

Rumah Coding Course



Flutter

Studi Kasus Aplikasi "SekolahKu"

Day 1



Course Breakdown



Day 1

1. Development Tool setup
2. Intro to Dart
3. Intro to Flutter
4. Project structure
5. Versioning
6. Widget
7. Layout Widget
8. Common widget
9. Challenge



Day 2

1. Conditional
2. Operator & default value
3. Setter and getter in Dart
4. App project structure
5. Basic Navigation in Flutter
7. Create Domain, Repository and Service (Basic SOLID Principle) For Student
8. Create dummy data using faker.
9. Create a screen to list all students and get a student
10. Import Image from network or local using Image component



Day 3

1. Create a screen to create a student Using Form widget
2. StatefulWidget & setState
3. Form validation
4. Using Snackbar in Flutter to show info
5. Simulate saving data & delete data
6. About Asynchronous in Flutter
7. Intro to SQLite
8. Install Sqflite and understanding the APIs
9. Create Model Config & Model Provider to manage Our sqlite database
10. Manage connection using WidgetBindingsObserver event listener
11. Update our existing Repository and screen to match our new implementation



Day 4

1. Edit student using TextEditingController
2. Create a form login
3. Create search function
4. Update homescreen to use search function
5. Passing data back to previous screen
6. Create splashscreen & app icon Image
7. Setup our splashscreen
8. Generate keystore
9. Prepare to create signed apk
10. Test our signed apk on a real device



Final Application






Splash Screen



Login Screen

3:04 [Settings] [Security] [Battery]

