Flutter ハンズオン

connpassからイベントを取得するアプリを作る

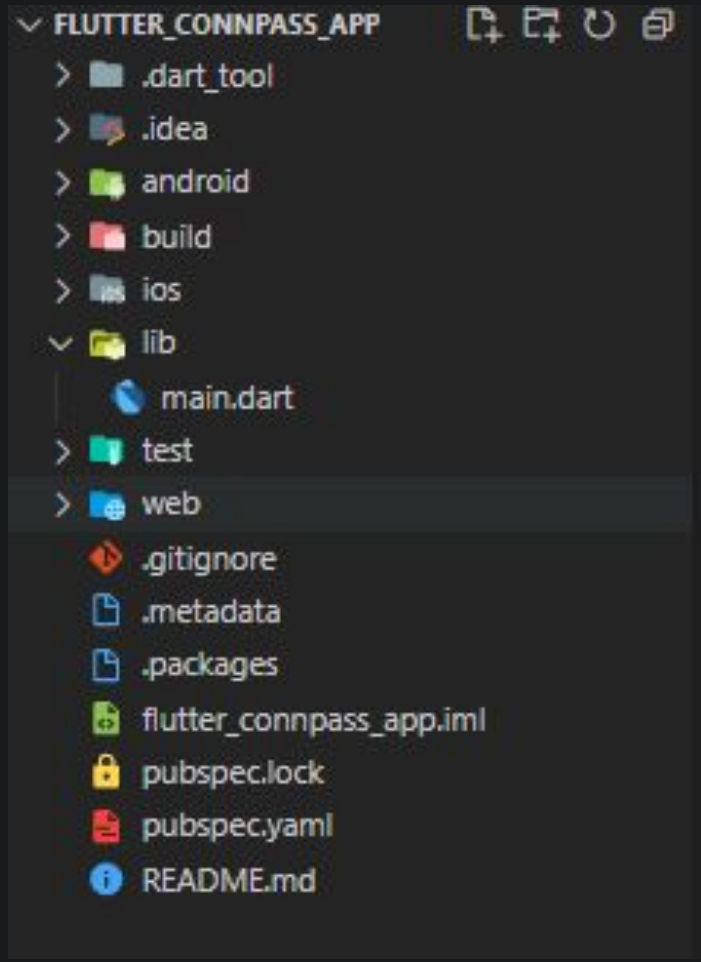
今日作るもの

connpassのイベント表示アプリ

- 一覧表示
- 詳細ページ
- 状態管理
- connpass APIの利用

connpass APIでサムネイル画像取れなかったんで見た目しょぼいです。。。

Create Project



プロジェクト構成

lib

コードを作成する場所

pubspec.yaml

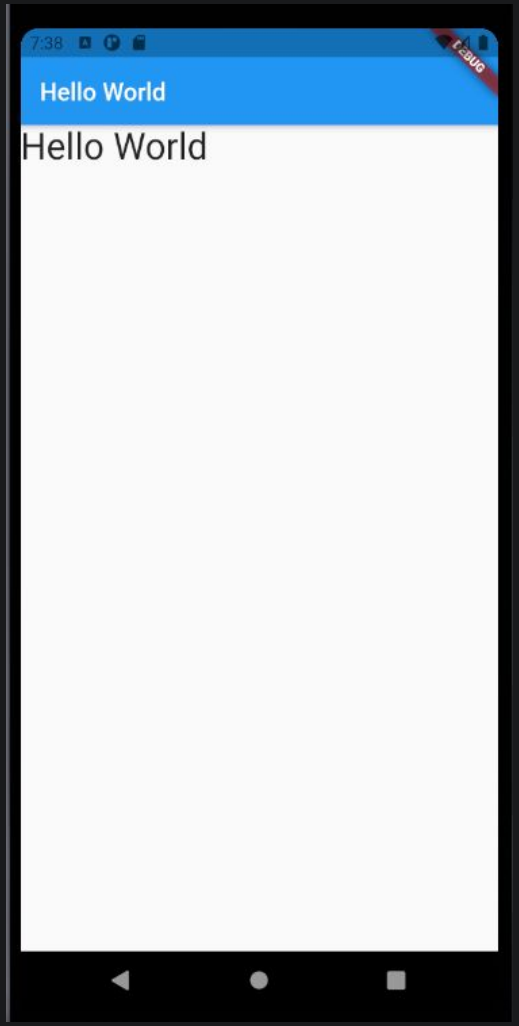
プロジェクトの設定ファイル

パッケージ管理やSDKのバージョン指定ができる

Work1

Hello World

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work1



main.dart

MaterialAppクラスのインスタンスを生成してrunApp()を実行

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

