**Materi Pembelajaran iCLOP**

1. Pengantar Pemrograman Android dengan Flutter

Data StatCounter

2021-2022

Perkembangan Smartphone telah mencapai kemajuan yang sangat besar, jumlah pengguna smartphone seiring berjalan waktu terus meningkat baik dari anak-anak bahkan sampai orang tua, karena saat menggunakan smartphone kegiatan yang dilakukan manusia lebih mudah dan cepat, Smartphone mempunyai berbagai macam sistem operasi untuk mengoperasikan aplikasi ataupun perangkat lunak sehingga aplikasi bisa berjalan dengan lancar, Menurut data yang didapatkan dari (Stat Counter , 2022) bahwa pengguna smartphone pada tahun 2019 - 2022 lebih dominan menggunakan sistem operasi dari android dengan persentase seperti pada gambar di atas.. Data tersebut membuktikan bahwa penggunaan sistem operasi android dan iOS sangat dibutuhkan saat ini dalam pembuatan serta pengembangan aplikasi mobile.

Android dikembangkan menggunakan bahasa Kotlin & Java yang umumnya digunakan dalam pembuatan aplikasi pada smartphone android, bedanya jika pengembangan aplikasi mobile android dikembangkan dengan bahasa (Java & Kotlin) maka jika pengembang ingin mengembangkan lagi ke sistem operasi iOS maka developer harus mengulang code menggunakan bahasa (Swift & Objective\_c), Flutter memudahkan pengembang aplikasi mobile dalam mengembangkan aplikasi multiplatform menggunakan satu bahasa (Dart). Flutter menyediakan fitur **hot\_reload** yang berfungsi jika kode berubah, tampilan output juga akan otomatis berubah secara realtime.

Android Studio diumumkan di Google I/O Conference pada tahun 2013 dan dirilis ke publik pada tahun 2014. Sebelum lahirnya Android Studio, aplikasi pada Android dikembangkan dengann Eclipse IDE yaitu IDE Java. Setelah adanya android studio yang open source dapat memudahkan bagi Anda yang ingin membuat aplikasi dengan Android Studio.

Android dapat menyediakan interface untuk Anda dalam membuat aplikasi serta mengelola manajemen filen aplikasi anda.  Untuk bahasa programman anda gunakan adalah Java. Dalam Android Studio, anda hanya tinggal menulis, mengedit, menyimpan  dan testing project beserta dan file lainnya yang ada dalam project itu hanya dengan android studio, Tidak hanya itu, keunggulan menggunakan Android Studio juga memberi Anda akses ke Android Software Development Kit (SDK). SDK adalah sebuah ekstensi dari kode Java yang memperbolehkannya untuk berjalan dengan mulus di device Android. Untuk, Java nya dibutuhkan untuk menulis program, Android SDK sangat diperlukan untuk menjalankan programnya di Android. Maka dari itu dengan menggabungkan keduanya, Anda memerlukan Android Studio. Sehingga ketika Anda menemukan bug pada aplikasi Anda, Anda bisa mengetahui bug tersebut dengan menggunakan Android Studio untuk memperbaikinya. Berikut ini adalah beberapa fitur Android Studio:

1. Environment yang mempermudah Anda untuk mengembangkan aplikasi untuk Android
2. Editor layout dengan interface drag-and-drop
3. Refactoring dan perbaikan cepat khusus Android
4. Dukungan build berbasis Gradle
5. Integrasi ProGuard
6. Emulator yang cepat dan berbagai fitur didalamnya
7. Dapat terintegrasi dengan Google Cloud Messaging dan App Engine
8. Dukungan program basic C++ dan NDK

Flutter dikembangkan oleh Google yang merupakan framework open source multiplatform dengan satu basis kode pemrograman yaitu bahasa Dart. Flutter menyediakan UI dan Widgets yang mudah digunakan dalam membangun aplikasi multi-platform secara efisien karena dapat digunakan di berbagai platform seperti iOS, Android, Desktop dan Web, Kemudahan tersebut ada kerena Flutter adalah SDK yang pastinya dapat digunakan di lintas platform misalnya rendering engine, Widgets line, API, dan command-line tools, adapun kelebihan dan kekurangan flutter meliputi :

Kelebihan flutter :

1. Penulisan kode
2. Satu kode untuk multiplatform
3. Tampilan desain
4. Mudah dipelajari
5. Hot Reloads

Kekurangan flutter :

1. Ukuran resource yang besar
2. Library masing kurang lengkap
3. Komunitas yang masih berkembang
4. Problem solving yang belum kompleks
5. Metode pengujian Flutter
6. UnitTesting

Pengujian unit adalah jenis pengujian pada aplikasi di mana pengujian difokuskan pada apa yang disebut unit. Unit adalah bagian terkecil yang membangun aplikasi. Contoh unit yang paling sederhana ketika kita berbicara tentang OOP (Pemrograman Berorientasi Objek) adalah Kelas dan Fungsi. Pengujian unit akan memastikan bahwa komponen kecil dari aplikasi berjalan dengan baik dan seperti yang diharapkan,Untuk melakukan pengujian unit kami menggunakan fungsi berikut:

**Main.dart**

import 'package:flutter/material.dart';  
  
class Followers extends ChangeNotifier {  
 final List<int> \_followerItems = [];  
  
 List<int> get items => \_followerItems;  
  
 void add(int itemNo) {  
 \_followerItems.add(itemNo);  
 notifyListeners();  
 }  
  
 void remove(int itemNo) {  
 \_followerItems.remove(itemNo);  
 notifyListeners();  
 }  
}

**Testing.dart**

void main() {  
 group('Test Rorvider', () {  
 var followers = Followers();  
 test('Test\_01\_menambah item baru', () {  
 var number = 10;  
 followers.add(number);  
 expect(followers.items.contains(number), true);  
 });  
 });  
}

Penjelasan :

var followers = Followers();

Membuat variabel case followers yang digunakan untuk memanggil fungsi Followers() dari provider

var number = 10;

followers.add(number);

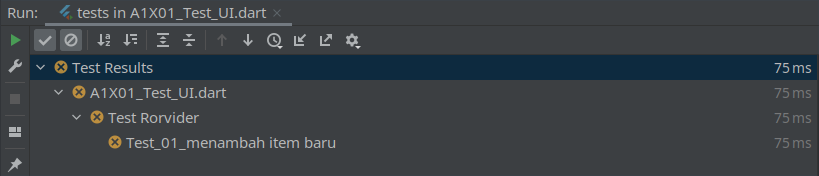
Variabel number digunakan untuk membuat spesifik angka 10 yang akan jalankan

expect(followers.items.contains(number), true);

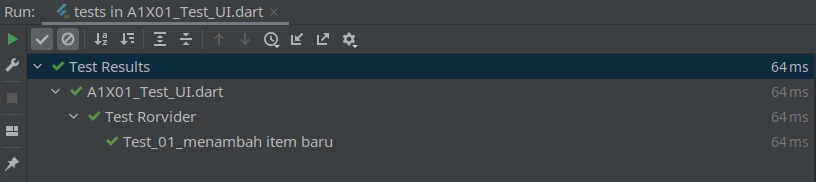
Digunakan untuk memferifikasi item dengan variabel angka 10 telah ditambahkan jika benar maka hasilnya akan true

Hasil :

Failed



Passed



1. WidgetsTesting

Pengujian Widgets atau dalam istilah lain disebut pengujian instrumentasi adalah jenis pengujian yang difokuskan pada bagian-bagian yang membangun Antarmuka Pengguna suatu aplikasi seperti tombol, bidang teks, dan sebagainya. Pengujian Widgets akan memastikan bahwa Antarmuka Pengguna ditata sesuai dengan desain UI aplikasi dan dapat berinteraksi dengan baik dengan pengguna.

Untuk melakukan pengujian Widgets kami menggunakan fungsi berikut:

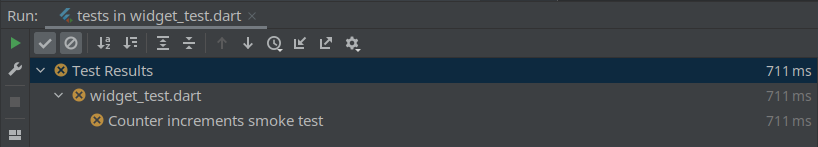
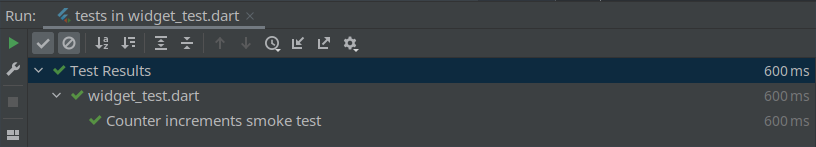
class MyHomePage extends StatefulWidgets {  
 const MyHomePage({Key? key, required this.title}) : super(key: key);  
 final String title;  
 @override  
 State<MyHomePage> createState() => \_MyHomePageState();  
}  
class \_MyHomePageState extends State<MyHomePage> {  
 int \_counter = 0;  
 int \_incounter = 0;  
 void \_incrementCounter() {  
 setState(() {  
 \_counter++;  
 });  
 }  
 void \_minCounter() {  
 setState(() {  
 \_counter--;  
 });  
 }  
 @override  
 Widgets build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text(Widgets.title),  
 ),

body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: <Widgets>[  
 const Text(  
 'Welcome To Flutter',style: TextStyle(  
 color: Colors.black, fontSize: 30  
 ),  
 ),const Text(  
 'Follow the next lesson',style: TextStyle(  
 color: Colors.black, fontSize: 40  
 )  
 ),  
 Text(  
 '$\_counter',  
 style: Theme.of(context).textTheme.headline4,  
 ),  
 ],  
 ),  
 ),  
 floatingActionButton: FloatingActionButton(  
 onPressed: \_incrementCounter,  
 tooltip: 'Increment',  
 child: const Icon(Icons.add),  
 ),  
 );  
 }  
}

void main() {  
 group('test counter', () {  
 testWidgetss('Counter increments smoke test', (WidgetsTester tester) async {  
 // Build our app and trigger a frame.  
 await tester.pumpWidgets(const MyApp());  
  