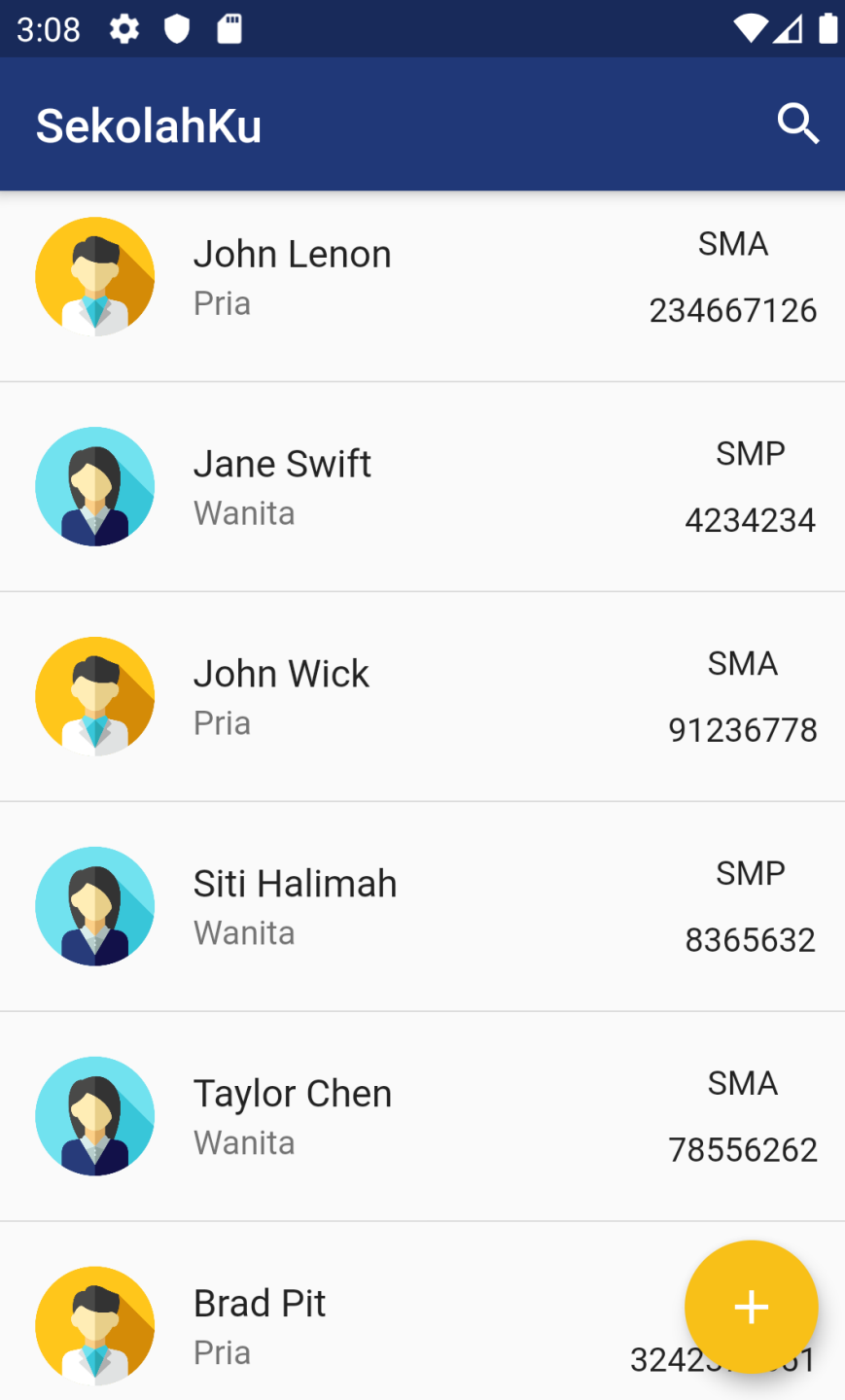


Username

Password

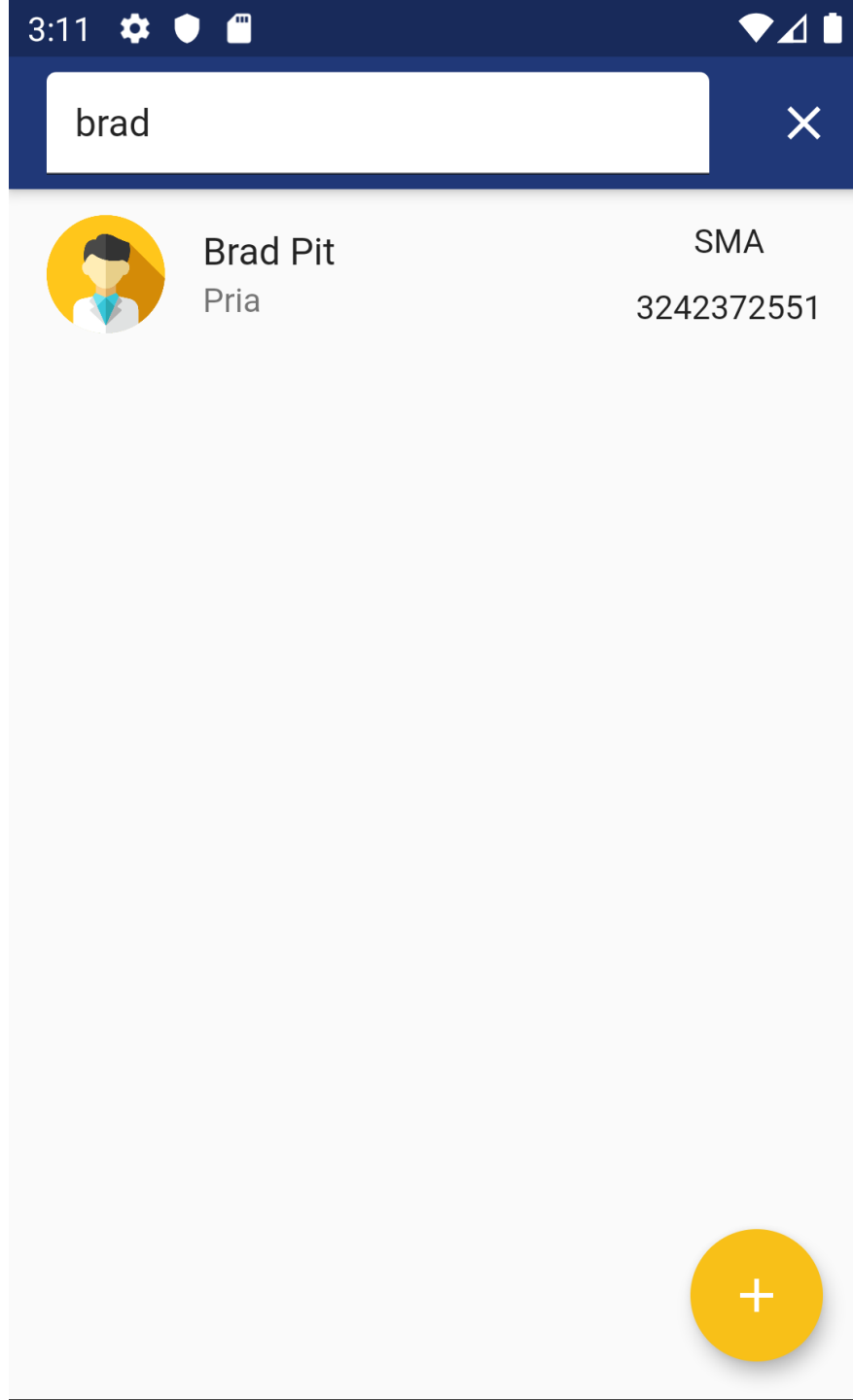
LOGIN





Home Screen





Home Screen



Add Form

3:09

← Buat Siswa

Nama Depan

Nama Belakang

No. Hp

Jenis Kelamin

☒ Pria ☐ Wanita

Pilih jenjang ▼


Hobi

☐ Membaca

☐ Menulis







☐ Menggambar

Alamat





Edit Form

3:15      

← Edit Siswa

Nama Depan
Brad

Nama Belakang
Pit

No. Hp
3242372551

Jenis Kelamin

☒ Pria ☐ Wanita

SMA


Hobi

☐ Membaca

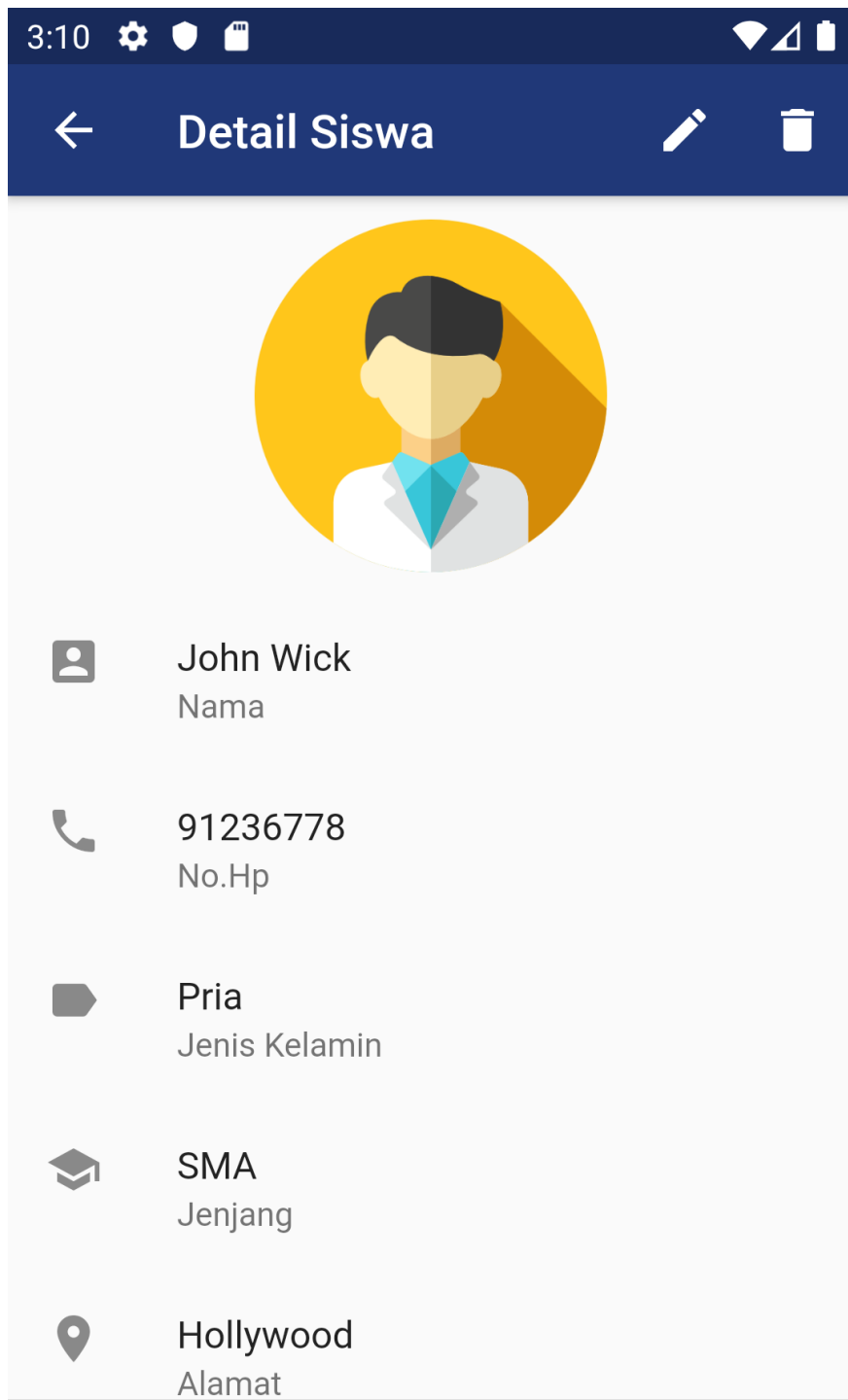
☒ Menulis

☒ Menggambar

Alamat
rwewer







Detail Screen



Development Tool setup



Development Tool

- Install Android Studio and SDK
- Setup Environment variable android (bit.ly/setupenv)
- Install flutter latest stable
- Setup environment variable for flutter (flutter.dev/docs/get-started/install/windows#update-your-path)
- Install Visual Studio Code
- Big Nox App Player

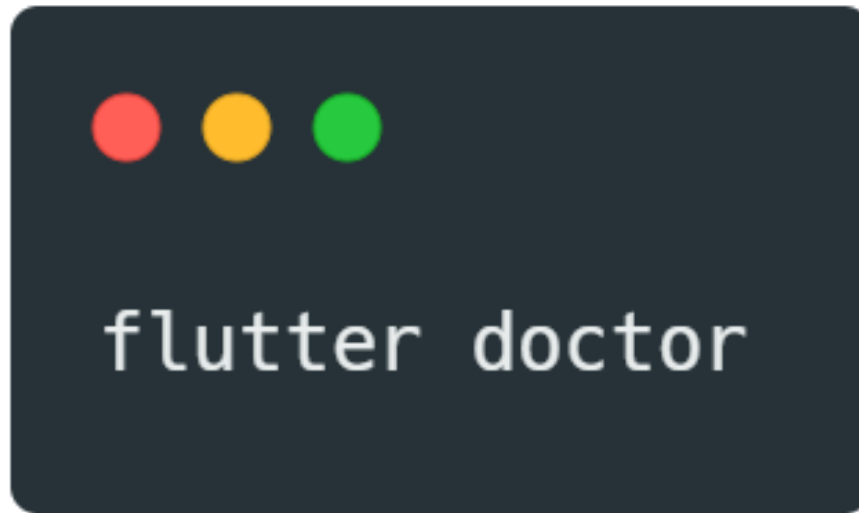


Visual Studio Code Extensions

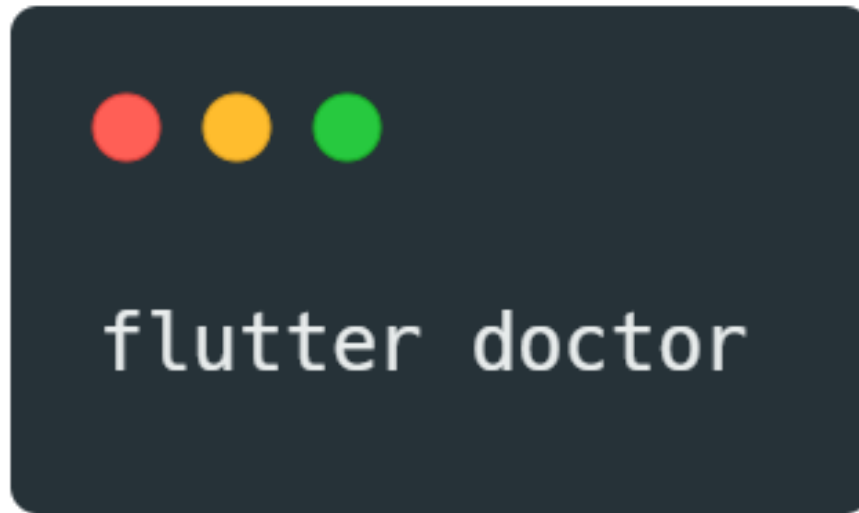
- Flutter
- Dart



Check complete setup



Check complete setup



Check complete setup



```
λ flutter doctor
```

```
Doctor summary (to see all details, run flutter doctor -v):
```

```
[✓] Flutter (Channel stable, v1.9.1+hotfix.6, on Microsoft Windows [Version 10.0.17763.864], locale en-ID)
```

```
[✓] Android toolchain - develop for Android devices (Android SDK version 29.0.2)
```

```
[✓] Android Studio (version 3.5)
```

```
[✓] VS Code
```

```
[✓] VS Code, 64-bit edition (version 1.40.2)
```

```
[!] Connected device
```

```
! No devices available
```

```
! Doctor found issues in 1 category.
```



Create New Project

- ▶ App Name: **sekolahku**
- ▶ Flutter: **1.9.1+hotfix.6**



```
flutter create --org=com.namamu --androidx --description="Aplikasi Sekolahku" sekolahku
```



Create New Project

- ▶ **--org** => organization name in reverse mode. Usually domain name. if we don't specify, defaults to "**com.example**". It's not like a real app, right?
- ▶ **--androidx** => to enable support Android X
- ▶ **--description** => set description for our app. defaults to "**A new Flutter project.**"
- ▶ **sekolahku** => our application name



