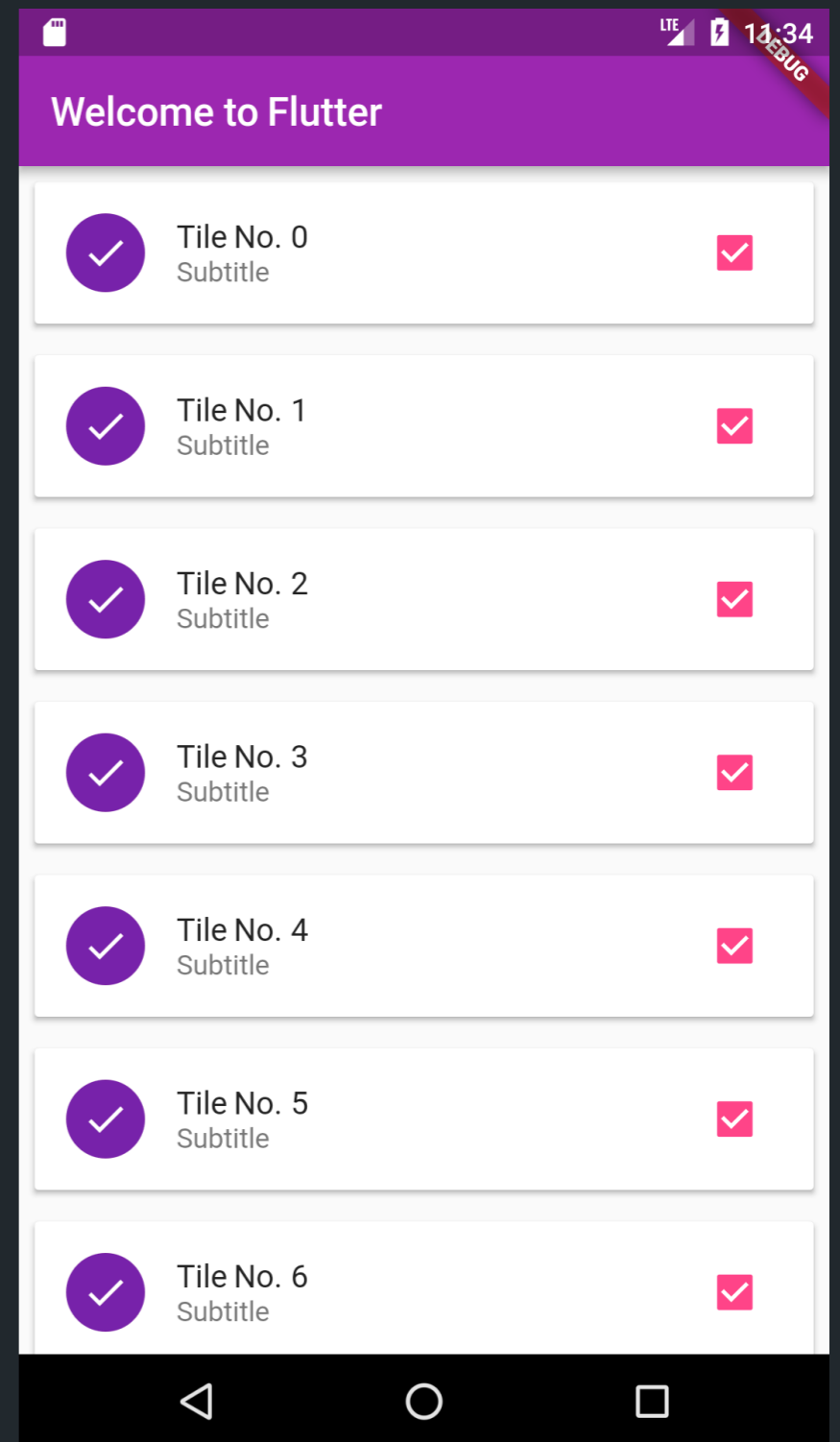


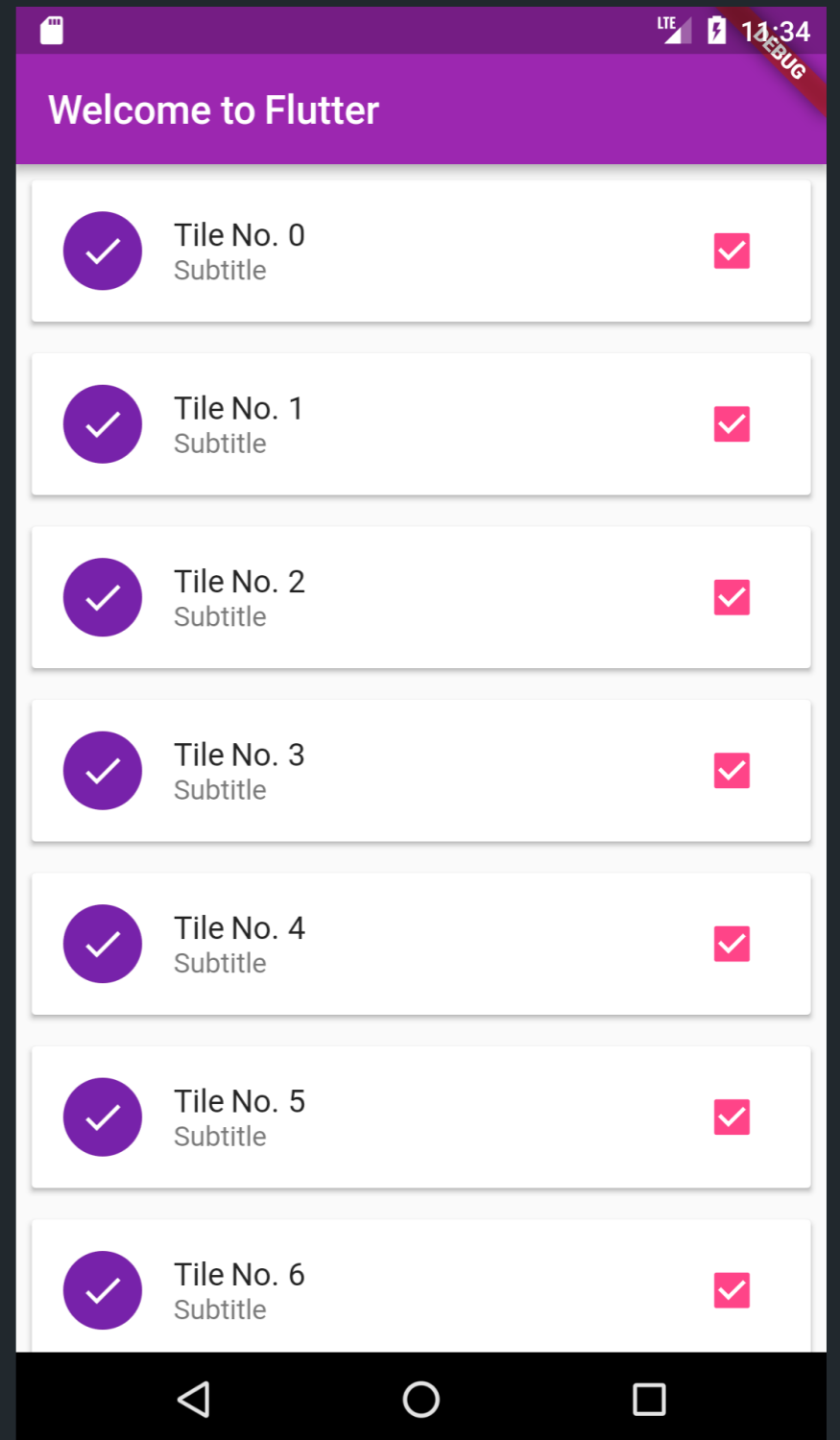
```
return MaterialApp(  
  title: 'FlutterApp',  
  home: Scaffold(  
    ...  
  ),  
);
```