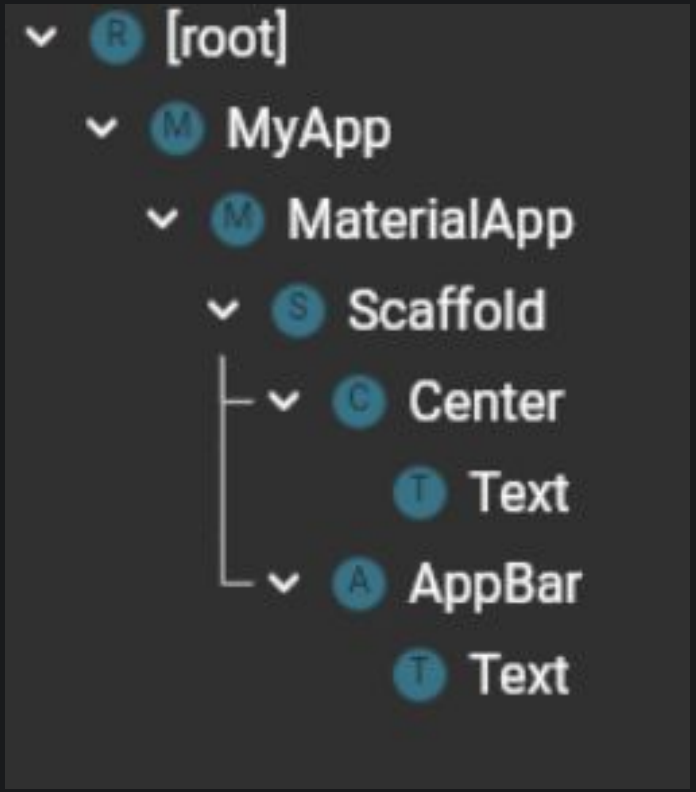
main.dart

まずはこれを理解する

MaterialAppクラスのインスタンスを生成してrunApp()を実行

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

Widget

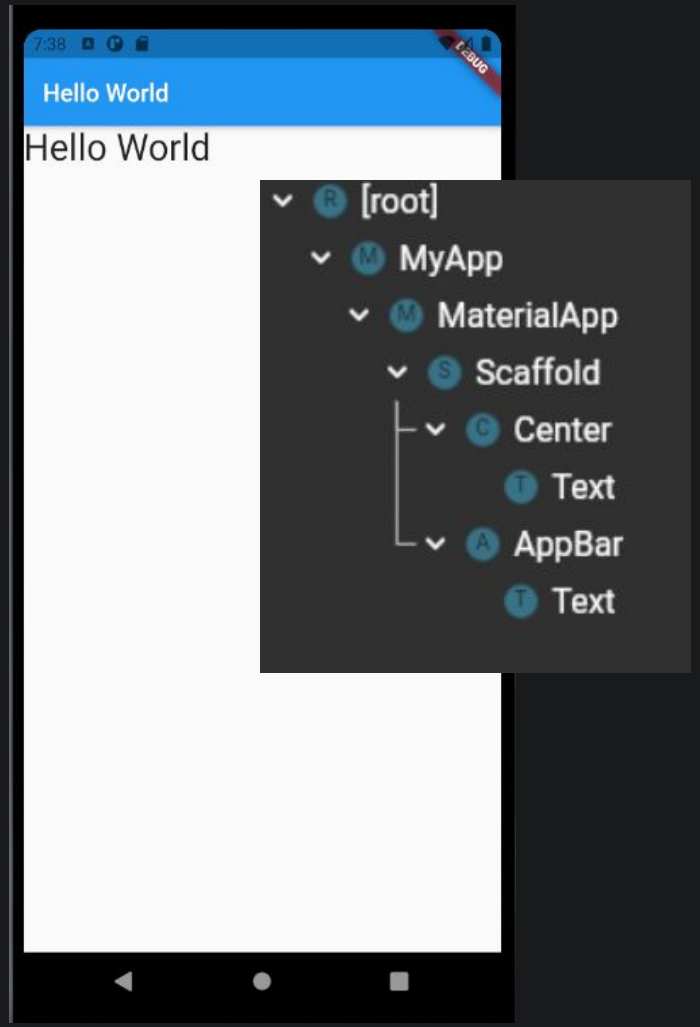


Widget
UIの構成要素のこと

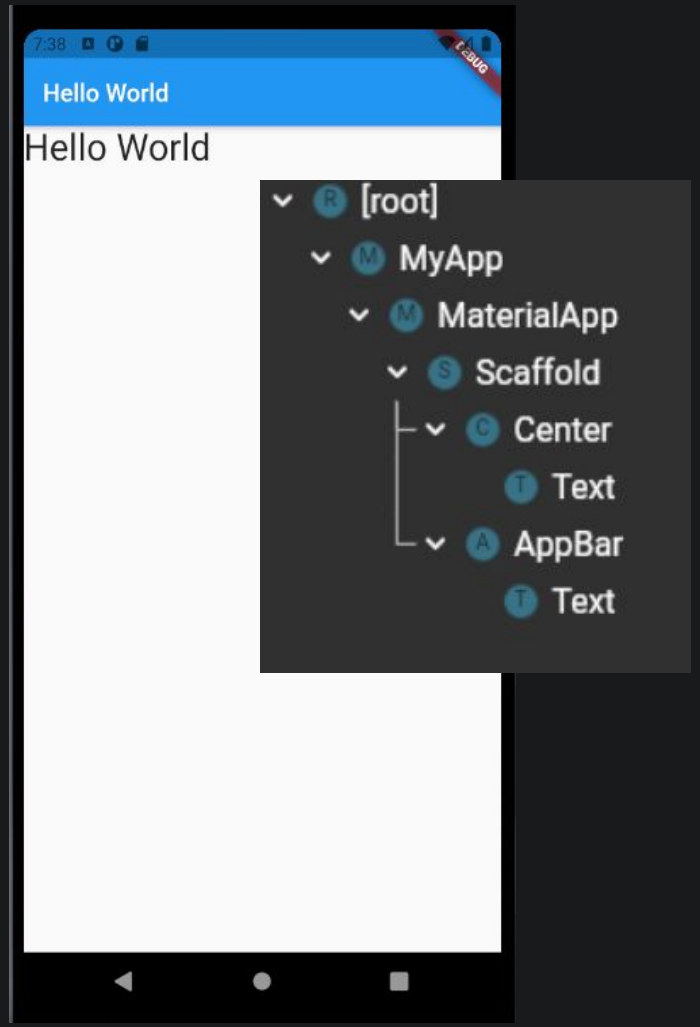
- 文字、ボタンのような部品
- レイアウト
- ジェスチャーの検出

すべてがWidget

Widgetの階層構造を構築することでUIを構成していく
（widget treeと呼ぶ）




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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```