Try to run, Open Emulator first



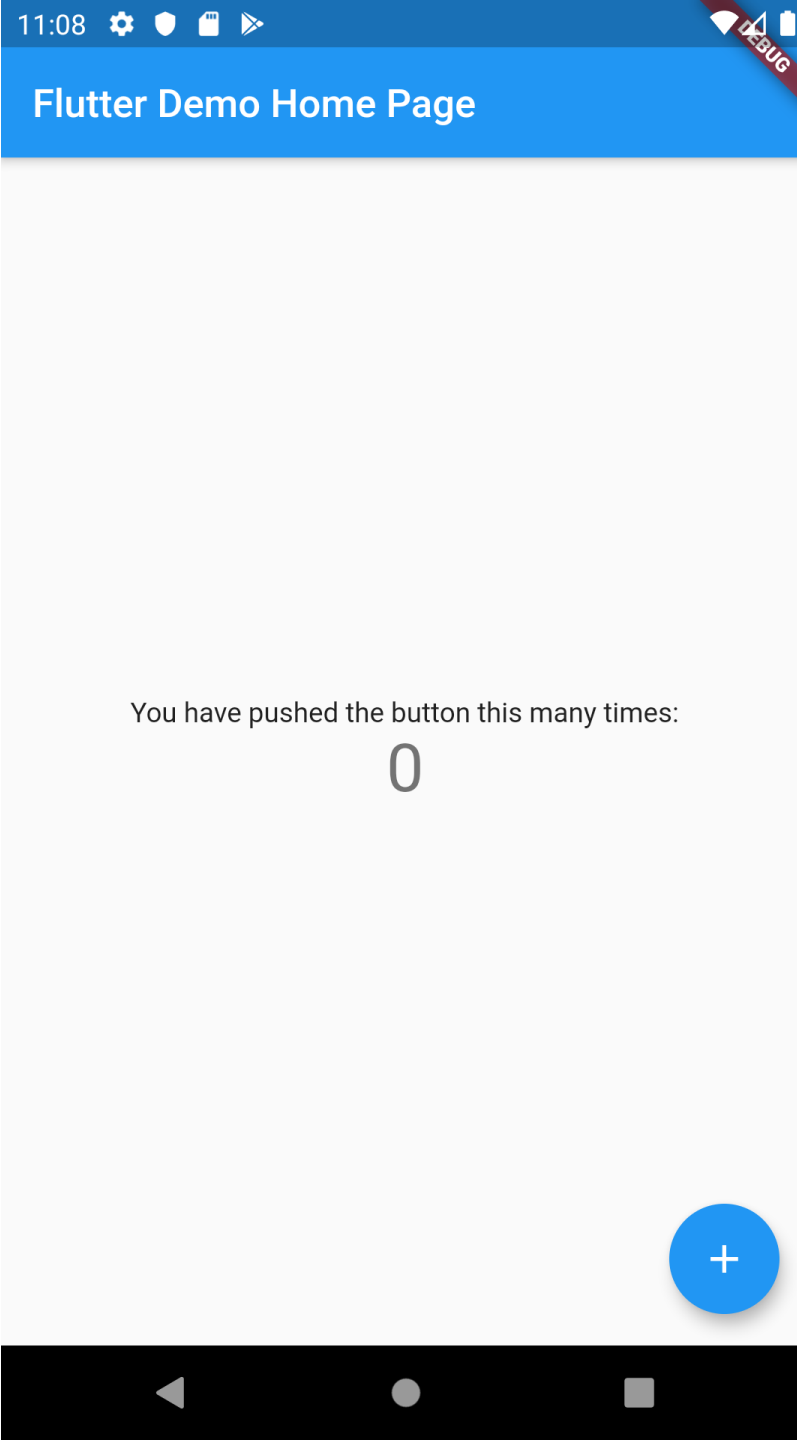
```
λ cd sekolahku  
λ flutter run
```



Try to run with VSCode

- ▶ Click **Ctrl + Shift + p**
- ▶ Type **Flutter**
- ▶ Select **Flutter: Select Device**
- ▶ **Select from list devices**
- ▶ Click **Ctrl + Shift + p**
- ▶ Type **Flutter**
- ▶ Select **Debug: attach to Flutter on Device**





Before We Start...



Introduction to Dart

- You can go to dartpad.dev

```
// before
void main() {
  for (int i = 0; i < 5; i++) {
    print('hello ${i + 1}');
  }
}

// after we remove for loop statement
void main() {
}
```



Dart Lang

- Object Oriented Programming (OOP)
- Compiled
- Need compiler
- Everything is Object
- Semicolon is a must!
- Strict type (static type support)
- Need a **main function** as a root to be executed
- Support runtime environment in debug mode.



OOP Pillars

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism



Abstraction

Abstraksi adalah proses menampilkan hanya fitur penting / yang diperlukan dari suatu entitas / objek ke dunia luar dan memerlukan informasi yang tidak relevan lainnya. Misalnya untuk TV (contoh), kami hanya memiliki tombol daya, Tidak perlu mengetahui bagaimana gelombang infra merah dihasilkan dalam TV remote control.



Encapsulation

Enkapsulasi berarti membungkus data dan fungsi anggota (Metode) bersama-sama menjadi satu unit yaitu kelas. Enkapsulasi secara otomatis mencapai konsep menyembunyikan data yang memberikan keamanan data dengan menjadikan variabel sebagai pribadi dan mengekspos properti untuk mengakses data pribadi yang akan menjadi publik.

Dalam kehidupan sehari-hari enkapsulasi dapat dimisalkan sebagai arus listrik pada generator, dan sistem perputaran generator untuk menghasilkan arus listrik. Kerja arus listrik tidak mempengaruhi kerja dari sistem perputaran generator, begitu pula sebaliknya. Karena didalam arus listrik tersebut, kita tidak perlu mengetahui bagaimana kinerja sistem perputaran generator, apakah generator berputar kebelakang atau ke depan atau bahkan serong. Begitu pula dalam sistem perputaran generator, kita tidak perlu tahu bagaimana arus listrik, apakah menyala atau tidak.



Inheritance

Kemampuan menciptakan kelas baru dari kelas yang ada. **Inheritance** adalah ketika suatu objek memperoleh properti dari objek lain. Ini membantu untuk menggunakan kembali, menyesuaikan dan meningkatkan kode yang ada. Jadi, membantu untuk menulis kode secara akurat dan mengurangi waktu pengembangan.



