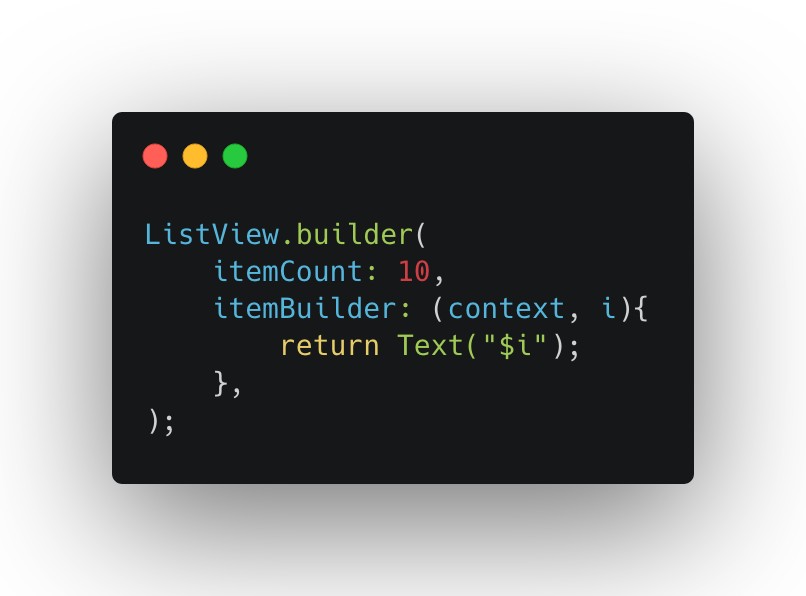
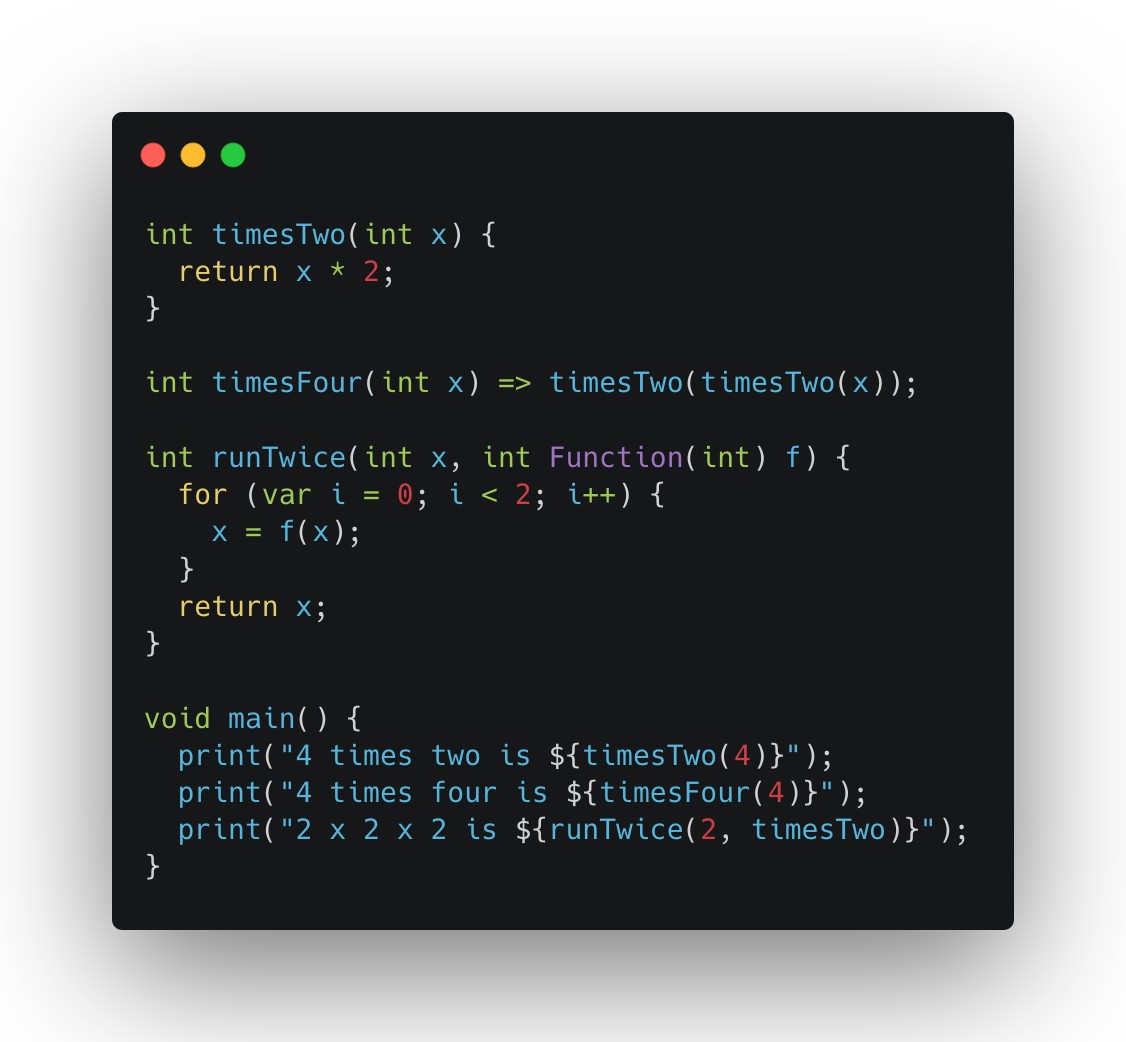
UAS

