



Flutter

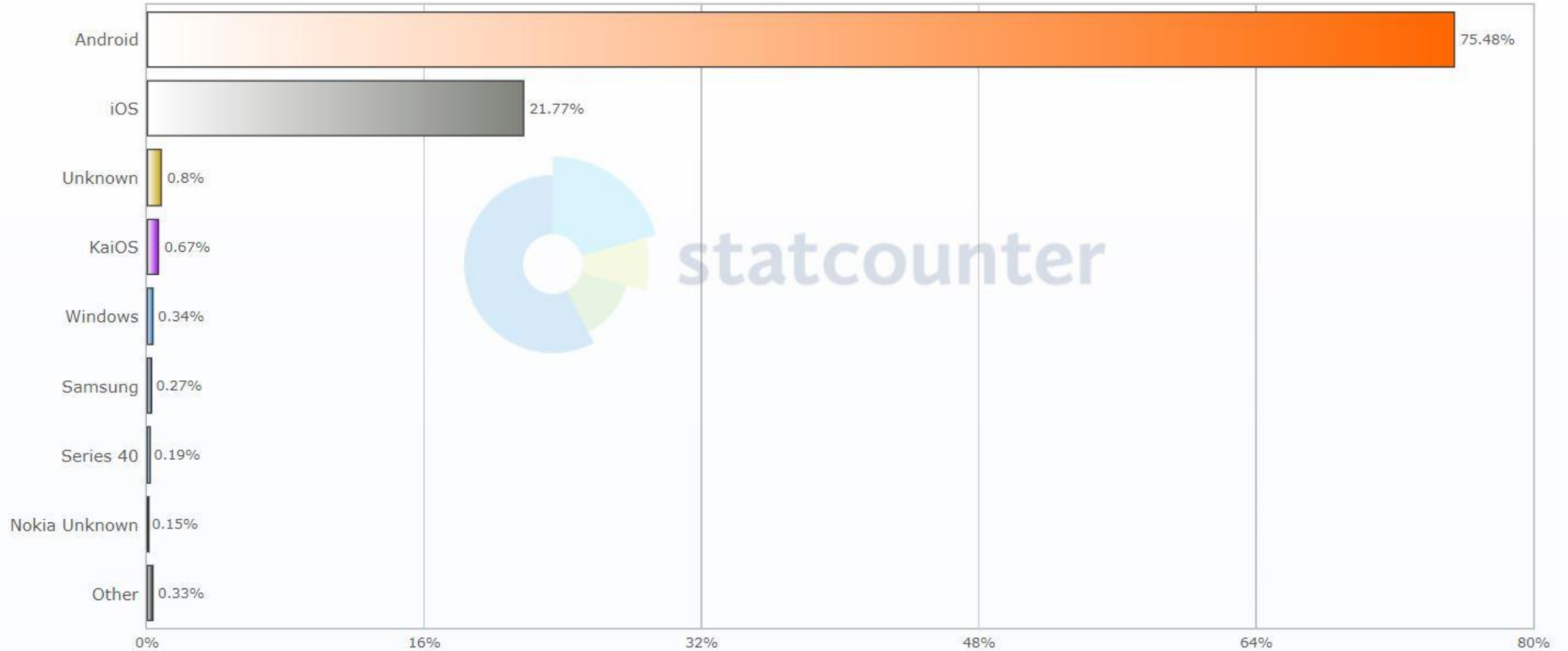
アジェンダ

1. モバイルのマーケットについて
2. アプリの種類について
3. Flutterについて
4. Dartについて
5. プログラムの例文
6. DEMOアプリ