Polymorphism

Polymorphism, sebuah istilah Yunani, kemampuan dari sebuah objek untuk membolehkan mengambil beberapa bentuk yang berbeda. Secara harfiah, “poli” berarti banyak sementara “morph” berarti bentuk. Polimorfisme memainkan peran penting dalam memungkinkan benda yang memiliki struktur internal yang berbeda untuk berbagi antarmuka eksternal yang sama.



by Srisipul for codecall.net



Rules

- File name should `snake_case` if more than one word.
- Variable name should `camelCase` if more than one word.
- Class name should `TitleCase`.



main() {}

- Sebuah fungsi
- Sebagai root
- Mesin kompiler akan melihat dan menjalankan fungsi utama
- Kode kita harus ditempatkan di dalam badan utama (main)



print()

- A function from `dart:core`
- To debug
- Print a value to the console



Variable Declaration



```
void main() {  
    var variableName;  
    final variableNameFinal = null;  
    String ordinaryVariable;  
    const Constant = null;  
  
    print(variableName);  
    print(variableNameFinal);  
    print(ordinaryVariable);  
    print(Constant);  
}
```

- Variabel adalah pengidentifikasi ulang suatu nilai. Misalnya nama Anda. Anda adalah manusia dan diidentifikasi dengan sebuah nama
- Bahasa lain memiliki kata kunci deklarasi. Di Dart, ada 4 cara untuk mendeklarasikan variabel.
- Saat Anda mendeklarasikan variabel, variabel itu akan tersedia dalam memori untuk Anda ingat atau ditugaskan kembali
- Sebagai contoh, ordinaryVariable perlu menulis tipe data secara eksplisit. Kami memberi tahu bahwa variabel adalah sebuah String. Karena kami hanya menulis nama tanpa menyatakan kata kunci apa pun
- `var` keyword akan menyiratkan tipe data, kita tidak perlu mendefinisikan suatu tipe.



const & final

Keduanya adalah constan (Konstanta) ,const & final adalah Variabel yang bersifat ***immutable*** tidak bisa dirubah nilainya setelah diinisiasi



void & null

- void adalah tipe tetapi ini bukan nilai, dan variabel tidak dapat mengaturnya
- null adalah sebuah objek. Representasi tanpa nilai.



String

- Mewakili kata atau kumpulan kata





```
void main() {  
    var variableName = "String with double quote";  
    final variableNameFinal = 'String with single quote';  
    String ordinaryVariable = ''  
    String with the triple quote,  
    for multiline purposes  
    ''';  
    const Constant = 'Always ahead of time';  
  
    print(variableName);  
    print(variableNameFinal);  
    print(ordinaryVariable);  
    print(Constant);  
}
```



String


- Explicit write the type

```
void main() {  
    var variableName = "String with double quote";  
    final String variableNameFinal = 'String with single quote';  
    String ordinaryVariable = ''  
    This is a String with triple quote,  
    for multiline purposes  
    ''';  
    const String Constant = 'Always ahead of time';  
  
    print(variableName);  
    print(variableNameFinal);  
    print(ordinaryVariable);  
}
```



bool is Boolean

- Represent truthy and falsey value




```
bool isFaith = true;
var isRight = false;
final bool isRaining = true;
const bool isSmart = false;

print(isFaith);
print(isRight);
print(isRaining);
print(isSmart);
```



Number

- Represent any number e.g: decimal (double), Integer (positive/negative)



```
50 // Positive Integer  
-50 // Negative Integer  
8.5 // Decimal  
9 + 5.5 // using '+' operator to addition
```



Maps

- Kumpulan key value. Dalam hal mengekspresikan anggota dari segala jenis objek.

```
var gifts = {  
  // Key:    Value  
  'first': 'partridge',  
  'second': 'turtledoves',  
  'fifth': 'golden rings'  
};  
  
var nobleGases = {  
  2: 'helium',  
  10: 'neon',  
  18: 'argon',  
};
```



List

- Koleksi segala jenis data dengan pengindeksan otomatis di setiap anggota. Dalam bahasa lain disebut Array.

```
// collection of string  
[ "array", "string" ]  
  
// collection of number  
[ 12, 32, -5, 1.5 ]
```



Membuat - Mengakses Peta (map) & Daftar(list)



```

// Map
var gifts = {
    // Key:    Value
    'first': 'partridge',
    'second': 'turtledoves',
    'fifth': 'golden rings'
};
var nobleGases = {
    2: 'helium',
    10: 'neon',
    18: 'argon',
};
var giftFirst = gifts['first']; // accessing key string
var nobleTen = nobleGases[10]; // accessing key number
print(giftFirst);
print(nobleTen);

// List
var list = ['array', 'string'];
var first = list[0]; // accessing index at 0
var last = list[1]; // accessing index at 1
print(first);
print(last);

```



Class

- Kelas adalah cetak biru untuk menciptakan objek. Pikirkan tentang skema bangunan untuk membuat bangunan. Bangunan adalah objek dan skema adalah cetak biru (kelas)
- Nama kelas harus ditulis dalam huruf besar
- Konstruktor kelas adalah opsional
- Inisialisasi objek dalam kata kunci `this`.
- Kata kunci `baru` adalah opsional di Dart v2 +
- Objek baru disebut turunan dari sebuah kelas
- Kelas memiliki properti dan metode
- awalan `_` untuk properti atau metode private



```
void main() {  
    // instance person  
    var john = new Person('John', 'Wick');  
    print(john);  
    print(john.greeting());  
}  
  
class Person {  
    // public class property  
    String firstName;  
    String lastName;  
    // private property  
    final String _level = 'A rank';  
  
    // constructor  
    Person(this.firstName, this.lastName);  
  
    // public method  
    greeting() {  
        return 'Hello, ${_getFullName()}'; // interpolation  
    }  
  
    // private method  
    _getFullName() {  
        return '$firstName $lastName'; // interpolation  
    }  
}
```

Class



```

void main() {
    // instance person
    var john = new Person('John', 'Wick');
    print(john);
    print(john.greeting());
    print(john.nationalAnthem());
}

class Person extends People {
    // public class property
    String firstName;
    String lastName;
    // private property
    final String _level = 'A rank';

    // constructor
    Person(this.firstName, this.lastName);

    // public method
    greeting() {
        return 'Hello, ${_getFullName()}'; // interpolation
    }

    // private method
    _getFullName() {
        return '$firstName $lastName'; // interpolation
    }

    // overriding method parent
    @override
    nationalAnthem() {
        // body method here
    }
}

class People {
    String nationality = 'Indonesia';

    nationalAnthem() {
        // body method here
    }
}

```

Class Inheritance



Parameters

- Parameter umum sebagai input dalam fungsi dan kelas
 - Ada 3 macam parameter, named parameter, parameter positional & parameter default.
 - Khusus untuk parameter default, dapat digabungkan dengan parameter bernama atau posisi.