MOBILE PROGRAMMING

Ade Putra Prima Suhendri, S.Kom, M.Kom SOAL

1.Jelaskan apa yang dimaksud dengan Mobile Programming? Point 5 2.Jelaskan apa yang dimaksud dengan User Interface (UI)? Point 5 3.Jelaskan apa yang dimaksud dengan API? jelaskan fungsinya! Point 5 4.Jelaskan perbedaan Native dan Hybrid pada mobile programming? Point 5

1. Jelaskan apa fungsi github! Point 5
2. Apa output dari script berikut ! Point 10:
3. Apa output dari script berikut ! Point 10:
4. Tuliskan sintak cara parsing JSON pada flutter ! Poin 55

Jawaban

1. ***Mobile Programming adalah pemrograman yang digunakan untuk perangkat mobile .***
2. User Interface adalah tampilan visual sebuah produk yang menjembatani sistem dengan pengguna (user).
3. *Application Programming Interface* atau API adalah sebuah antarmuka yang digunakan untuk menghubungkan antara satu aplikasi dengan aplikasi yang lain. Peran dari API adalah untuk sebagai perantara yang menghubungkan aplikasi berbeda, baik dari platform yang sama maupun lintas platform

*Fungsi API*

**Fungsi API**

Setelah mengetahui jenis dari API, berikutnya kita akan masuk pada pembahasan mengenai fungsi API. Di dalam proses penerapannya, API memiliki berbagai kemudahan dan keuntungan yang dapat dirasakan oleh setiap Developer. Berikut merupakan beberapa fungsi utama dari penggunaannya.

**1. Membantu beban kerja dari server**

Fungsi pertama dari API sendiri adalah untuk membantu tugas dari server. Dimana, dengan menggunakan sebuah antarmuka khusus ini, maka server tidak perlu mencari dan menyimpan semua data. Cukup dengan memanggil atau meminta API untuk mendapatkan data dari server asal.

### 2. Mengembangkan aplikasi lebih cepat dan efektif

API memberikan kemudahan dan manfaat dari sisi pengembangan aplikasi. Anda tidak perlu melakukan menghubungkan dua aplikasi untuk melakukan komunikasi.

### 3. Menciptakan aplikasi yang bersifat fungsional

Manfaat lain dari penggunaan API ini adalah menciptakan aplikasi yang lebih fungsional dan memiliki struktur yang kompleks.

4 Aplikasi **native** adalah aplikasi yang dibangun dengan bahasa pemrograman yang spesifik untuk platform tertentu. Contoh populernya yakni penggunaan bahasa pemrograman Objective-C atau Swift untuk platform iOS (Apple). Adapun platform Android yang menggunakan bahasa pemrograman Java.

Aplikasi hybrid adalah aplikasi web yang ditransformasikan menjadi kode native pada platform seperti iOS atau [**Android**](https://www.codepolitan.com/tag/android/). Aplikasi hybrid biasanya menggunakan browser untuk mengijinkan aplikasi web mengakses berbagai fitur di device mobile seperti Push Notification, Contacts, atau Offline Data Storage. Beberapa tools untuk mengembangkan aplikasi hybrid antara lain Phonegap, Rubymotion dan lain-lain.

5 GitHub adalah manajemen proyek dan sistem *versioning code* sekaligus platform jaringan sosial yang dirancang khusus bagi para developer. Lalu, apa saja fungsi GitHub? Dengan platform ini, Anda bisa bekerja bersama-sama dengan rekan dari berbagai berlahan dunia, merencanakan proyek, dan bahkan *tracking* (melacak) pekerjaan Anda.

9 Jika saya memiliki string JSON sederhana seperti ini:

{

"name" : "Mary",

"age" : 30

}

# Mendirikan

Impor convertpustaka bawaan , yang menyertakan metode jsonDecode()dan jsonEncode():

import 'dart:convert';

class Person {

Person(this.name, this.age);

final String name;

final int age;

}

Melakukannya dengan cara ini sangat ringkas dan semuanya dibangun dengan dart:convertperpustakaan. Kerugiannya adalah kami tidak memiliki keamanan tipe yang banyak, dan mudah untuk membuat kesalahan ketik saat menulis nama kunci peta.