Mobile Operating System Market Share Worldwide

June 2018 - June 2019

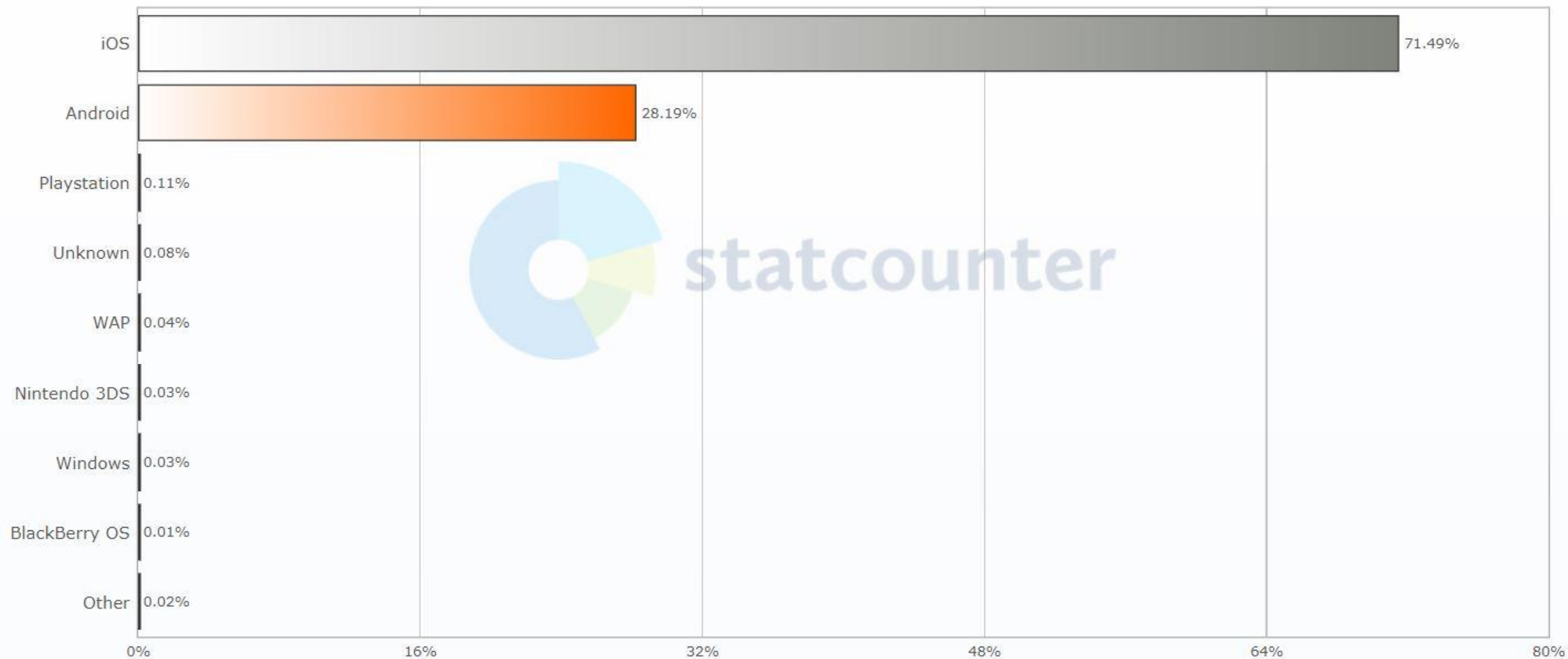
[Edit Chart Data](#)



Mobile Operating System Market Share Japan

June 2018 - June 2019

[Edit Chart Data](#)



Progressive Web Apps
HTML/CSS, React, Angular, Vue

マイナス:

本当のアプリではないので:

- ブラウザ上で実行します。
- マーケット（ストア）に出版できません。
- ショートカットを作成できません。
- デバイスの機能（NFC、Bluetoothなど）使えないです。

プラス:

- 制作は簡単です。

Hybrid
(PhoneGap, Cordova, Sencha)

マイナス:

- **WebView**には実行するのでアプリのパフォーマンスが悪いです。

プラス:

- **HTML**や**JAVASCRIPT**使用するので**WEB**デベロッパーは早く慣れてます。
- プラグイン利用してデバイスの機能にアクセスできます。
- マーケット（ストア）に出版できます。
- ショートカットを作成できます。

Native Solutions
(React Native, Flutter, Xamarin)

マイナス:

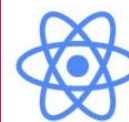
- 新しいフレームワークを勉強しないといけません。

プラス:

- パフォーマンスは最高です。
- デバイスの機能（NFC、Bluetoothなど）を使えます。
- マーケット（ストア）に出版できます。
- ショートカットを作成できます。