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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

次はこれを理解する

StatelessWidget と StatefulWidget

どちらかのクラスを継承してUIを作っていく

StatelessWidget

状態(state)を持たないWidget

一度インスタンスされたら変わらない

StatefulWidget

状態(state)を持つWidget

状態が変更されると再描画される

厳密には違うけど最初の理解はこれで OK

main.dart

StatelessWidgetは
build()メソッドに実装したWidge
Treeを表示する

StatefulWidgetは今回使わな
いので省略

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

main.dart

StatelessWidgetは
build()メソッドに実装したWidge
Treeを表示する

StatefulWidgetは今回使わな
いので省略

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

次はこれを理解する

Material Components

Google Material Designでアプリ開発する際に便利なWidget

MaterialApp

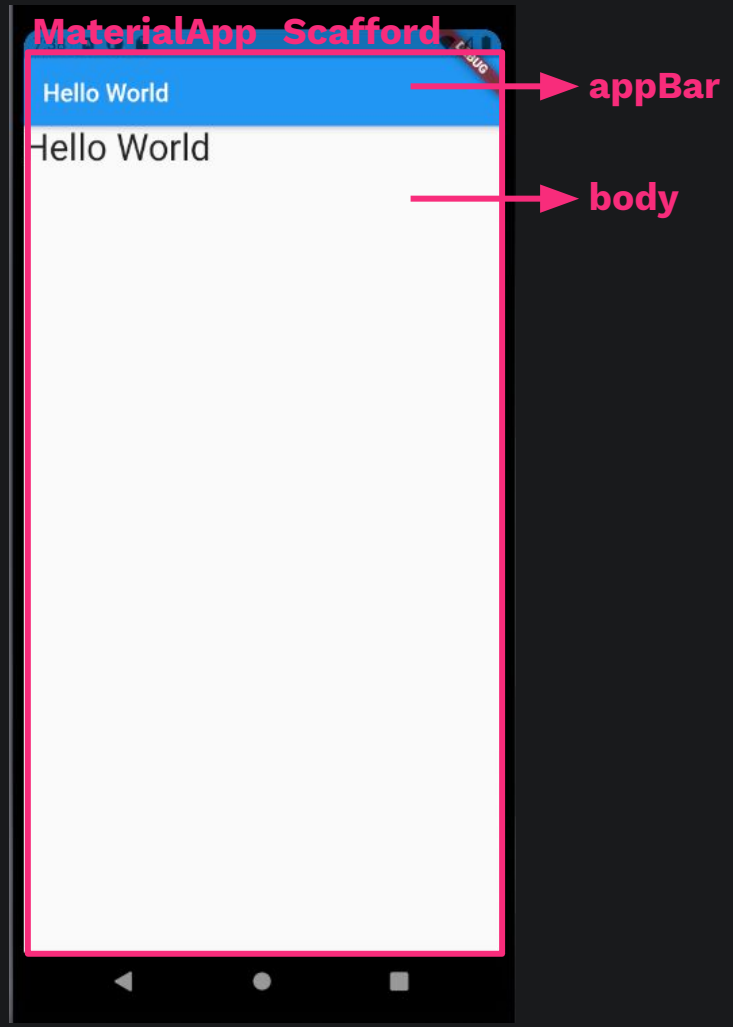
マテリアルデザインでアプリを開発する際に必要なWidgetをラップした便利Widget

