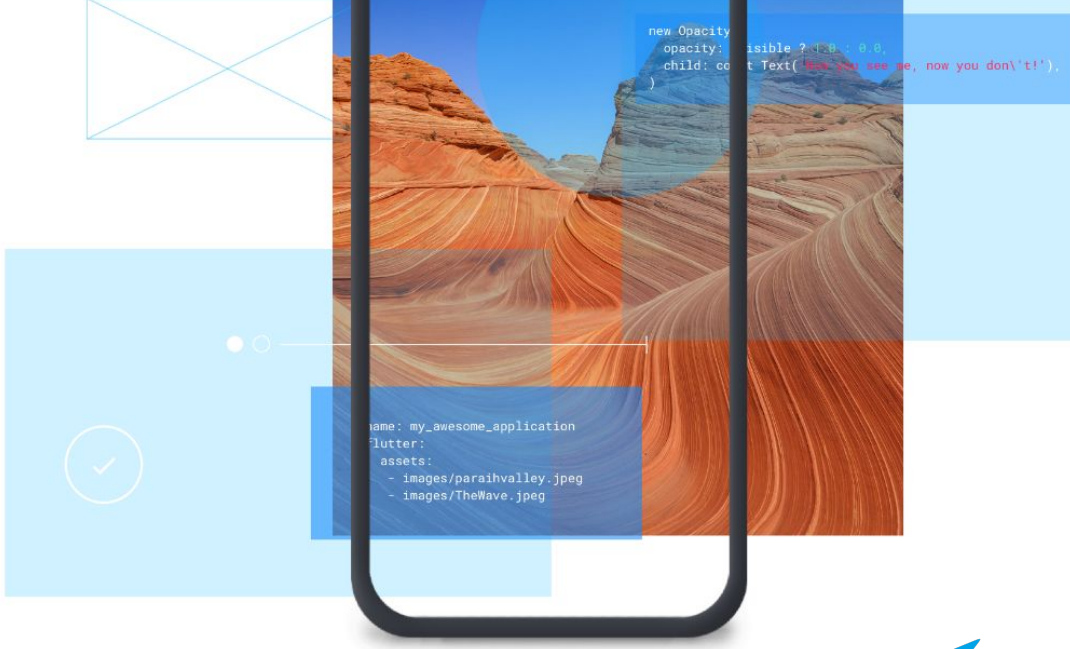
GDG Bari

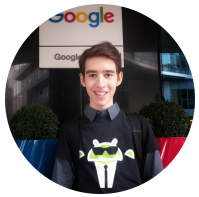
Flutter

Study Jam



Anna Labellarte

@anna_labe



Paolo Rotolo

@paolorotolo

Andrea Lops

@lopsandrea

Study Jam Agenda

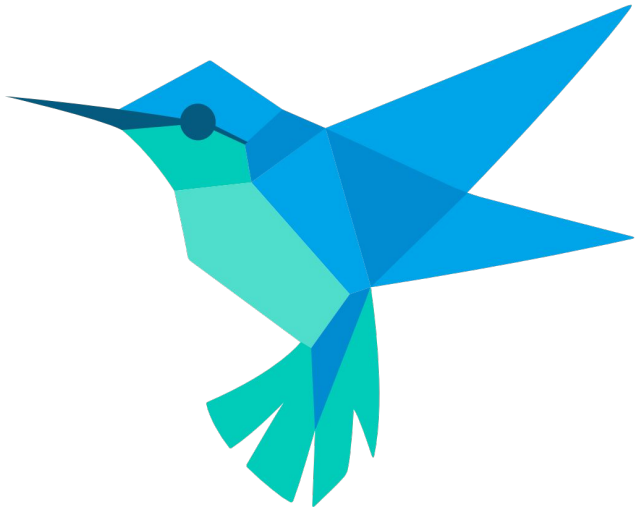
Study Jam

- **Lunedì 19 Novembre**
- Mercoledì 21 Novembre
- Lunedì 26 Novembre
- Mercoledì 28 Novembre

Consegna attestati - Watch Party Flutter Live

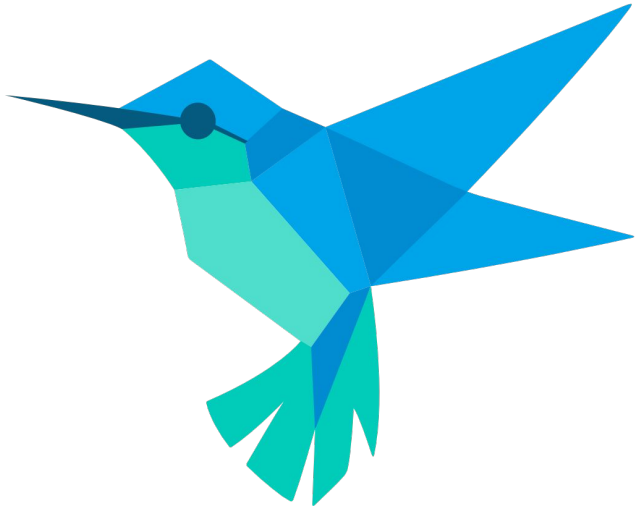
- **Martedì 4 Dicembre - 17:00**

Flutter Introduction



What is **Flutter**

Flutter is a mobile app SDK, complete with framework, widgets, and tools, that gives developers an easy and productive way to build and deploy beautiful mobile apps on both Android and iOS.



Mobile development

	Compiled to native	Interpreted (JavaScript)
MVC / MVVM		
Reactive		

Mobile development

	Compiled to native	Interpreted (JavaScript)
MVC / MVVM	iOS SDK Android SDK	
Reactive		

Mobile development

	Compiled to native	Interpreted (JavaScript)
MVC / MVVM	iOS SDK Android SDK	Cordova, Ionic, PhoneGap...
Reactive		

Mobile development

	Compiled to native	Interpreted (JavaScript)
MVC / MVVM	iOS SDK Android SDK	Cordova, Ionic, PhoneGap...
Reactive		React Native

Mobile development

	Compiled to native	Interpreted (JavaScript)
MVC / MVVM	iOS SDK Android SDK	Cordova, Ionic, PhoneGap...
Reactive	 FLUTTER	React Native

Native code
vs
Cross Platform

Challenges of mobile development today

“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

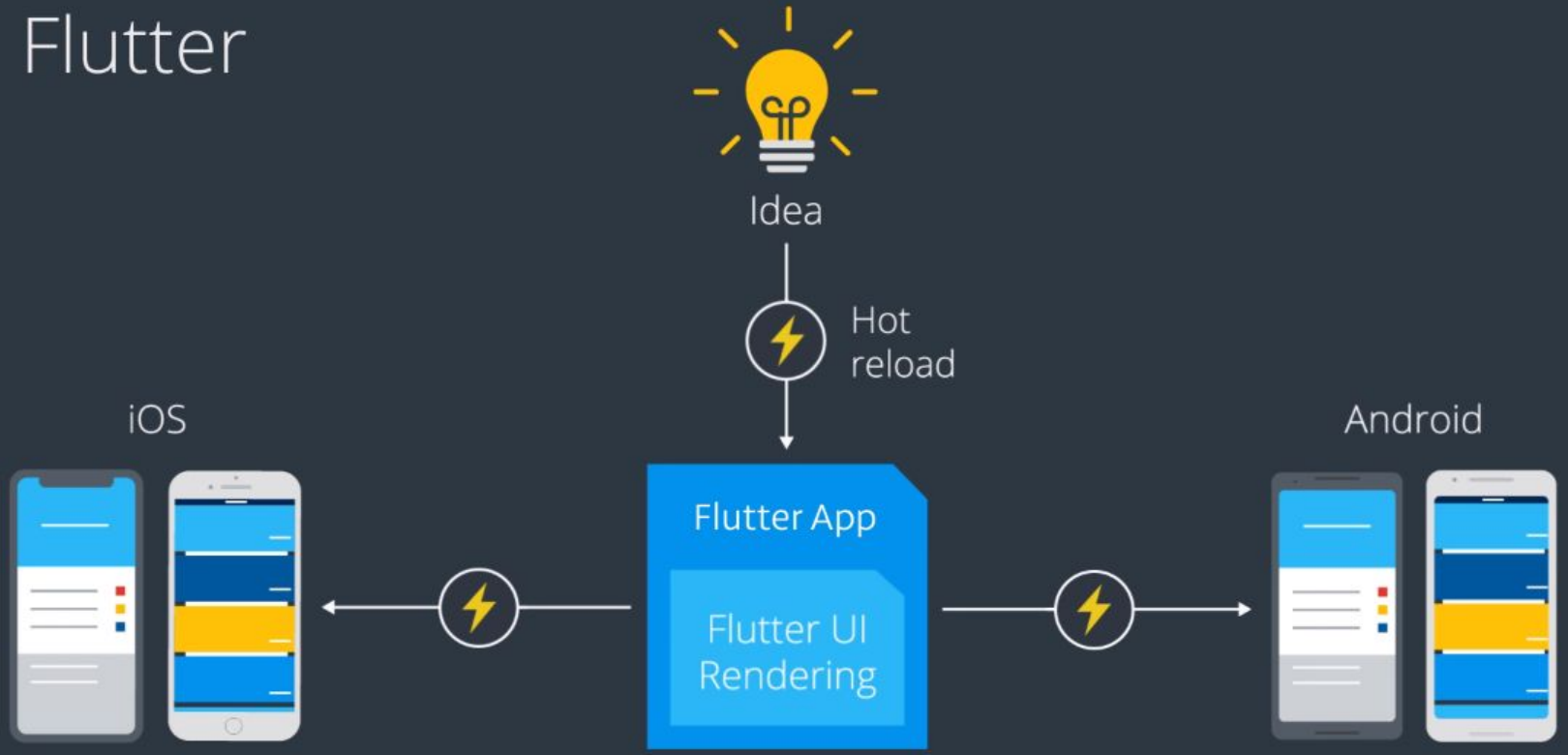
“Cross platform” approaches

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

Flutter Features

- Flutter includes a modern reactive framework
- Tools for fast development,
- Ready-made widgets which are the building blocks of the UI components in an app
- Compiles to native arm code
- Fast 2D rendering engine,

