LeanCode

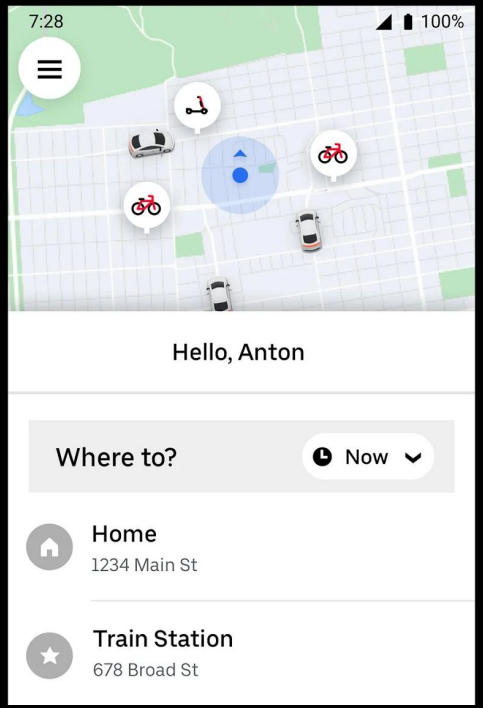
Flutter intro / Layouting 1

LeanCode

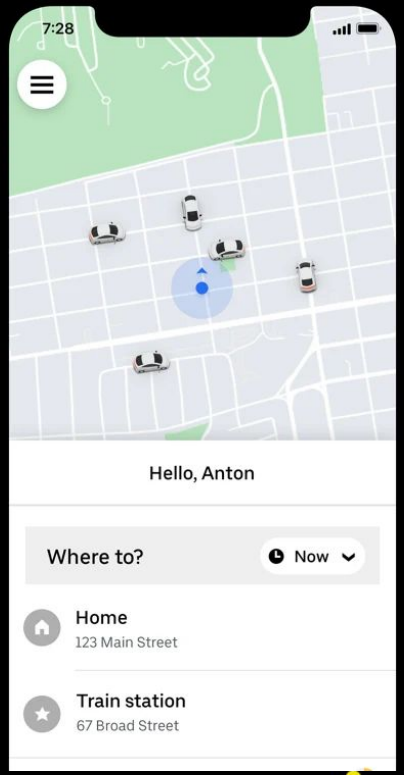
Introduction

LeanC

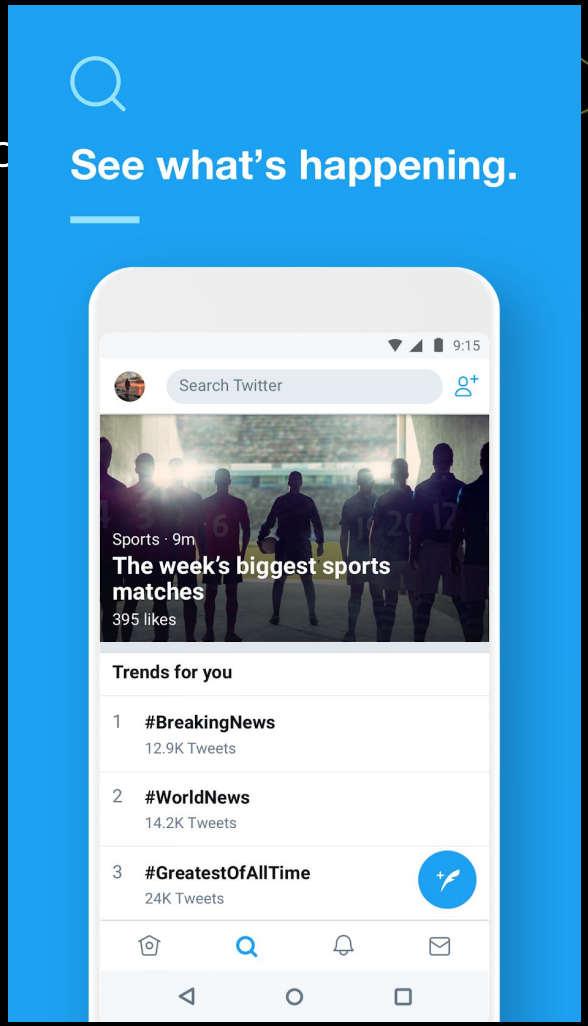
Choose your destination

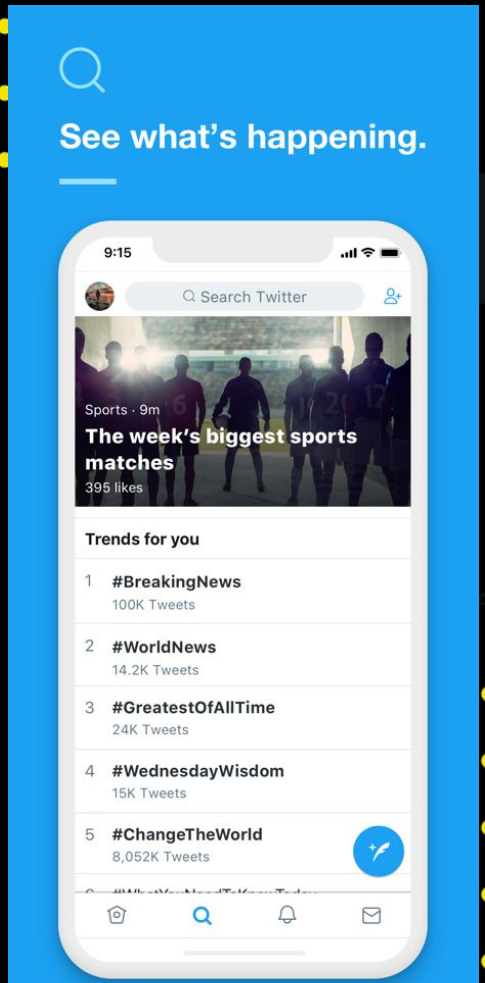


Choose your destination



LeanCo


See what's happening.
9:15
Search Twitter
Sports · 9m
The week's biggest sports matches
395 likes
Trends for you
1 #BreakingNews
12.9K Tweets
2 #WorldNews
14.2K Tweets
3 #GreatestOfAllTime
24K Tweets


See what's happening.
9:15
Search Twitter
Sports · 9m
The week's biggest sports matches
395 likes
Trends for you
1 #BreakingNews
100K Tweets
2 #WorldNews
14.2K Tweets
3 #GreatestOfAllTime
24K Tweets
4 #WednesdayWisdom
15K Tweets
5 #ChangeTheWorld
8,052K Tweets