Scaffold Widgetとともに使用する

Scaffold

マテリアルデザインの基本的なレイアウトを実装したWidget

各プロパティを設定するだけでマテリアルデザインを実現できる




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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      home: Scaffold(
        appBar: AppBar(title: Text('Hello World')),
        body: Text(
          'Hello World',
          style: TextStyle(fontSize: 30),
        ),
      ),
    );
  }
}
```

雰囲気わかったところで
さっそくアプリ開発😉

Work2

アプリのレイアウトを作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work2

Layout

Row

レイアウト系の基本Widget

要素を横並びに配置する

要するにflexboxのrow

引用 : https://api.flutter.dev/flutter/widgets/Row-class.html#widgets.Row.1

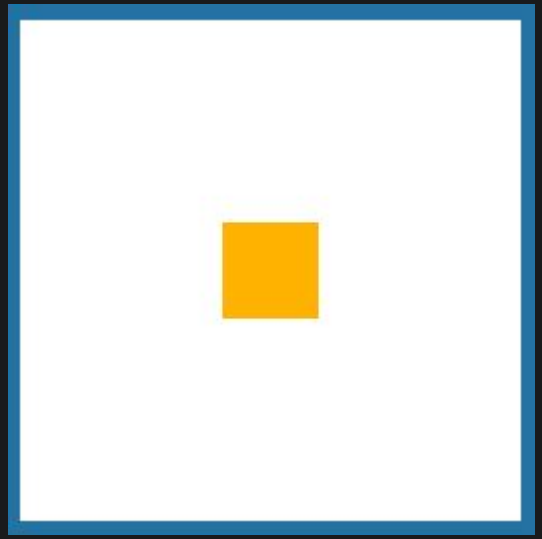
Column

レイアウト系の基本Widget

要素を縦並びに配置する

要するにflexboxのcolumn

引用 : https://api.flutter.dev/flutter/widgets/Column-class.html



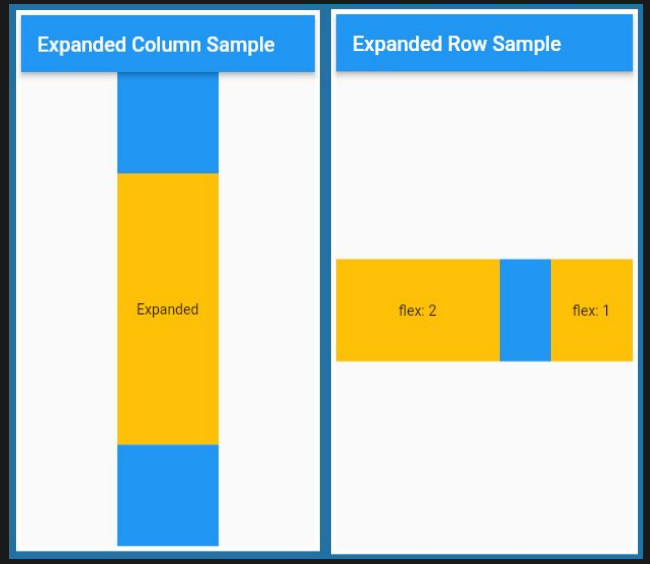
Container

レイアウト系の基本Widget

Widgetを入れるコンテナ

サイズ指定や枠線等の装飾ができる

引用：https://api.flutter.dev/flutter/widgets/Container-class.html



Expanded

レイアウト系の基本Widget

余白いっぱいまで要素を広げる

Row、Columnと併用する

引用：https://api.flutter.dev/flutter/widgets/Expanded-class.html

Work2

アプリのレイアウトを作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work2

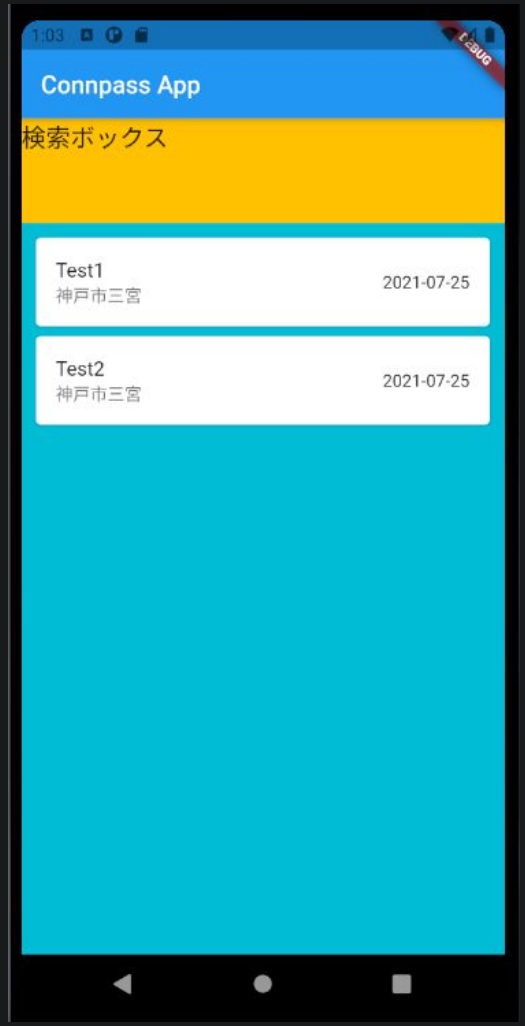
