

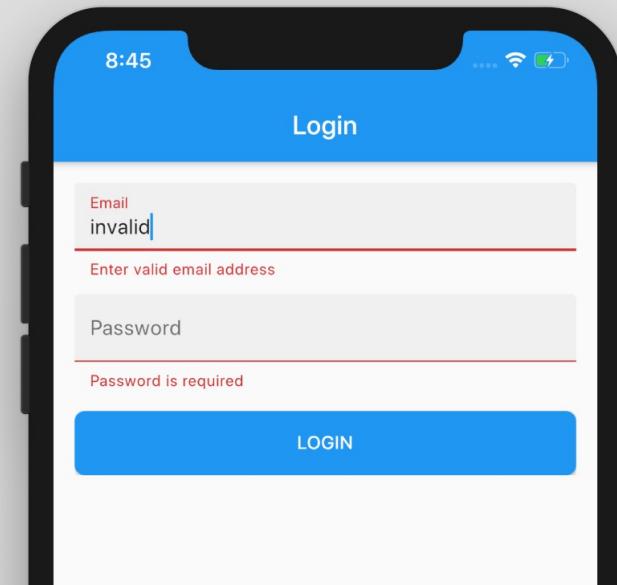
FLUTTER WEEK4

PATIPAN WATJANAPRON

การจัดการฟอร์มและการรับข้อมูลจากผู้ใช้

FORM គីអោនា

- Form គីអោនគម្រោងរបៀបទីមួយនៃការប្រើប្រាស់សាមគរណី Interact ក្នុងរបៀប។
- តាមដែនរបៀបនេះការប្រើប្រាស់សាមគរណី (Input) មិនត្រូវបានប្រើប្រាស់ឡើងទៅប្រើប្រាស់សាមគរណី (Input) ដែលបានបង្កើតឡើង។

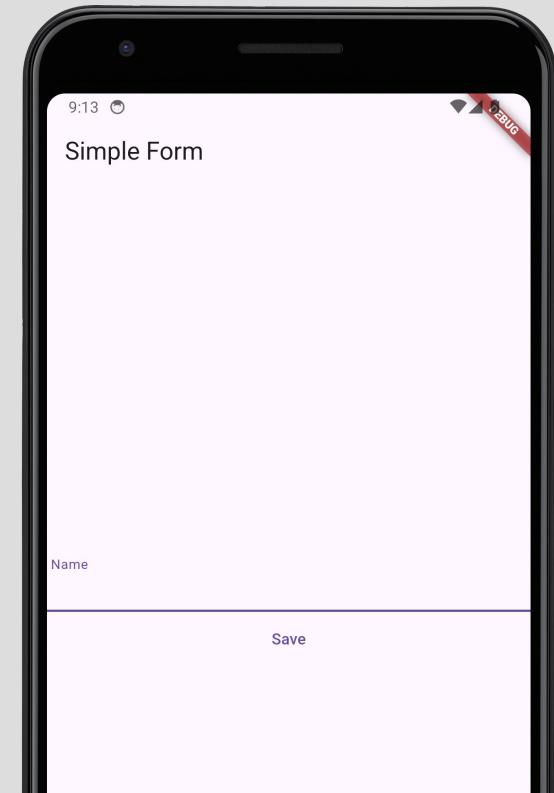


FORM WIDGET

TEXTFORMFIELD (ช่องกรอกข้อมูล)

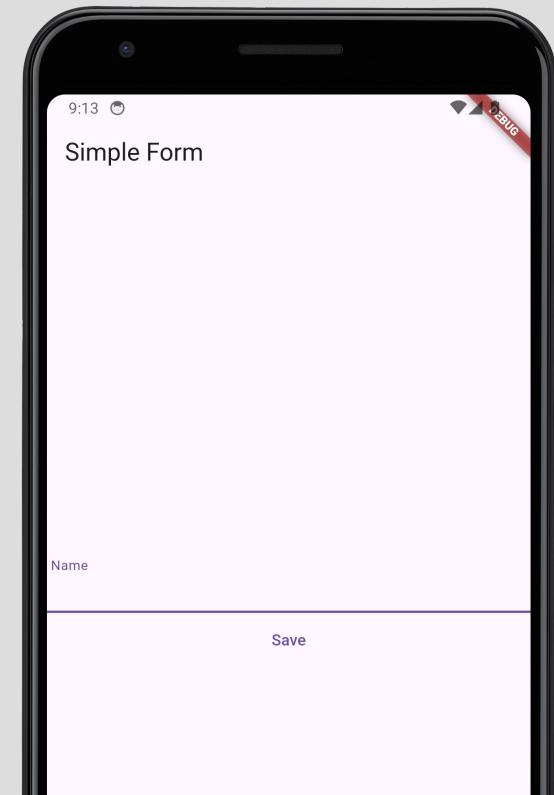
- ใช้สำหรับรับค่าจากผู้ใช้ เช่น ชื่อ อีเมล รหัสผ่าน
- มี **validator** สำหรับตรวจสอบค่าที่ป้อน
- สามารถใช้ **TextEditingController**

ในการจัดการค่าได้



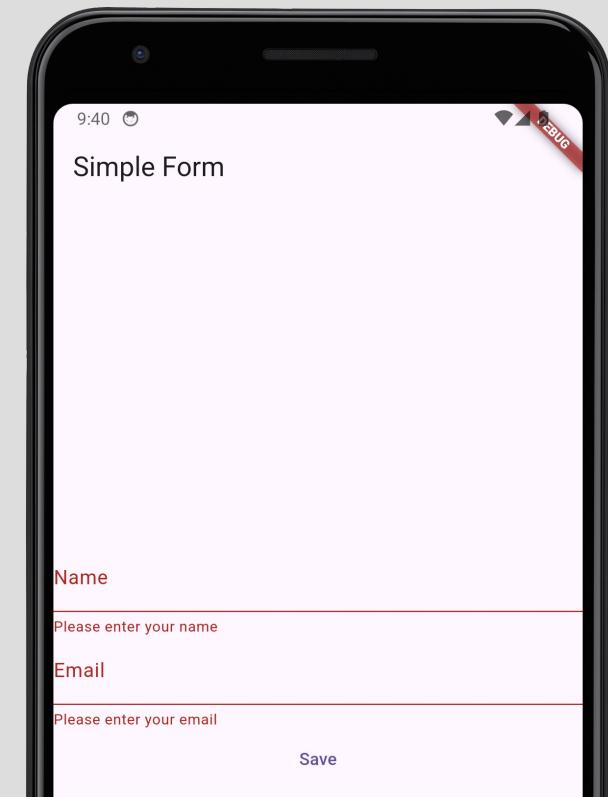
TEXTFORMFIELD (ช่องกรอกข้อความ)

```
TextField(  
    decoration: const InputDecoration(labelText: ' Name') ,  
    validator: (String? value) {  
        if (value == null || value.isEmpty) {  
            return 'Please enter your name';  
        }  
        return null;  
    },  
    onChanged: (String value) {  
        print(object: value);  
    },  
) , // TextFormField
```



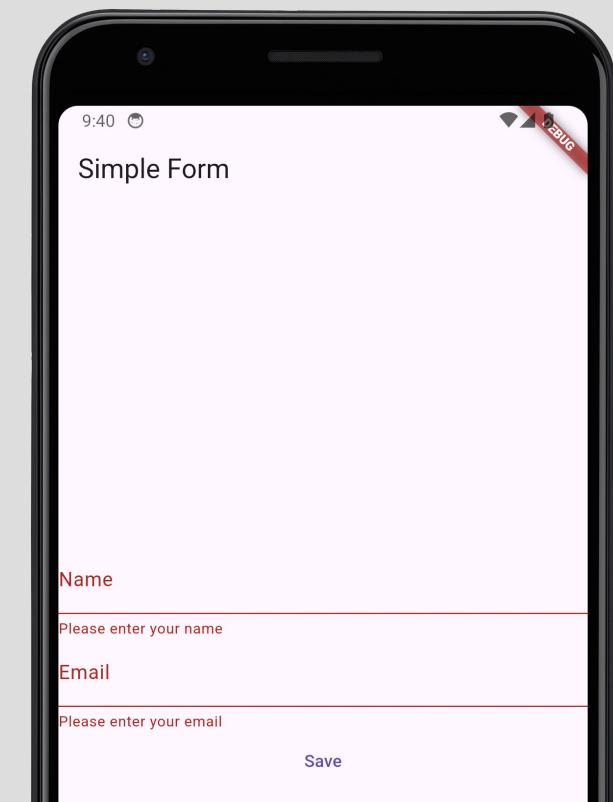
FORM (ตัวจัดการฟอร์มทั้งหมด)

