

UAS  
MOBILE PROGRAMMING  
Ade Putra Prima Suhendri, S.Kom, M.Kom

Nama : Danil Tanjung  
NIM : 181011401719  
Kelas : 06TPLE022  
Mata Kuliah : Mobile Programming

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
6. Jelaskan apa fungsi github! Point 5
7. Apa output dari script berikut ! Point 10:



```
ListView.builder(  
    itemCount: 10,  
    itemBuilder: (context, i){  
        return Text("$i");  
    },  
);
```

8. Apa output dari script berikut ! Point 10:

```
int timesTwo(int x) {  
    return x * 2;  
}  
  
int timesFour(int x) => timesTwo(timesTwo(x));  
  
int runTwice(int x, int Function(int) f) {  
    for (var i = 0; i < 2; i++) {  
        x = f(x);  
    }  
    return x;  
}  
  
void main() {  
    print("4 times two is ${timesTwo(4)}");  
    print("4 times four is ${timesFour(4)}");  
    print("2 x 2 x 2 is ${runTwice(2, timesTwo)}");  
}
```

9. Tuliskan sintak cara parsing JSON pada flutter ! Poin 55

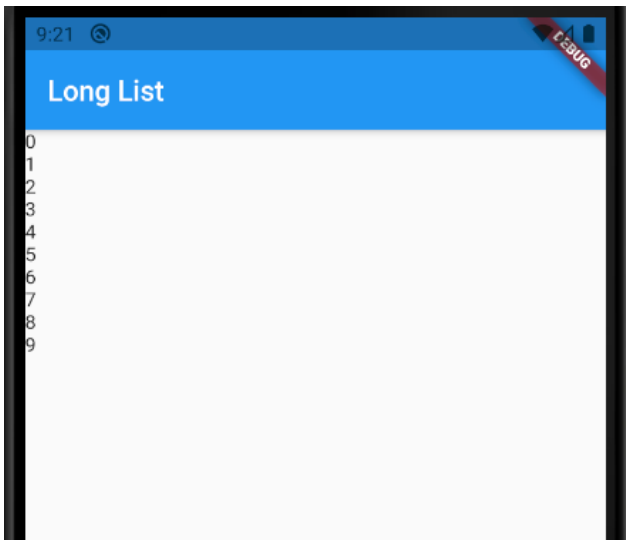
# UJIAN AKHIR SEMESTER

Nama : Danil Tanjung  
NIM : 181011401719  
Kelas : 06TPLE022  
Mata Kuliah : Mobile Programming

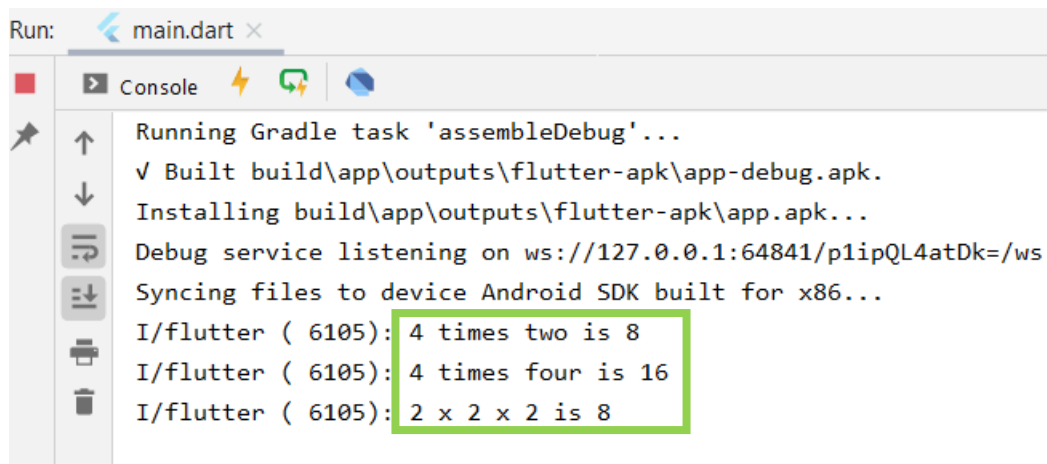
Jawab :