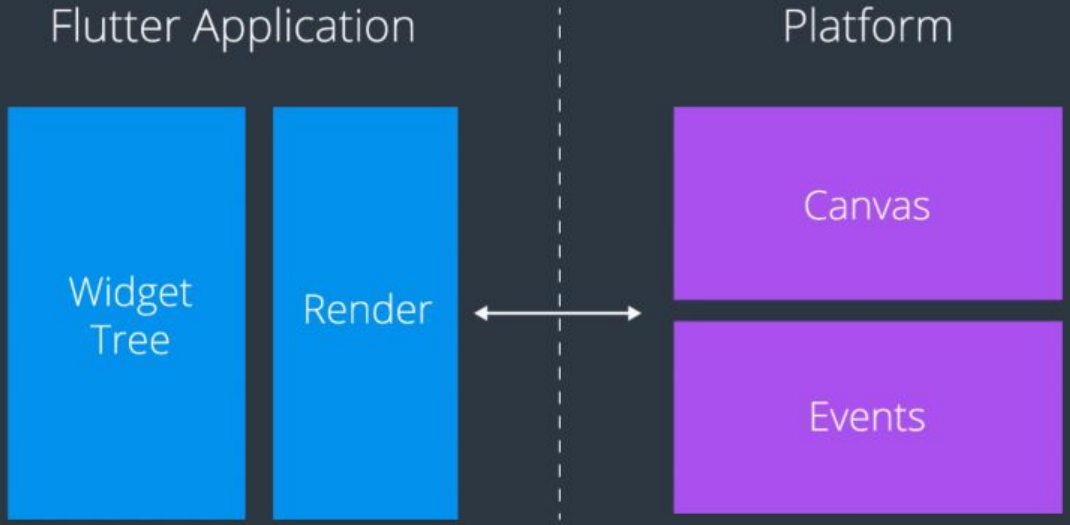
Flutter


Idea
Hot reload
iOS
Android
Flutter App
Flutter UI Rendering

Flutter Features

- Flutter includes a modern reactive framework
- Tools for fast development,
- Ready-made widgets which are the building blocks of the UI components in an app
- Compiles to native arm code
- Fast 2D rendering engine,

Using Flutter

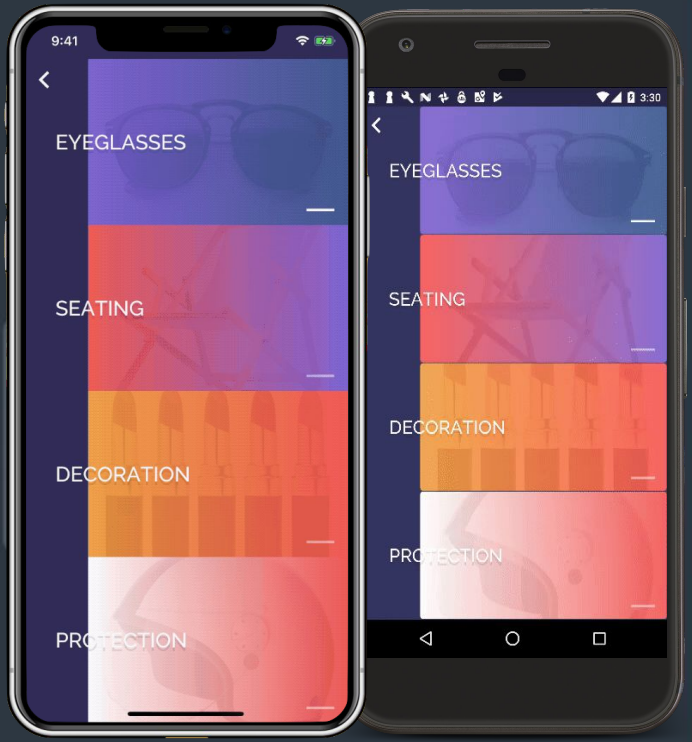


Flutter

Idea

Flutter App

Flutter UI Rendering

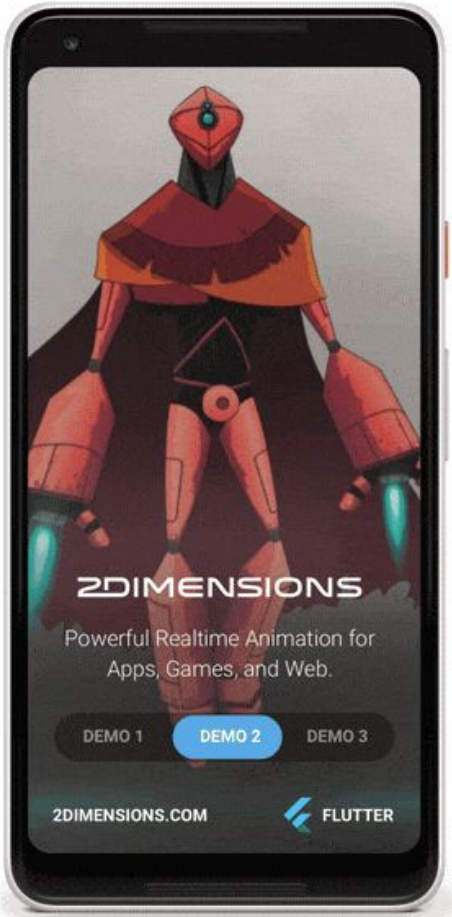


Flutter Features

- Flutter includes a modern reactive framework
- Tools for fast development,
- Ready-made widgets which are the building blocks of the UI components in an app
- Compiles to native arm code
- Fast 2D rendering engine,

Flutter’s Render Engine

Flutter seamlessly combines user interface widgets with 60fps animated graphics generated in real time, with the same code running on iOS and Android



Works with popular tools and platforms

Android Studio

Xcode

VS Code

Firebase

3rd-party Android SDKs

Android APIs

iOS APIs

Material Design

Redux

3rd-party iOS SDKs

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.

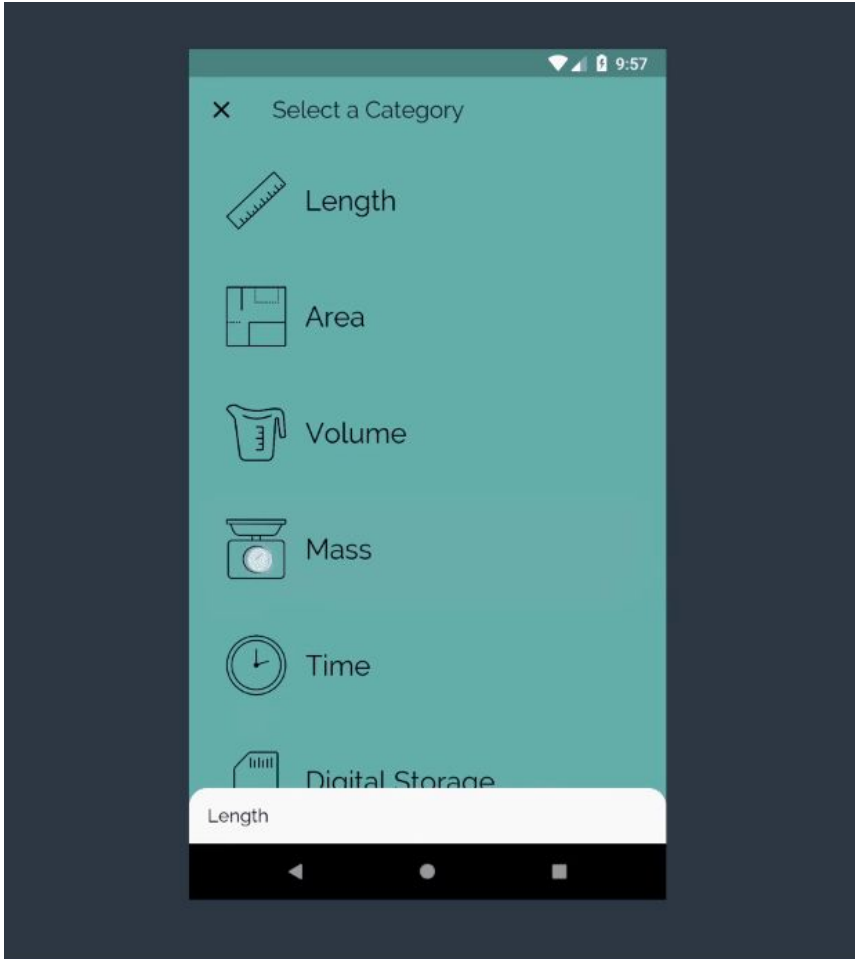
Bring your app to the *other* platform

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.

What will we build



Dart

Compilation and execution

Static languages
(Fortran, C, Java)

vs

Dynamic languages
(Smalltalk or JavaScript)

Compilation and execution

Static languages (Fortran, C, Java)	vs	Dynamic languages (Smalltalk or JavaScript)
compiled to native machine code		executed by an interpreter

Compilation and execution

Static languages
(Fortran, C, Java)

compiled to
native machine code

translated in
intermediate
language,
executed on VM

vs

Dynamic languages
(Smalltalk or JavaScript)

executed by an
interpreter

compiled
on the fly
(JIT)

Just In Time compilation

faster development cycles

slower execution

Ahead Of Time compilation

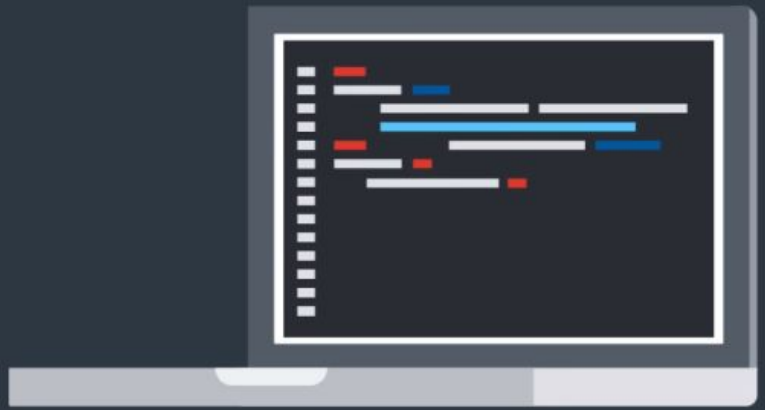
Faster and more predictably execution

slower development cycles

AOT compilation

JIT compilation

Hot Reload!



Avoiding **THE BRIDGE**

Preemptive scheduling, time slicing, and shared resources

Most languages with **multiple concurrent execution** (Java, Kotlin, Objective-C, and Swift) are **preemptive**.