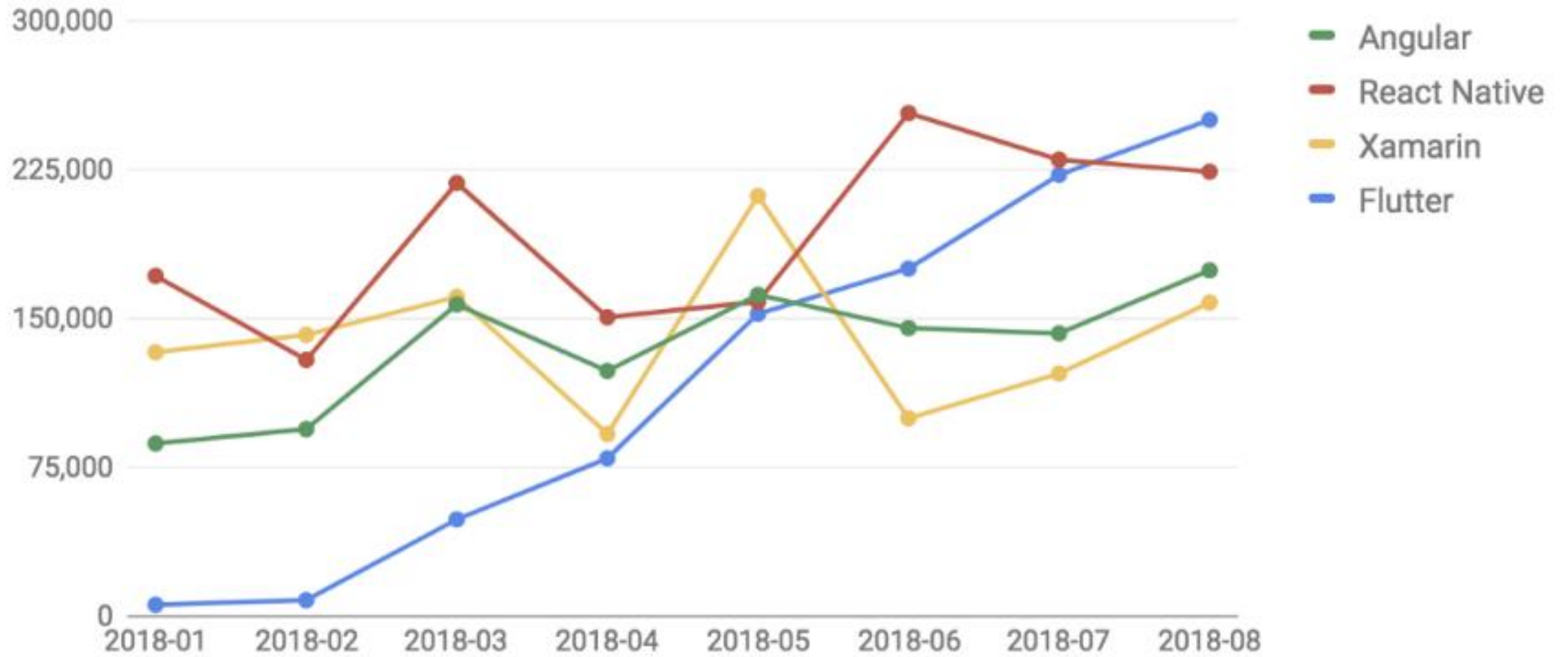


React Native



	Xamarin	React Native	Flutter
会社	Microsoft	Facebook	Google
リリース年	2011年	2015年	2018年
利用言語	XAML、Xamarin.forms	JavaScript	Dart
人気のレベル	高	高	低
人気のトレンド	だんだん下がります	だんだん上がります	どんどん上がります
コスト	ライセンス必要	無料	無料

StackOverflowの質問数





Flutter について

Flutter（フラッター）は、Googleによって開発された**フリーかつオープンソース**のモバイルアプリケーションフレームワークである。

Flutterエンジン

主にC++で書かれており、GoogleのSkiaを使用した低レベルのレンダリングをサポートしている。AndroidやiOS固有のSDKとも連携ができるようになっている

基本ライブラリ

Dartで書かれており、エンジンと通信を行うためのAPIなど、Flutterを利用してアプリケーションを構築するために必要な基本的なクラスと関数を提供している

ウィジェット

FlutterのUI設計は、様々なウィジェットによって構成されている。ウィジェットはUIのテキスト・図形などを含む多くの単純なウィジェットを組み合わせることで、複雑なウィジェットを作成することができる。