## JSON → objek

String rawJson = '{"name":"Mary","age":30}';

Map<String, dynamic> map = jsonDecode(rawJson);

String name = map['name'];

int age = map['age'];

Person person = Person(name, age);

final map = jsonDecode(rawJson) as Map<String, dynamic>;

Kami akan menggunakan jsonEncode(), tetapi hanya menerima angka, boolean, string, null, list atau peta dengan kunci string. Jadi kita harus mengubah objek kita menjadi peta terlebih dahulu.

Person person = Person('Mary', 30);

Map<String, dynamic> map = {

'name': person.name,

'age': person.age

};

String rawJson = jsonEncode(map);

Kami akan memperbarui kelas model kami untuk memasukkan logika konversi Peta:

class Person {

Person(this.name, this.age);

final String name;

final int age;

// named constructor

Person.fromJson(Map<String, dynamic> json)

: name = json['name'],

age = json['age'];

// method

Map<String, dynamic> toJson() {

return {

'name': name,

'age': age,

};

}

}

## JSON → objek

String rawJson = '{"name":"Mary","age":30}';

Map<String, dynamic> map = jsonDecode(rawJson);

Person person = Person.fromJson(map);

Person person = Person('Mary', 30);

Map<String, dynamic> map = person.toJson();

String rawJson = jsonEncode(map);

Tim Dart memiliki paket bernama [json\_serializable](https://translate.google.com/website?sl=auto&tl=id&nv=1&u=https://pub.dartlang.org/packages/json_serializable)yang berfungsi di Flutter dan di server Dart. Ini pada dasarnya sama dengan Metode 2 di atas, tetapi ini menghasilkan kode fromJsondan toJsonuntuk Anda. Ini membantu Anda mengurangi kesalahan yang disebabkan oleh kesalahan ketik yang konyol.

Tambahkan dependensi ke pubspec.yaml:

dependencies:

json\_annotation: ^2.0.0

dev\_dependencies:

build\_runner: ^1.0.0

json\_serializable: ^2.0.0

import 'package:json\_annotation/json\_annotation.dart';

part 'person.g.dart';

@JsonSerializable()

class Person {

Person(this.name, this.age);

final String name;

final int age;

factory Person.fromJson(Map<String, dynamic> json) =>

\_$PersonFromJson(json);

Map<String, dynamic> toJson() => \_$PersonToJson(this);

}

flutter packages pub run build\_runner build

pub run build\_runner build

## JSON → objek

String rawJson = '{"name":"Mary","age":30}';

Map<String, dynamic> map = jsonDecode(rawJson);

Person person = Person.fromJson(map);

Person person = Person('Mary', 30);

Map<String, dynamic> map = person.toJson();

String rawJson = jsonEncode(map);

Konversi JSON hanya berfungsi dengan peta, daftar, string, angka, boolean, atau null. Bagaimana jika objek Anda berisi tipe lain seperti DateTime?

## Objek → JSON

Mari kita mulai dengan objek berikut:

class Person {

Person(this.name, this.birthdate);

String name;

DateTime birthdate;

}

final person = Person('Bob', DateTime(2020, 2, 25));

Map<String, dynamic> map = {

'name': person.name,

'birthdate': person.birthdate,

};

**Solusi 1**

Anda bisa membuat serial sendiri terlebih dahulu seperti ini:

Map<String, dynamic> map = {

'name': person.name,

'birthdate': person.birthdate.toIso8601String(),

};

final jsonString = json.encode(map);

Inilah perbedaan antara toStringdan toIso8601String:

2020-02-25 14:44:28.534 // toString()

2020-02-25T14:44:28.534 // toIso8601String()

**Solusi 2**

Anda dapat menggunakan toEncodableparameter fungsi opsional di jsonEncode.

import 'dart:convert';

void main() {

final person = Person('Bob', DateTime(2020, 2, 25));

Map<String, dynamic> map = {

'name': person.name,

'birthdate': person.birthdate,

};

final toJson = json.encode(map, toEncodable: myDateSerializer);

}

class Person {

Person(this.name, this.birthdate);

String name;

DateTime birthdate;

}

dynamic myDateSerializer(dynamic object) {

if (object is DateTime) {

return object.toIso8601String();

}

return object;

}

## JSON → objek

Tidak ada yang spesial disini. Anda hanya perlu mengurai string menjadi jenis yang Anda butuhkan. Dalam kasus DateTime, Anda dapat menggunakan metode parseatau tryParse.

final myMap= json.decode(jsonString);

final name = myMap['name'];

final birthdateString = myMap['birthdate'];

final birthdate = DateTime.parse(birthdateString);

final decodedPerson = Person(name, birthdate);

## Contoh kelas model

Dan inilah logika di atas dalam kelas model:

class Person {

Person(this.name, this.birthdate);

String name;

DateTime birthdate;

Person.fromJson(Map<String, dynamic> json)

: name = json['name'],

birthdate = DateTime.tryParse(json['birthdate']),

Map<String, dynamic> toJson() {

return {

'name': name,

'birthdate': birthdate.toIso8601String(),

};

}

}

Link Github Ari Yulianto

https://github.com/Akungithub1665