Preemptive scheduling, time slicing, and shared resources

Most languages with **multiple concurrent execution** (Java, Kotlin, Objective-C, and Swift) are **preemptive**.

Can generate **race conditions**.

Preemptive scheduling, time slicing, and shared resources

Most languages with **multiple concurrent execution** (Java, Kotlin, Objective-C, and Swift) are **preemptive**.

Can generate **race conditions** fixed with locks.

Preemptive scheduling, time slicing, and shared resources

Most languages with **multiple concurrent execution** (Java, Kotlin, Objective-C, and Swift) are **preemptive**.

Can generate **race conditions** fixed with locks that can generate **deadlock** and **starvation**.

Preemptive scheduling, time slicing, and shared resources

Dart is **single threaded**, which means it does not allow preemption at all.

Dart code runs in the context of an **isolate**.

No other code in the same isolate can run concurrently.

Isolates **do not** share memory.

Unified layout

In Flutter, layouts are defined using **Dart code only**.

There is no XML/templating language.
There's no visual designer/storyboarding tool either.

Unified layout

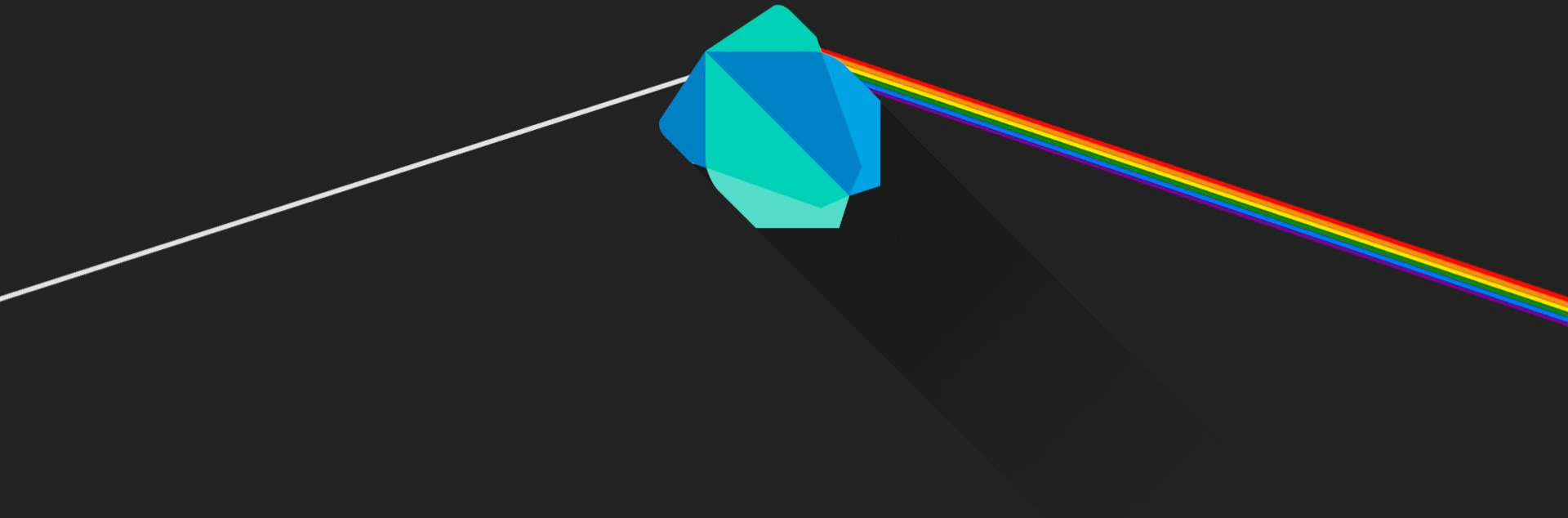
In Flutter, layouts are defined using **Dart code only**.

There is no XML/templating language.
There's no visual designer/storyboarding tool either.

```
return new ListView.builder(itemBuilder: (context, i) {
  if (i.isOdd) return new Divider();
  // rest of function
});
```


OPEN SOURCE =
memegenerator.net

The **Dart** side of the code



Comments

```
// This is a comment

/*
    This is a
    multiline comment.

    Such comment
    very documented
    so explanatory

    WOW

*/
```

Variables

```dart
String fact = "Flutter è un'opera D'art";
int answer = 42;

// or

var fact = "Flutter è un'opera D'art";
var answer = 42;

// Constants
const pi = 3.14;
```

Variables (and classes?)

```
void main() {
  print(Values.pi);
}

class Values {
    static const double pi = 3.14;
}
```

Variables (yep, again)

```
const pi = 3.14;


var list = [1, 2, 3];
// Implicit type is List<int>

print(list[0]);
```

Variables (I promise it's the last one)

```
var languages = {
  'Flutter': 'Dart',
  'Android': 'Java',
  'iOS' : 'Swift',
};


print(languages['Flutter']);
```

Variables (I promise it's the last one)

```
var languages = {
  'Flutter': 'Dart',
  'Android': 'Java',
  'iOS' : 'Swift',
};


print(languages['Flutter']);
languages['Android'] = 'Kotlin';
```

Variables (I lied)

```
List<int> numbers = [1, 2, 3, 4];

List<String> strings = ["Paolo", "Anna", "Lops"];

List<String> imConfused = [1, 2, 3, "4"];
```

Variables (I lied)

```
List<int> numbers = [1, 2, 3, 4];

List<String> strings = ["Paolo", "Anna", "Lops"];

List<dynamic> imConfused = [1, 2, 3, "4"];
```

String concatenation

```
var topic = "Flutter";


var welcome = "Welcome to $topic Study Jam!";
// This works too
var welcome = "Welcome to " + topic + " Study Jam!";

print(welcome);
// Welcome to Flutter Study Jam!
```

Functions

```
void main() {
  print(welcome("Paolo"));
}

String welcome(String name) {
  return "Welcome $name";
}

// OUTPUT: "Welcome Paolo"
```

Functions (even without types)

```
void main() {
  print(welcome("Paolo"));
}

welcome(String name) {
  return "Welcome $name";
}

// OUTPUT: "Welcome Paolo"
```

Functions (Dart is so clever)

```dart
void main() {
  print(welcome("Paolo"));
}

welcome(name) {
  return "Welcome $name";
}

// OUTPUT: "Welcome Paolo"
```

Functions in functions

```
void main() {

  welcome(name) {
    return "Welcome $name";
  }

  print(welcome("Paolo"));
}

// OUTPUT: "Welcome Paolo"
```

Functions (so terse, wow)

```
void main() {

  welcome(name) => "Welcome $name";


  print(welcome("Paolo"));
}

// OUTPUT: "Welcome Paolo"
```

Named parameters (spoilers, they're coming)

```
void main() {

  writeText(true, false);
}

void writeText(bool bold, bool hidden) { }
```

Named parameters (you called them ;))

```
void main() {

  writeText(bold: true);
}

void writeText({bool bold, bool hidden}) { }
```

Default parameters values (because, who likes null?)

```
/// Sets the [bold] and [hidden] flags ...
void writeText({bool bold = false, bool hidden = false}) {...}

// bold will be true; hidden will be false.
writeText(bold: true);
```

Assertions

```
// Make sure the variable has a non-null value.
assert(text != null);

// Make sure the value is less than 100.
assert(number < 100);

// Make sure this is an https URL.
assert(urlString.startsWith('https'));
```

Flutter <3 Dart

```
Text(
    "SPOILERS: I'm a Widget.",
    textAlign: TextAlign.center,
    textScaleFactor: 3,
)
```

```
const Text(this.data, {
    Key key,
    this.style,
    this.textAlign,
    this.textDirection,
    this.textScaleFactor,
    this.maxLines,
}) : assert(data != null),
       textSpan = null,
       super(key: key);
```

Everything is an object

```
void printElement(int element) {
  print(element);
}

var list = [1, 2, 3];

// Pass printElement as a parameter.
list.forEach(printElement);
```

Anonymous Functions (they're probably GDPR compliant)

```
var list = ['apples', 'bananas', 'oranges'];
list.forEach((item) {
  print('${list.indexOf(item)}: $item');
});


OUTPUT:
0: apples
1: bananas
2: oranges
```

Anonymous Functions (they're probably GDPR compliant)

```
var list = ['apples', 'bananas', 'oranges'];

list.forEach(
    (item) => print('${list.indexOf(item)}: $item')
);



OUTPUT:
0: apples
1: bananas
2: oranges
```

Classes (still boring stuff)

```
class Point {
  num x, y;

  Point(num x, num y) {
    // There's a better way to do this, stay tuned.
    this.x = x;
    this.y = y;
  }
}
```

Classes

```dart
class Point {
  num x, y;

  // Syntactic sugar for setting x and y
  // before the constructor body runs.
  Point(this.x, this.y);
}
```

Named constructors

```
class Point {
  num x, y;

  Point(this.x, this.y);

  // Named constructor
  Point.origin() {
    x = 0;
    y = 0;
  }
}
```

Extend classes

```
class Television {
  void turnOn() {
    _illuminateDisplay();
    _activateIrSensor();
  }
  // ...
}

class SmartTelevision extends Television {
  void turnOn() {
    super.turnOn();
    _bootNetworkInterface();
    _initializeMemory();
    _upgradeApps();
  }
  // ...
}
```

Extended classes

```
class Person {
  String firstName;

  Person.fromJson(Map data) {
    print('Person');
  }
}

class Employee extends Person {
  // Person does not have a default constructor;
  // you must call super.fromJson(data).
  Employee.fromJson(Map data) : super.fromJson(data) {
    print('Employee');
  }
}

main() {
  var emp = new Employee.fromJson({});
}
```

Abstract class

```
abstract class Doer {
  // Define instance variables and methods...

  void doSomething(); // Define an abstract method.
}

class EffectiveDoer extends Doer {
  void doSomething() {
    // Provide an implementation, so the method is not abstract here...
  }
}
```

Dart Language

Asynchrony Support

We'll see it in the **Future**
Await for it :)

Quiz: Write some Dart code!