1. Mobile programming adalah proses pembuatan program / aplikasi yang diperuntukkan bagi device / perangkat yang bersifat mobile seperti smartphone dan juga tablet. Mobile programming juga memadupadankan antara aplikasi desktop dan aplikasi web, yang berarti dapat mengonlinekan aplikasi desktop pada perangkat mobile.
2. User Interface adalah tampilan antarmuka yang divisualisasikan dari sebuah device ke penggunaanya secara langsung.
3. API atau Application Programming Interface merupakan interface yang menjadi penghubung antara aplikasi satu dengan yang lainnya baik dalam platform sejenis maupun lintas platform.
4. Perbedaannya, Aplikasi native dibangun dengan Bahasa tertentu dan untuk platform tertentu saja, misalnya Bahasa pemrograman java untuk platform android saja , dan Bahasa pemrograman objective C atau swift untuk platform apple saja. Sedangkan Aplikasi Hybrid dibangun untuk lintas platform (bisa digunakan di android, bisa juga di apple) dengan satu Bahasa pemrograman yaitu Dart.
5. **Tidak ada soal pada no. 5, lanjut ke no. 6**
6. Beberapa fungsi github yang saya ketahui :
  - 1) Bisa digunakan untuk menyimpan repository dari project yang telah dibuat.
  - 2) Sebagai sosial medianya para programmer untuk kolaborasi dan mengerjakan project.
  - 3) Sebagai wadah penyimpanan project project yang telah dikerjakan, jika projectnya ingin dibagikan kita bisa membuat repositorynya diakses public, namun jika dirasa projectnya bersifat rahasia kita bisa membuat repositorynya menjadi private, hanya kita yang bisa mengakses.
  - 4) github mencatat semua perubahan yang dilakukan pada file sehingga riwayat perubahan akan terekam untuk dapat ditinjau kembali.

7. Outputnya merupakan list angka dari 0 sampai dengan 9 seperti yang telah saya coba pada gambar berikut :



8. Setelah saya coba, outputnya dapat dilihat pada kotak hijau yang saya tandai berikut ini



## 9. Sintak cara parsing JSON

```
void _onTap() async {
  var client = http.Client();
  try {
    var param = {"q": _lokasi, "APPID": "db789012c45a4e8556e63e24478c274c"};
    var uri = Uri.https('api.openweathermap.org', '/data/2.5/weather', param);
    var uriResponse = await client.get(uri); //membuat Network Request
    setState(() {
      _weather =
        WeatherModel.fromJson(json.decode(uriResponse.body.toString())); //Convert
Json
    });
  } finally {
    client.close();
  }
}
```

---

*//membuat Class WeatherModel*

```
class WeatherModel {
  Coord coord;
  List<Weather> weather;
  String base;
  Main main;
  int visibility;
  Wind wind;
  Clouds clouds;
  int dt;
  Sys sys;
  int timezone;
  int id;
  String name;
  int cod;

  WeatherModel(
    {this.coord,
     this.weather,
     this.base,
     this.main,
     this.visibility,
     this.wind,
     this.clouds,
     this.dt,
     this.sys,
     this.timezone,
     this.id,
     this.name,
     this.cod});

  WeatherModel.fromJson(Map<String, dynamic> json) {
    coord = json['coord'] != null ? new Coord.fromJson(json['coord']) : null;
```

```

    if (json['weather'] != null) {
        weather = new List<Weather>();
        json['weather'].forEach((v) {
            weather.add(new Weather.fromJson(v));
        });
    }
    base = json['base'];
    main = json['main'] != null ? new Main.fromJson(json['main']) : null;
    visibility = json['visibility'];
    wind = json['wind'] != null ? new Wind.fromJson(json['wind']) : null;
    clouds =
    json['clouds'] != null ? new Clouds.fromJson(json['clouds']) : null;
    dt = json['dt'];
    sys = json['sys'] != null ? new Sys.fromJson(json['sys']) : null;
    timezone = json['timezone'];
    id = json['id'];
    name = json['name'];
    cod = json['cod'];
}

Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    if (this.coord != null) {
        data['coord'] = this.coord.toJson();
    }
    if (this.weather != null) {
        data['weather'] = this.weather.map((v) => v.toJson()).toList();
    }
    data['base'] = this.base;
    if (this.main != null) {
        data['main'] = this.main.toJson();
    }
    data['visibility'] = this.visibility;
    if (this.wind != null) {
        data['wind'] = this.wind.toJson();
    }
    if (this.clouds != null) {
        data['clouds'] = this.clouds.toJson();
    }
    data['dt'] = this.dt;
    if (this.sys != null) {
        data['sys'] = this.sys.toJson();
    }
    data['timezone'] = this.timezone;
    data['id'] = this.id;
    data['name'] = this.name;
    data['cod'] = this.cod;
    return data;
}
}

class Coord {
    double lon;
    double lat;
}

