

Nama : Nahrowi
Nim : 181011401652
Kelas : 06TPLE017

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

Kartu UAS :



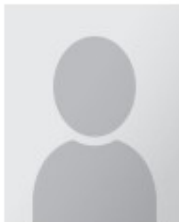
UNIVERSITAS PAMULANG
KARTU UJIAN AKHIR SEMESTER GENAP 2020/2021
NOMOR UJIAN : 293507716755

FAK/PROG : TEKNIK / TEKNIK INFORMATIKA
NAMA : NAHROWI
NIM : 181011401652
SHIFT : REGULER C

| NO | HARI / TANGGAL | WAKTU | RUANG | KELAS | MATA KULIAH | PARAF |
|----|----------------|-------|-------|-----------|----------------------------|-------|
| 1 | - | | | 06TPLE017 | SISTEM INFORMASI MANAJEMEN | |
| 2 | - | | | 06TPLE017 | KERJA PRAKTEK | |
| 3 | - | | | 06TPLE017 | 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

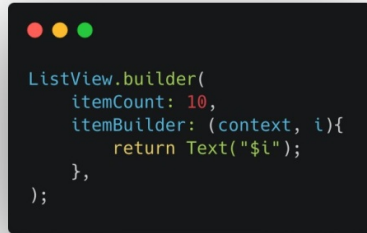


Pamulang, 07 April 2021
Ketua Panitia Ujian

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

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

Jawaban :

1. Mobile Programming adalah pemrograman yang digunakan untuk perangkat mobile. Adapun beberapa bahasa yang digunakan untuk pemrograman perangkat mobile di antaranya:

- J2ME
- C++ dalam symbian framework
- Flash Lite

- Objective C (mirip C, tapi struktur bahasa program), ini untuk iPhone dan hanya bisa di-develop via OS Macintosh
- C++ dalam BREW framework (untuk HP CDMA)
- C# .NET (untuk HP dgn OS Windows Mobile)
- Javafx mobile (masih sedikit yang support, kemungkinan saat hanya di HP Android)
- PHP

Software yang diperlukan:

- Paket Apache + MySQL + PHP: xampp, appserver, phptriad, wamp, dan lain-lain
- Web Browser (IE, Mozilla Firefox, atau yang lainnya): Untuk mengecek server dari laptop/komputer.
- Emulator WAP: Browser berbentuk handphone yang digunakan untuk mencoba program wap yang kita buat.
- Editor: Editplus, Notepad++, Macromedia Dreamweaver atau yang lainnya.

2. User interface (antar muka pengguna) merupakan bentuk tampilan yang berfungsi untuk menghubungkan antara pengguna (user) dengan aplikasi atau sistem operasi, sehingga pengguna (user) dapat mengerti dan berkomunikasi. User interface dapat berupa teks maupun grafis. Untuk aplikasi yang menekankan pada kecepatan proses dan ditujukan untuk pengguna (user) dengan pengetahuan lebih baik, biasanya dibuat bentuk teks. Sedangkan untuk aplikasi yang ditujukan untuk pengguna akhir (end user) dan lebih menekankan user friendly, maka digunakan bentuk grafis.

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 :

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. Dengan kondisi tersebut, server yang anda gunakan tidak akan terbebani tugas terlalu berat.

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. Cukup dengan menggunakan bantuan API, maka komunikasi dapat terjalin dengan baik.

Kemudian, proses integrasi dan penambahan beberapa fitur pada aplikasi akan menjadi lebih cepat. Anda tidak perlu lagi untuk mengupdate beberapa fitur secara manual. Dengan bantuan API, permasalahan tersebut dapat teratasi dengan cepat dan tepat.

3. Menciptakan aplikasi yang bersifat fungsional

Manfaat lain dari penggunaan API ini adalah menciptakan aplikasi yang lebih fungsional dan memiliki struktur yang kompleks. Maksudnya, dalam menambahkan informasi tidak perlu melakukan input secara manual. Cukup dengan menggunakan bantuan API dapat menampilkan fitur yang sama dengan aplikasi tujuan.

Sebagai contoh penerapannya pada platform layanan transportasi seperti Gojek dan Grab. Kedua platform tersebut tidak perlu untuk membuat fitur untuk menampilkan peta pada aplikasi. Cukup dengan mengintegrasikan dengan Google Maps API, seluruh data terkait pemetaan wilayah dapat terakses secara otomatis melalui platform tersebut.

4. Native

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.