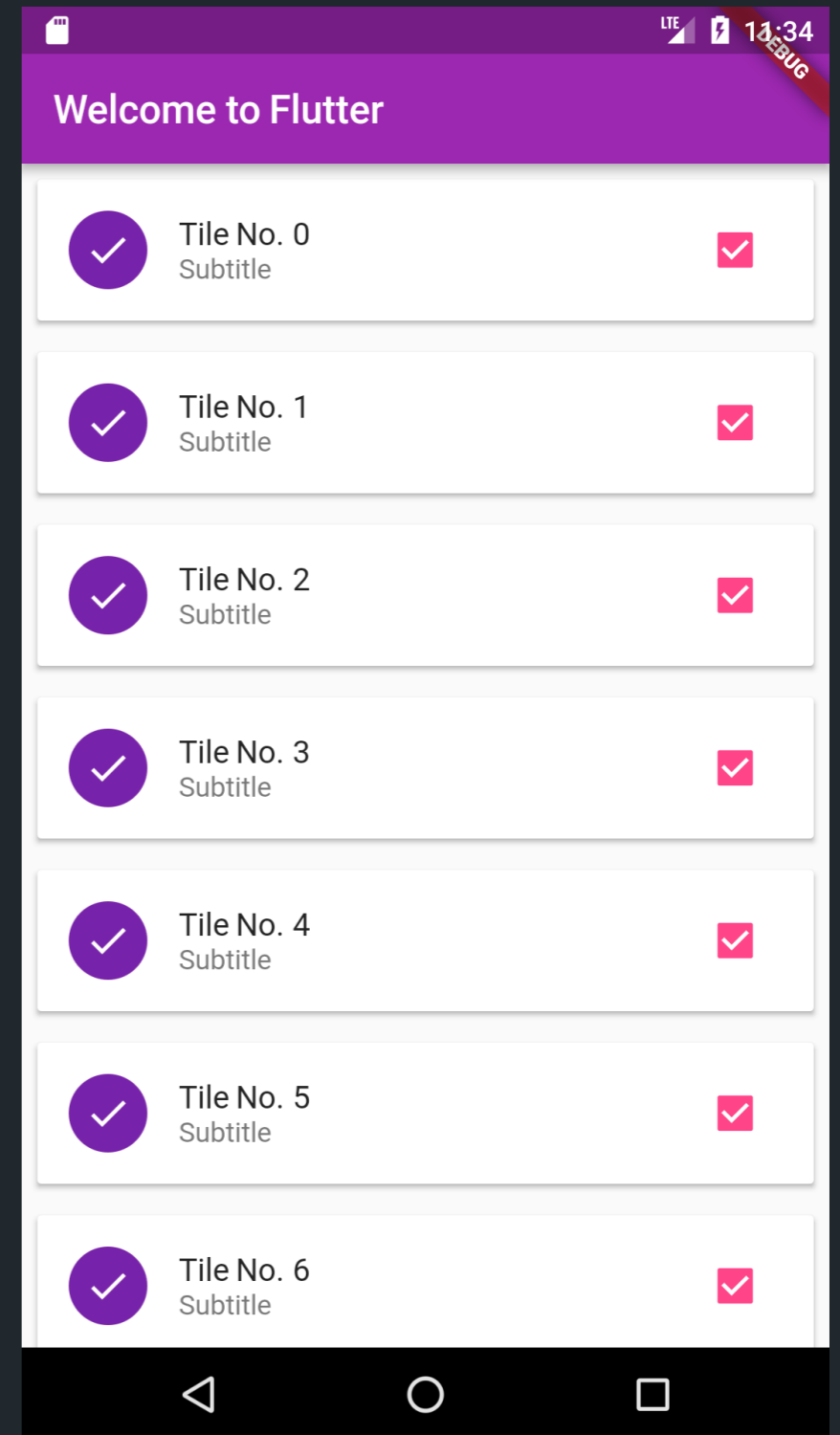
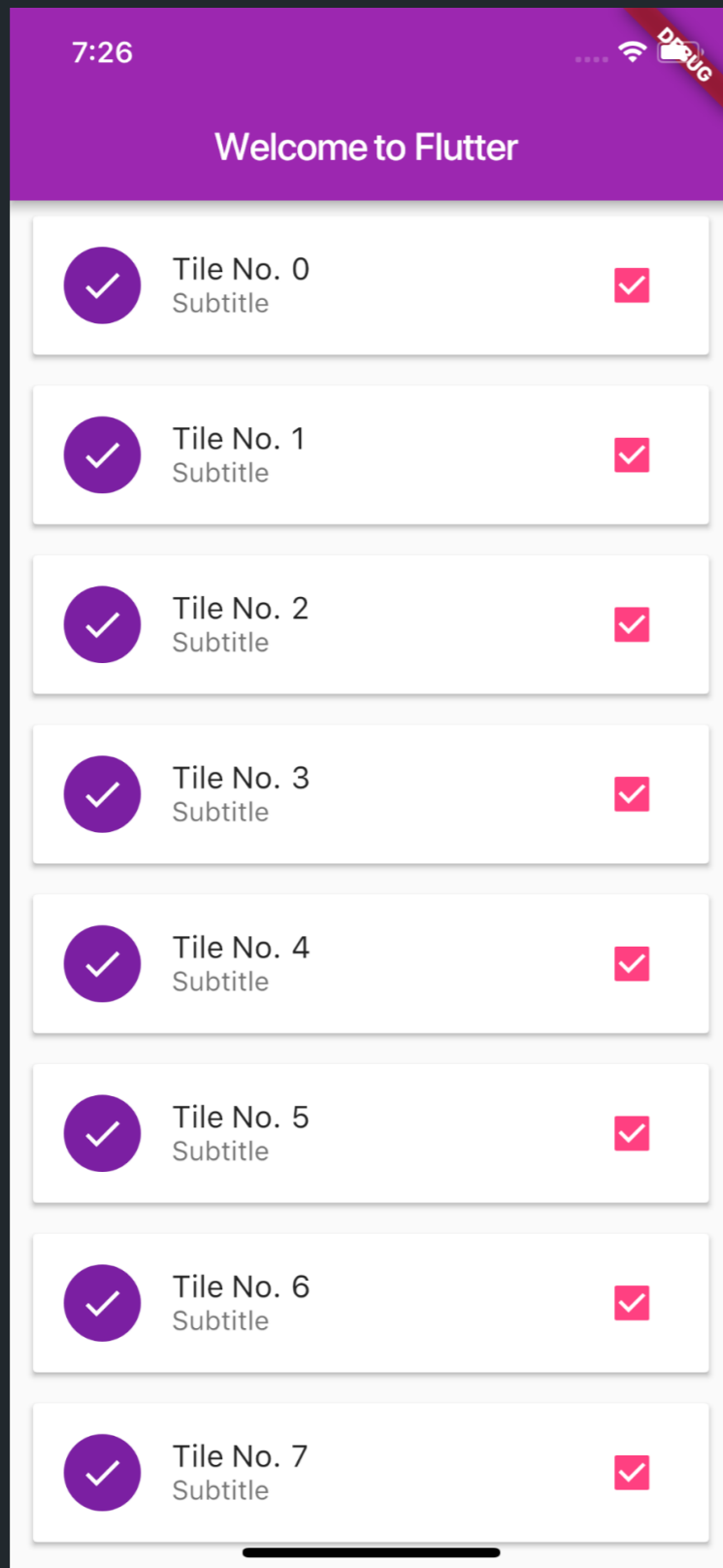
```
return MaterialApp(  
  title: 'FlutterApp',  
  theme: ThemeData(  
    primarySwatch: Colors.blue,  
    accentColor: Colors.pinkAccent,  
  ),  
  home: Scaffold(  
    ...  
  ),  
);
```