dartpad.dartlang.org

class **Quad**
int side1, side2, side3, side4

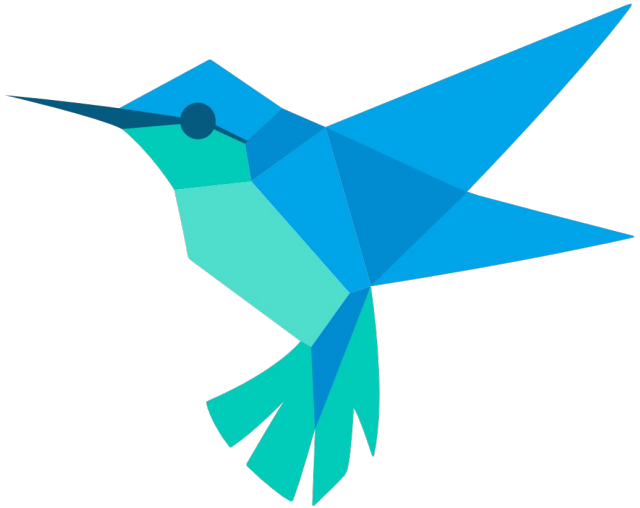
abstract class **QuadArea**
area()

class **Square(int lato)**
class **Rectangle(int lato1, int lato2)**
Extends **Quad,** implements **QuadArea**

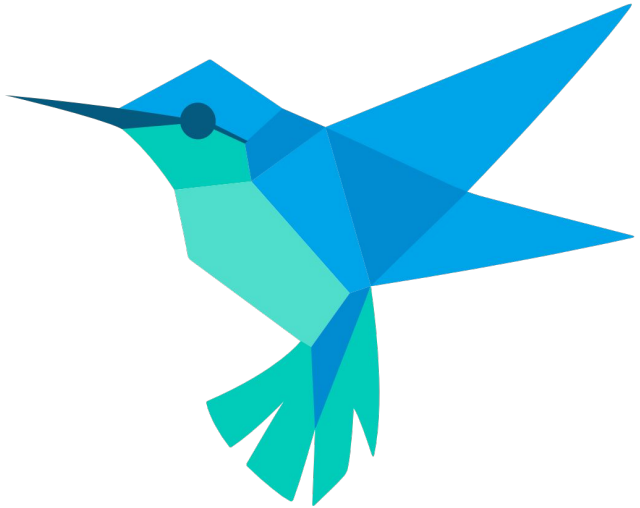
Main: Create a Square and Rectangle and print their area

Build your first Flutter app

10 minutes
Break

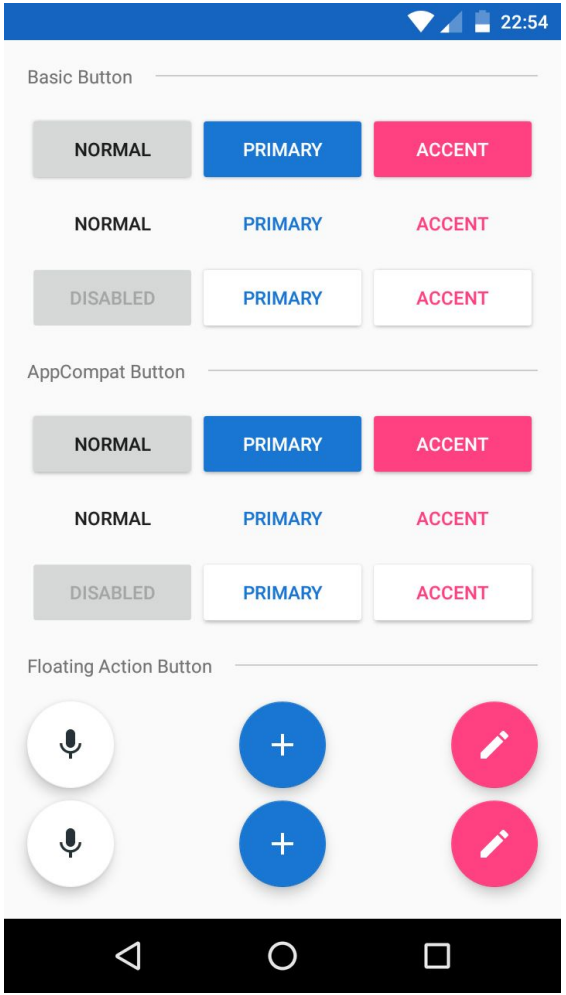


Flutter’s reactive framework



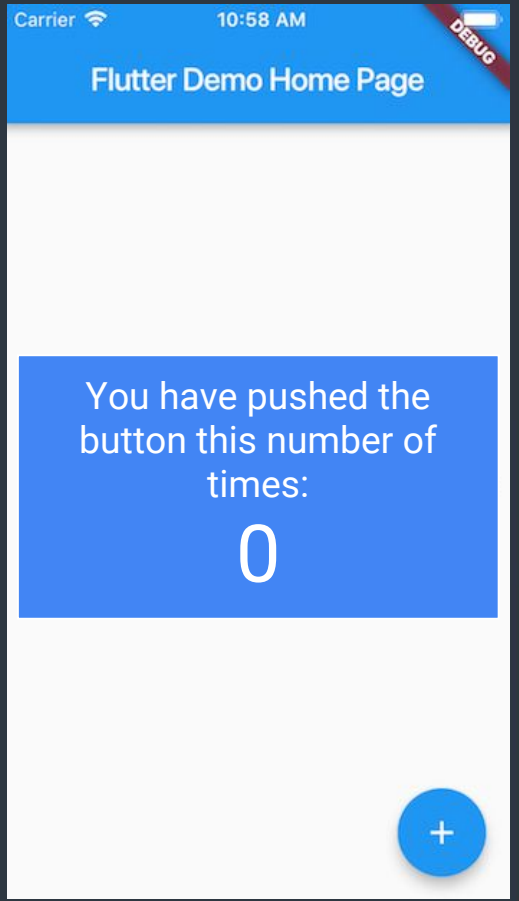
Widget

A widget is a description of part of a user interface and nearly everything is a widget, written in Dart.