LeanCode

Design systems are platform-independent

LeanCode

Two separate dev teams

...and separate bugs, release cycles, deployments

LeanCode

It costs a lot

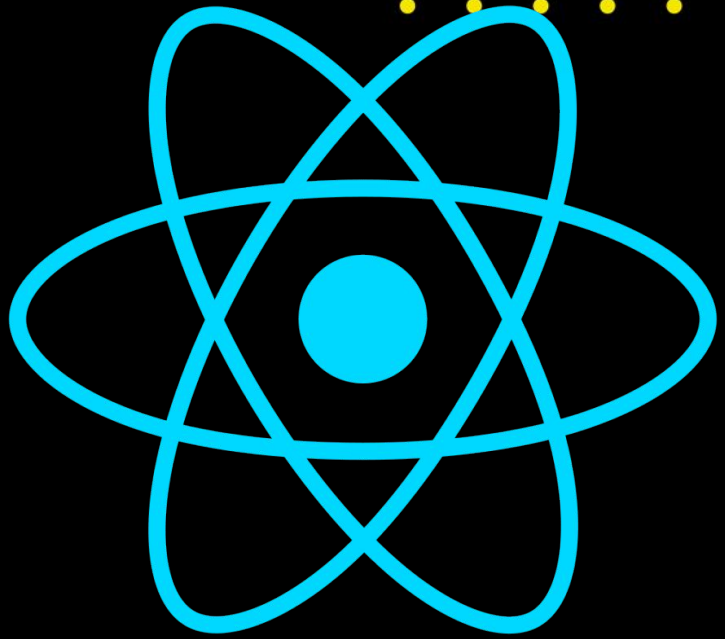
LeanCode

Xamarin

LeanCode

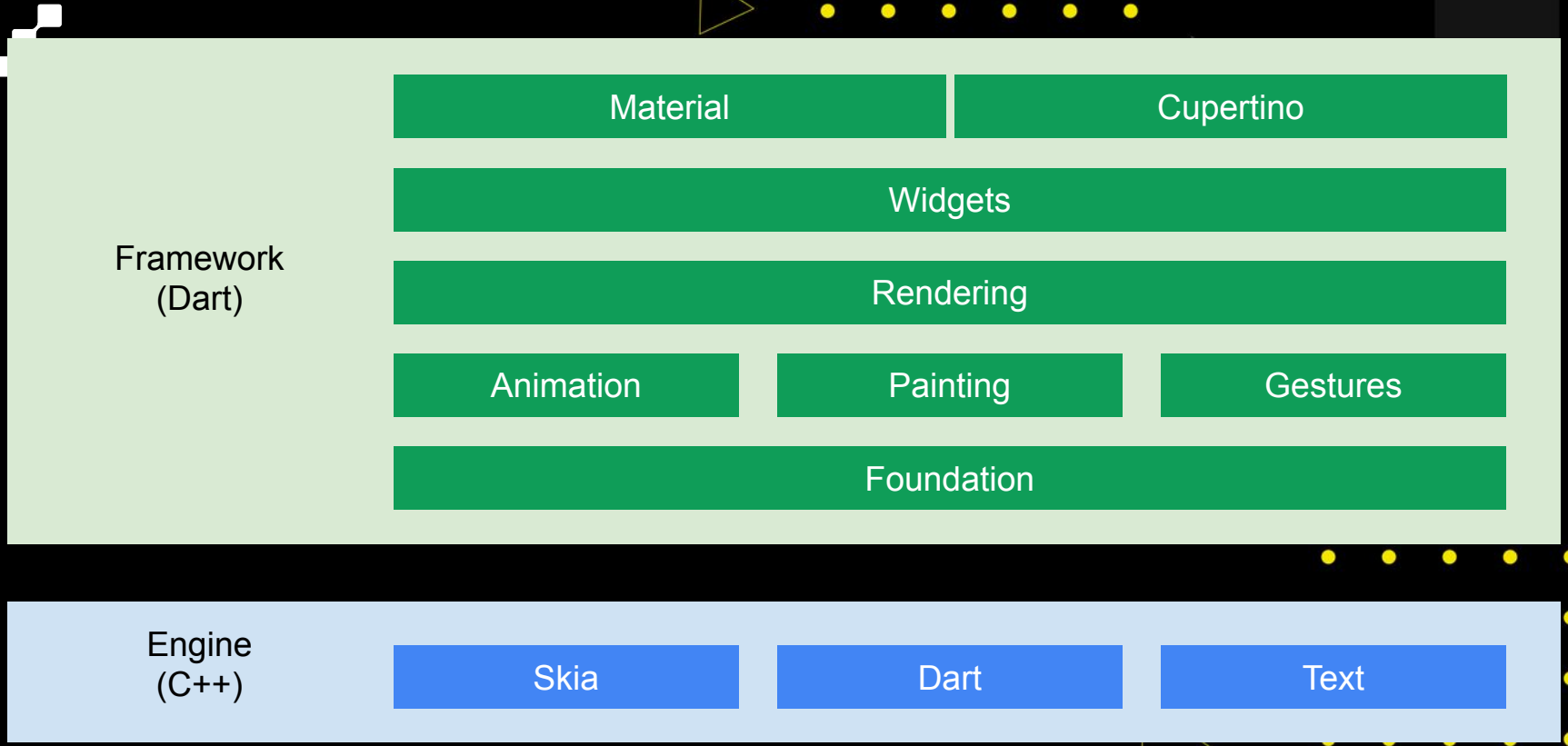
APACHE CORDOVA™

LeanCode



React Native

Flutter


Framework
(Dart)
Material
Cupertino
Widgets
Rendering
Animation
Painting
Gestures
Foundation
Engine
(C++)
Skia
Dart
Text

LeanCode

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, Mozilla Firefox and Firefox OS, and many other products.