レイアウトの実現パターンは様々
他のLayout Widgetはこちら💁

https://flutter.dev/docs/development/ui/widgets/layout

Work3

Event List
を作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work3



List

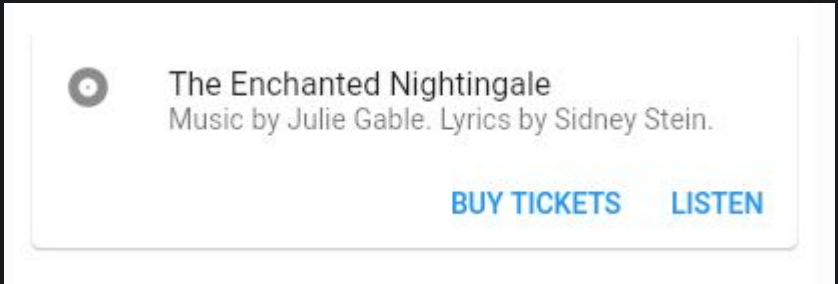
ListView

スクロール系の基本Widget

一覧表示するときに使う

Columnと違い画面上に表示されている要素だけレンダリングされる

引用：https://api.flutter.dev/flutter/widgets/ListView-class.html

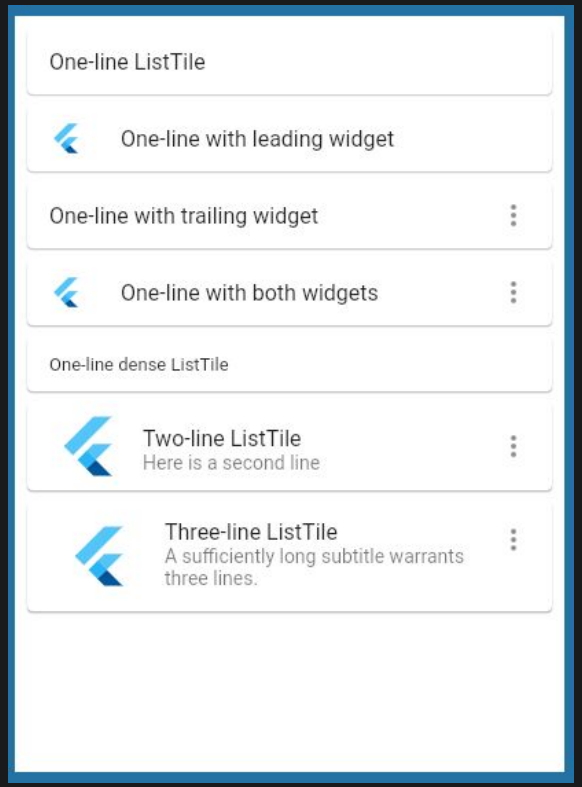


Card

Material ComponentsのWidget

マテリアルデザインのCardsを実現する

引用 : https://api.flutter.dev/flutter/material/Card-class.html



ListTile

高さが固定された行

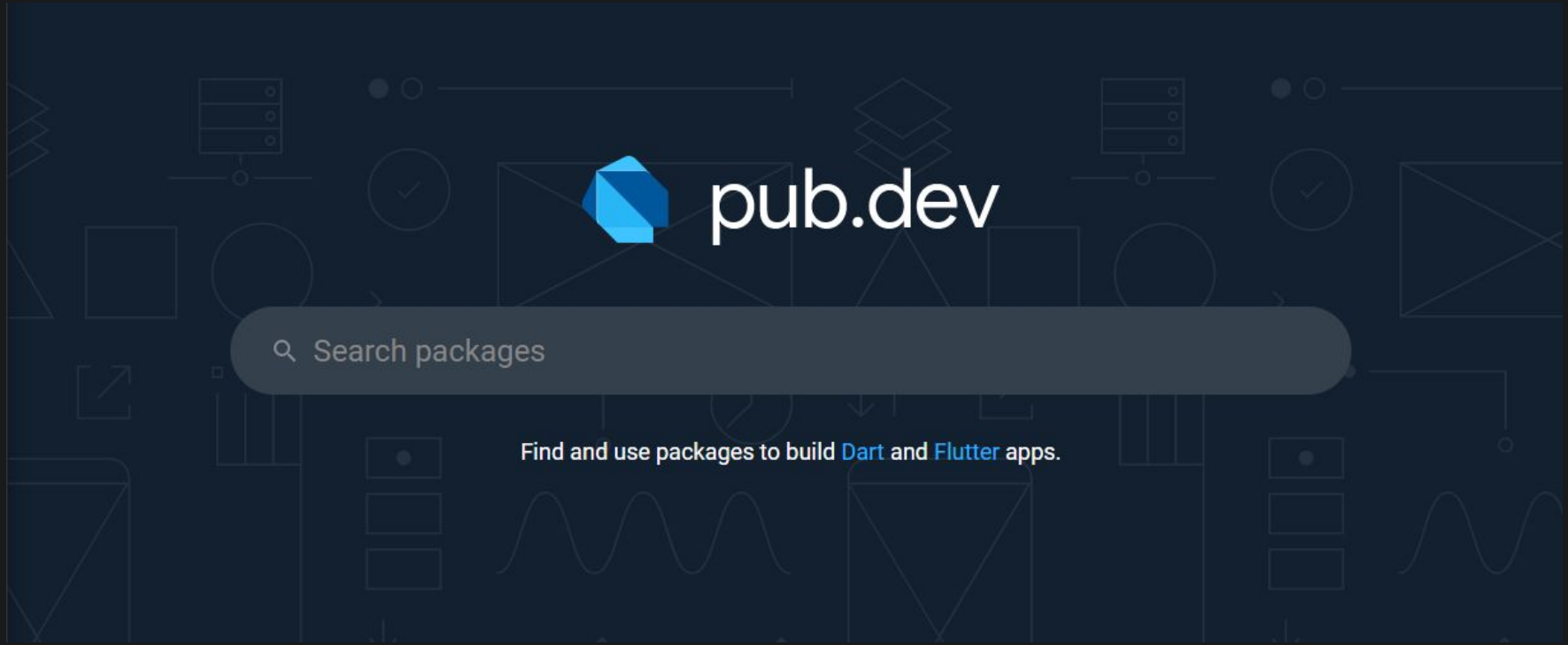
マテリアルデザインのListsを実現する

CardやListViewの中で使うことが多い

引用 : https://api.flutter.dev/flutter/material/ListTile-class.html

Date Format

パッケージを追加する



引用 : https://pub.dev/

intl

多言語対応に便利な機能を提供

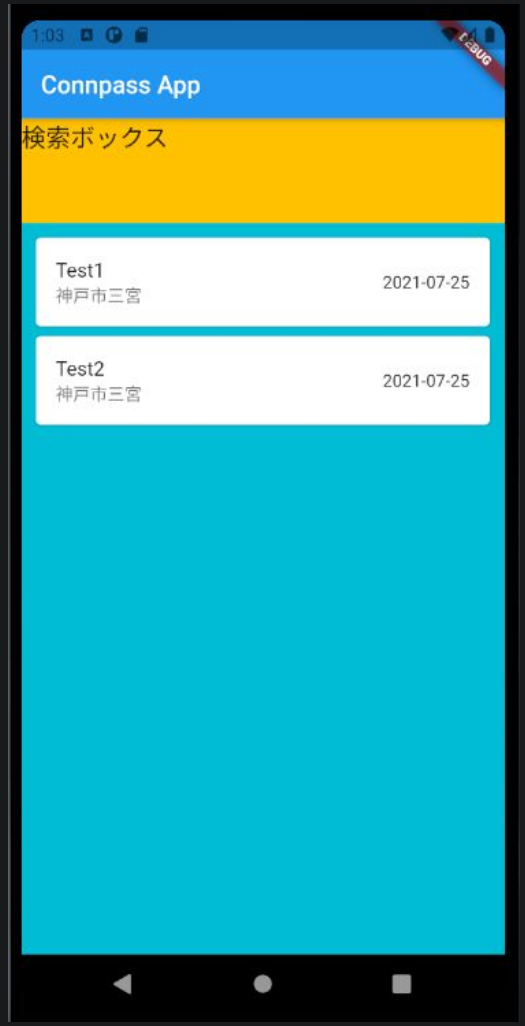
DateTimeを扱う場合に高確率で使用

引用 : https://pub.dev/packages/intl

Work3

Event List
を作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work3



一覧表示には
Scrolling Widgetが便利

https://flutter.dev/docs/development/ui/widgets/scrolling

Work4

Event詳細ページを作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work4

Routing

Navigate with named routes

引用：https://flutter.dev/docs/cookbook/navigation/named-routes