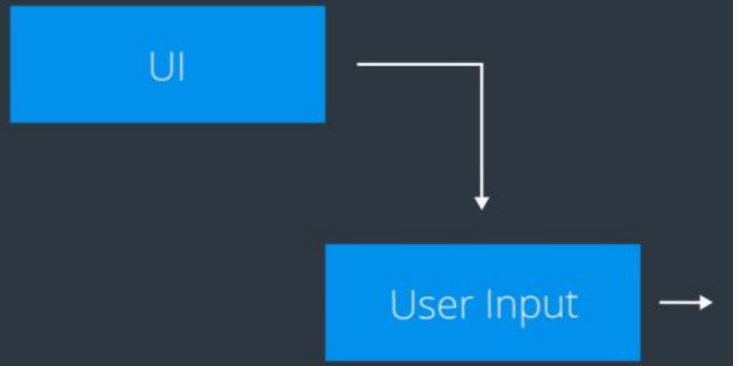


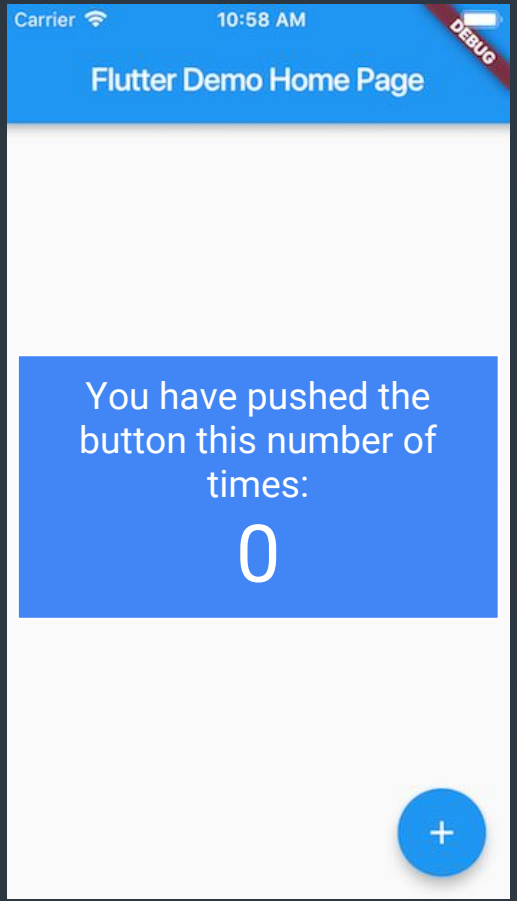
Widget lifecycle

UI

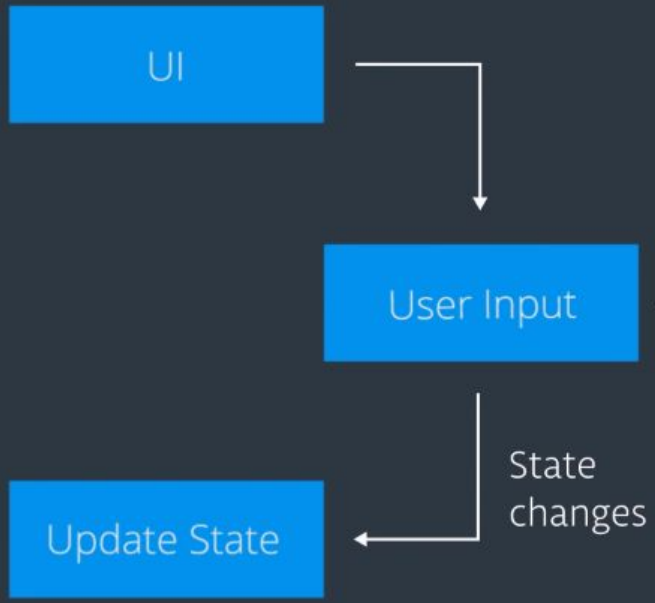


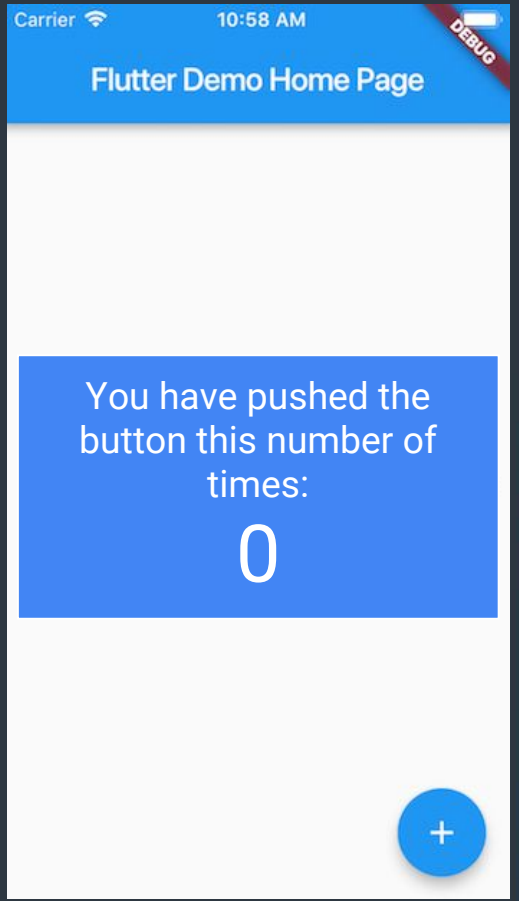
Widget lifecycle



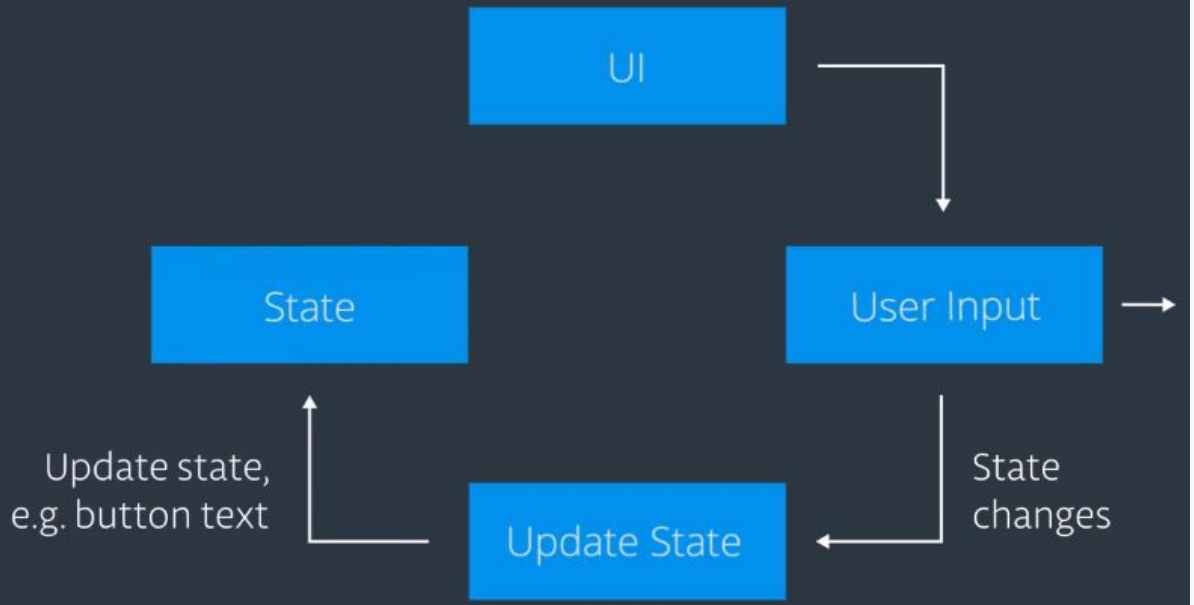


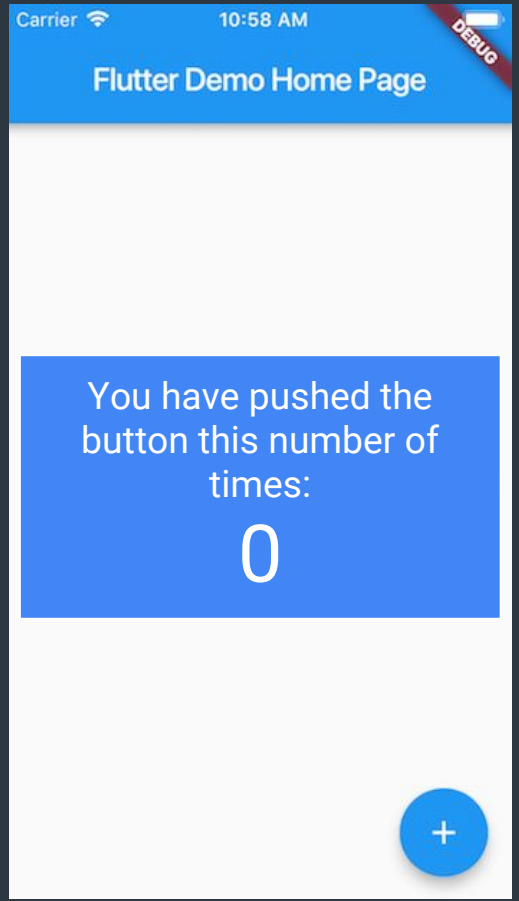
Widget lifecycle



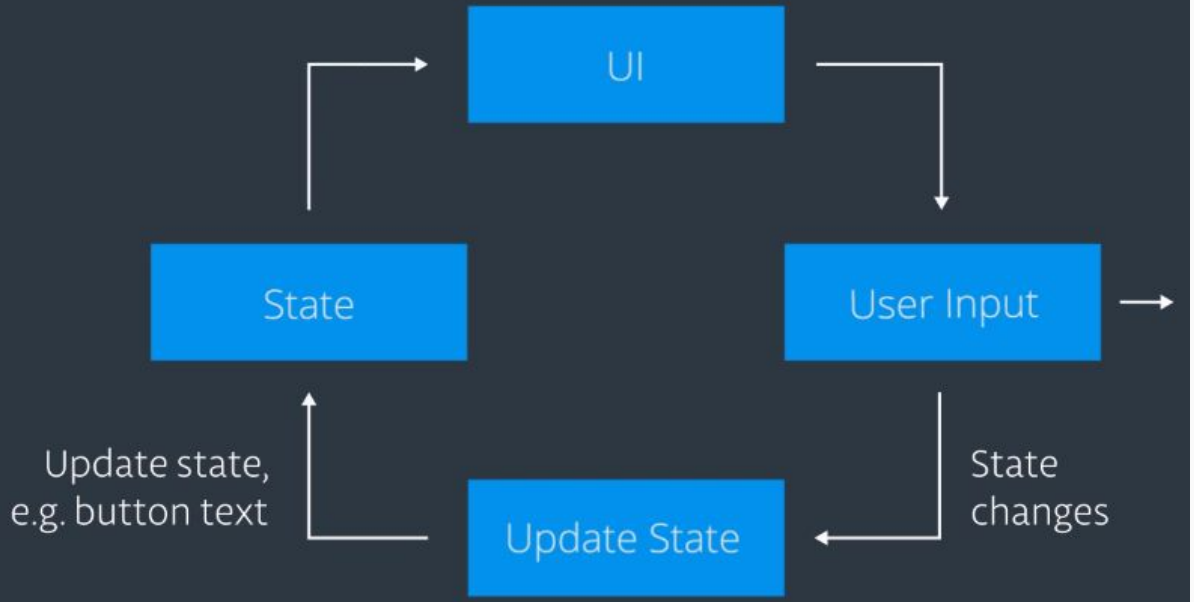


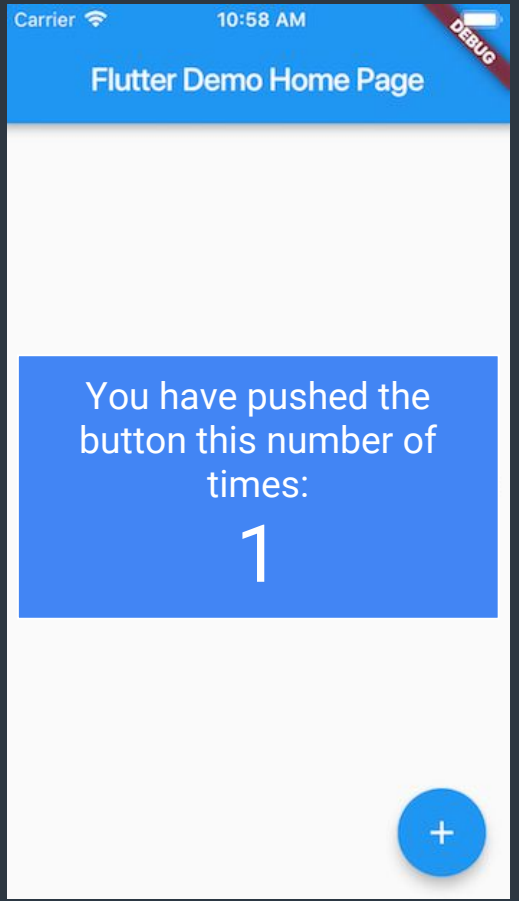
Widget lifecycle

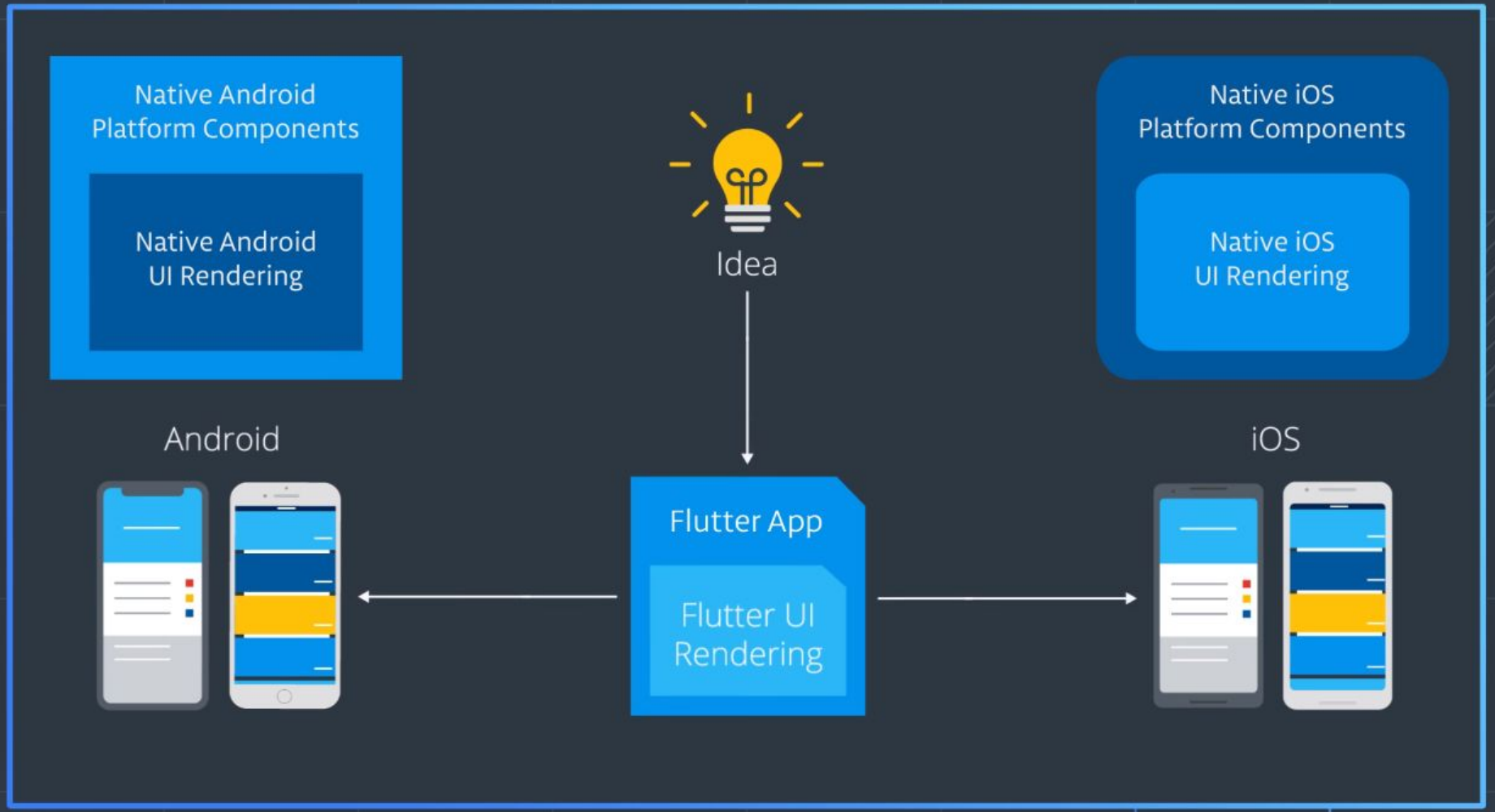