- ใช้ครอบ **TextField** กรณีที่มีหลายฟิลด์
- ใช้ **GlobalKey<FormState>** เพื่อตรวจสอบ
และบันทึกค่า



FORM (ตัวจัดการฟอร์มทั่วหมด)

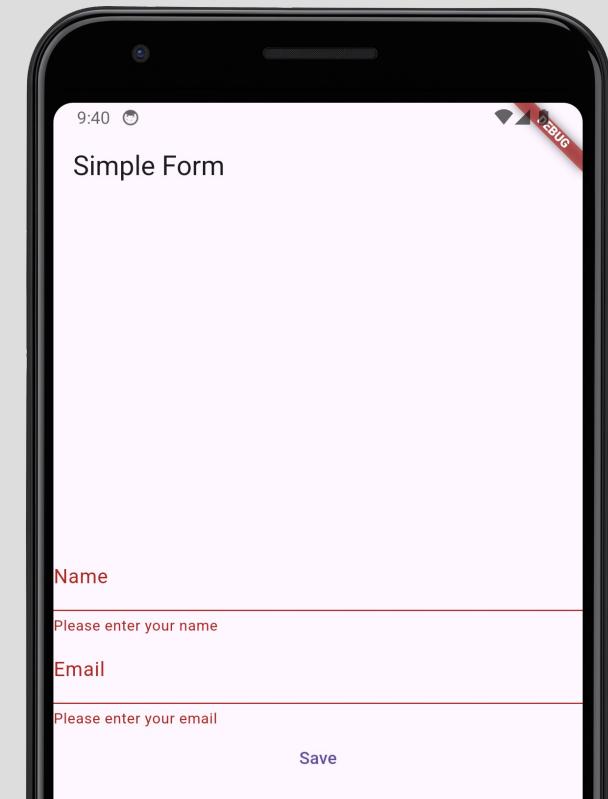
```
final GlobalKey<FormState> _formKey = GlobalKey<FormState>();  
  
Form(  
  key: _formKey,  
  child: Column(  
    mainAxisAlignment: MainAxisAlignment.center,  
    children: <Widget>[  
      TextFormField(  
        decoration: const InputDecoration(labelText: 'Name'),  
        validator: (String? value) {  
          if (value == null || value.isEmpty) {  
            return 'Please enter your name';  
          }  
          return null;  
        },  
      ), // TextFormField  
      TextFormField(  
        decoration: const InputDecoration(labelText: 'Email'),  
        validator: (String? value) {  
          if (value == null || value.isEmpty) {  
            return 'Please enter your email';  
          }  
          return null;  
        },  
      ), // TextFormField  
      TextButton(  
        onPressed: () {  
          if (_formKey.currentState!.validate()) {  
            print('Form is valid');  
          }  
        },  
        child: const Text(data: 'Save'),  
      ), // TextButton  
    ],  
  ) // Column
```



TEXTEDITINGCONTROLLER

(จัดการค่าของ TEXTFORMFIELD)

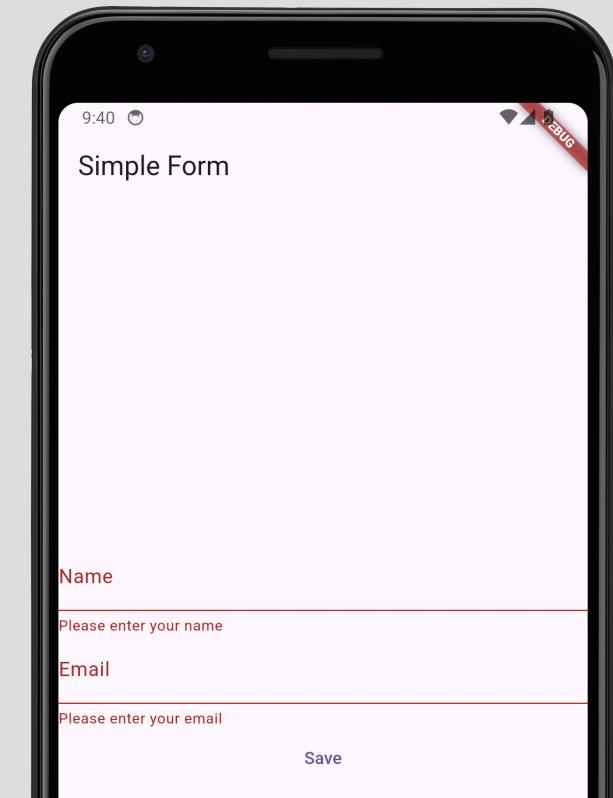
- ใช้เก็บค่าจากช่องกรอกข้อมูล
- ใช้ดึงและอัปเดตค่าใน **TextField**



TEXTEDITINGCONTROLLER

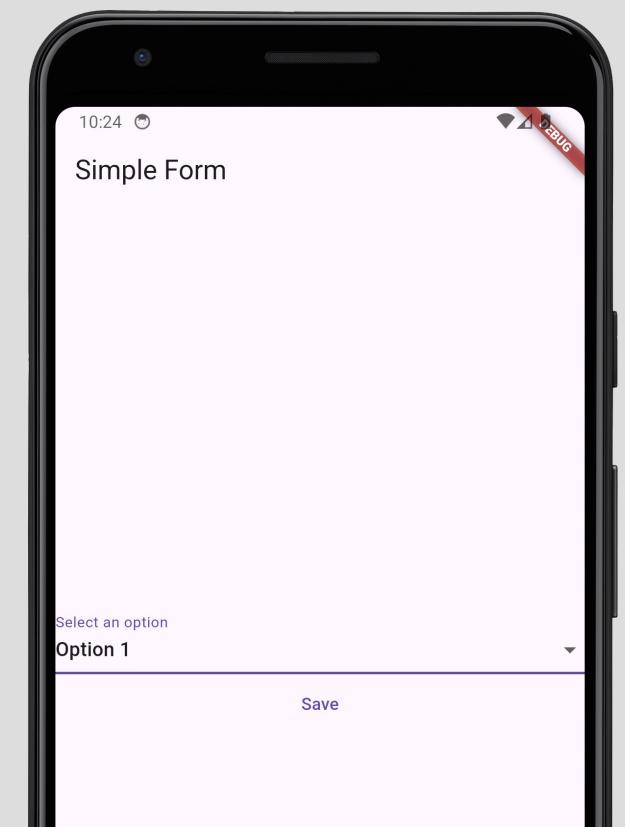
(จัดการค่าของ TEXTFORMFIELD)

```
final TextEditingController _nameController = TextEditingController();  
  
TextFormField(  
    controller: _nameController,  
    decoration: const InputDecoration(labelText: 'Name'),  
    validator: (String? value) {  
        if (value == null || value.isEmpty) {  
            return 'Please enter your name';  
        }  
        return null;  
    },  
    onChanged: (String value) {  
        print(object: value);  
    },  
, // TextFormField  
TextButton(  
    onPressed: () {  
        print(object: 'User input: ${_nameController.text}');  
    },  
    child: const Text(data: 'Save'),  
, // TextButton
```



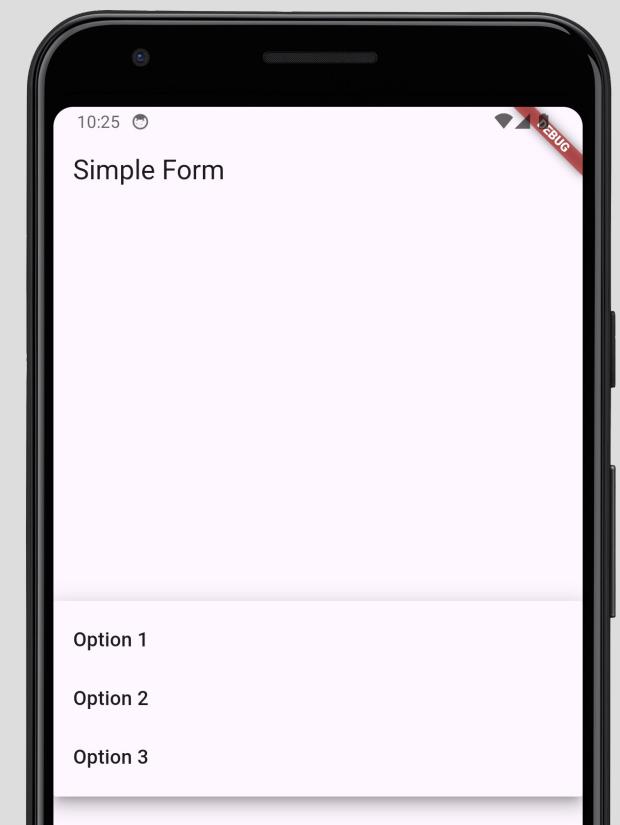
DROPDOWNBUTTONONFORMFIELD (INPUT แบบตัวเลือก DROPDOWN)