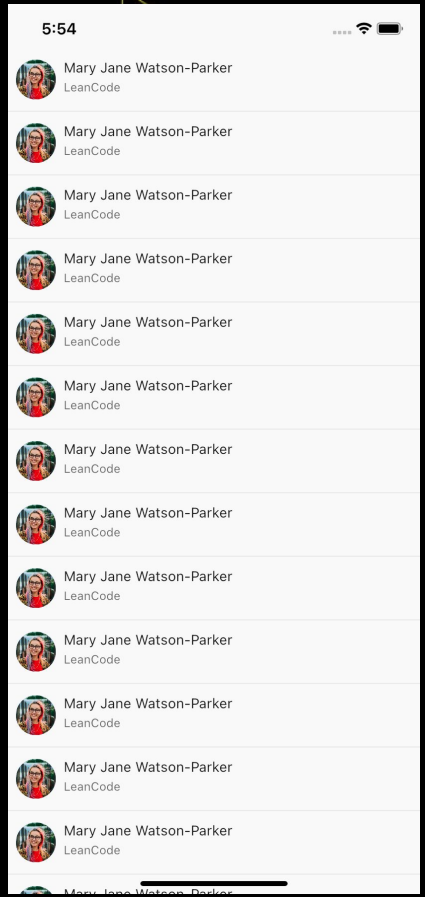
LeanCode

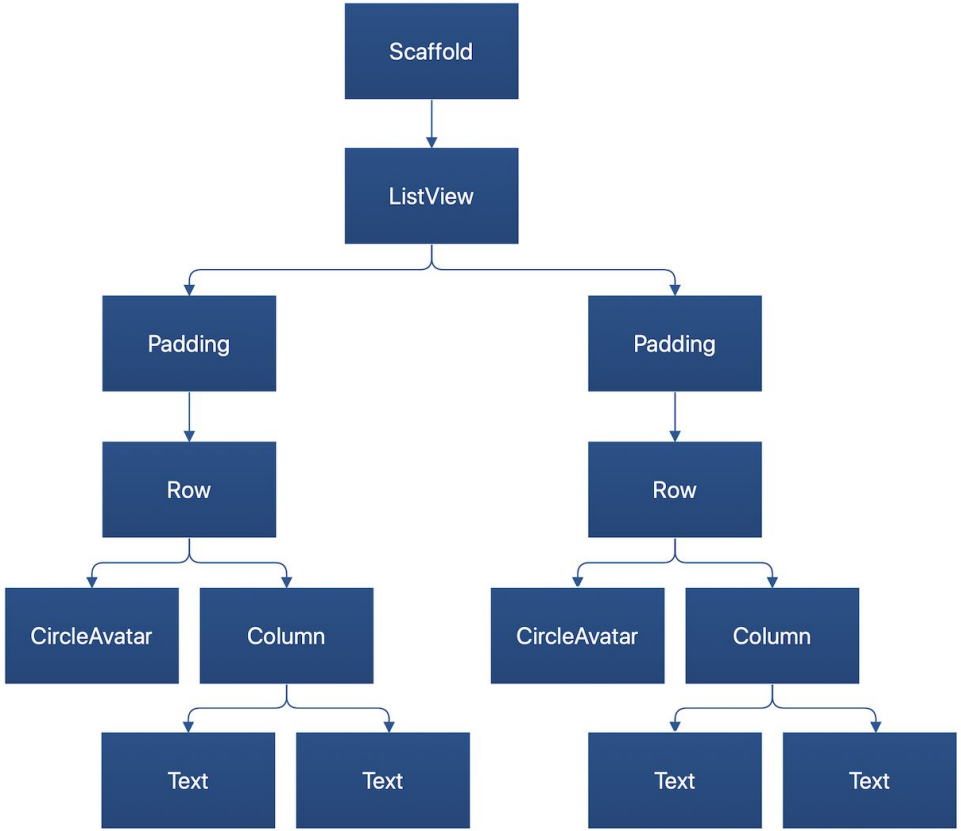
(Almost) everything is a widget

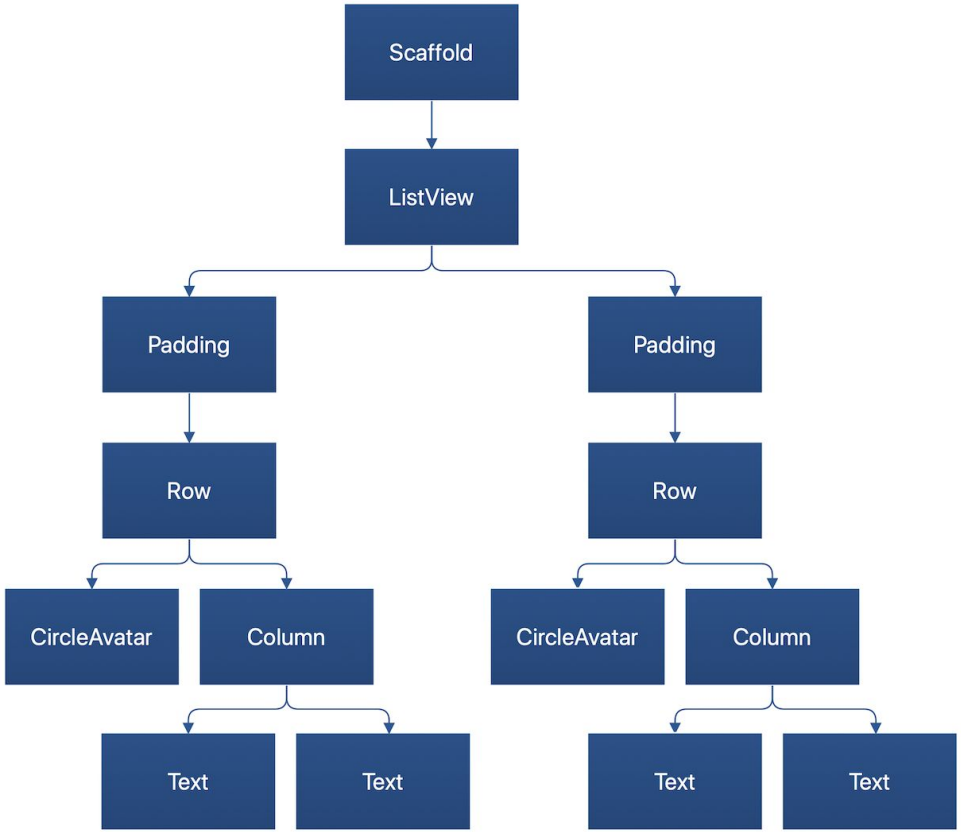
LeanCode

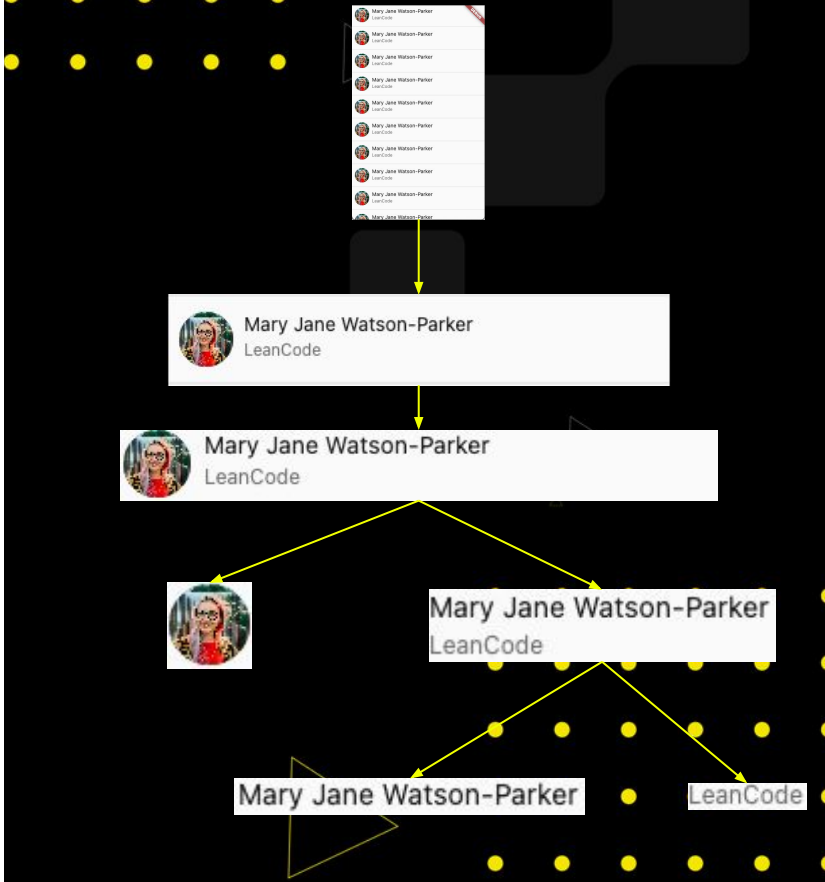
Widget in a widget

LeanCode






Scaffold
ListView
Padding
Padding
Row
Row
CircleAvatar
Column
CircleAvatar
Column
Text
Text
Text
Text


Mary Jane Watson-Parker
LeanCode
Mary Jane Watson-Parker
LeanCode
Mary Jane Watson-Parker
LeanCode
Mary Jane Watson-Parker
LeanCode

LeanCode

Flutter Desktop

macOS + Windows + Linux

LeanCode

Flutter Web

LeanCode

Flutter Embedded

Imperative UI

Windows Forms / Android / iOS / GTK

Add callback which on change does:

- Set color
- Remove child
- Add child
- Set position

Declarative UI

React / Flutter / Jetpack Compose / SwiftUI

For this state return a View with red background color and children consisting of a text message and a button.

```
final title = Text();
title.data = 'Please tap the button to finish';

final button = Button();
button.text = 'Finish';
button.onPressed = () {
  print('Button pressed!');
};

view.backgroundColor = Colors.white;
view.children = [];
view.children.add(title);
view.children.add(button);
```

```
return View(
  children: [
    Text('Please tap the button to finish'),
    Button(
      text: 'Finish',
      onPressed: () {
        print('Button pressed!');
      }
    ),
  ],
);
```

Let’s make some Hello World!

