Modul 10

Access Device Features

Managing App Permissions, Importing Contact, Integrating Device Camera

Module Overview

Membuat proyek dengan menerapkan permintaan akses terhadap pembacaan kontak dan pengaktifan kamera pada *smartphone*.

Module Objectives

Setelah mempelajari dan mempraktikkan modul ini, mahasiswa diharapkan dapat:

- Mengetahui mengenai *app permissions* dalam Flutter
- Melakukan pembacaan kontak pada *smartphone*
- Melakukan permintaan akses untuk menggunakan kamera

Dalam beberapa kasus Anda mungkin perlu mengakses fungsionalitas-fungsionalitas yang tersedia pada *smartphone* Anda untuk menyelesaikan masalah-masalah tertentu pada aplikasi Anda. Penempatan koordinat posisi pada *maps* menggunakan GPS, pembacaan kontak pada *smartphone* dan pengaktifan kamera untuk mengambil sebuah gambar merupakan beberapa contoh fungsionalitas yang disediakan pada *smartphone* Anda yang mungkin perlu Anda tambahkan ketika pengguna sedang mengakses aplikasi Anda.

Secara *default* Anda tidak dapat mengakses secara langsung fungsionalitas perangkat yang disebutkan di atas tanpa melakukan permintaan terhadap hak akses fungsionalitas tersebut. Tujuan hal ini dilakukan adalah untuk menjaga keamanan dan privasi dari masing-masing pengguna aplikasi Anda di mana pengguna harus mengetahui hal-hal apa saja yang dapat diakses oleh aplikasi Anda serta mereka sendiri jugalah yang menentukan tentang apa yang boleh atau tidak boleh diakses oleh aplikasi Anda.

Pada pertemuan ini Anda akan belajar bagaimana melakukan manajemen hak akses terhadap fungsionalitas perangkat secara khusus fungsionalitas pada pembacaan kontak dan pengaktifan kamera *smartphone*. Untuk mempermudah Anda dalam melakukan hal ini maka di sini Anda dapat menggunakan sebuah *package* **permission_handler** yang berfungsi untuk mengatur manajemen hak akses perangkat yang Anda pergunakan. Tambahkan *package* di bawah ini pada pubspec proyek milik Anda.

```
permission_handler: ^10.2.0
```

Sebelumnya Anda perlu merancang sebuah tampilan sederhana yang berisikan 2 buah tombol yang masing-masing berfungsi untuk mengakses halaman kontak dan halaman kamera. Buatlah sebuah file baru **screen.dart** dan rancang tampilan berikut ini.

```dart
import 'package:flutter/material.dart';
import 'package:m10/camera.dart';
import 'package:permission_handler/permission_handler.dart';

import 'contact.dart';

class Screen extends StatefulWidget {
  const Screen({super.key});

  @override
  State<Screen> createState() => _ScreenState();
}

class _ScreenState extends State<Screen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Screen")),
        body: Center(
          child: Column(mainAxisAlignment: MainAxisAlignment.center, children: [
            ElevatedButton(onPressed: contact, child: Text("Contact")),
            ElevatedButton(onPressed: camera, child: Text("Camera"))
          ]), // Column
        )); // Center // Scaffold
  }
}
```

Membaca Kontak

Untuk membaca kontak maka Anda perlu menambahkan *package* di bawah ini pada proyek Anda. *Package* **contacts_service** merupakan sebuah *plugin* yang dapat Anda pergunakan untuk mengakses dan mengatur kontak yang terdapat pada perangkat Anda.

```
contacts_service: ^0.6.3
```

1. Tambahkan perintah permintaan hak akses kontak di bawah ini pada **AndroidManifest.xml** Anda yang terdapat pada folder **android/app/src/main/AndroidManifest.xml**. Pastikan Anda menempatkannya di atas bagian **<application>**.

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

Permintaan hak akses di atas akan memberitahukan kepada perangkat Anda bahwa aplikasi Anda mungkin akan membutuhkan permintaan akses untuk membaca kontak (READ_CONTACTS).