- ใช้เลือกค่าจากการแบบ Drop down
- รองรับ **validator** สำหรับตรวจสอบค่า



DROPDOWNBUTTONONFORMFIELD (INPUT แบบตัวเลือก DROPDOWN)

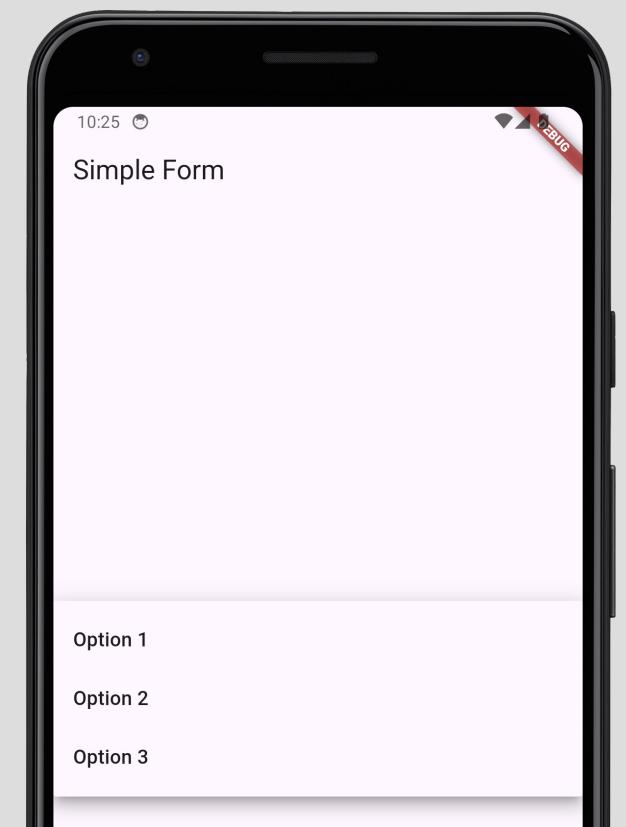
- ใช้เลือกค่าจากการแบบ Drop down
- รองรับ **validator** สำหรับตรวจสอบค่า



DROPODOWNBUTTONFORMFIELD (INPUT แบบตัวเลือก DROPODOWN)

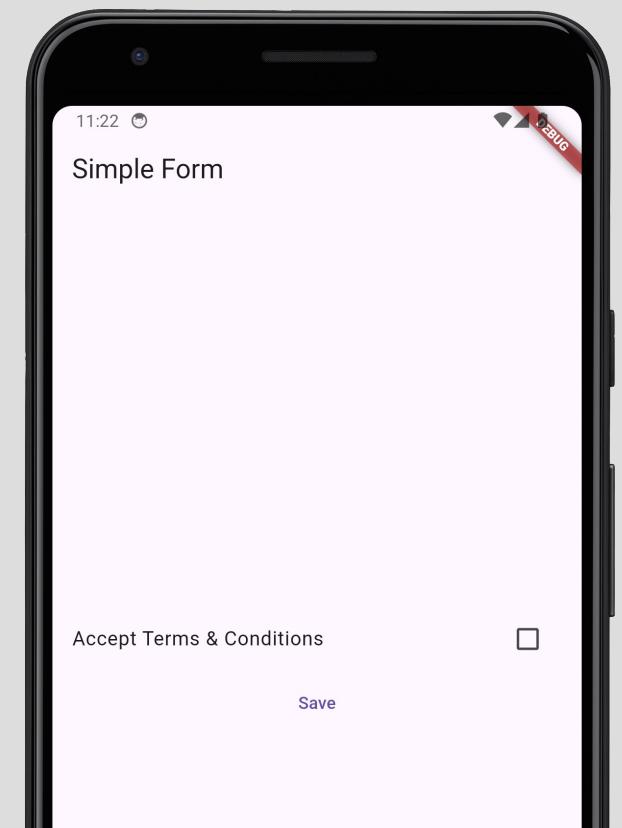
```
String? _selectedItem;

DropdownButtonFormField<String>(
    decoration:
        const InputDecoration(labelText: 'Select an option'),
    value: _selectedItem,
    items: <String>['Option 1', 'Option 2', 'Option 3']
        .map<DropdownMenuItem<String>>(toElement: (String item) =>
            DropdownMenuItem<String>(value: item, child: Text(data: item)))
        .toList(),
    onChanged: (String? value) {
        _selectedItem = value;
    },
    validator: (String? value) =>
        value == null ? 'Please select an option' : null,
) // DropdownButtonFormField
```



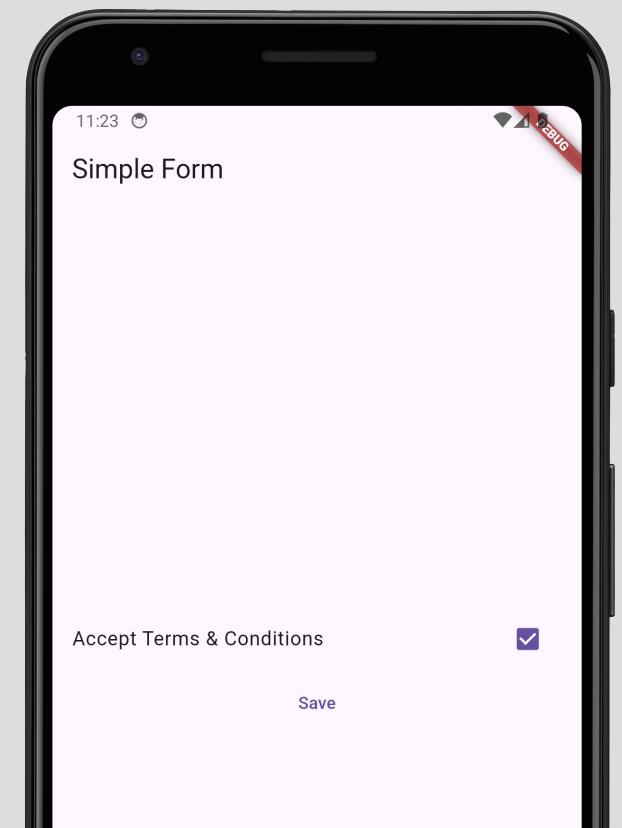
CHECK BOX (INPUT แบบ CHECKBOX)

- ใช้เลือกค่า true/false (Boolean)
- ใช้หลายตัวเลือกได้



CHECK BOX (INPUT แบบ CHECKBOX)

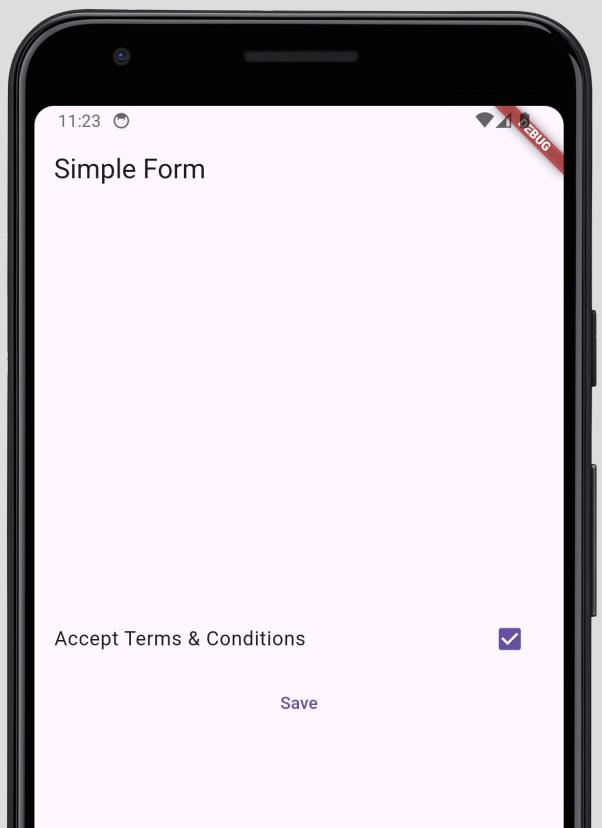
- ใช้เลือกค่า true/false (Boolean)
- ใช้หลายตัวเลือกได้