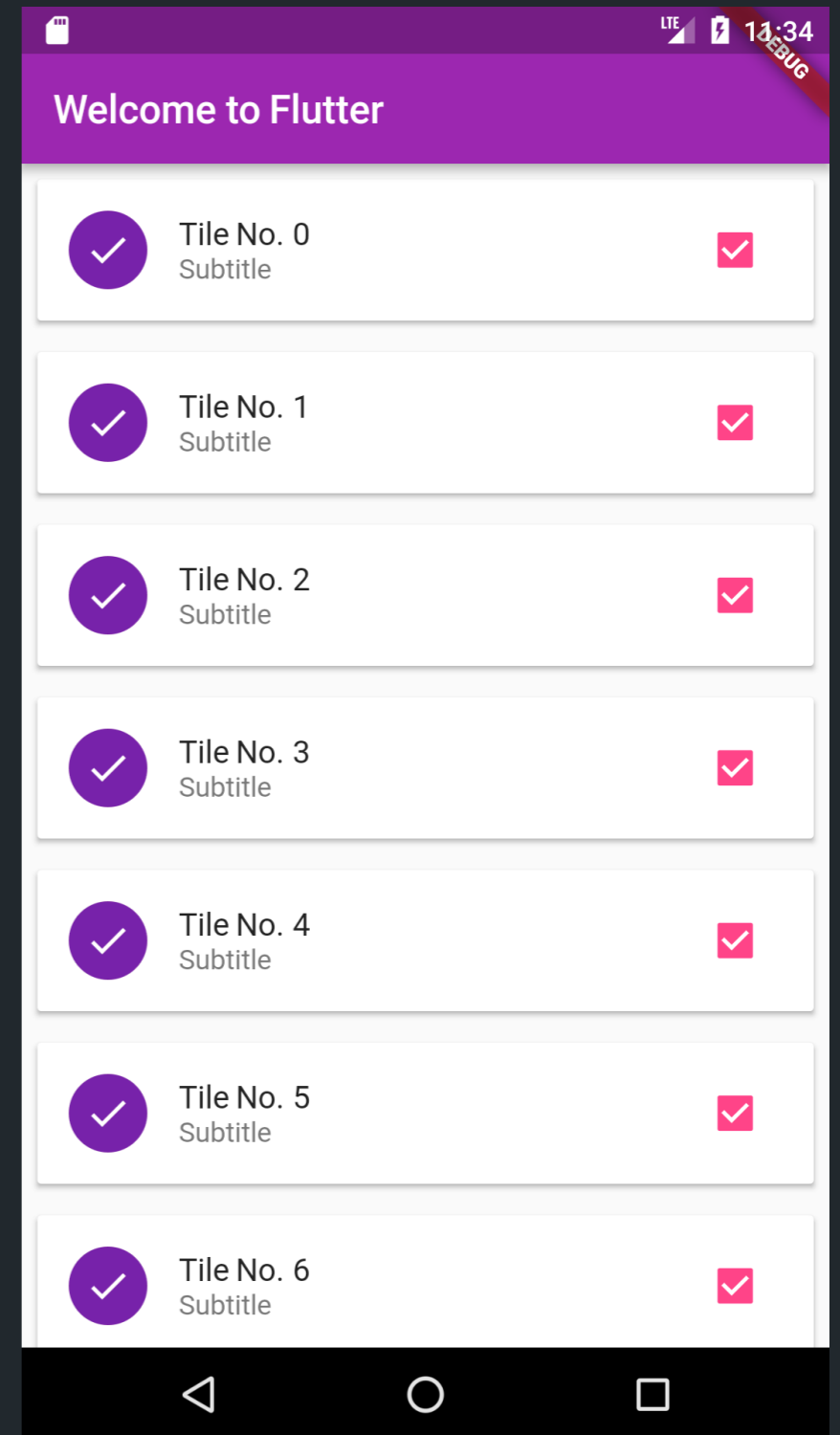
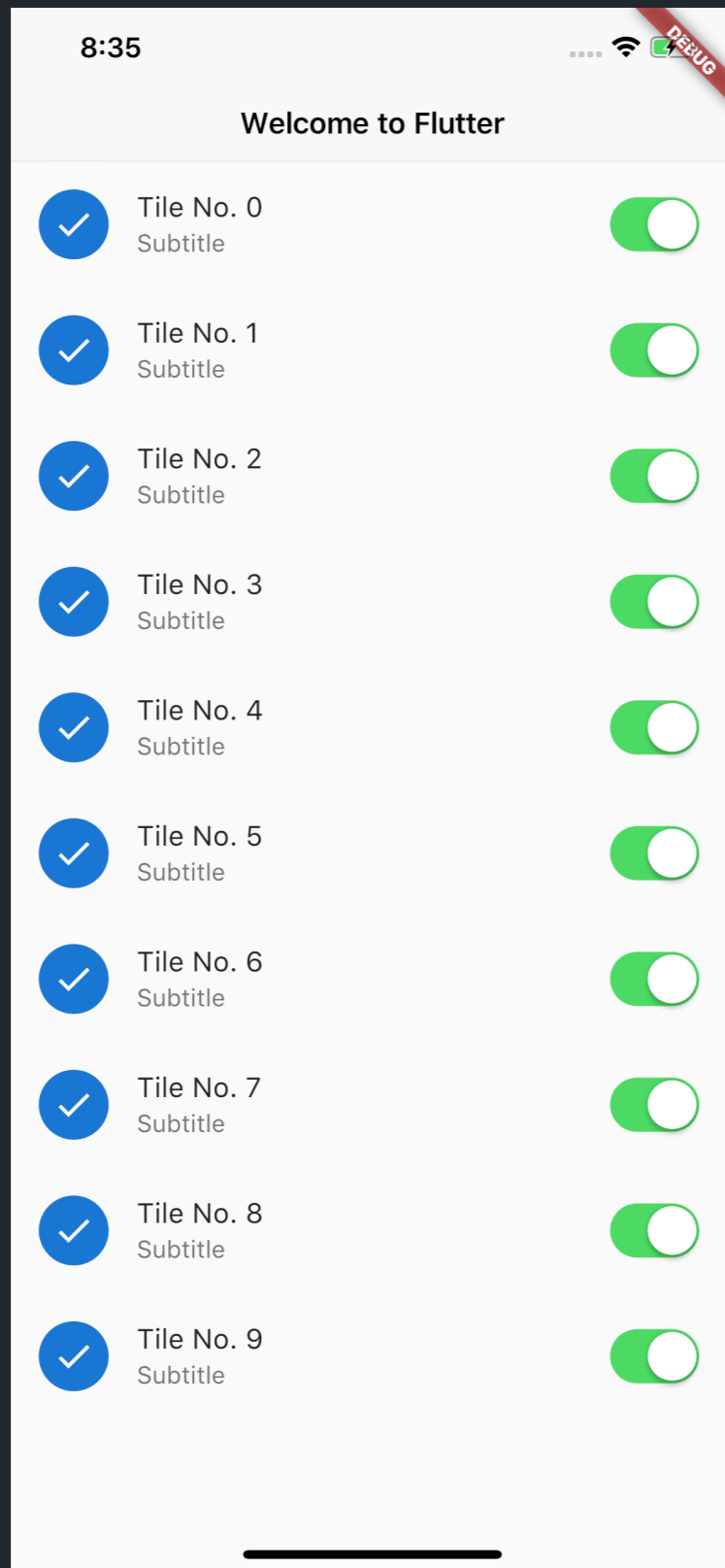


```
return MaterialApp(  
  title: 'FlutterApp',  
  theme: ThemeData(  
    primarySwatch: Colors.purple,  
    accentColor: Colors.pinkAccent,  
  ),  
  home: Scaffold(  
    ...  
  ),  
);
```