予定

Desktop

Linux

Windows

MacOS

Mobile

Android

IOS

Web

Chrome

Firefox

etc



Dart

はGoogleによって開発されたWEB向けのオブジェクト指向プログラミング言語です。

コンパイルできる環境：

- 機械語ARM & x64
- JavaScript

```
class Spacecraft {
  String name;
  DateTime launchDate;

  // Constructor, with syntactic sugar for assignment to members.
  Spacecraft(this.name, this.launchDate) {
    // Initialization code goes here.
  }

  // Named constructor that forwards to the default one.
  Spacecraft.unlaunched(String name) : this(name, null);

  int get launchYear =>
    launchDate?.year; // read-only non-final property

  // Method.
  void describe() {
    print('Spacecraft: $name');
    if (launchDate != null) {
      int years =
        DateTime.now().difference(launchDate).inDays ~/
        365;
      print('Launched: $launchYear ($years years ago)');
    } else {
      print('Unlaunched');
    }
  }
}
```

Widgets

Stateful Widget

Widgetの変数
が変わる

Stateless Widget

Widgetの変数
が変わらない

Inherited Widget

チャイルドの
Widgetに値を
上げます。

Stateless Widget

コンストラクタ

Statelessなので関数だけ使います。

Containerのスタイル

Containerのチャイルド

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

class MyAppBar extends StatelessWidget {
  MyAppBar({this.title});

  // Fields in a Widget subclass are always marked "final".
  final Widget title;

  @override
  Widget build(BuildContext context) {
    return Container(
      height: 56.0, // in logical pixels
      padding: const EdgeInsets.symmetric(horizontal: 8.0),
      decoration: BoxDecoration(color: Colors.blue[500]),
      // Row is a horizontal, linear layout.
      child: Row(
        // <Widget> is the type of items in the list.
        children: <Widget>[
          IconButton(
            icon: Icon(Icons.menu),
            tooltip: 'Navigation menu',
            onPressed: null, // null disables the button
          ),
          // Expanded expands its child to fill the available space.
          Expanded(
            child: title,
          ),
          IconButton(
            icon: Icon(Icons.search),
            tooltip: 'Search',
            onPressed: null,
          ),
        ],
      ),
    );
  }
}
```

Stateful Widget

ステータスを作成

ステータスの変数

ステータスの操作

ステータスの変数があるWidget

```
class Counter extends StatefulWidget {  
  @override  
  _CounterState createState() => _CounterState();  
}  
  
class _CounterState extends State<Counter> {  
  int _counter = 0;  
  
  void _increment() {  
    setState(() {  
      _counter++;  
    });  
  }  
  
  @override  
  Widget build(BuildContext context) {  
    return Row(  
      children: <Widget>[  
        RaisedButton(  
          onPressed: _increment,  
          child: Text('Increment'),  
        ),  
        Text('Count: $_counter'),  
      ],  
    );  
  }  
}
```

Inherited Widget

Widgetのデータ