CHECK BOX

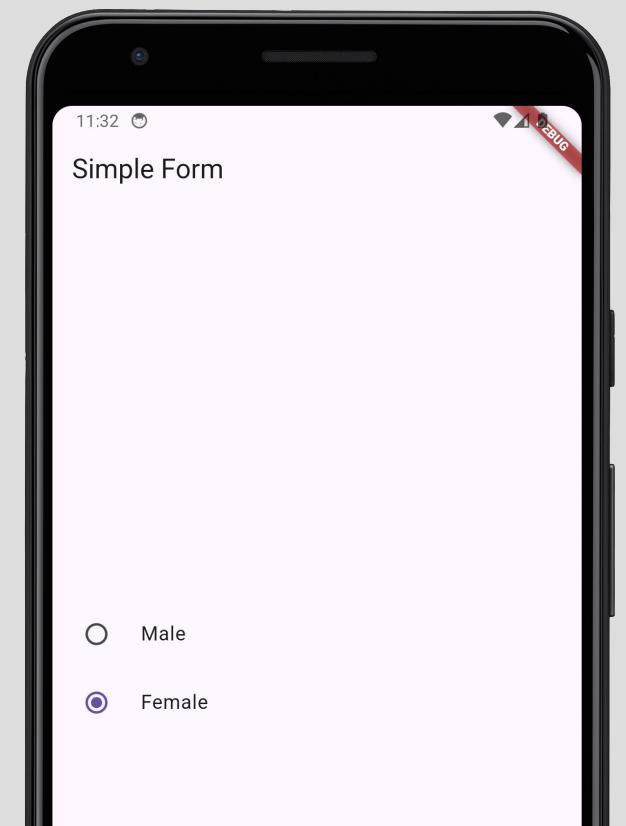
(INPUT ԱՅՍ ՉԵԿԲՈԽ)

```
bool _isChecked = false;  
  
CheckboxListTile(  
    title: const Text(data: 'Accept Terms & Conditions'),  
    value: _isChecked,  
    onChanged: (bool? value) {  
        setState(fn: () {  
            _isChecked = value!;  
        });  
    },  
, // CheckboxListTile
```



RADIO (INPUT แบบ RADIO)

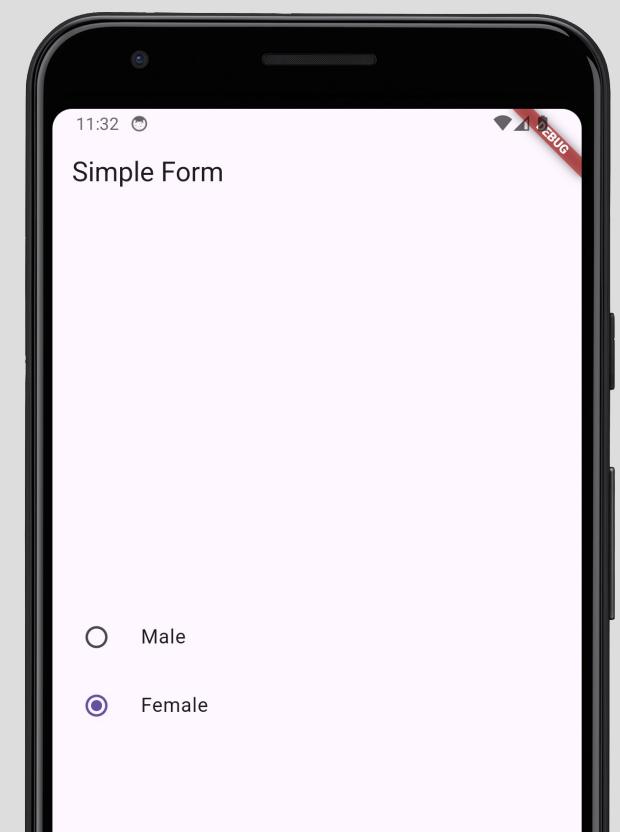
- ใช้สำหรับเลือกได้เพียงตัวเลือกเดียวจากหลายตัวเลือก เช่นการเลือกเพศ



RADIO (INPUT ԱՅՍ ԲԱՏԻ)

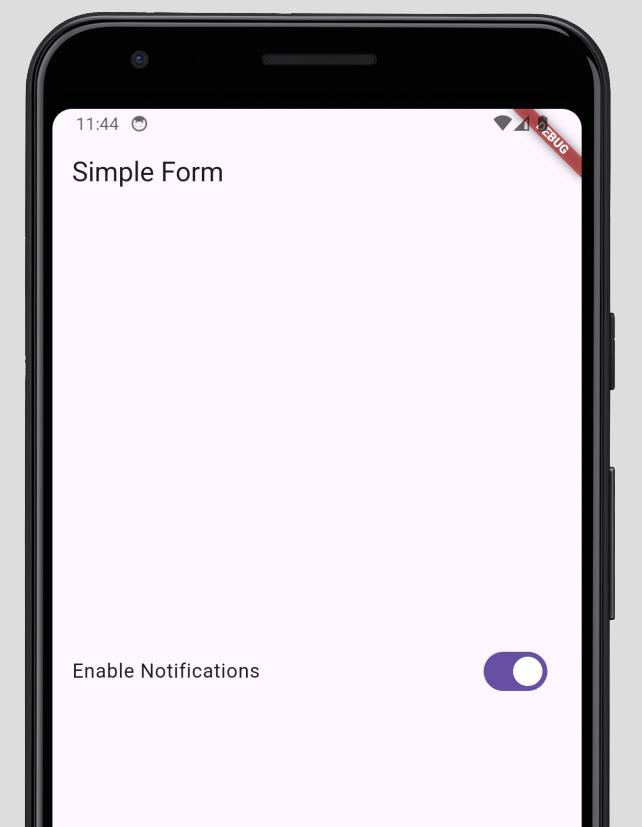
```
String? _gender = 'Female';

Column(
  children: <Widget>[
    RadioListTile<String>(
      title: const Text(data: 'Male'),
      value: 'Male',
      groupValue: _gender,
      onChanged: (String? value) {
        setState(fn: () {
          _gender = value.toString();
        });
      },
    ), // RadioListTile
    RadioListTile<String>(
      title: const Text(data: 'Female'),
      value: 'Female',
      groupValue: _gender,
      onChanged: (String? value) {
        setState(fn: () {
          _gender = value.toString();
        });
      },
    ), // RadioListTile
  ], // Column
```



SWITCH (INPUT แบบเปิด/ปิด)