 // Verify that our counter starts at 0.  
 expect(find.text('0'), findsOneWidgets);  
 expect(find.text('1'), findsNothing);  
  
 // Tap the '+' icon and trigger a frame.  
 await tester.tap(find.byIcon(Icons.add));  
 await tester.pump();  
  
 // Verify that our counter has incremented.  
 expect(find.text('0'), findsNothing);  
 expect(find.text('1'), findsOneWidgets);  
 });  
 });

Hasil :

Failed Passed

1. Rencana Pembelajaran

|  |  |
| --- | --- |
| TextWidgets | Siswa mampu mempelajari macam-macam jenis text pada widgets seperti TextSpan, TextRich, TextStyle, dan ButtonText. Widgets digunakan untuk menampilkan text kedalam layar Text Widgets dibagi menjadi dua properties yaitu : String yang merupakan data dan nantinya akan ditampilkan kelayar, lalu properti kedua merupakan argument berbentuk Object yang dimana berfungsi untuk memanipulasi tampilan seperti warna, ukuran, posisi. |
| ContainerWidgets | Siswa mampu memahami penggunaan ContainerWidgets meliputi : Padding, dan Border yang digunakan untuk mengatur posisi dan merancang struktur tampilan pada view |
| StatefullWidgets | Siswa mampu memahami konsep StatefullWidgets dimana Pembelajaran Widgets yang dapat berubah ubah, StateFull widgets yang dapat mengupdate tampilan,merubah warna, dan menambah jumlah baris baru. |
| StatelessWidgets | Siswa mampu memahami konsep widgets statis yang tidak dapat dirubah atau tidak akan pernah berubah karena Stateless hanya digunakan untuk mengubah tampilan seperti icon,warna dan boxDecoration. |

1. Studi Kasus

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **Topik Pembelajaran** | **Studi Kasus** | **Tujuan Pembelajaran** | **Materi** |
| 1. | TextSpan | Siswa mampu memahami konsep penggunaan TextSpa n pada Aplikasi Android | 1. Memahami cara membuat jarak text pada view 2. Memahami fungsi InlineSpan pada view | A1 |
| 2. | Text.Rich | Siswa mampu memahami konsep penggunaan TextRich pada Aplikasi Android | 1. Memahami cara menampilkan rentang text pada view 2. Memahami cara membuat bentuk text pada view |
| 3. | TextStyle | Siswa mampu memahami konsep penggunaan TextStyle pada Aplikasi Android | 1. Memahami cara menampilkan rentang text pada view 2. Memahami cara membuat bentuk text pada view |
| 4. | ButtonText | Siswa mampu memahami konsep penggunaan ButtonText pada Aplikasi Android | 1. Memahami cara membuat button 2. Memahami cara mengubah styleButton |
| 5. | Image | Siswa mampu memahami konsep penggunaan Image pada Aplikasi Android | 1. Memahami cara menambahkan gambar ke view 2. Memahami penggunaan local Asset dan Network Image |
| 6. | Padding, Border | Siswa mampu memahami konsep penggunaan Padding dan Border pada Aplikasi Android | 1. Siswa memahami penggunaan Padding untuk mengatur border pada view |
|  | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7. | ListView | Siswa mampu memahami konsep penggunaan ListView pada Aplikasi Android | 1. Siswa mampu membuat listView dan memahaminya 2. Siswa memahami cara membuat scrollingView | B1 |
| 8. | Icon | Siswa mampu memahami konsep penggunaan Icons pada Aplikasi Android | 1. Memahami penggunaan icon pada view 2. Siswa mampu mahami cara membuat jenis icon seperti IconButton dan TextButton.icon pada view |
| 9. | Snackbar | Siswa mampu memahami konsep penggunaan Snackbar pada Aplikasi Android | 1. Siswa memahami cara membuat popUp notifikasi pada view |
| 10. | State | Siswa mampu memahami konsep penggunaan state management pada Aplikasi Android | 1. Siswa mampu memahami cara kerja state pada view flutter 2. Siswa mampu memahami perbedaan state management pada statefull dan stateless widgets |
| 11. | Provider | Siswa mampu memahami konsep penggunaan Provider pada Aplikasi Android | 1. Siswa mampu memahami cara mengalokasikan resource secara sederhana 2. Memahami konsep penggunaan provider pada view |
| 12. | Route | Siswa mampu memahami konsep penggunaan state management pada Aplikasi Android | 1. Siswa mampu memahami cara menghubungan page pada view |