```

```

Coord({this.lon, this.lat});

Coord.fromJson(Map<String, dynamic> json) {
  lon = json['lon'];
  lat = json['lat'];
}

Map<String, dynamic> toJson() {
  final Map<String, dynamic> data = new Map<String, dynamic>();
  data['lon'] = this.lon;
  data['lat'] = this.lat;
  return data;
}
}

```

---

//Buat class untuk JSON Object

```

class Weather {
  int id;
  String main;
  String description;
  String icon;

  Weather({this.id, this.main, this.description, this.icon});

  Weather.fromJson(Map<String, dynamic> json) {
    id = json['id'];
    main = json['main'];
    description = json['description'];
    icon = json['icon'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['id'] = this.id;
    data['main'] = this.main;
    data['description'] = this.description;
    data['icon'] = this.icon;
    return data;
  }
}

class Main {
  double temp;
  double feelsLike;
  double tempMin;
  double tempMax;
  int pressure;
  int humidity;

  Main(

```

```

        {this.temp,
         this.feelsLike,
         this.tempMin,
         this.tempMax,
         this.pressure,
         this.humidity});

Main.fromJson(Map<String, dynamic> json) {
    temp = json['temp'];
    feelsLike = json['feels_like'];
    tempMin = json['temp_min'];
    tempMax = json['temp_max'];
    pressure = json['pressure'];
    humidity = json['humidity'];
}

Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['temp'] = this.temp;
    data['feels_like'] = this.feelsLike;
    data['temp_min'] = this.tempMin;
    data['temp_max'] = this.tempMax;
    data['pressure'] = this.pressure;
    data['humidity'] = this.humidity;
    return data;
}
}

class Wind {
    double speed;
    int deg;

    Wind({this.speed, this.deg});

    Wind.fromJson(Map<String, dynamic> json) {
        speed = json['speed'];
        deg = json['deg'];
    }

    Map<String, dynamic> toJson() {
        final Map<String, dynamic> data = new Map<String, dynamic>();
        data['speed'] = this.speed;
        data['deg'] = this.deg;
        return data;
    }
}

class Clouds {
    int all;

    Clouds({this.all});

    Clouds.fromJson(Map<String, dynamic> json) {
        all = json['all'];
    }
}

```



```

Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['all'] = this.all;
    return data;
}
}

class Sys {
    int type;
    int id;
    String country;
    int sunrise;
    int sunset;

    Sys({this.type, this.id, this.country, this.sunrise, this.sunset});

    Sys.fromJson(Map<String, dynamic> json) {
        type = json['type'];
        id = json['id'];
        country = json['country'];
        sunrise = json['sunrise'];
        sunset = json['sunset'];
    }

    Map<String, dynamic> toJson() {
        final Map<String, dynamic> data = new Map<String, dynamic>();
        data['type'] = this.type;
        data['id'] = this.id;
        data['country'] = this.country;
        data['sunrise'] = this.sunrise;
        data['sunset'] = this.sunset;
        return data;
    }
}

```



**UNIVERSITAS PAMULANG**  
**KARTU UJIAN AKHIR SEMESTER GENAP 2020/2021**  
**NOMOR UJIAN : 422307039342**

FAK/PROG : TEKNIK / TEKNIK INFORMATIKA  
NAMA : DANIL TANJUNG  
NIM : 181011401719  
SHIFT : REGULER C

NO	HARI / TANGGAL	WAKTU	RUANG	KELAS	MATA KULIAH	PARAF
1	-			06TPLE022	SISTEM INFORMASI MANAJEMEN	
2	-			06TPLE022	KERJA PRAKTEK	
3	-			06TPLE022	MOBILE PROGRAMMING	

**Peraturan dan Tata Tertib Peserta Ujian**

1. Peserta ujian harus berpakaian rapi, sopan dan memakai jaket Almamater
2. Peserta ujian sudah berada di ruangan sepuluh menit sebelum ujian dimulai
3. Peserta ujian yang terlambat diperkenankan mengikuti ujian setelah mendapat ijin, tanpa perpanjangan waktu
4. Peserta ujian hanya diperkenankan membawa alat-alat yang ditentukan oleh panitia ujian
5. Peserta ujian dilarang membantu teman, mencontoh dari teman dan tindakan-tindakan lainnya yang mengganggu peserta ujian lain
6. Peserta ujian yang melanggar tata tertib ujian dikenakan sanksi akademik



Pamulang, 06 April 2021  
Ketua Panitia Ujian

**Dr. E. NURZAMAN AM, M.M, M. Si**  
**NIDK. 8811520016**