- ใช้สำหรับเปลี่ยนสถานะ **ON/OFF** เช่นการตั้งค่า การเปิดปิดพัฟฟ์ชันในระบบ

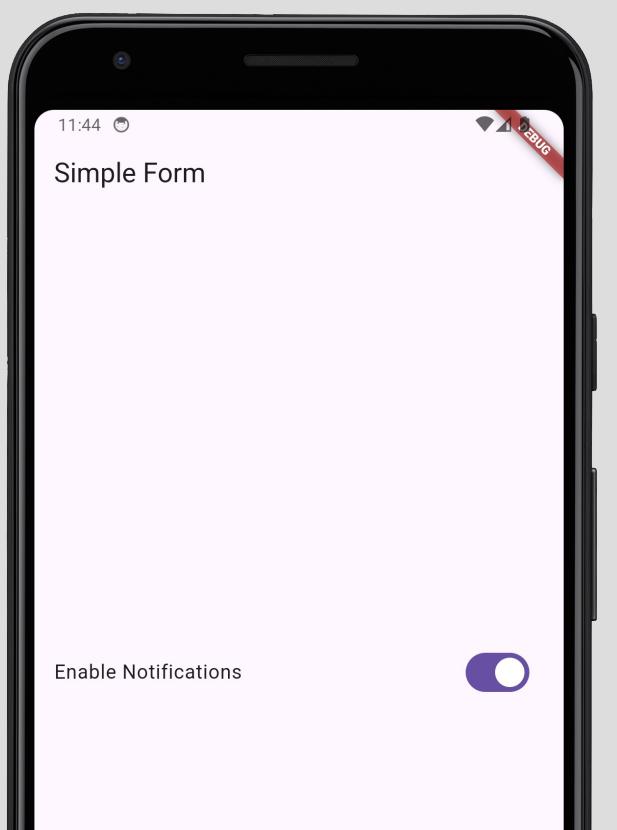


SWITCH

(INPUT แบบเปิด/ปิด)

```
bool _isSwitched = true;
```

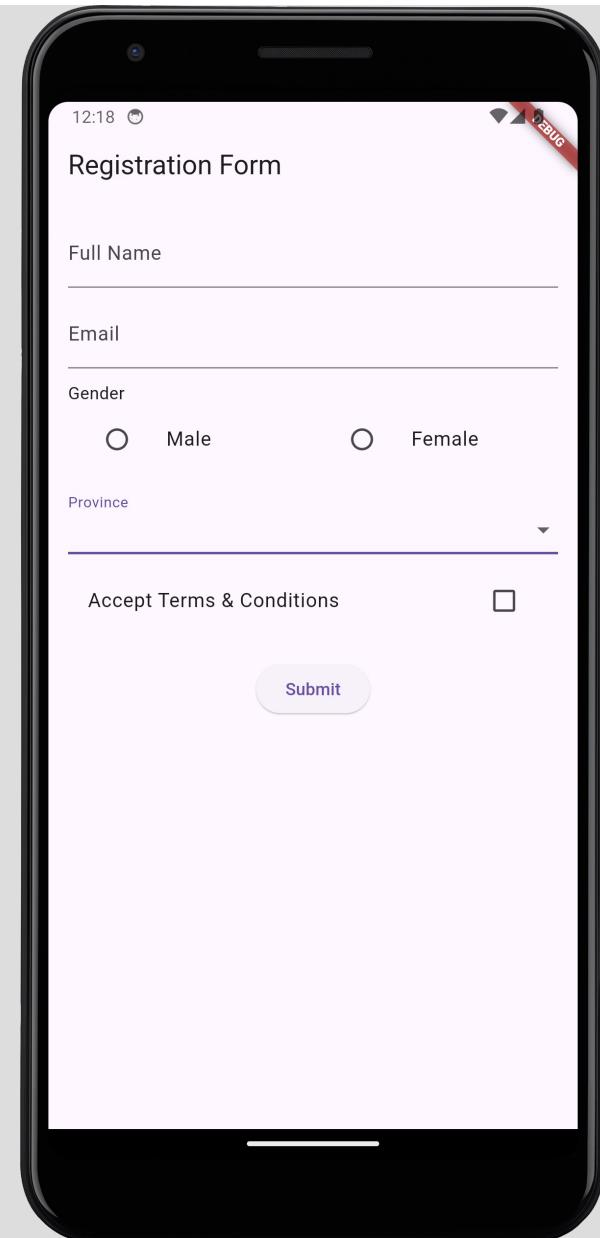
```
SwitchListTile(  
    title: const Text(data: 'Enable Notifications'),  
    value: _isSwitched,  
    onChanged: (bool value) {  
        setState(fn: () {  
            _isSwitched = value;  
        });  
    },  
) // SwitchListTile
```



ลองสร้าง FORM INPUT กัน

โจทย์: สร้างฟอร์มลงทะเบียนโดยมีองค์ประกอบดังนี้

- ชื่อ-นามสกุล (TextField)
- อีเมล (TextField)
- เพศ (Radio Button)
- จังหวัด (Dropdown) (ตัวเลือกแค่ 4 จังหวัด ['Bangkok', 'Chiang Mai', 'Phuket', 'Khon Kaen'])
- ยอมรับเงื่อนไข (Checkbox)
- ปุ่ม Submit (กด Submit ได้เมื่อ validate แต่ละ field สำเร็จ)
- *จัดวาง Layout ให้ตามความเหมาะสม หรือวาง Layout ตามตัวอย่างภาพหน้าจอ



การเชื่อมต่อ API

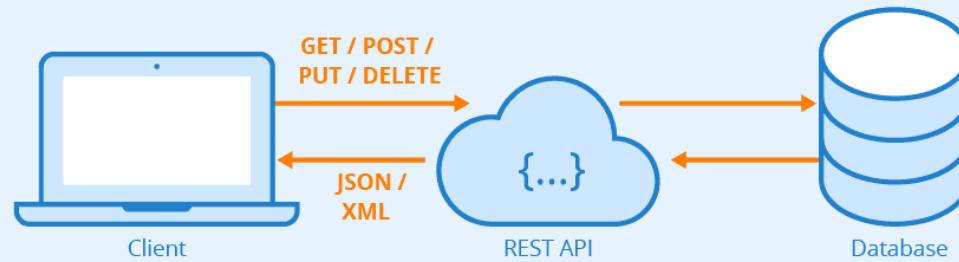
API คืออะไร?

- API (Application Programming Interface) คือ ชุดของกฎและโปรโตคอลที่กำหนดวิธีการสื่อสารระหว่างซอฟต์แวร์หรือระบบต่างๆ โดย API ทำหน้าที่เป็นตัวกลางที่ช่วยให้อแอปพลิเคชันสามารถแลกเปลี่ยนข้อมูลกันได้โดยไม่ต้องรู้ถึงโครงสร้างภายในของกันและกัน

API ทำงานอย่างไร?

1. Client (ผู้ร้องขอ) → แอปของเรางส่ง request ไปที่ API (เช่น ขอข้อมูลผู้ใช้)
2. Server (ผู้ให้บริการ API) → API ประมวลผล request และดึงข้อมูลจากฐานข้อมูล
3. Response (ผลลัพธ์ที่ได้กลับมา) → API ส่งข้อมูลกลับมาเป็น JSON หรือ XML ให้กับ Client
4. Client นำข้อมูลไปแสดงผล เช่น แสดงใน ListView หรือบน UI อื่นๆ

API ทำงานอย่างไร?



RESTFUL API

RESTful API Methods



GET

Retrieve a
resource



POST

Create a
resource



PUT

Replace a
resource



PATCH

Update a
resource



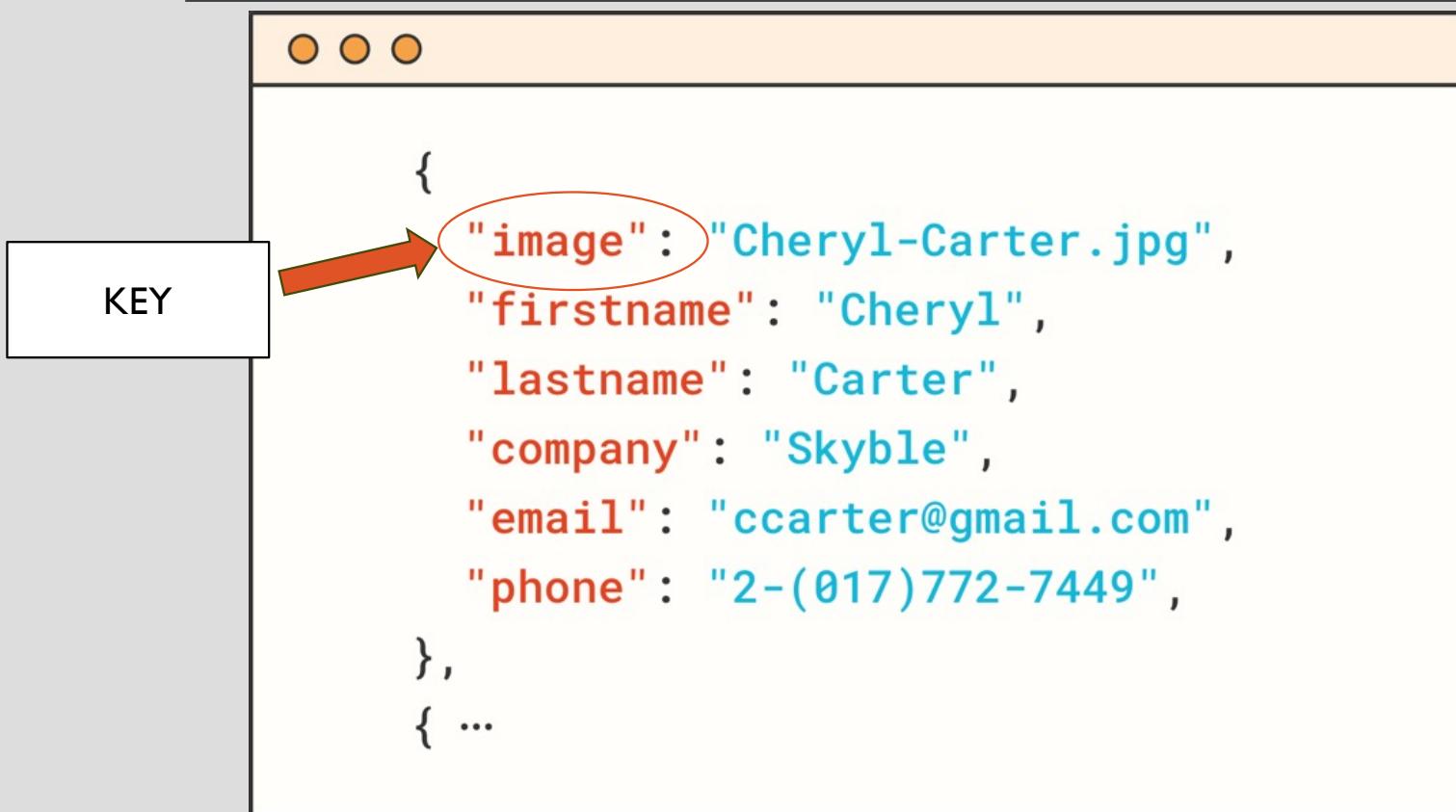
DELETE

Delete a
resource

JSON คืออะไร

- JSON (JavaScript Object Notation) เป็นรูปแบบข้อมูลที่ใช้แลกเปลี่ยนข้อมูลระหว่างระบบต่างๆ ที่มีโครงสร้างแบบ Key-Value Pair และสามารถอ่านและเขียนได้ง่าย