Testing?

```
void main() {
  testWidgets('widget has first and last elements',
    (WidgetTester tester) async {

    await tester.pumpWidget(MyApp());
    expect(find.byType(ListView), findsOneWidget);
    expect(find.text("Title No. 0"), findsOneWidget);

    await tester.drag(find.byType(ListView), Offset(0.0, -200.0));
    await tester.pump();

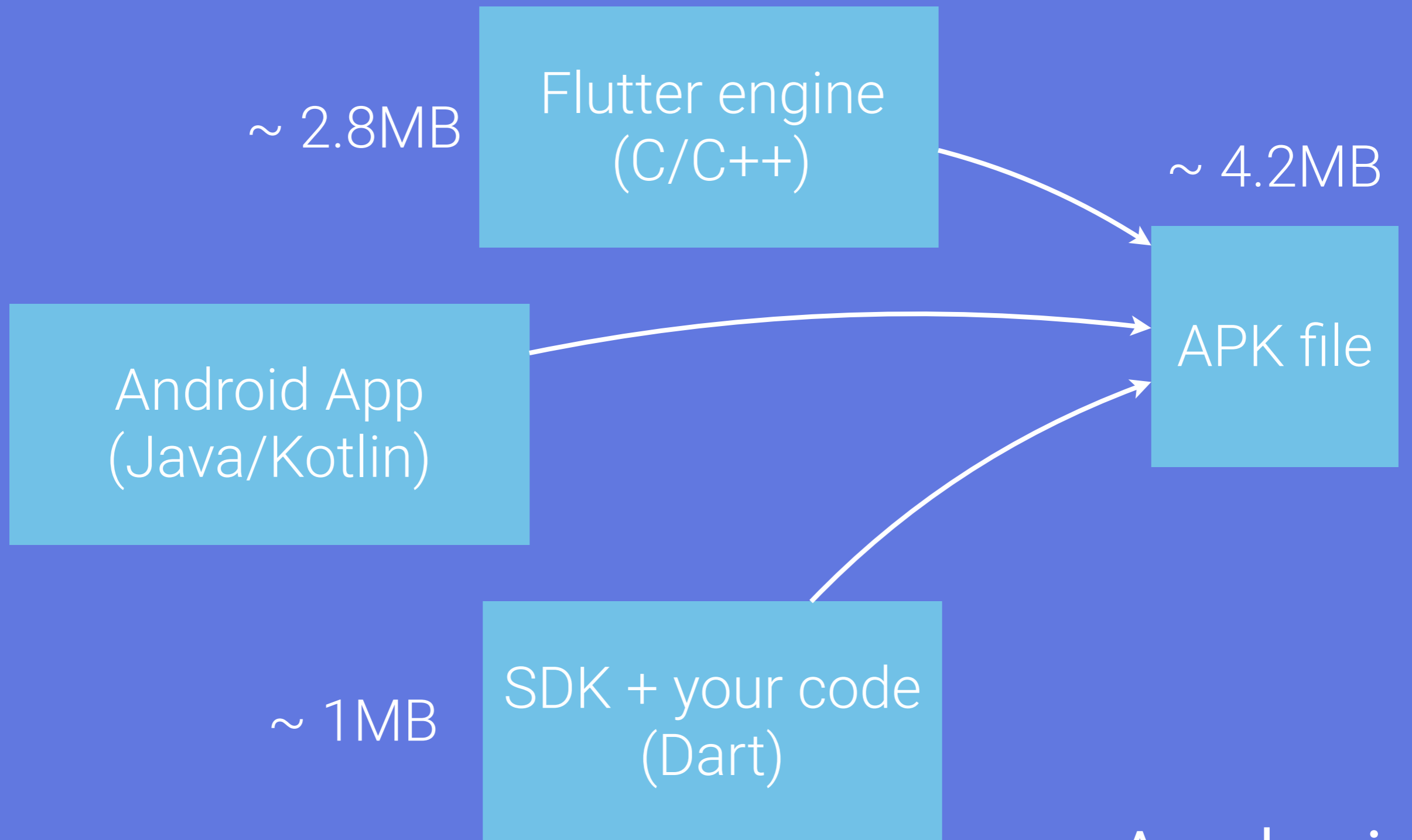
    expect(find.text("Title No. 9"), findsOneWidget);
  });
}
```

How does it work?