Parameters



```
// named parameter
enableFlags(bold: true, hidden: false);

/// actual function declaration
/// Sets the [bold] and [hidden] flags ...
void enableFlags({bool bold, bool hidden}) {...}
```



Parameters

```

// positiona parameter with optional parameter
// Wrapping a set of function parameters in [] marks them as optional positional parameters:
String say(String from, String msg, [String device]) {
    var result = '$from says $msg';
    if (device != null) {
        result = '$result with a $device';
    }
    return result;
}

// invoke
say('Gojek', 'Pesanan sesuai aplikasi'); // 3rd parameter is optional
say('Grab', 'Pesanan sesuai aplikasi', 'Android'); // we pass 3rd parameter

```



Parameters



```
// named parameter
/// Sets the [bold] and [hidden] flags ...
/// default parameter with =
void enableFlags({bool bold = false, bool hidden = false}) {...}

// bold will be true; hidden will be false.
enableFlags(bold: true);
```



Parameters



```
// positional parameter, optional parameter + default parameter
String say(String from, String msg,
    [String device = 'carrier pigeon', String mood]) {
    var result = '$from says $msg';
    if (device != null) {
        result = '$result with a $device';
    }
    if (mood != null) {
        result = '$result (in a $mood mood)';
    }
    return result;
}
```



Import

Every file are exported automatically

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

Prefix

Package name /
folder name

File name



Import



```
import 'dart:async'; // import specific dart APIs  
// more specific async library from dart  
import 'package:async/src/async_memoizer.dart';
```



Challenge 1



Challenge 1

- Buat variabel bernama myNumber dan tentukan jenis nomor yang Anda suka
- Buat variabel bernama myArray dan tetapkan jenis Array yang Anda suka
- Buat variabel bernama myObject dan tentukan jenis objek yang Anda suka
- Buat variabel bernama myWord dan tetapkan jenis string yang Anda suka
- Buat variabel bernama myBool dan tetapkan jenis Boolean yang Anda suka
- Cobalah untuk menetapkan ulang variabel itu dengan tipe apa pun
- Membuat fungsi dan mengembalikan tipe nilai Boolean, array, objek, string
- Buat kelas bernama Mamalia dan tambahkan perilaku sebagai metode
- Buat kelas bernama Sapi dan mewarisi kelas Mamalia dan tambahkan perilaku tertentu



Static Type



Keuntungan Menggunakan Sistem Tipe Statis

- Mencegah kode yang salah agar tidak dijalankan
- Mendokumentasikan basis kode saat ini
- Menawarkan dukungan IDE yang lebih kaya, misalnya, penyelesaian otomatis



Visual Studio Code Custom Settings

- Add .vscode directory in our project
- Create file settings.json
- Paste this code

```
{  
  "editor.formatOnSave": true,  
  "editor.tabSize": 2,  
  "javascript.validate.enable": false  
}
```



What's Widget

Bagi teman-teman yang belum mengerti apa itu widget, sederhananya pada Flutter seluruh tampilan seperti tombol, gambar, teks, list, ikon, bahkan satu layar pada handphone tersebut merupakan sekumpulan dari banyak widget.

Komponen pada Widget, Setiap widget memiliki komponen, misalnya kita membuat sebuah tombol, lalu kita ingin agar warna backgroundnya itu hijau, dan warna tulisannya putih, lalu teksnya kita tebalkan.

- Untuk melakukan itu semua, maka tombol kita atur melalui propertinya.

```
return Scaffold(  
  appBar: AppBar(  
    title: Text("Aplikasi Pertamaku"),  
    backgroundColor: Colors.purple,  
  ), // AppBar  
); // Scaffold
```



Widget




Widget

1. Login screen is one Widget
2. Login Screen has Image widget
3. Login Screen has Input widget
4. Login Screen has Button widget
5. Button component has Icon widget
6. Button Component has Text widget
7. Input Component has Text widget



Declare Flutter Widget

- Widget is a Class and inherit Flutter widget



```
class FormLabel extends StatelessWidget {  
  final String data;  
  
  const FormLabel(this.data, {Key key}) : super(key: key);  
  
  @override  
  Widget build(BuildContext context) {  
    return Padding(  
      padding: const EdgeInsets.only(left: 5.0),  
      child: Text(data),  
    );  
  }  
}
```