JSON DATA



JSON DATA

The diagram illustrates a JSON object structure. At the top, a large box contains the text "JSON DATA". Below it, a window-like frame represents a browser or data viewer. Inside the frame, a JSON object is displayed:

```
{  
  "image": "Cheryl-Carter.jpg",  
  "firstname": "Cheryl",  
  "lastname": "Carter",  
  "company": "Skyble",  
  "email": "ccarter@gmail.com",  
  "phone": "2-(017)772-7449",  
},  
{ ...}
```

A red oval highlights the string value "Cheryl-Carter.jpg". A red arrow points from this oval to a white box on the right labeled "VALUE".

JSON DATA

The diagram illustrates a JSON object structure. At the top, a large box contains the text "JSON DATA". Below it, a window-like frame represents a browser or data viewer. Inside the frame, a JSON object is displayed:

```
{  
  "image": "Cheryl-Carter.jpg",  
  "firstname": "Cheryl",  
  "lastname": "Carter",  
  "company": "Skyble",  
  "email": "ccarter@gmail.com",  
  "phone": "2-(017)772-7449",  
},  
{ ...}
```

A red oval highlights the string value "Cheryl-Carter.jpg". A red arrow points from this oval to a white box on the right labeled "VALUE".

JSON ใน FLUTTER (DART)

- การแปลง JSON เป็น Dart Object

```
import 'dart:convert';

void main() {
    String jsonString = '{"name": "John Doe", "age": 25}';

    Map<String, dynamic> user = jsonDecode(jsonString);
    print(user['name']); // Output: John Doe
}
```

JSON ใน FLUTTER (DART)

- การแปลง Dart Object เป็น JSON

```
import 'dart:convert';

void main() {
    Map<String, dynamic> user = {"name": "Alice", "age": 22};

    String jsonString = jsonEncode(user);
    print(jsonString); // Output: {"name": "Alice", "age": 22}
}
```

JSON ใน FLUTTER (DART)

- แปลง JSON Array เป็น List ของ Dart Objects

สร้าง Class สำหรับใช้
Dart Object

```
import 'dart:convert';

// Model Class
class Student {
    final int id;
    final String name;

    // Constructor
    Student(this.id, this.name);

    // แปลง JSON เป็น Object
    Student.fromJson(Map<String, dynamic> json)
        : id = json['id'],
          name = json['name'];

    // แปลง Object เป็น JSON Map
    Map<String, dynamic> toJson() {
        return {
            'id': id,
            'name': name,
        };
    }
}
```

JSON ใน FLUTTER (DART)

- แปลง JSON Array เป็น List ของ Dart Objects

การแปลงค่าโดยใช้
Dart Object

```
void main() {
    String jsonString = '''
    [
        {"id": 1, "name": "Alice"},
        {"id": 2, "name": "Bob"},
        {"id": 3, "name": "Charlie"}
    ]
''';

// แปลง JSON String เป็น List ของ Object
List<dynamic> jsonList = jsonDecode(jsonString);
List<Student> students = jsonList.map((item) => Student.fromJson(item)).toList();

// แสดงข้อมูลของนักศึกษา
for (var student in students) {
    print("ID: ${student.id}, Name: ${student.name}");
}

// แปลง Object กลับเป็น JSON Array
String newJsonString = jsonEncode(students.map((e) => e.toJson()).toList());
print("JSON ใหม่: $newJsonString");
}
```

ประโยชน์ของ API

- ช่วยให้แอปพลิเคชันเชื่อมต่อกัน เช่น Flutter App ติดต่อกับ Backend เช่น Node.js, FastAPI, Django
- ลดการทำงานซ้ำซ้อน ไม่ต้องเขียนโค้ดทุกอย่างในแอปเอง สามารถใช้ API จากผู้ให้บริการ เช่น Google Maps, OpenWeather
- อัปเดตข้อมูลแบบ Real-time ดึงข้อมูลจาก API ทำให้แอปแสดงข้อมูลที่อัปเดตอยู่เสมอ
- รองรับการขยายตัวของแอป สามารถเพิ่มฟีเจอร์ได้ง่ายโดยใช้ API ของระบบอื่น

ทดสอบการใช้งาน API

- ทดสอบเบรียก API เพื่อ Fetch ข้อมูลจาก DB มาดูผ่าน Public API ตัวอย่าง
- <https://jsonplaceholder.typicode.com/users/1>

The screenshot shows the homepage of [JSONPlaceholder](#). At the top, there's a navigation bar with links to 'Check my new project' (with a MistCSS badge), 'Guide', 'Sponsor this project', 'Blog', and 'My JSON Server'. The main title 'JSONPlaceholder' is centered above a sub-header 'Free fake and reliable API for testing and prototyping.' Below this, it says 'Powered by [JSON Server](#) + [LowDB](#)'. A note states 'Serving ~3 billion requests each month.' On the right side, there's a small advertisement for PubNub.

Check my new project ▾ MistCSS write React components with 50% less code

JSONPlaceholder

Guide Sponsor this project Blog My JSON Server

{JSON} Placeholder

Free fake and reliable API for testing and prototyping.

Powered by [JSON Server](#) + [LowDB](#).

Serving ~3 billion requests each month.

Build real-time interactions on the world's most feature-complete PubSub network

www.pubnub.com

Ads by EthicalAds

ทดสอบการใช้งาน API

- ทดสอบเรียก API ผ่าน URL บน Browser ซึ่งจะถูกเรียกโดย method GET

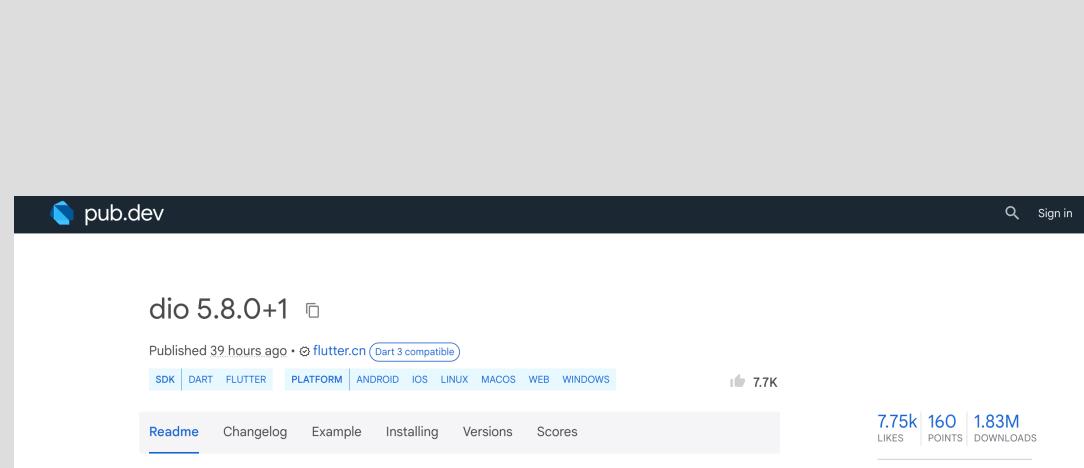
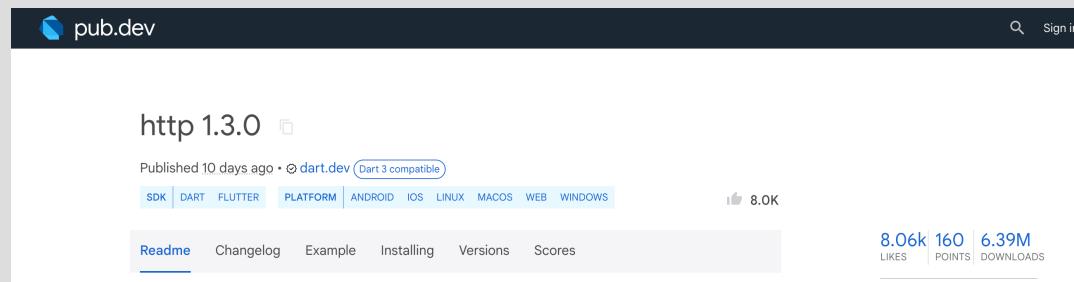