How does it work?

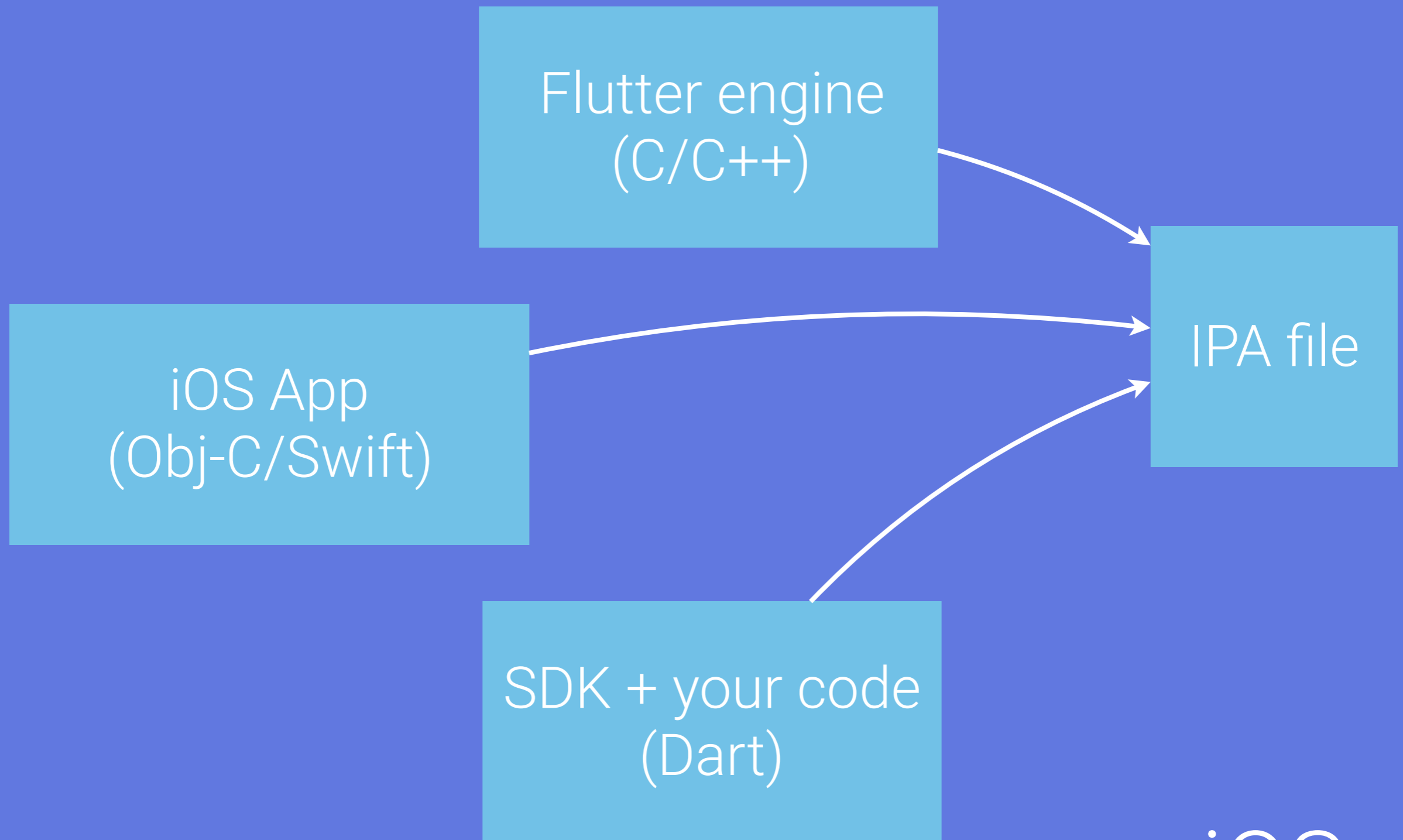


Building app process



Android

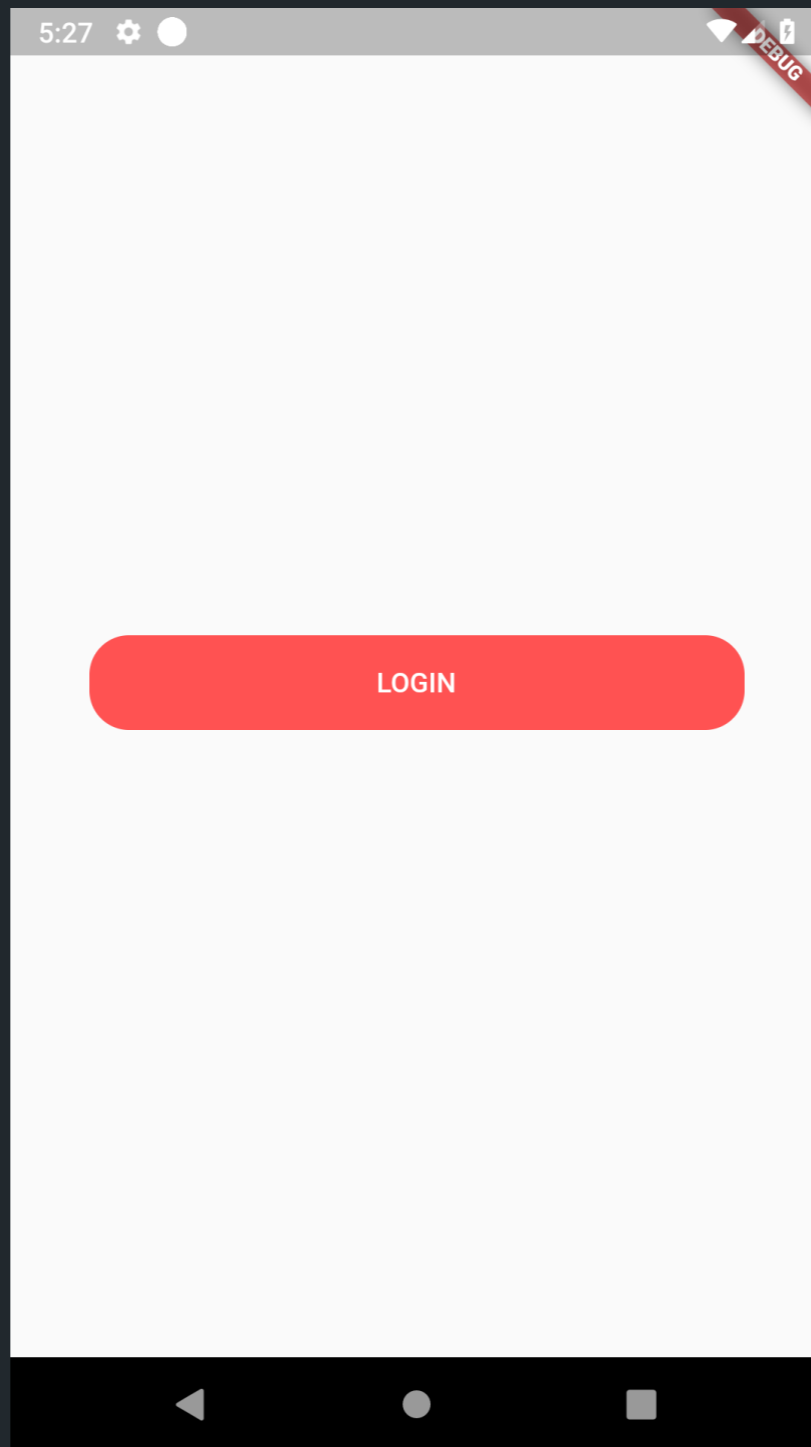
Building app process



iOS

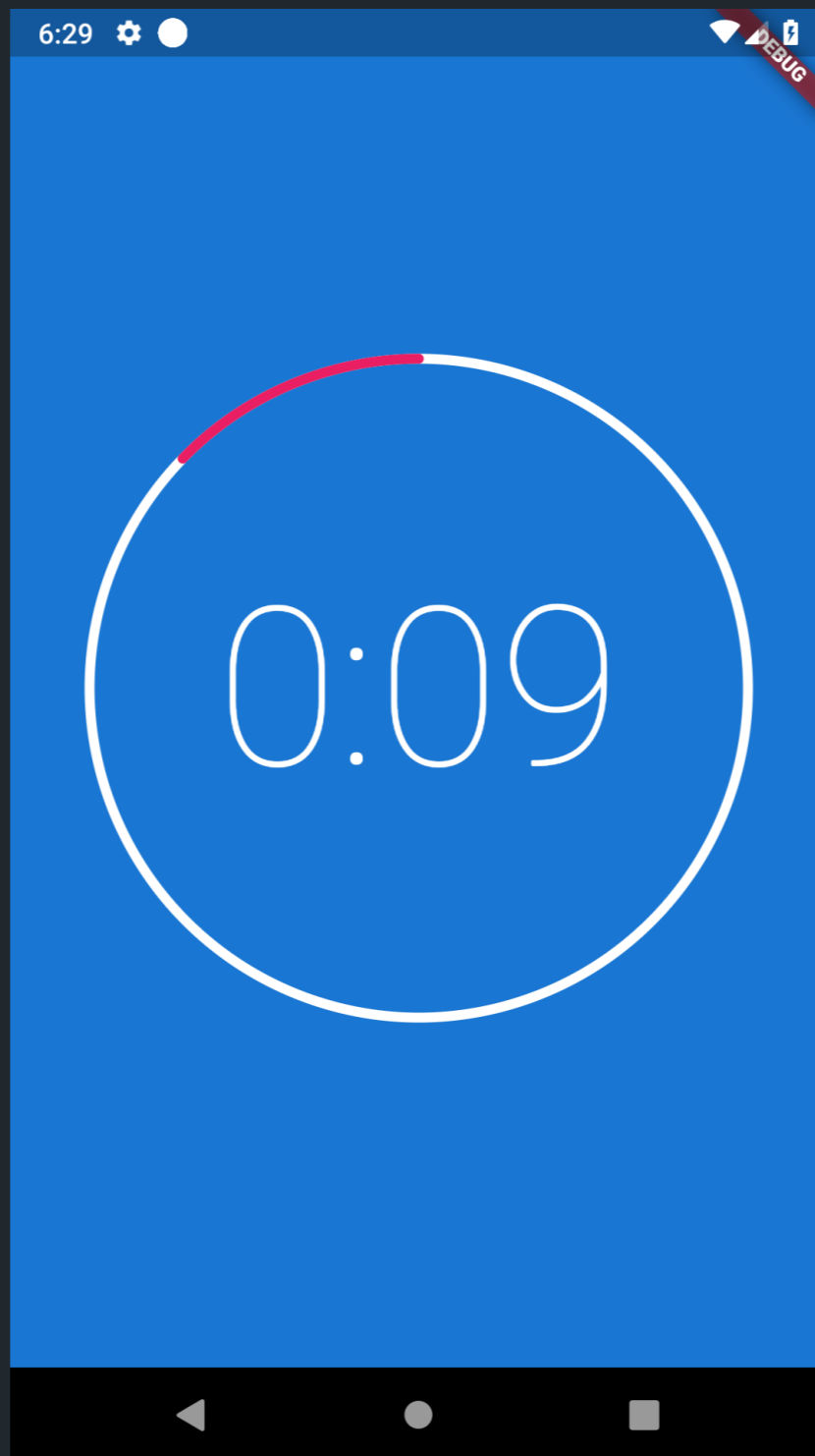
Cool, but...

What about custom UI?



```
class SimpleButton extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return FlatButton(
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(20.0),
      ),
      color: Colors.redAccent,
      child: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Text(
          "LOGIN",
          style: TextStyle(color: Colors.white),
        ),
      ),
      onPressed: () => {},
    );
  }
}
```

```
class SimpleButton extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return FlatButton(
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(20.0),
      ),
      color: Colors.redAccent,
      child: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Text(
          "LOGIN",
          style: TextStyle(color: Colors.white),
        ),
      ),
      onPressed: () => {},
    );
  }
}
```



```
class ProgressPainter extends CustomPainter {  
  
    @override  
    void paint(Canvas canvas, Size size) {  
        Paint paint = new Paint()  
            ..color = Colors.white  
            ..strokeWidth = 5.0  
            ..strokeCap = StrokeCap.round  
            ..style = PaintingStyle.stroke;  
  
        Offset center = size.center(Offset.zero);  
        double radius = size.width / 2.0;  
        canvas.drawCircle(center, radius, paint);  
  
        paint.color = Colors.pink;  
        paint.style = PaintingStyle.stroke;  
        canvas.drawArc(  
            Rect.fromCircle(center: center, radius: radius),  
            math.pi * 1.5, -0.8, false, paint);  
    }  
}
```

What about camera, BLE,
other services?