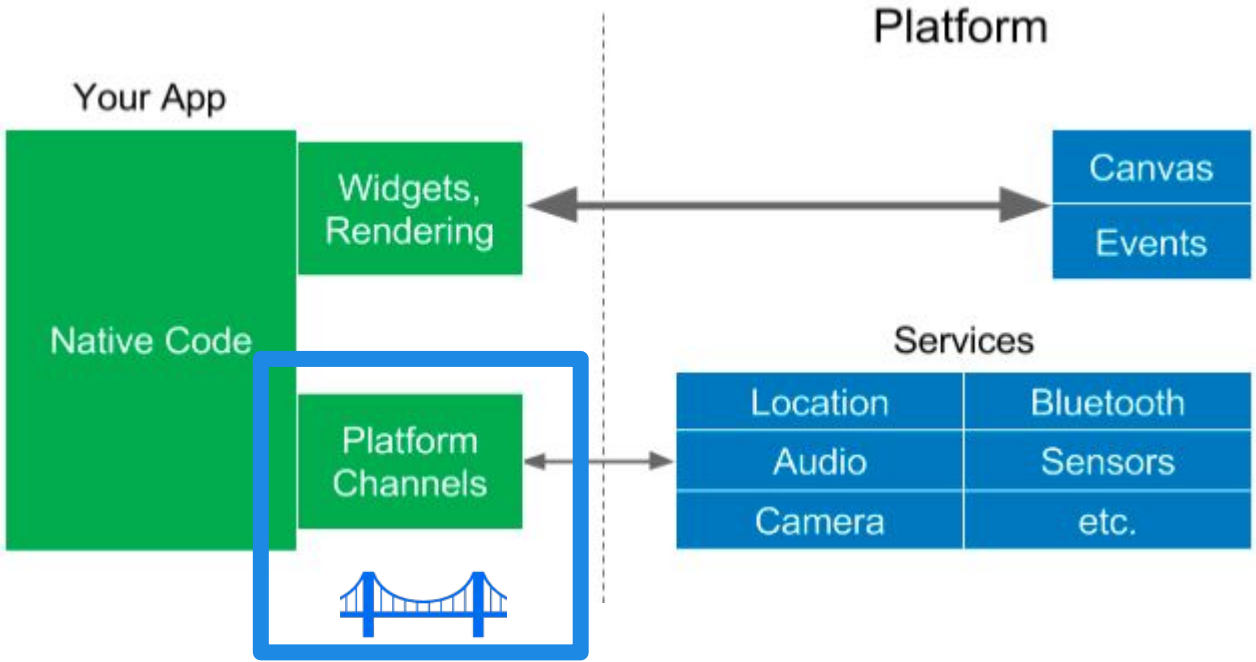
Widget lifecycle

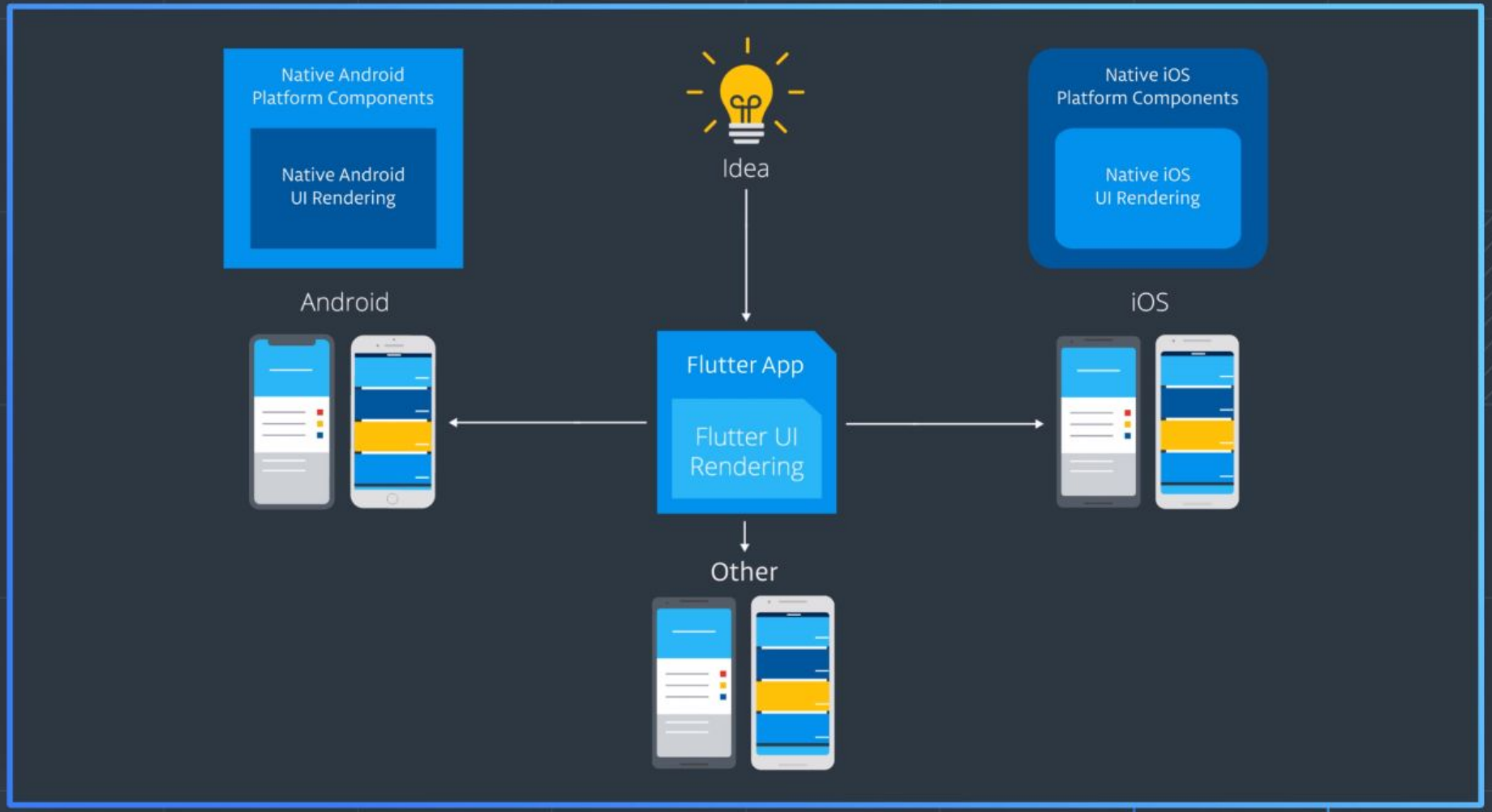





Native Android
Platform Components
Native Android
UI Rendering
Idea
Native iOS
Platform Components
Native iOS
UI Rendering
Android
iOS
Flutter App
Flutter UI
Rendering

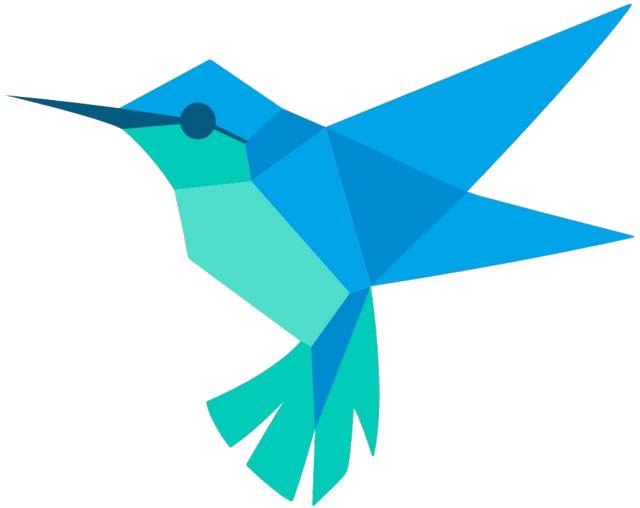
Inside a Flutter app


Platform
Your App
Widgets, Rendering
Canvas
Events
Native Code
Services
Location
Bluetooth
Platform Channels
Audio
Sensors
Camera
etc.