Membangun aplikasi native harus menyediakan pengalaman produk yang optimal pada perangkat mobile. Meskipun begitu, budget yang tinggi dibutuhkan untuk membangun aplikasi cross platform yang mampu mempertahankan aplikasi native tetap update.

Hybrid

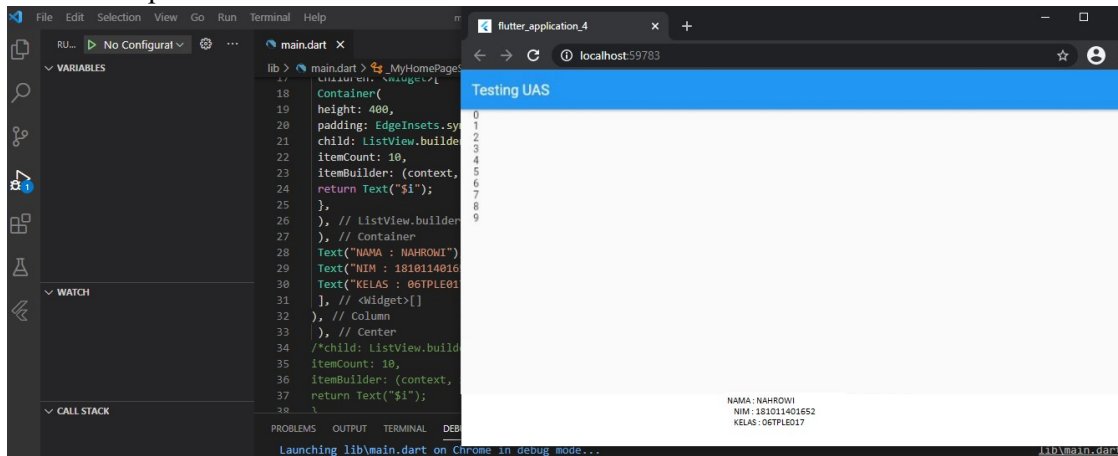
Aplikasi hybrid adalah aplikasi web yang ditransformasikan menjadi kode native pada platform seperti iOS atau Android. Aplikasi hybrid biasanya menggunakan browser untuk memungkinkan aplikasi web mengakses berbagai fitur di device mobile seperti Push Notification, Contacts, atau Offline Data Storage. Beberapa tools untuk mengembangkan aplikasi hybrid antara lain Phonegap, Rubymotion dan lain-lain.

Keuntungan membangun aplikasi hybrid diantaranya pemeliharaan project menjadi semakin mudah jika dibandingkan dengan aplikasi native. Aplikasi hybrid juga, bisa dibangun secara cepat untuk keperluan cross platform dan dana yang bisa menjadi lebih hemat jika dibandingkan dengan native.

6. Github berfungsi sebagai tempat penyimpanan source code dan alat kolaborasi sebuah project. Dan Fungsi lainnya yang bisa dilakukan oleh Github adalah sebagai berikut:

1. Bisa mengikuti programmer lain sehingga mengetahui apa saja yang dilakukannya.
2. Ada bagian star yang fungsinya sama dengan bookmark.
3. Github juga memiliki fungsi Watch, yakni untuk mengawasi repositori tertentu sehingga ketika terjadi perubahan Anda akan memperoleh pemberitahuan.
4. Selain fungsi di atas, Github juga memiliki fungsi Fork yang kerjasanya sama dengan copy-paste. Hal ini memudahkan ketika Anda membutuhkan source code programmer lain sehingga tidak perlu susah mencarinya.

7. Hasil output :



Script :

```
import 'package:flutter/material.dart';
void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Soal UAS No.6',
```

```

home: Scaffold(
  appBar: AppBar(
    title: Text('Soal UAS No.6'),
    centerTitle: false,
  ),
  body: Container(
    child: Center(
      child: Column(
        children: <Widget>[
          Container(
            height: 400,
            padding: EdgeInsets.symmetric(horizontal: 18),
            child: ListView.builder(
              itemCount: 10,
              itemBuilder: (context, i) {
                return Text("$i");
              },
            ),
          ),
          Text("NAMA : NAHROWI"),
          Text("NIM : 181011401652"),
          Text("KELAS : 06TPLE017"),
        ],
      ),
    ),
  /*child: ListView.builder(
    itemCount: 10,
    itemBuilder: (context, i) {
      return Text("$i");
    },
  ),*/
),
);
}

class MyHomePage extends StatefulWidget {
  MyHomePage({Key key, this.title}) : super(key: key);

  // This widget is the home page of your application. It is stateful, meaning
  // that it has a State object (defined below) that contains fields that affect
  // how it looks.
  // This class is the configuration for the state. It holds the values (in this
  // case the title) provided by the parent (in this case the App widget) and
  // used by the build method of the State. Fields in a Widget subclass are
  // always marked "final".

  final String title;

  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  int _counter = 0;

  void _incrementCounter() {