Accessing external services

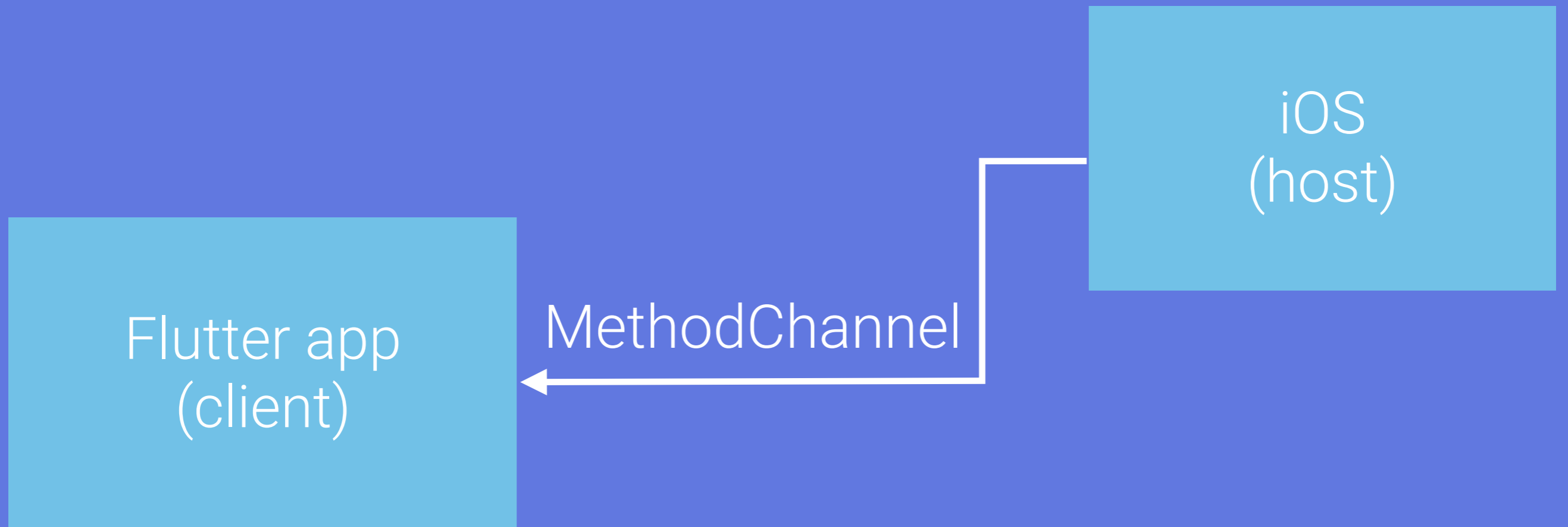
MethodChannel

Accessing external services

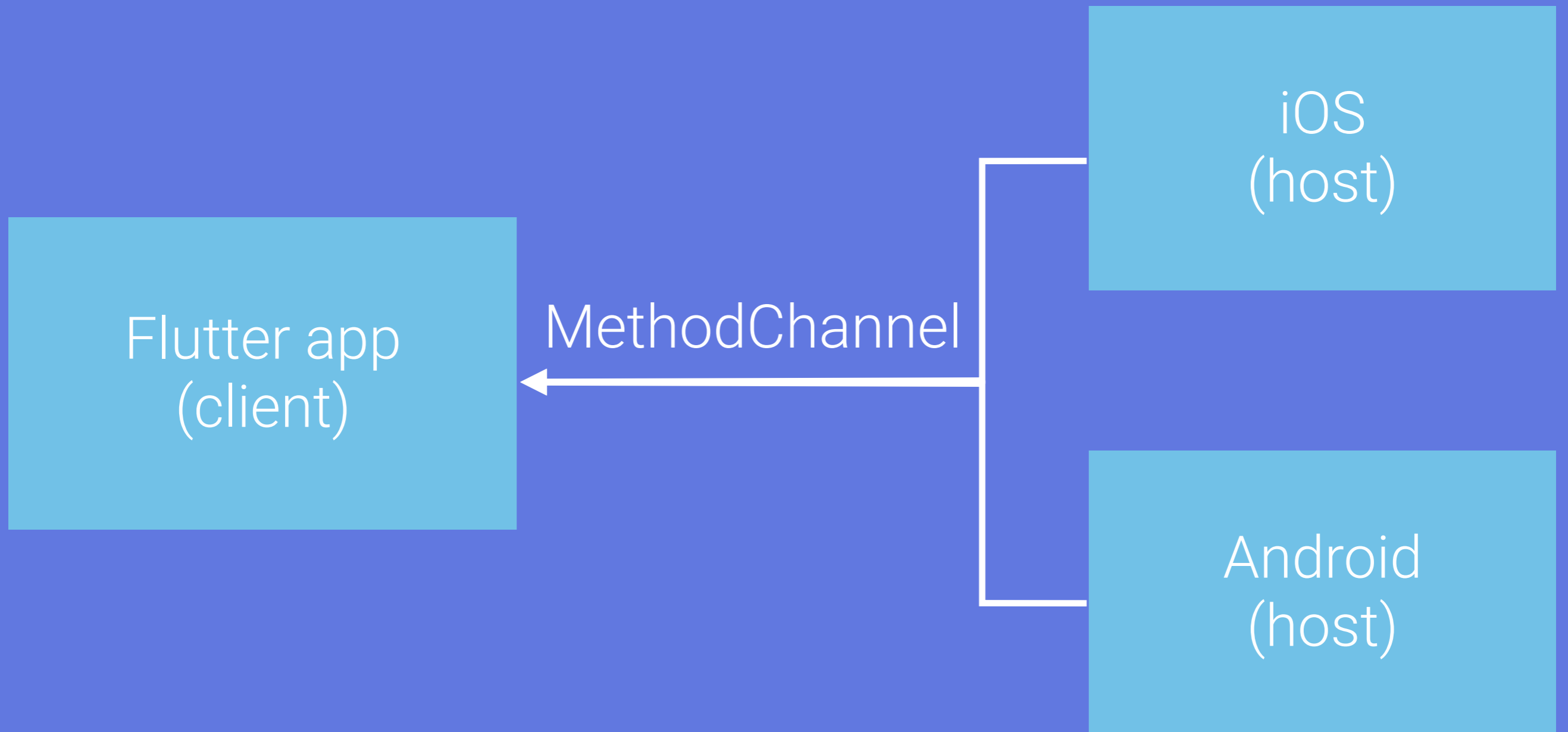
Flutter app
(client)

MethodChannel

Accessing external services



Accessing external services



```
class MainActivity() : FlutterActivity() {
    private val CHANNEL = „samples.flutter.io/battery”

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        GeneratedPluginRegistrant.registerWith(this)

        MethodChannel(flutterView, CHANNEL)
            .setMethodCallHandler { call, result ->
                // TODO
            }
    }
}
```

```
class MainActivity() : FlutterActivity() {
    private val CHANNEL = „samples.flutter.io/battery”

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        GeneratedPluginRegistrant.registerWith(this)

        MethodChannel(flutterView, CHANNEL)
            .setMethodCallHandler { call, result ->
                // TODO
            }
    }
}
```

```
if (call.method.equals("getBatteryLevel")) {  
    int batteryLevel = getBatteryLevel();  
    if (batteryLevel != -1) {  
        result.success(batteryLevel);  
    } else {  
        result.error("UNAVAILABLE",  
            "Battery level not available.", null);  
    }  
} else {  
    result.notImplemented();  
}
```

```
if (call.method.equals("getBatteryLevel")) {  
    int batteryLevel = getBatteryLevel();  
    if (batteryLevel != -1) {  
        result.success(batteryLevel);  
    } else {  
        result.error("UNAVAILABLE",  
            "Battery level not available.", null);  
    }  
} else {  
    result.notImplemented();  
}
```



```
if (call.method.equals("getBatteryLevel")) {
    int batteryLevel = getBatteryLevel();
    if (batteryLevel != -1) {
        result.success(batteryLevel);
    } else {
        result.error("UNAVAILABLE",
            "Battery level not available.", null);
    }
} else {
    result.notImplemented();
}
```

```
if (call.method.equals("getBatteryLevel")) {  
    int batteryLevel = getBatteryLevel();  
    if (batteryLevel != -1) {  
        result.success(batteryLevel);  
    } else {  
        result.error("UNAVAILABLE",  
            "Battery level not available.", null);  
    }  
} else {  
    result.notImplemented();  
}
```

```
const platform =  
    const MethodChannel('samples.flutter.io/battery');  
final int result =  
    await platform.invokeMethod('getBatteryLevel');  
String batteryLevel = "Battery level at $result % .";
```