Native Android
Platform Components
Native Android
UI Rendering
Idea
Native iOS
Platform Components
Native iOS
UI Rendering
Android
iOS
Flutter App
Flutter UI
Rendering
Other

Let’s explore
Flutter’s dev tools

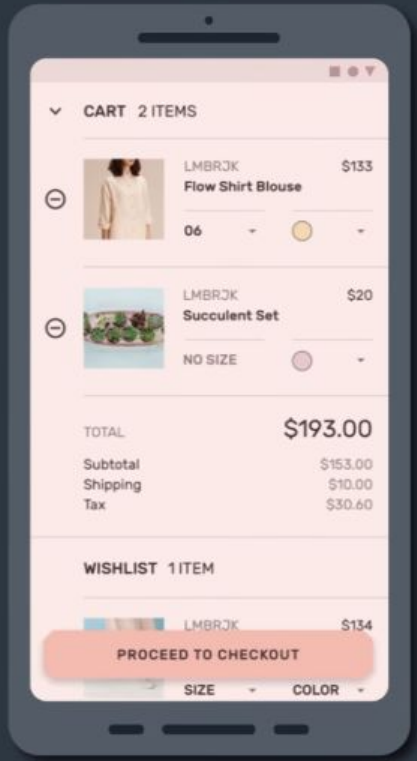
The wonderful things about Widget



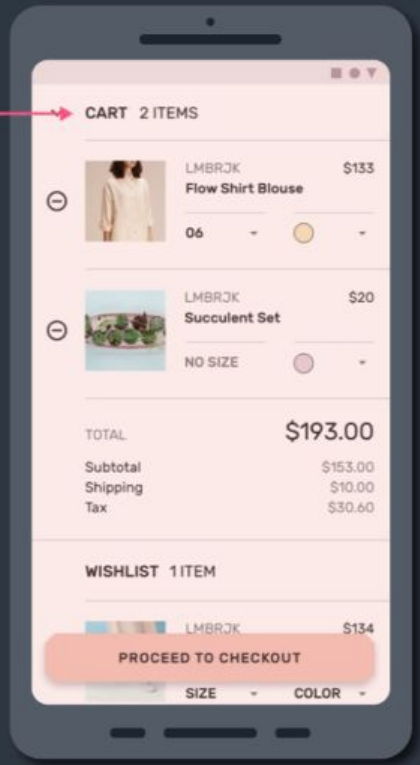
Widget

Widgets are the foundation of Flutter apps.

- Container
- Row/Column
- Icon
- IconButton
- Offstage
- Stack
- Expanded

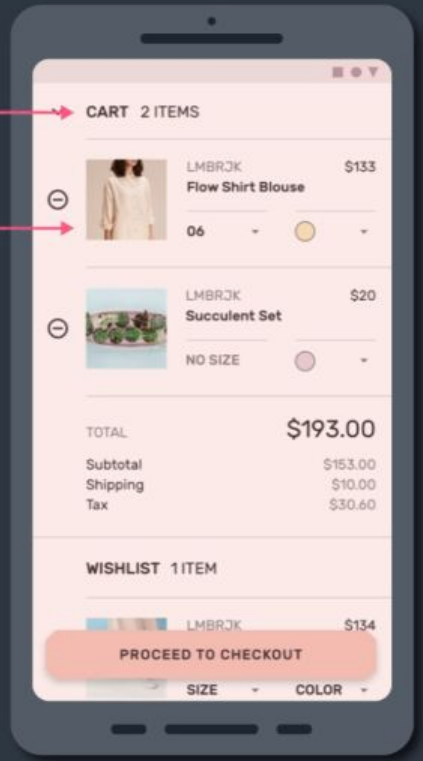

CART 2 ITEMS
LMBRJK
$133
Flow Shirt Blouse
06
LMBRJK
$20
Succulent Set
NO SIZE
TOTAL
$193.00
Subtotal
$153.00
Shipping
$10.00
Tax
$30.60
WISHLIST 1 ITEM
LMBRJK
$134
PROCEED TO CHECKOUT
SIZE
COLOR

Text Widget


CART 2 ITEMS
LMBRJK
$133
Flow Shirt Blouse
06
LMBRJK
$20
Succulent Set
NO SIZE
TOTAL
$193.00
Subtotal
$153.00
Shipping
$10.00
Tax
$30.60
WISHLIST 1 ITEM
LMBRJK
$134
PROCEED TO CHECKOUT
SIZE
COLOR