LeanCoc

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

void main() {
  runApp(
    const Center(
      child: Text(
        'Hello world!',
        textDirection: TextDirection.ltr,
      ),
    ),
  );
}
```

Constraints go down. Sizes go up. Parent sets position.

LeanCode

BoxConstraints({double minWidth, double maxWidth, double minHeight, double maxHeight})

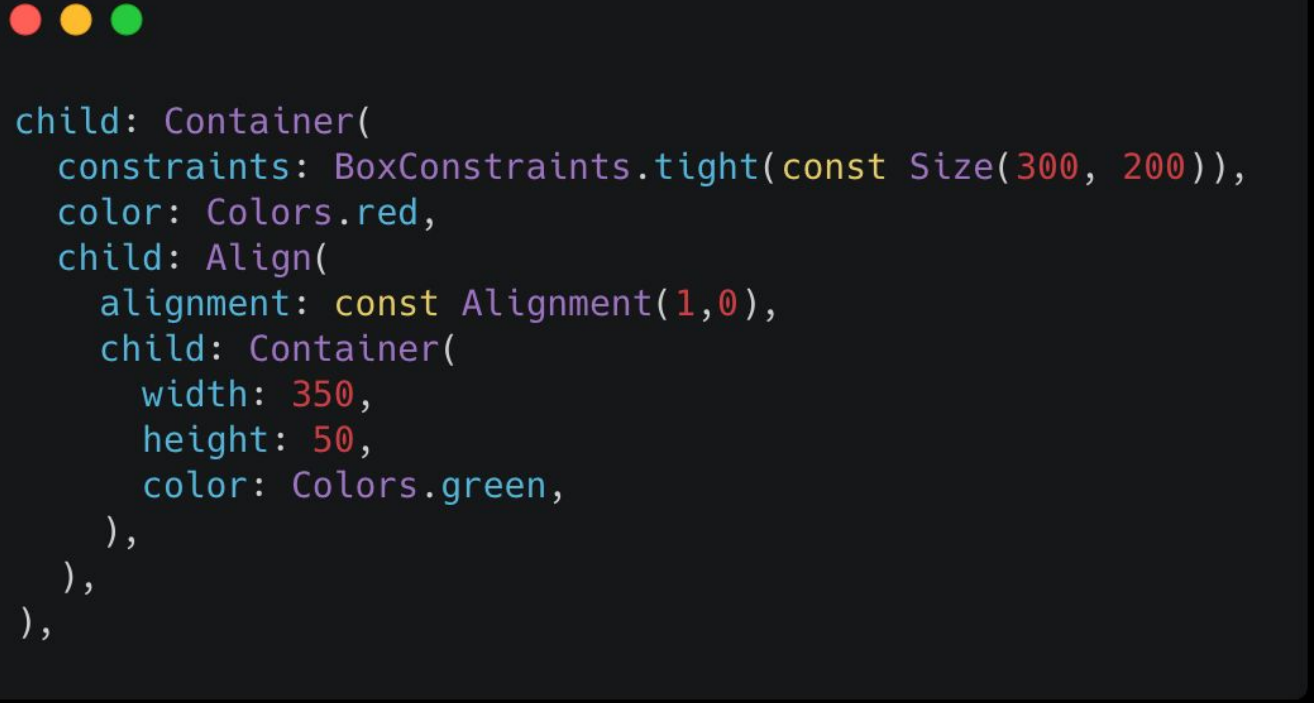
Creates box constraints with the given constraints.

const

Layout algorithm

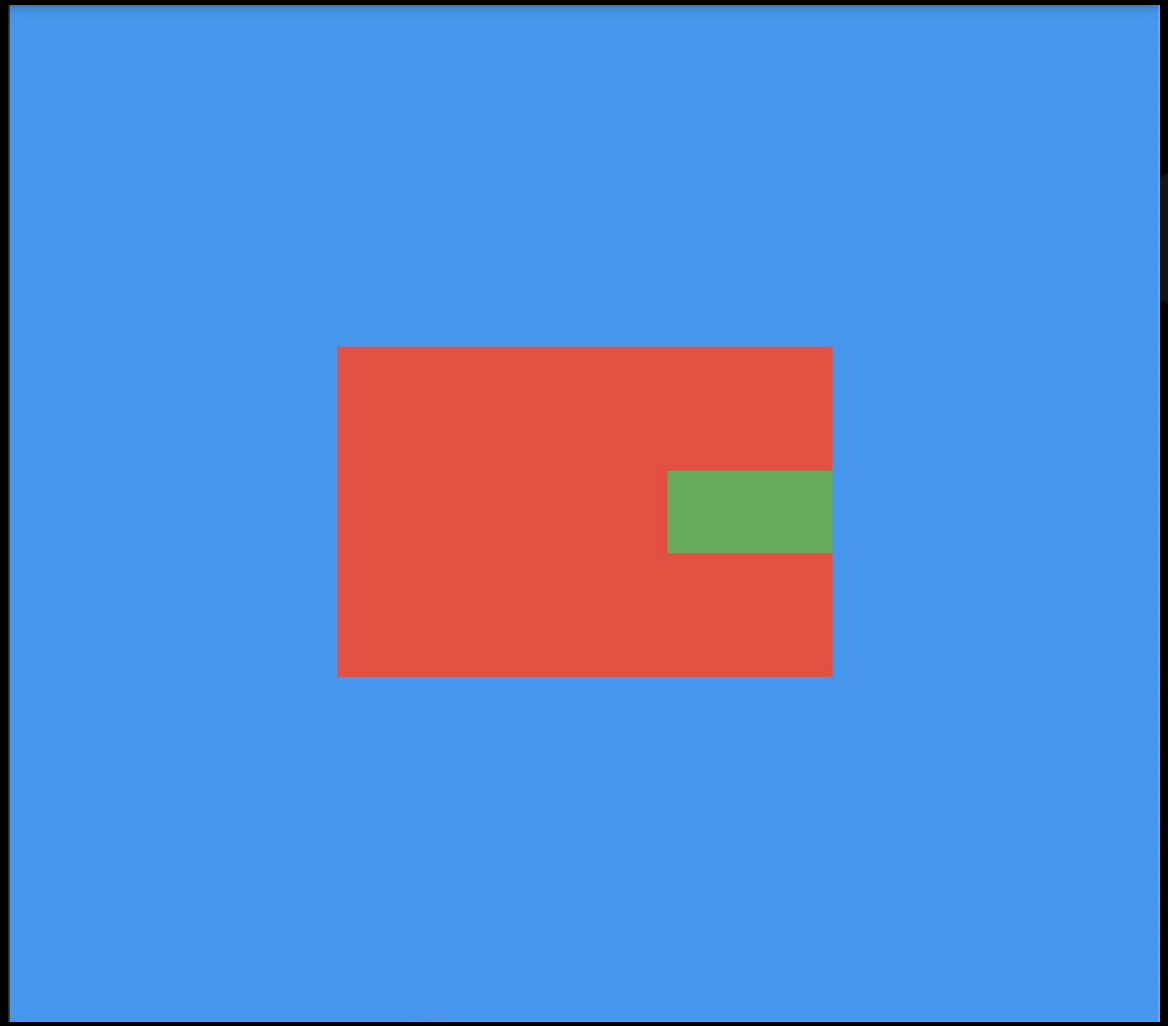
1. Widget gets constraints from its parent
2. For every child, it requests its size within given constraints (could be different from the first constraints)