```
//routesに定義していく
MaterialApp(
  title: 'Named Routes Demo',
  initialRoute: '/',
  routes: {
    '/': (context) => const FirstScreen(),
    '/second': (context) => const SecondScreen(),
  },
)

//ページ遷移はNavigatorクラスを使用する（SecondScreenに値を渡すことも可能）
onPressed: () {
  Navigator.pushNamed(context, '/second');
}
```

Work4

Event詳細ページを作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work4

その他の実装パターンは公式cookbookを参照

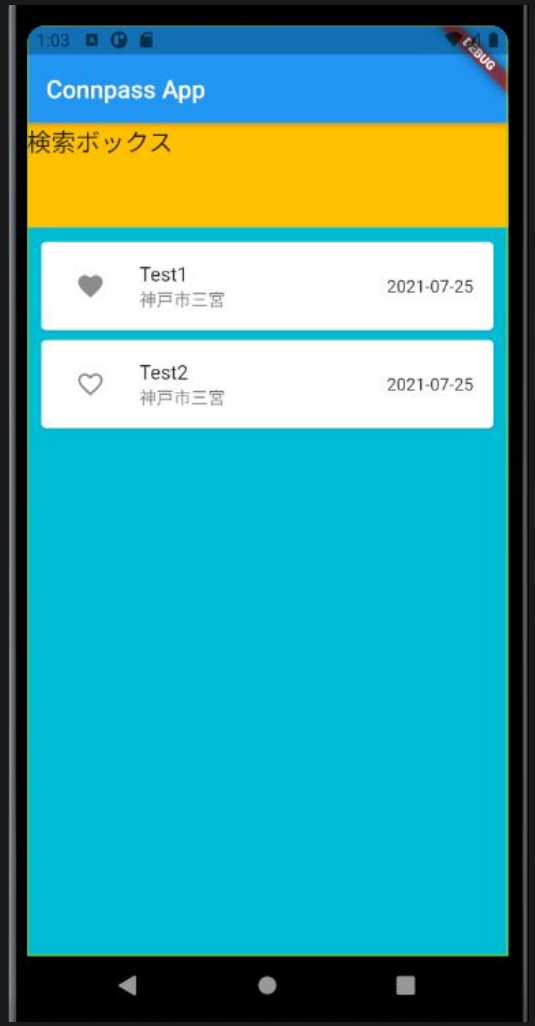
https://flutter.dev/docs/cookbook

（Navigationの項目）

Work5

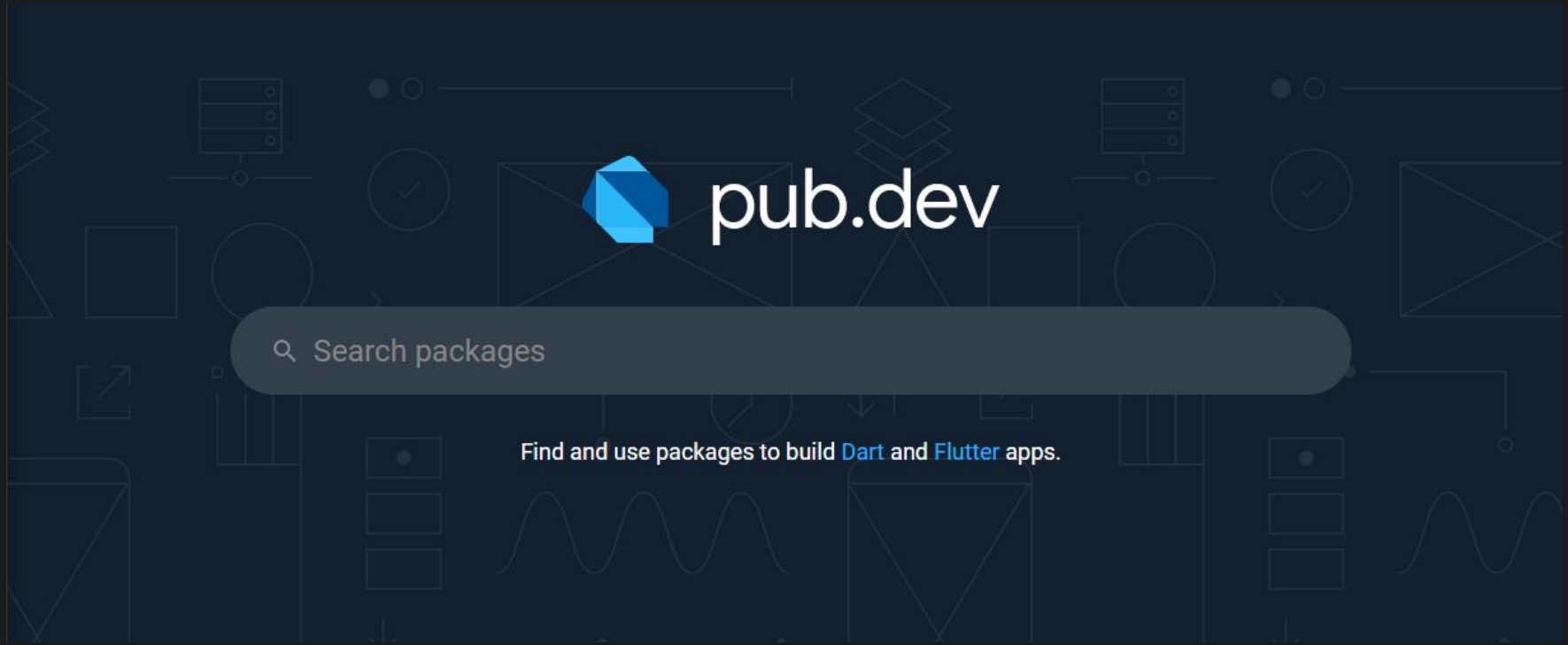
お気に入り機能を作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work5



State Management

パッケージを追加する



引用 : https://pub.dev/

provider

アプリケーションの状態管理

どこからでも参照できるストアと変更通知の仕組みを提供してくれる

引用 : https://pub.dev/packages/intl

Providerの基本的な使い方