Text Widget

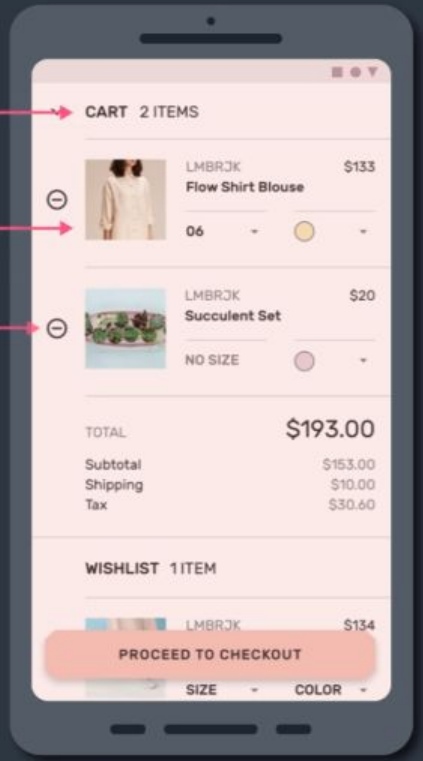
Image Widget

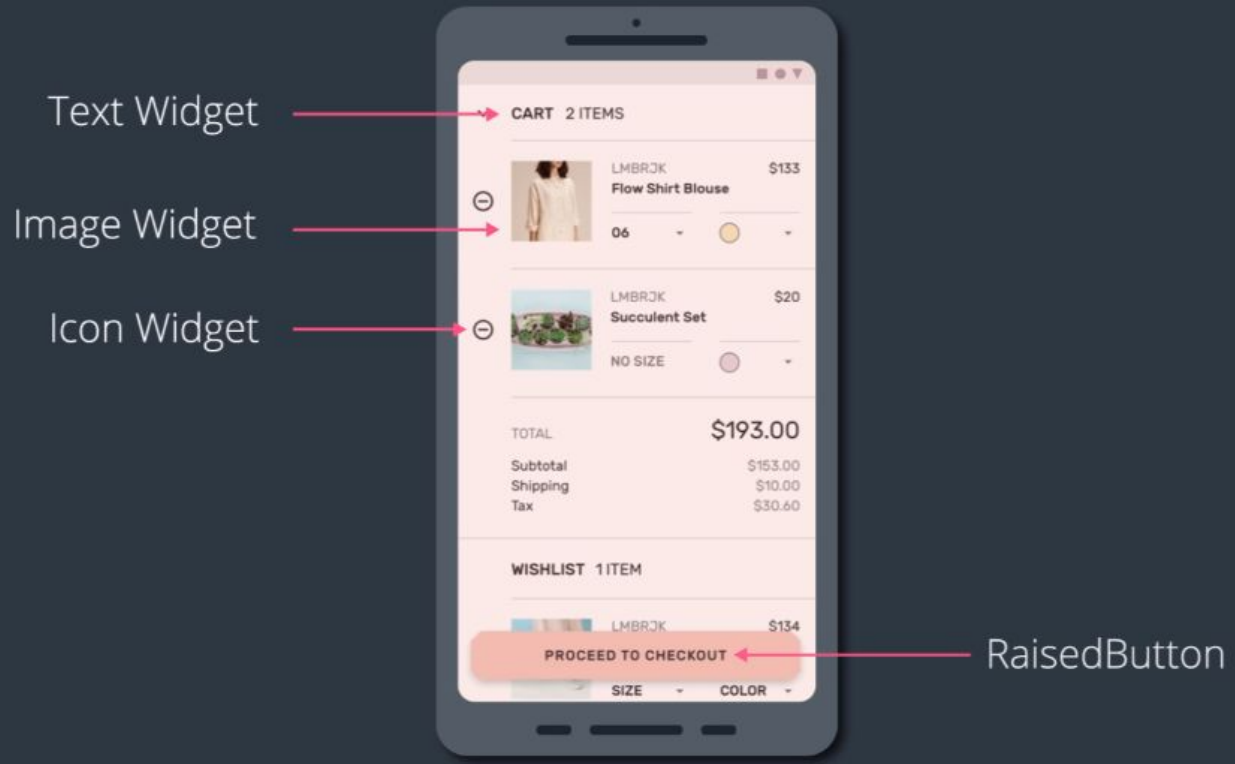


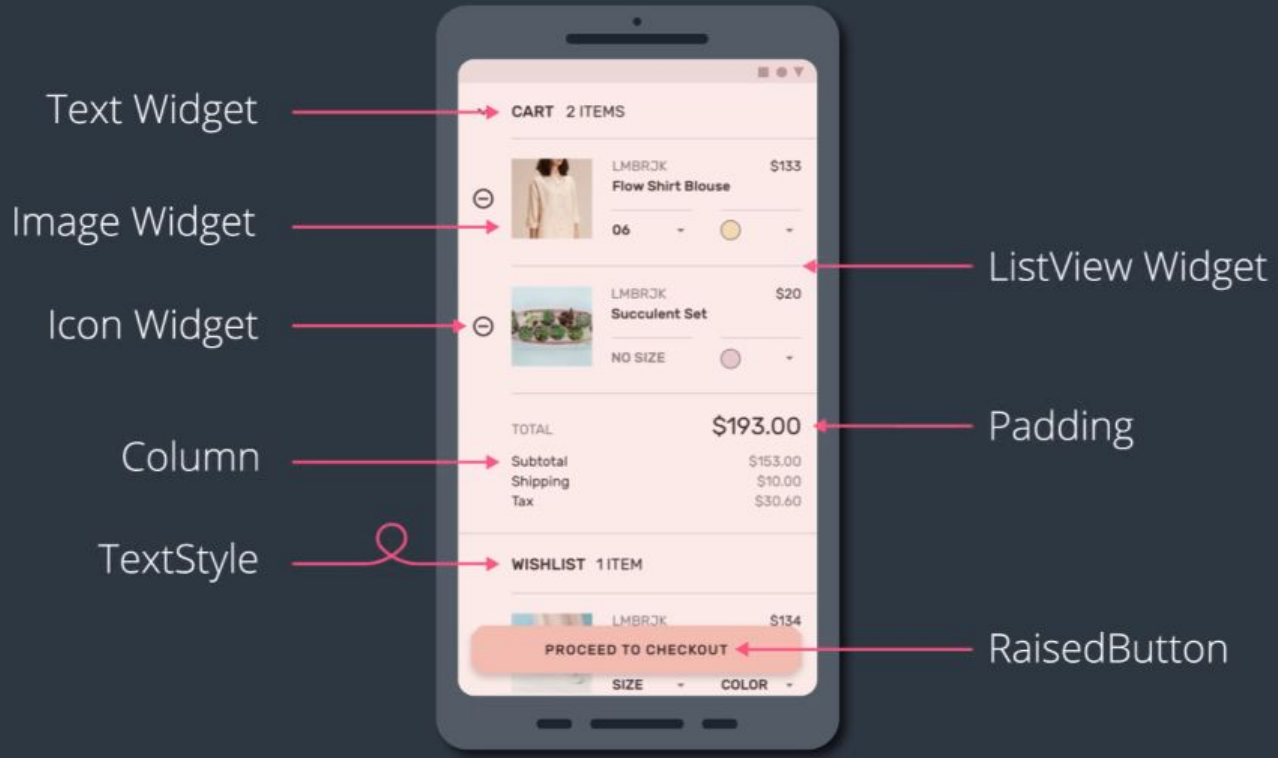
Text Widget

Image Widget

Icon Widget

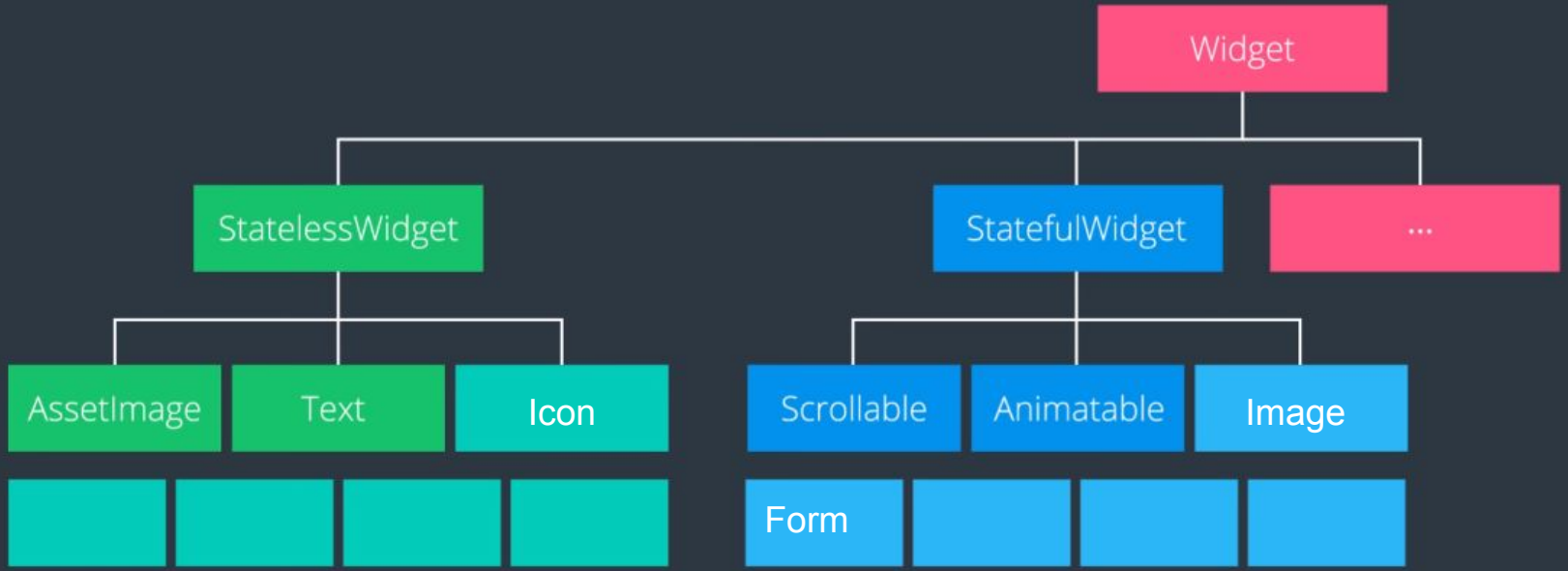



Text Widget
Image Widget
Icon Widget
CART 2 ITEMS
LMBRJK
$133
Flow Shirt Blouse
06
LMBRJK
$20
Succulent Set
NO SIZE
TOTAL
$193.00
Subtotal
$153.00
Shipping
$10.00
Tax
$30.60
WISHLIST 1 ITEM
LMBRJK
$134
PROCEED TO CHECKOUT
SIZE
COLOR
RaisedButton

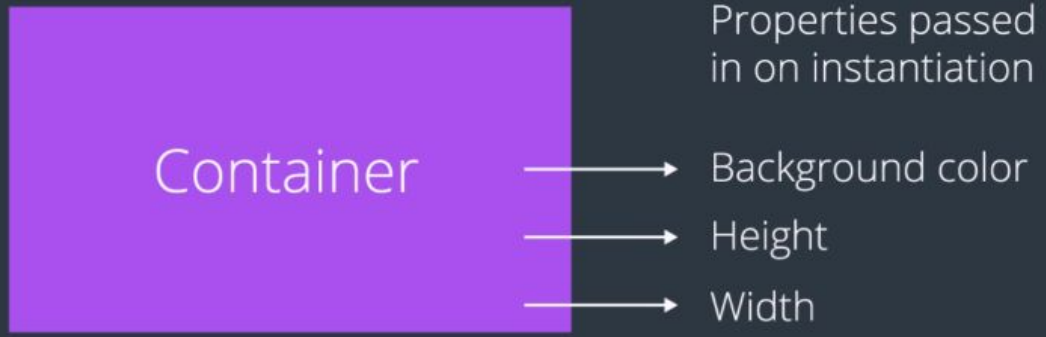

Text Widget
Image Widget
Icon Widget
Column
TextStyle
ListView Widget
Padding
RaisedButton
CART 2 ITEMS
LMBRJK
Flow Shirt Blouse
$133
06
LMBRJK
Succulent Set
$20
NO SIZE
TOTAL
$193.00
Subtotal
Shipping
Tax
$153.00
$10.00
$30.60
WISHLIST 1 ITEM
LMBRJK
$134
PROCEED TO CHECKOUT
SIZE
COLOR


WIDGETS EVERYWHERE
FLUTTER
imgflip.com

Everything is a widget



Stateless widgets



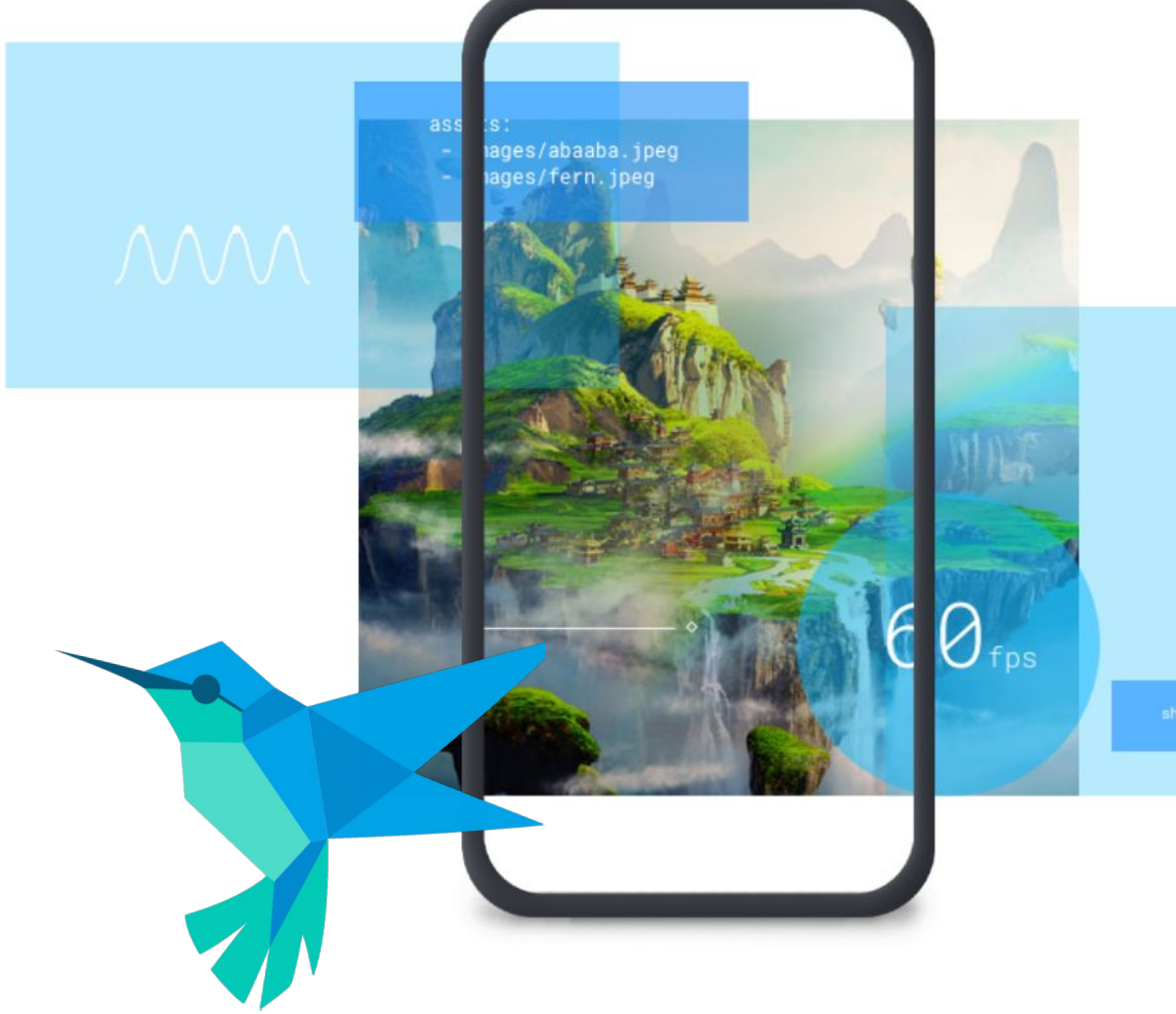
Let’s explore a Container Widget

Sign up
on Udacity

Quiz: Explore Flutter Widgets

Let’s examine Hello Rectangle

Questions?



How can we improve?

goo.gl/forms/RqinKD2LcVkzQoIm2