A screenshot of a web browser window. The address bar shows the URL: jsonplaceholder.typicode.com/users/1. Below the address bar, there is a 'pretty-print' checkbox. The main content area displays a JSON object representing a user profile:

```
{  
  "id": 1,  
  "name": "Leanne Graham",  
  "username": "Bret",  
  "email": "Sincere@april.biz",  
  "address": {  
    "street": "Kulas Light",  
    "suite": "Apt. 556",  
    "city": "Gwenborough",  
    "zipcode": "92998-3874",  
    "geo": {  
      "lat": "-37.3159",  
      "lng": "81.1496"  
    }  
  },  
  "phone": "1-770-736-8031 x56442",  
  "website": "hildegard.org",  
  "company": {  
    "name": "Romaguera-Crona",  
    "catchPhrase": "Multi-layered client-server neural-net",  
    "bs": "harness real-time e-markets"  
  }  
}
```

FLUTTER API PACKAGE

- ใน Flutter มี package สำหรับใช้เชื่อมต่อ API อัญญาตัวบัน <https://pub.dev/> โดยที่ได้รับความนิยมใช้จะได้แก่ http, dio โดย http จะใช้การเรียก API ทั่วไป ส่วน dio จะใช้ในโปรเจคที่ซับซ้อนขึ้นมา



FLUTTER API PACKAGE

- ติดตั้ง Package http สำหรับเข้ามต่อ API

With Flutter:

```
$ flutter pub add http
```

This will add a line like this to your package's pubspec.yaml (and run an implicit dart pub get):

```
dependencies:  
  http: ^1.3.0
```

FLUTTER API PACKAGE

- การเรียก API ด้วย package http

```
import 'package:http/http.dart' as http;
import 'dart:convert';

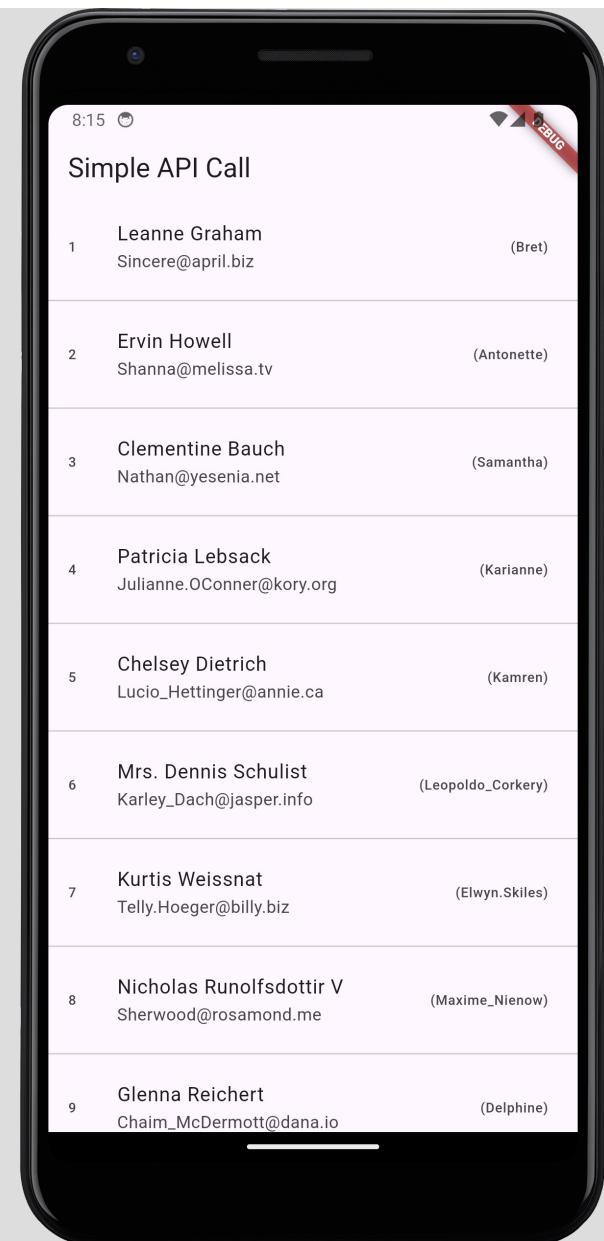
void main() {
    void fetchUser() async {
        try {
            var response = await http
                .get(Uri.parse('https://jsonplaceholder.typicode.com/users/1'));
            if (response.statusCode == 200) {
                var data = jsonDecode(response.body);
                print('Name: ${data["name"]}');
            } else {
                print('Failed to fetch data');
            }
        } catch (e) {
            print('Error: $e');
        }
    }

    fetchUser();
}
```

ลองสร้างหน้าจอดึงข้อมูลจาก API

โจทย์: สร้างหน้าจอดึงข้อมูลจาก API

- ดึงข้อมูลจาก <https://jsonplaceholder.typicode.com/users/> โดยใช้ http package
- สร้าง Class User มาเก็บข้อมูล JSON Array จาก Data ที่ได้จาก API โดยกำหนดพิล็อตใน Class User ได้แก่ id, name, username, email
- สร้าง ListView แสดง User ทั้งหมด โดยแต่ละรายการใช้ ListTile แสดงผลดังปรากฏในภาพจอย่อๆ
- Hint : ใช้ ListView.separated



ASSIGNMENT

โจทย์: สร้างหน้าจอดึงข้อมูลจาก API

- ดึงข้อมูลจาก <https://aqicn.org/data-platform/token/> โดยใช้ http package (ดูวิธีรับ link api ที่มี Token ของตนในหน้าถัดไป)
- ใช้ข้อมูลจาก API แสดงผลคุณภาพอากาศในปัจจุบัน ทั้งหมด 3 ค่าได้แก่ (Aqi, City, Temperature) ขอให้แสดงข้อมูลให้ครบ 3 ค่านี้
- สามารถออกแบบหน้าจอได้ตามต้องการ (jintna) ไม่จำเป็นต้องเหมือนกับตัวอย่าง หรือหากคิดไม่ออกก็สามารถออกแบบตามหน้าจอตัวอย่างได้
- กำหนดส่งภายในวันอาทิตย์ที่ 9 กุมภาพันธ์ 2568 เวลา 23.59 น. โดย capture หน้าจอแอปและ push code ขึ้น git



วิธีรับ TOKEN สำหรับใช้งาน API (การบ้าน)

- <https://aqicn.org/data-platform/token/>
- กรอก Email และ Name เพื่อขอ Token สำหรับใช้งาน API โดยข้อความยืนยันจะส่งไปที่ Email

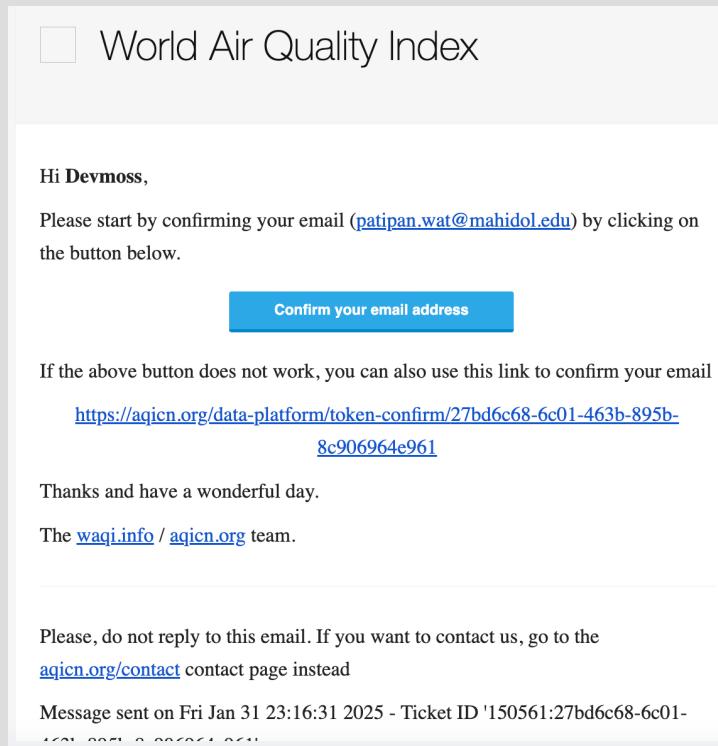
The screenshot shows a web browser window with the URL <https://aqicn.org/data-platform/token/> in the address bar. The page title is "Air Quality Open Data Platform API Token Request Form". The form contains the following fields:

- A checkbox labeled "I agree with the [Terms of Service](#)".
- An input field for "Enter your email address" containing "name@example.com".
- An input field for "Enter your name" containing "Full name".
- A blue "Submit" button.