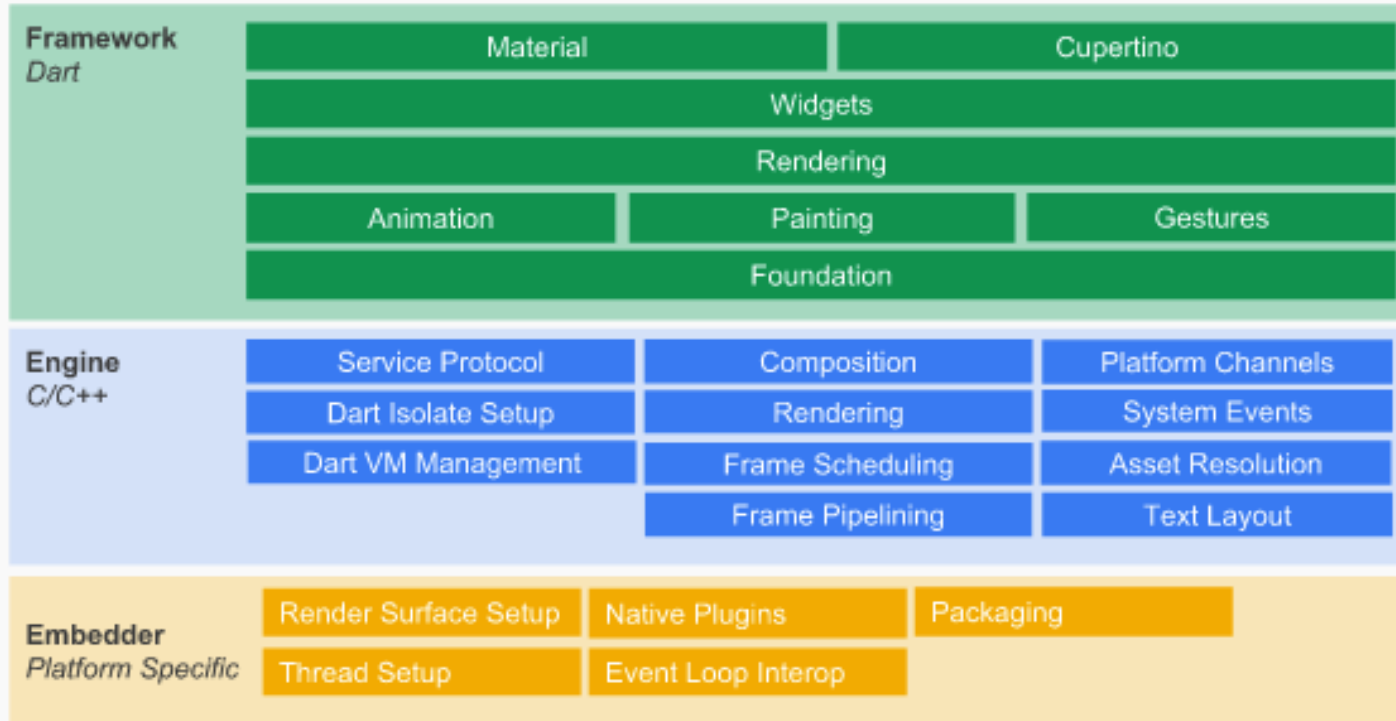


Perbedaan kelas dan Komponen fungsi

- Komponen kelas memiliki state & fungsi. Komponen belum.
- Komponen kelas memiliki referensi & fungsi. Komponen belum.
- Komponen kelas memiliki metode siklus hidup & fungsi. Komponen belum.
- Sederhananya, function Component hanyalah fungsi render (metode) saja.



Flutter System Overview

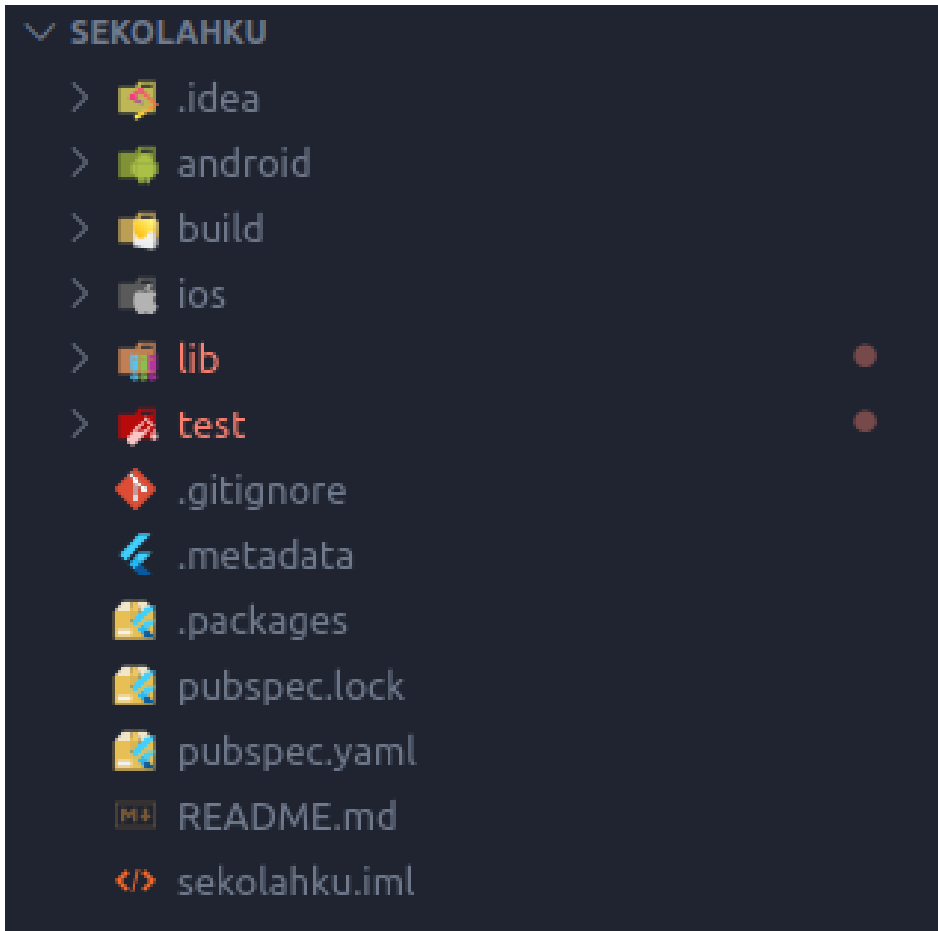


Flutter Layer

Flutter Framework berinteraksi dengan Flutter Engine (berwarna biru), melalui lapisan abstraksi, yang disebut Window. Lapisan abstraksi ini memaparkan serangkaian API untuk berkomunikasi, secara tidak langsung, dengan perangkat.



Project Structure



When we create a new project:

- ios – Native platform specific ios
- Android – Native Platform specific android
- Pubspec – Info & depedensi Proyek