```

```

    setState() {
// This call to setState tells the Flutter framework that something has
// changed in this State, which causes it to rerun the build method below
// so that the display can reflect the updated values. If we changed
// _counter without calling setState(), then the build method would not be
// called again, and so nothing would appear to happen.
    _counter++;
    });
}

@override
Widget build(BuildContext context) {
// This method is rerun every time setState is called, for instance as done
// by the _incrementCounter method above.
//
// The Flutter framework has been optimized to make rerunning build methods
// fast, so that you can just rebuild anything that needs updating rather
// than having to individually change instances of widgets.
    return Scaffold(
      appBar: AppBar(
// Here we take the value from the MyHomePage object that was created by
// the App.build method, and use it to set our appBar title.
        title: Text(widget.title),
      ),
      body: Center(
// Center is a layout widget. It takes a single child and positions it
// in the middle of the parent.
        child: Column(
// Column is also a layout widget. It takes a list of children and
// arranges them vertically. By default, it sizes itself to fit its
// children horizontally, and tries to be as tall as its parent.
//
// Invoke "debug painting" (press "p" in the console, choose the
// "Toggle Debug Paint" action from the Flutter Inspector in Android
// Studio, or the "Toggle Debug Paint" command in Visual Studio Code)
// to see the wireframe for each widget.
//
// Column has various properties to control how it sizes itself and
// how it positions its children. Here we use mainAxisAlignment to
// center the children vertically; the main axis here is the vertical
// axis because Columns are vertical (the cross axis would be
// horizontal).
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              'You have pushed the button this many times:',
            ),
            Text(
              '$_counter',
              style: Theme.of(context).textTheme.headline4,
            ),
          ],
        ),
        floatingActionButton: FloatingActionButton(
          onPressed: _incrementCounter,
          tooltip: 'Increment',
          child: Icon(Icons.add),

```

```

), // This trailing comma makes autoformatting nicer for build methods.
);
}
}
}

```

8. Output :

```

lib > main.dart > runTwice
1 //@AUTHOR : NAHROWI
2 // @NIM : 181011401652
3 int timesTwo(int x) {
4   return x * 2;
5 }
6 int timesFour(int x) => timesTwo(timesTwo(x));
7 int runTwice(int x, int Function(int) f) {
8   for (var i = 0; i < 2; i++) {
9     x = f(x);
10  }
11  return x;
12 }
13
Run | Debug
14 void main () {
15   print("4 times two is ${timesTwo(4)}");
16   print("4 times four is ${timesFour(4)}");
17   print("2 x 2 x 2 is ${runTwice(2, timesTwo)}");
18 }
19
PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE
Launching lib/main.dart on Edge in debug mode...
Debug service listening on ws://127.0.0.1:61504/QNK_D-Iq1zc-/ws
Running with unsound null safety
For more information see https://dart.dev/null-safety/unsound-null-safety
Connecting to VM Service at ws://127.0.0.1:61504/QNK_D-Iq1zc-/ws
4 times two is 8
4 times four is 16
2 x 2 x 2 is 8
lib/main.dart:1

```

9.

```

lib > main.dart > ...
1 //@AUTHOR : NAHROWI
2 // @NIM : 181011401652
3 import 'dart:convert';
4 List decodedList = jsonDecode("[\"Nahrowi\", \"181011401652\", \"06TPLE017\"]");
5
Run | Debug
6 void main() {
7   print("NAMA = ${decodedList[0]}");
8   print("NIM = ${decodedList[1]}");
9   print("KELAS = ${decodedList[2]}");
10 }
11
12
PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE
Launching lib/main.dart on Edge in debug mode...
Debug service listening on ws://127.0.0.1:62160/HfSXzbVYHqs-/ws
Running with unsound null safety
For more information see https://dart.dev/null-safety/unsound-null-safety
Connecting to VM Service at ws://127.0.0.1:62160/HfSXzbVYHqs-/ws
NAMA = Nahrowi
NIM = 181011401652
KELAS = 06TPLE017
lib/main.dart:1

```