At the bottom of the page, there is a footer with links: HOME, HERE, MAP, MASK, FAQ, SEARCH, CONTACT, LINKS, and a link to the Terms of Service.

วิธีรับ TOKEN สำหรับใช้งาน API (การบ้าน)

- เปิด Email และ Confirm your email จากนั้นจะขึ้นหน้าเว็บให้ยืนยันว่าไม่ใช่ Robot



วิธีรับ TOKEN สำหรับใช้งาน API (การบ้าน)

- ใช้ url ที่ได้สำหรับเรียก API ใช้งาน

The screenshot shows a web browser window for the "Air Quality Open Data Platform". The title bar reads "Air Quality Open Data Platform API Token Confirmation". The page content includes a message "Congratulation, your registration already validated.", a placeholder "Your token is [REDACTED]", and a link "https://api.waqi.info/feed/beijing/?token=[REDACTED]". A large blue gear icon with the word "API" is positioned on the right. A red oval highlights the token URL. At the bottom, there is a JSON snippet and a navigation bar.

Congratulation, your registration already validated.

Your token is [REDACTED]

You can now try, for instance, to get the beijing feed using:
[https://api.waqi.info/feed/beijing/?token=\[REDACTED\]](https://api.waqi.info/feed/beijing/?token=[REDACTED])

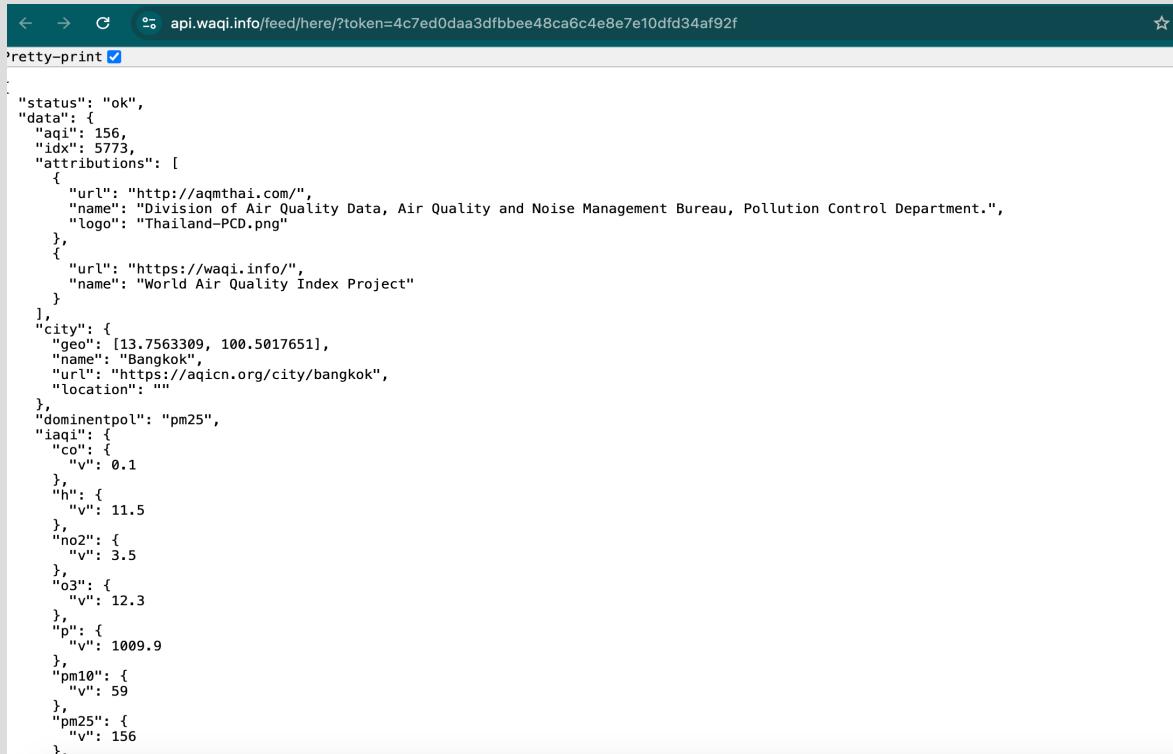
And you will get this result:

```
{
  "status": "ok",
  "data": [
    {
      "aqi": 77,
      "idx": 1451,
      "attributions": [
        {
          "url": "http://www.bjmemc.com.cn/",
          "name": "Beijing Environmental Protection Monitoring Center (北京市环境保护监测中心)"
        },
        {
          "url": "https://waqi.info/",
          "name": "World Air Quality Index Project"
        }
      ],
      "city": {
        "geo": [
          39.954592,
          116.468117
        ]
      }
    }
  ]
}
```

| HOME | HERE | MAP | MASK | FAQ | SEARCH | CONTACT | LINKS |

วิธีรับ TOKEN สำหรับใช้งาน API (การบ้าน)

- ลองทดสอบเรียก API ผ่าน Postman หรือ Browser



```
status: "ok",
data: {
  aqi: 156,
  idx: 5773,
  attributions: [
    {
      url: "http://aqmthai.com/",
      name: "Division of Air Quality Data, Air Quality and Noise Management Bureau, Pollution Control Department.",
      logo: "Thailand-PCD.png"
    },
    {
      url: "https://waqi.info/",
      name: "World Air Quality Index Project"
    }
  ],
  city: {
    geo: [13.7563309, 100.5017651],
    name: "Bangkok",
    url: "https://aqicn.org/city/bangkok",
    location: ""
  },
  dominantpol: "pm25",
  iaqi: {
    co: {
      v: 0.1
    },
    h: {
      v: 11.5
    },
    no2: {
      v: 3.5
    },
    o3: {
      v: 12.3
    },
    p: {
      v: 1009.9
    },
    pm10: {
      v: 59
    },
    pm25: {
      v: 156
    }
  }
}
```

คำแนะนำเพิ่มเติม (การบ้าน)

- การเข้าถึงค่าแต่ละค่าที่ต้องการจาก response ของ API
- 1. AQI = {response}.data.aqi
- 2. CITY = {response}.data.city.name
- 3. Temperature = {response}.data.iaqi.t.v

คำแนะนำเพิ่มเติม (การบ้าน)

- <https://aqicn.org/scale/> หน้าเว็บสำหรับกำหนดเงื่อนไข scale สี และขอความระดับสภาพอากาศ

The table below defines the Air Quality Index scale as defined by the US-EPA 2016 standard:

| AQI | Air Pollution Level | Health Implications | Cautionary Statement (for PM2.5) |
|----------|--------------------------------|--|---|
| 0 - 50 | Good | Air quality is considered satisfactory, and air pollution poses little or no risk | None |
| 51 - 100 | Moderate | Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution. | Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion. |
| 101-150 | Unhealthy for Sensitive Groups | Members of sensitive groups may experience health effects. The general public is not likely to be affected. | Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion. |
| | | Everyone may begin to experience health effects. | Active children and adults, and people with respiratory disease, such as asthma, should avoid prolonged outdoor exertion. |

[HOME](#) | [HERE](#) | [MAP](#) | [MASK](#) | [FAQ](#) | [SEARCH](#) | [CONTACT](#) | [LINKS](#)

Example Code ระบุเงื่อนไขตาม AQI

```
String getAqiScale(int aqi) {  
    if (aqi <= 50) {  
        return "Good";  
    } else if (aqi <= 100) {  
        return "Moderate";  
    } else if (aqi <= 150) {  
        return "Unhealthy for Sensitive Groups";  
    } else if (aqi <= 200) {  
        return "Unhealthy";  
    } else if (aqi <= 300) {  
        return "Very Unhealthy";  
    } else {  
        return "Hazardous";  
    }  
}
```