```
class FrogColor extends InheritedWidget {
```

```
  const FrogColor({
```

```
    Key key,
```

```
    @required this.color,
```

```
    @required Widget child,
```

```
  }) : super(key: key, child: child);
```

```
  final Color color;
```

```
  static FrogColor of(BuildContext context) {
```

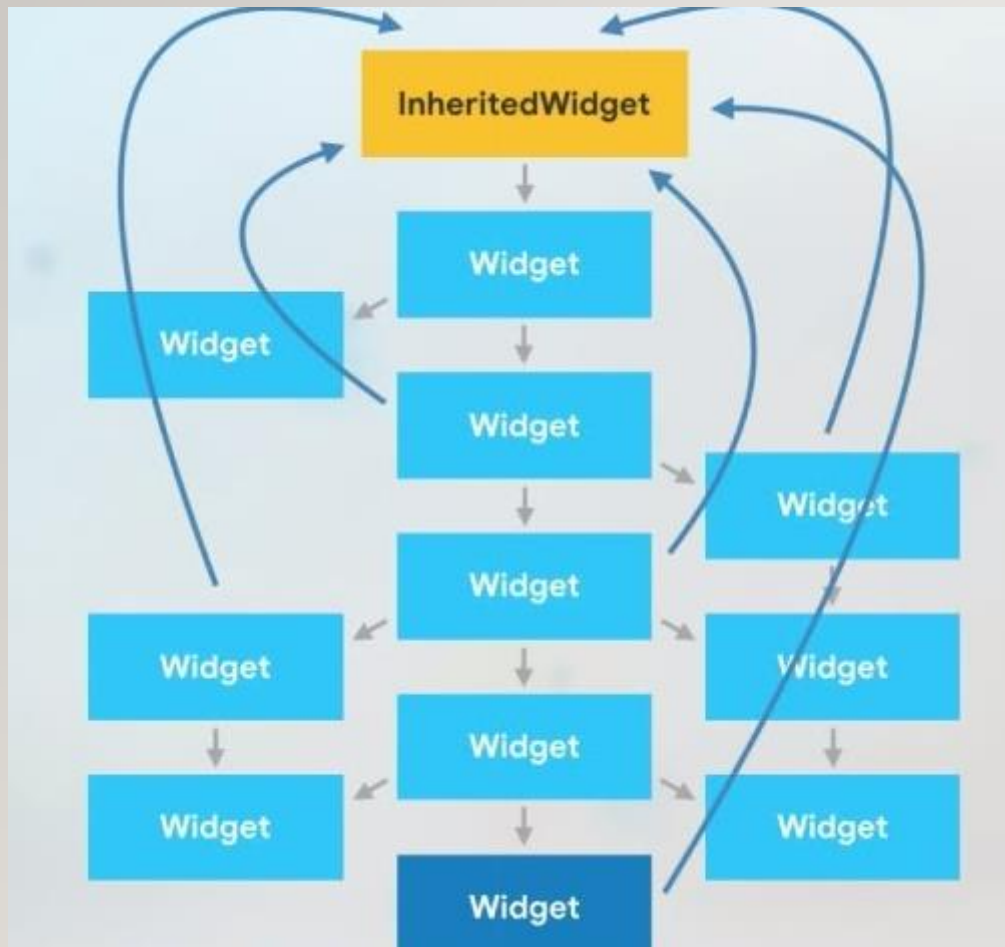
```
    return context.inheritFromWidgetOfExactType(FrogColor) as FrogColor;
```

```
  }
```

```
  @override
```

```
  bool updateShouldNotify(FrogColor old) => color != old.color;
```

```
}
```



Product Card

```
ProductCard
├── ProductCard(this.product, {Key key})
├── product
├── build(BuildContext context) → Widget
│   ├── Card margin: EdgeInsets.all(SizeConfig.blockSizeHorizontal * 1)
│   │   ├── Container constraints: ..., padding: EdgeInsets.all(5.0)
│   │   │   ├── Column mainAxisAlignment: MainAxisAlignment.min
│   │   │   │   ├── Image name: product.imgURL, fit: BoxFit.cover, scale: 0.6
│   │   │   │   ├── Text product.name, style: ...
│   │   │   │   ├── Align alignment: Alignment.centerLeft
│   │   │   │   │   ├── Text product.description
│   │   │   │   │   ├── Align alignment: FractionalOffset.bottomRight
│   │   │   │   │   │   ├── Row mainAxisAlignment: MainAxisAlignment.end
│   │   │   │   │   │   │   ├── Text product.price + "\$ ", style: ...
│   │   │   │   │   │   │   ├── Padding padding: const EdgeInsets.fromLTRB(2, 5, 5, 5)
│   │   │   │   │   │   │   │   ├── RaisedButton onPressed: () { ... }, color: Colors.deepOrange
│   │   │   │   │   │   │   │   │   ├── Icon Icons.add_shopping_cart
│   │   │   │   │   │   │   │   │   ├── Text "Add to cart"
```



Fedora Hat

Awesome Fedora Hat Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

500\$


 Add to cart

Image Widget Text Widget

```
Image.asset(  
  product.imgURL,  
  fit: BoxFit.cover,  
  scale: 0.6,  
),
```

```
Text(product.name,  
  style: DefaultTextStyle.of(context)  
    .style  
    .apply(fontSizeFactor: 3.5, fontWeightDelta: 2)),
```

```
RaisedButton.icon(  
  onPressed: () {  
    CartModel.of(context).addProducts(product);  
  },  
  icon: Icon(Icons.add_shopping_cart),  
  label: Text("Add to cart"),  
  color: Colors.deepOrange,  
),
```

RaisedButton Widget

DEMO TIME



まとめ

Flutterの利点

- どこでも実行できる
- 開発のスピードは高い
- パフォーマンスは良い
- OSのバージョン異なってもUIが同じです

Flutterの欠点

- 新しい言語を勉強しないと将来は不明
- ライブラリはまだ少ない
- OSのバージョン異なってもUIが同じです