3. Knowing children sizes, now the widget positions every of them
4. Knowing children sizes and positions, now the widget can pass its own size to its parent

LeanCode



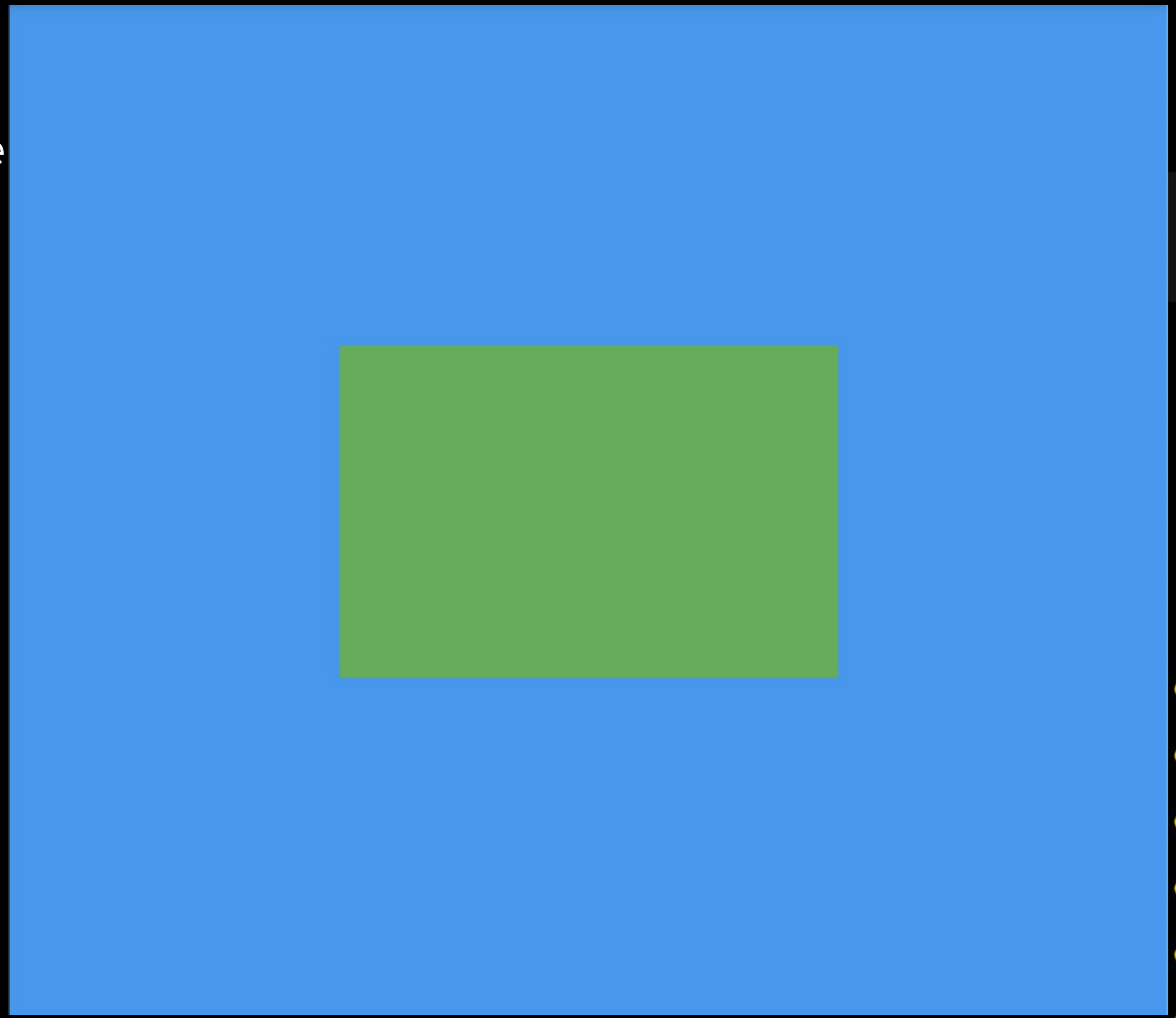
```
child: Container(
  constraints: BoxConstraints.tight(const Size(300, 200)),
  color: Colors.red,
  child: Align(
    alignment: const Alignment(1,0),
    child: Container(
      width: 350,
      height: 50,
      color: Colors.green,
    ),
  ),
),
```

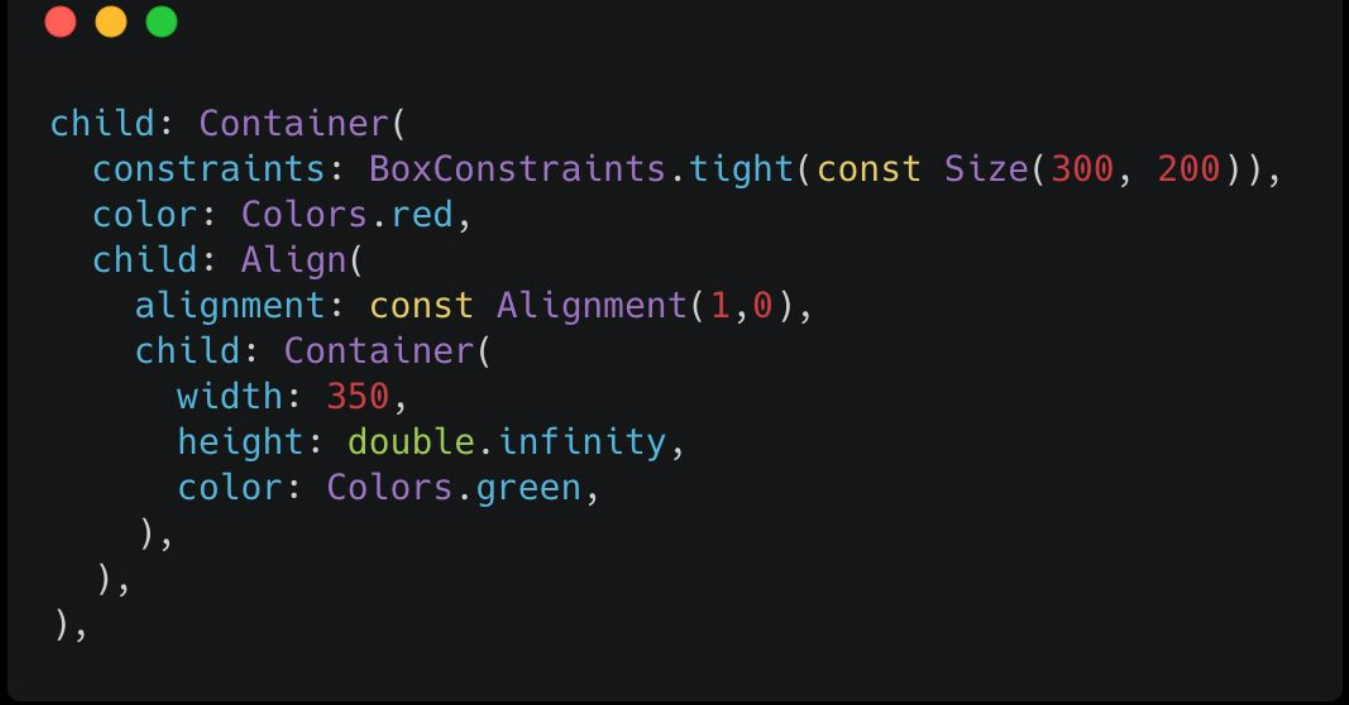
LeanCode



LeanC

```
Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        constraints: const BoxConstraints(minWidth: 400, minHeight: 400),
        color: Colors.blue,
        child: Center(
          child: Container(
            constraints: BoxConstraints.tight(const Size(300, 200)),
            color: Colors.red,
            child: Align(
              alignment: const Alignment(1,0),
              child: Container(
                width: 100,
                height: 50,
                color: Colors.green,
              ),
            ),
          ),
        ),
      ),
    );
  }
```





```
child: Container(
  constraints: BoxConstraints.tight(const Size(300, 200)),
  color: Colors.red,
  child: Align(
    alignment: const Alignment(1,0),
    child: Container(
      width: 350,
      height: double.infinity,
      color: Colors.green,
    ),
  ),
),
```

Let’s go to the lab

source: flutter.dev

Thanks!