2. Rancanglah sebuah tampilan baru semisal **contact.dart** yang berfungsi untuk menampilkan seluruh daftar kontak pada perangkat Anda seperti di bawah ini. Jika pada perangkat Anda terdapat minimal satu kontak maka Anda akan menampilkan *avatar* (gambar profil atau inisial kontak) serta nama kontak tersebut sedangkan jika pada perangkat Anda tidak memiliki kontak maka Anda akan menampilkan teks "Kontak Kosong".

```dart
import 'package:contacts_service/contacts_service.dart';
import 'package:flutter/material.dart';

class ContactScreen extends StatefulWidget {
  const ContactScreen({super.key});

  @override
  State<ContactScreen> createState() => _ContactScreenState();
}

class _ContactScreenState extends State<ContactScreen> {
  var _contacts;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Contact")),
      body: _contacts != null && _contacts.length != 0
          ? ListView.builder(
              itemCount: _contacts?.length ?? 0,
              itemBuilder: (BuildContext context, int index) {
                Contact? contact = _contacts?.elementAt(index);
                return ListTile(
                  contentPadding:
                      const EdgeInsets.symmetric(vertical: 2, horizontal: 18),
                  leading:
                      (contact?.avatar != null && contact!.avatar!.isNotEmpty)
                          ? CircleAvatar(
                              backgroundImage: MemoryImage(contact.avatar!),) // CircleAvatar
                          : CircleAvatar(
                              backgroundColor: Theme.of(context).primaryColor,
                              child: Text(contact!.initials()),), // CircleAvatar
                  title: Text(contact.displayName ?? ''),
                ); // ListTile
              },
            ) // ListView.builder
          : Center(child: Text('Kontak Kosong')),
    ); // Scaffold
  }
}
```

3. Tambahkan sebuah fungsi yang bekerja secara asinkron untuk mendapatkan seluruh daftar kontak dengan menggunakan *package* yang sebelumnya Anda tambahkan dengan menggunakan perintah seperti di bawah ini.

```
20  Future<void> getContacts() async {
21    List<Contact> contacts = await ContactsService.getContacts();
22    print(contacts);
23    setState(() {
24      _contacts = contacts;
25    });
26  }
```

   Fungsi di atas akan melakukan pengaksesan terhadap daftar kontak serta mengembalikan objek **Contact**. Informasi lebih lanjut mengenai objek **Contact** dapat Anda baca pada dokumentasi berikut https://pub.dev/packages/contacts_service.

4. Panggil fungsi yang sebelumnya telah Anda tambahkan pada bagian **iniState**. Hal ini ditujukan agar fungsi tersebut pertama sekali dijalankan sebelum tampilan dibangun.

```
14  @override
15  void initState() {
16    getContacts();
17    super.initState();
18  }
```

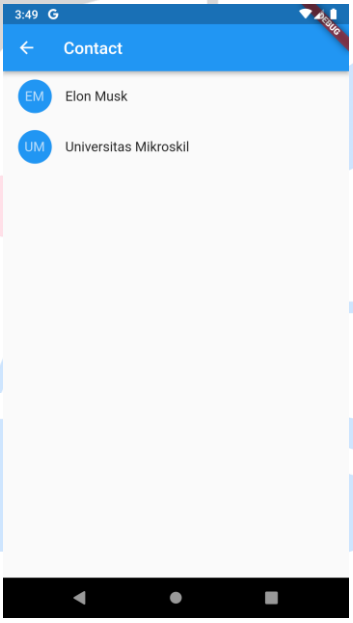
5. Pada **screen.dart** tambahkan sebuah fungsi **contact** yang berisi pengecekan akses terhadap kontak.

```
27  void contact() async {
28    if (await Permission.contacts.status.isGranted) {
29      Navigator.of(context)
30          .push(MaterialPageRoute(builder: (context) => ContactScreen()));
31    } else {
32      var status = await Permission.contacts.request();
33      print(status);
34      if (status == PermissionStatus.granted) {
35        Navigator.of(context)
36            .push(MaterialPageRoute(builder: (context) => ContactScreen()));
37      } else if (status == PermissionStatus.permanentlyDenied) {
38        openAppSettings();
39      }
40    }
41  }
```

Jika akses diberikan (*granted*) maka pengguna akan langsung diarahkan menuju ke halaman kontak sedangkan jika akses belum diberikan maka pengguna akan diminta untuk memberikan akses kontak terhadap aplikasi.

Jika pengguna mencentang "Don't ask again" (*permanentlyDenied*) maka pengguna akan diarahkan menuju pengaturan untuk memberikan akses kontak secara manual. Hal ini dikarenakan ketika pengguna menolak untuk memberikan akses selamanya maka aplikasi tidak lagi dapat meminta akses melalui tampilan/*widget*.