```
//まずはChangeNotifierを継承したmodelを定義する
import 'package:flutter/material.dart';

class Item with ChangeNotifier {
  final bool isFavorite;

  Item(this.isFavorite);
}
```

Providerの基本的な使い方

```
//親WidgetでChageNotifierProviderを呼び出す
ListView.builder(
  itemBuilder: (ctx, i) => ChangeNotifierProvider.value(
    value: items[i],  //List<Item>
    child: ItemDetail(),
  ),
  itemCount: events.length,
);

//子Widgetでitems[i]の値が参照可能になる
class ItemDetail extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final item = Provider.of<Item>(context);
```

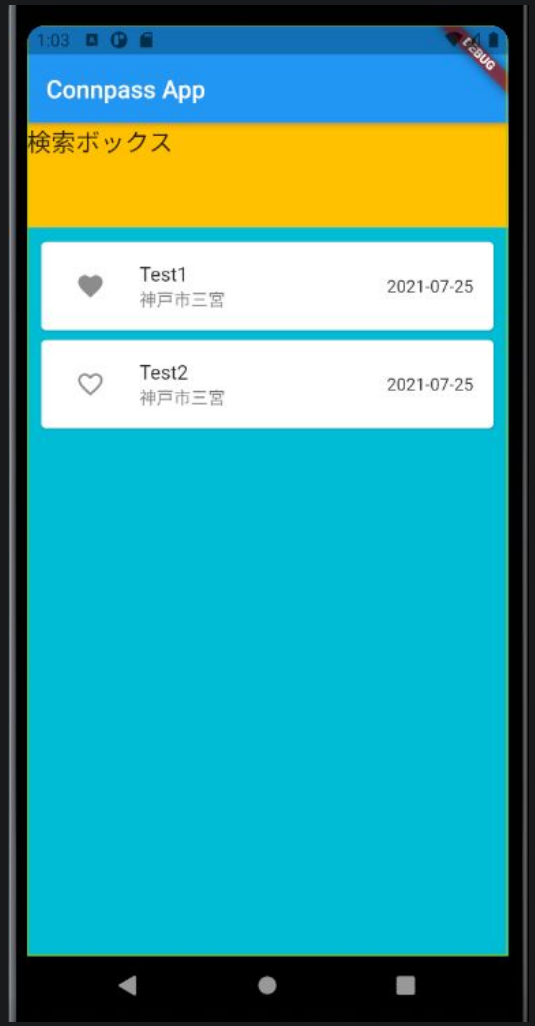
Providerの基本的な使い方

```
//notifyListerners()で変更を通知できる
import 'package:flutter/material.dart';

class Item with ChangeNotifier {
  final bool isFavorite;

  Item(this.isFavorite);

  void toggleFavoriteStatus() {
    isFavorite = !isFavorite;
    notifyListeners();  //ココ
  }
}