Dart Packages

firebase_auth

FLUTTER

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

google_sign_in

FLUTTER

Flutter plugin for Google Sign-In, a secure authentication system for signing in with a Google account 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.

connectivity

FLUTTER

Flutter plugin for discovering the state of the network (WiFi & mobile/cellular) connectivity on Android and iOS.

path_provider

FLUTTER

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

shared_preferences

FLUTTER

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

firebase_analytics

FLUTTER

Flutter plugin for Google Analytics for Firebase, an app measurement solution that provides insight on app usage and user engagement on Android and iOS.

video_player

FLUTTER

Flutter plugin for displaying inline video with other Flutter widgets on Android and iOS.

url_launcher

FLUTTER

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

firebase_database

FLUTTER

Flutter plugin for Firebase Database, a cloud-hosted NoSQL database with realtime data syncing across Android and iOS clients, and offline access.

firebase_storage

FLUTTER

Flutter plugin for Firebase Cloud Storage, a powerful, simple, and cost-effective object storage service for Android and iOS.

device_info

FLUTTER

Flutter plugin providing detailed information about the device (make, model, etc.), and Android or iOS version the app is running on.

Dart Packages

shared_preferences 0.4.3

Published Oct 1, 2018

FLUTTER

[README.md](#)

[CHANGELOG.md](#)

[Example](#)

[Installing](#)

[Versions](#)

100

Popularity:

100

Health:

100

Maintenance:

100

Overall:

100

Learn more about [scoring](#).

We analyzed this package on Oct 26, 2018, and provided a score, details, and suggestions below. Analysis was completed with status *completed* using:

- Dart: 2.1.0-dev.7.1.flutter-b99bcfd309
- pana: 0.12.5
- Flutter: 0.10.1

About

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

[Homepage](#)

[Repository \(GitHub\)](#)

[Issue Tracker](#)

[Report an issue \(GitHub\)](#)

[API Docs](#)

Author

Flutter Team

Uploader

zarah@google.com

goderbauer@google.com

jackson@google.com

mravn@google.com

bkonyi@google.com

What about limitations?

Limitations

- not possible to use native UI components (Google Maps)
- no web equivalent
- no support for 3D
- ARM only (no support for x86 in release)
- min. 4.2MB

Limitations

- ~~not possible to use native UI components (Google Maps)~~
- ~~no web equivalent~~
- no support for 3D
- ARM only (no support for x86 in release)
- min. 4.2MB

* since Flutter Live, 4th December 2018



Useful Resources

Flutter UI Codelabs

<https://codelabs.developers.google.com/codelabs/flutter/>

1 Overview

- 1 Overview
- 2 Set up your Flutter environment
- 3 Start a new Flutter project
- 4 Build the main user interface
- 5 Add a UI for composing messages
- 6 Add a UI for displaying messages
- 7 Animate your app
- 8 Apply finishing touches
- 9 Next steps
- Optional: Get the sample

Did you find a mistake? [Please file a bug.](#)

← Building Beautiful UIs with Flutter

🕒 86 min remaining

1. Overview

Flutter is an open source SDK for creating high-performance, high-fidelity mobile apps for iOS and Android. The Flutter framework makes it easy for you to build user interfaces that react smoothly in your app, while reducing the amount of code required to synchronize and update your app's view.

Flutter makes it easy to get started building beautiful apps, with its rich set of [Material Design](#) and Cupertino (iOS) widgets and behaviors. Your users will love your app's natural look and feel, because Flutter implements platform-specific scrolling, navigational patterns, fonts, and more. You'll feel powerful and productive with Flutter's functional-reactive framework and our extremely fast hot reloads on devices and emulators.

You'll write your Flutter apps in [Dart](#). Dart syntax should look familiar if you already know Java, JavaScript, C#, or Swift. Dart is compiled using the standard [Android](#) and [iOS](#) toolchains for the specific mobile platform where your app needs to run. You get all the benefits of the Dart language, including familiar and terse [syntax](#), [first-class functions](#), [async/await](#), [rich standard libraries](#), and more.

What you'll learn

- ✓ How to write a Flutter app that looks natural on both iOS and Android.
- ✓ How to debug your Flutter app.
- ✓ How to run your Flutter app on a simulator/emulator and on a device.



Useful Resources

Flutter Firebase Codelabs

<https://codelabs.developers.google.com/codelabs/flutter-firebase>

← Firebase for Flutter 🕒 78 min remaining

1. Overview

In this codelab, you'll learn how to enable Firebase features in an existing simple Flutter app, Friendlychat. You'll also enhance the app to send and receive images stored on a mobile device, in addition to text.

The steps for building the Flutter version of Friendlychat are described in an earlier codelab, [Building Beautiful UIs with Flutter](#). Both codelabs use the same software environment: the Flutter SDK, a set of development tools for iOS and another set for Android, and an IntelliJ IDE. If your environment is already set up for Flutter development, you can skip to [Open Friendlychat in IntelliJ](#).

If you're familiar with the Firebase codelabs, the Flutter codelabs demonstrate how to build a similar chat client also named Friendlychat, using Flutter. With Flutter, you can build Friendlychat for iOS and Android at the same time, from a single code base, and iterate quickly using developer tools like hot reload.

1 Overview

2 Set up your Flutter environment

3 Get the sample code

4 Open Friendlychat in IntelliJ

5 Set up Firebase integration

6 Sign users into your app

7 Track user activity

8 Authenticate users

9 Enable data syncing

Did you find a mistake? [Please file a bug.](#)

>

Facts

1. Flutter is cool.
2. Flutter is very fast (development, application).
3. Flutter is super fun.
4. Flutter is easy to start.
5. Flutter is beautiful.

Useful Resources

Do you want to...

See other Flutter applications?

<https://itsallwidgets.com>

Know more about state management?

<https://www.youtube.com/watch?v=zKXz3pUkw9A>

Know more about technical details?

<https://flutter.io/technical-overview/>

Read about 5 reasons why you may love Flutter?

<https://hackernoon.com/flutter-5-reasons-why-you-may-love-it-55021fdbf1aa>

Thanks!
Questions?