6. Setelah semuanya telah sesuai maka seharusnya ketika tombol kontak dipilih akan menampilkan tampilan sebagai berikut.



Mengakses Kamera

Untuk mengakses kamera maka Anda perlu menambahkan *package* di bawah ini pada proyek Anda. *Package* **camera** merupakan sebuah *plugin* yang dapat Anda

pergunakan untuk mengakses dan mengatur kontak yang terdapat pada perangkat Anda.

```
camera:
```

1. Tambahkan kembali perintah permintaan hak akses kamera di bawah ini pada **AndroidManifest.xml** Anda yang terdapat pada folder **android/app/src/main/AndroidManifest.xml**. Pastikan Anda menempatkannya di atas bagian **<application>.**

   <uses-permission android:name="android.permission.CAMERA" />

   Permintaan hak akses di atas akan memberitahukan kepada perangkat Anda bahwa aplikasi Anda mungkin akan membutuhkan permintaan akses untuk menggunakan kamera.
2. Rancanglah sebuah tampilan baru semisal **camera.dart** yang berfungsi untuk menampilkan tampilan yang ditangkap oleh kamera Anda.

```dart
import 'package:camera/camera.dart';
import 'package:flutter/material.dart';

class CameraScreen extends StatefulWidget {
  const CameraScreen({super.key});

  @override
  State<CameraScreen> createState() => _CameraScreenState();
}

class _CameraScreenState extends State<CameraScreen> {
  late CameraController _controller;
  late List<CameraDescription> cameras;
  late CameraDescription camera;
  Widget? cameraPreview;
  late Image image;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Camera")),
      body: cameraPreview,
    ); // Scaffold
  }
}

```

3. Tambahkan sebuah fungsi yang bekerja secara asinkron untuk menentukan kamera mana yang hendak Anda pergunakan. Hal ini diperlukan mengingat perangkat-perangkat saat ini biasanya memiliki kamera lebih dari 1.

```
36    Future setCamera() async {
37      cameras = await availableCameras();
38      camera = cameras.first;
39    }
```

4. Pada **initState** lakukan pengaksesan terhadap kamera yang sebelumnya telah dipilih di mana data/snapshot gambar yang diperoleh dari kamera akan ditampilkan pada *body* menggunakan metode setState.

```
18    @override
19    void initState() {
20      WidgetsFlutterBinding.ensureInitialized();
21      setCamera().then((_) {
22        _controller = CameraController(
23          camera,
24          ResolutionPreset.medium,
25        );
26        _controller.initialize().then((snapshot) {
27          cameraPreview = Center(child: CameraPreview(_controller));
28          setState(() {
29            cameraPreview = cameraPreview;
30          });
31        });
32      });
33      super.initState();
34    }
```

5. Kembali lagi pada **screen.dart** tambahkan sebuah fungsi **camera** yang berisi pengecekan akses terhadap kamera.

```
43    void camera() async {
44      if (await Permission.camera.status.isGranted) {
45        Navigator.of(context)
46            .push(MaterialPageRoute(builder: (context) => CameraScreen()));
47      } else {
48        var status = await Permission.camera.request();
49        print(status);
50        if (status == PermissionStatus.granted) {
51          Navigator.of(context)
52              .push(MaterialPageRoute(builder: (context) => CameraScreen()));
53        } else if (status == PermissionStatus.permanentlyDenied) {
54          openAppSettings();
55        }
56      }
57    }
```

Jika akses diberikan (*granted*) maka pengguna akan langsung diarahkan menuju ke halaman kamera sedangkan jika akses belum diberikan maka pengguna akan diminta untuk memberikan akses kamera terhadap aplikasi.

Jika pengguna mencentang "Don't ask again" (*permanentlyDenied*) maka pengguna akan diarahkan menuju pengaturan untuk memberikan akses kamera secara manual. Hal ini dikarenakan ketika pengguna menolak untuk memberikan akses selamanya maka aplikasi tidak lagi dapat meminta akses melalui tampilan/*widget*.

6. Setelah semuanya telah sesuai maka seharusnya ketika tombol kamera dipilih akan menampilkan tampilan sebagai berikut.

Catatan: Jika Anda menggunakan emulator maka Anda akan mendapatkan hasil seperti yang ada di atas di mana tampilan dari kamera yang Anda dapatkan hanya contoh yang menandakan pengaksesan terhadap kamera berhasil. Namun jika Anda menggunakan *smartphone* Anda maka Anda dapat melihat secara langsung apa yang ditangkap oleh kamera *smartphone* Anda.

Latihan

1. Tambahkan sebuah permintaan hak akses fungsionalitas perangkat pada aplikasi Anda di luar dari pembacaan kontak dan pengaktifan kamera. Anda bebas merancang serta menentukan bagaimana penggunaan dari fungsionalitas yang Anda tentukan.

   Silakan baca dokumentasi dari permission_hander berikut untuk mengetahui daftar permission yang dapat Anda pergunakan https://pub.dev/documentation/permission_handler_platform_interface/latest/permission_handler_platform_interface/Permission-class.html#constants.