}
```

Work5

お気に入り機能を作ってみる

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work5



Providerの上位互換である Riverpodも存在する

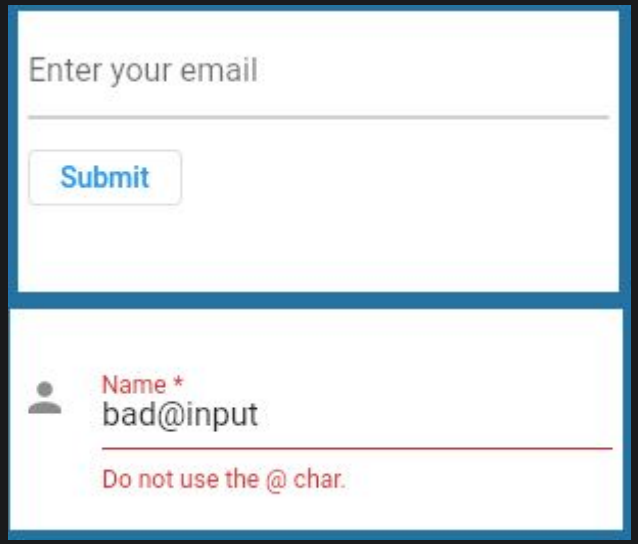
https://pub.dev/packages/riverpod

Work6

connpass API からデータを取得する

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work6

Form



Form
TextFormField

フォーム入力

TextFormFieldはmaterial component

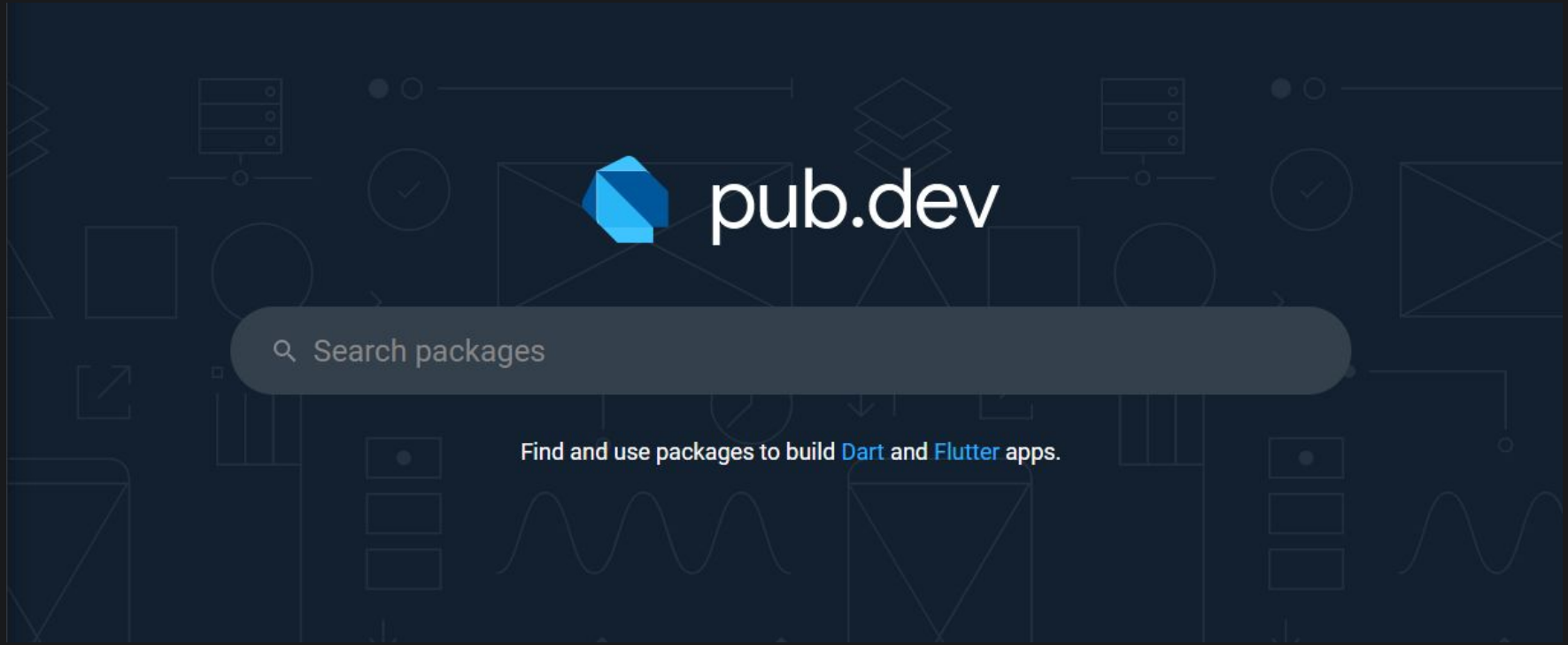
TextField Widgetもある

引用 : https://api.flutter.dev/flutter/widgets/Form-class.html

引用 : https://api.flutter.dev/flutter/material/TextFormField-class.html

Http Requests

パッケージを追加する

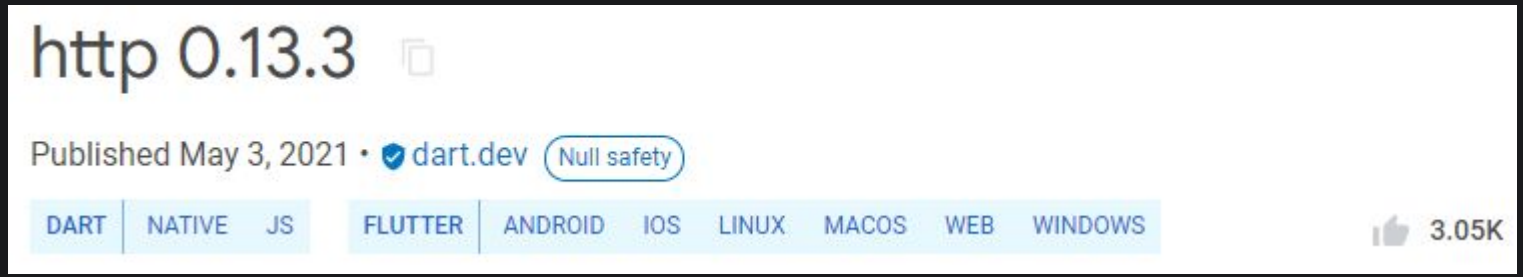


引用 : https://pub.dev/

http

httpリクエストをするためのパッケージ

今回はGETだけ使ってみる



引用 : https://pub.dev/packages/http

httpの基本的な使い方

```
import 'package:http/http.dart' as http;
...
  // Futureはjavascriptでいうpromise
  Future<void> fetch(String keyword) async {
    final params = {
      'keyword': keyword,
    };
    // URIを生成
    final uri = Uri.https('connpass.com', 'api/v1/event', params);

    // GETリクエスト
    final response = await http.get(uri);
    final data = json.decode(response.body) as Map<String, dynamic>;
...
```

Work6

connpass API からデータを取得する

https://github.com/tenkawa0/flutter_handson/tree/master/sample_code/work6

これにてハンズオンは終了です
お疲れさまでした🎉