Root function and file

main.dart

```
void main() => runApp(MyApp());
```

Data type

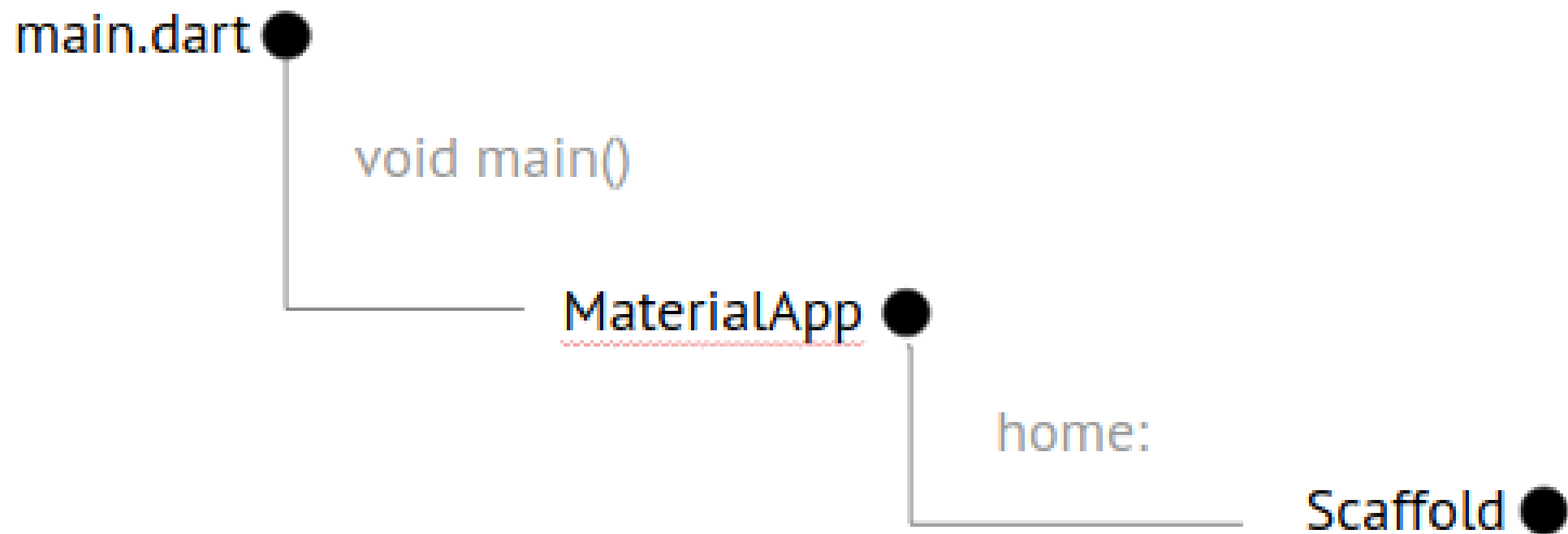
Function name

the init function
from flutter

Our root class / app



Structure tree



Versioning

Project Version

version: 1.0.0+1

Bag. pertama versi project, baik itu di android maupun di iOS, biasa disebut build name. `versionName` untuk Android dan `CFBundleShortVersionString` untuk iOS

Bag. kedua versi project, baik itu di android maupun di iOS, biasa disebut build code. `versionCode` untuk Android dan `CFBundleVersion` untuk iOS

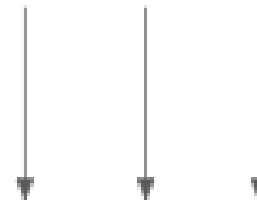


Versioning

Package Version

cupertino_icons: **^0.1.2**

Caret symbol, di cari range yang lebih jauh dari major ke minor (left to right). tapi ada kemungkinan breaking changes jika perbedaan 1 digit di major atau minor release. eg: 0.1.2 ke 0.3.0



Major Minor Patch

~

Tilde symbol, dicari range terdekat, dari patch dulu baru ke minor. (right to left)



Let's Start



Layout Widget

- Column
- Row
- Stack
- Container
- Wrap
- Expanded / SizedBox

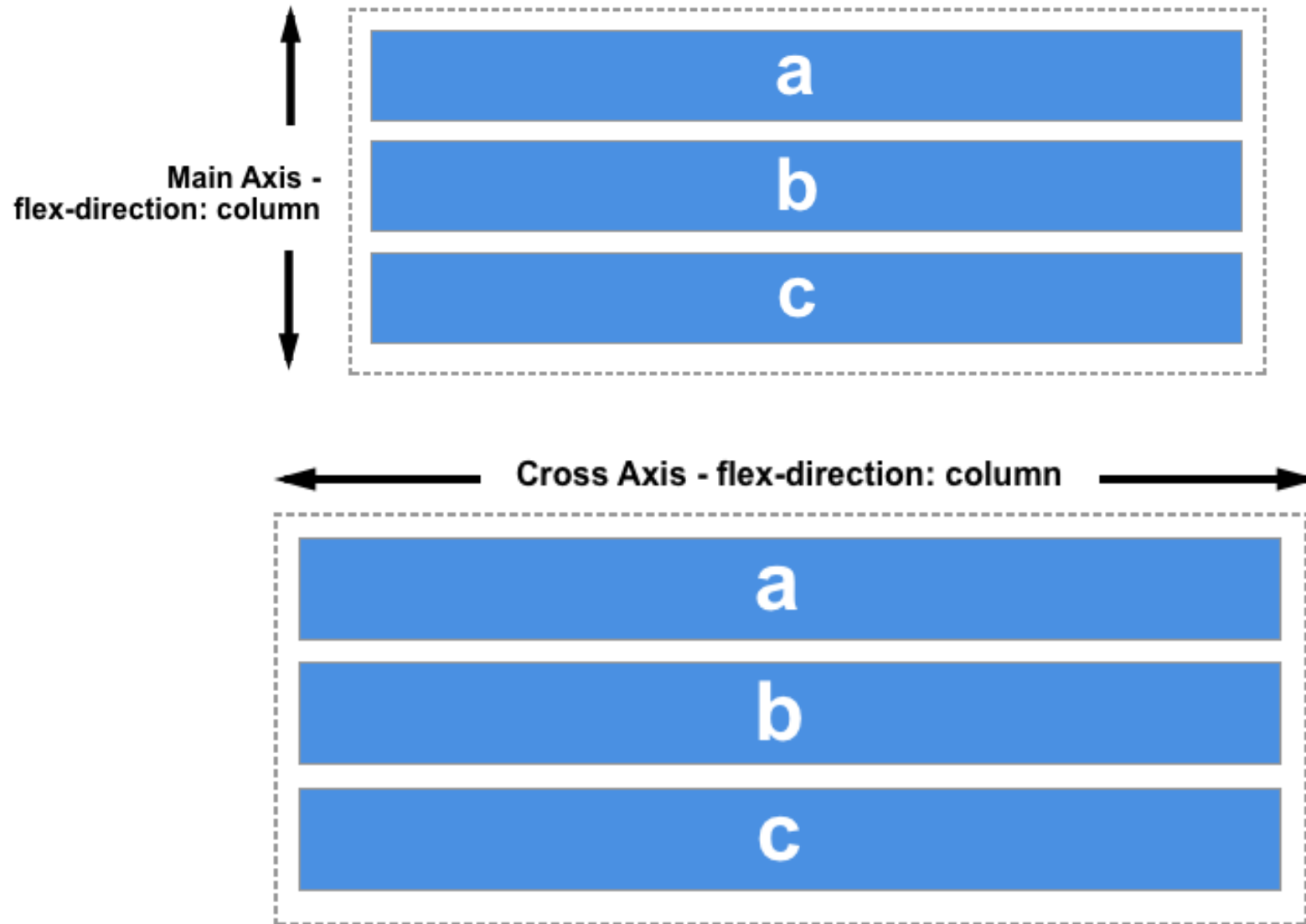


Common Widget

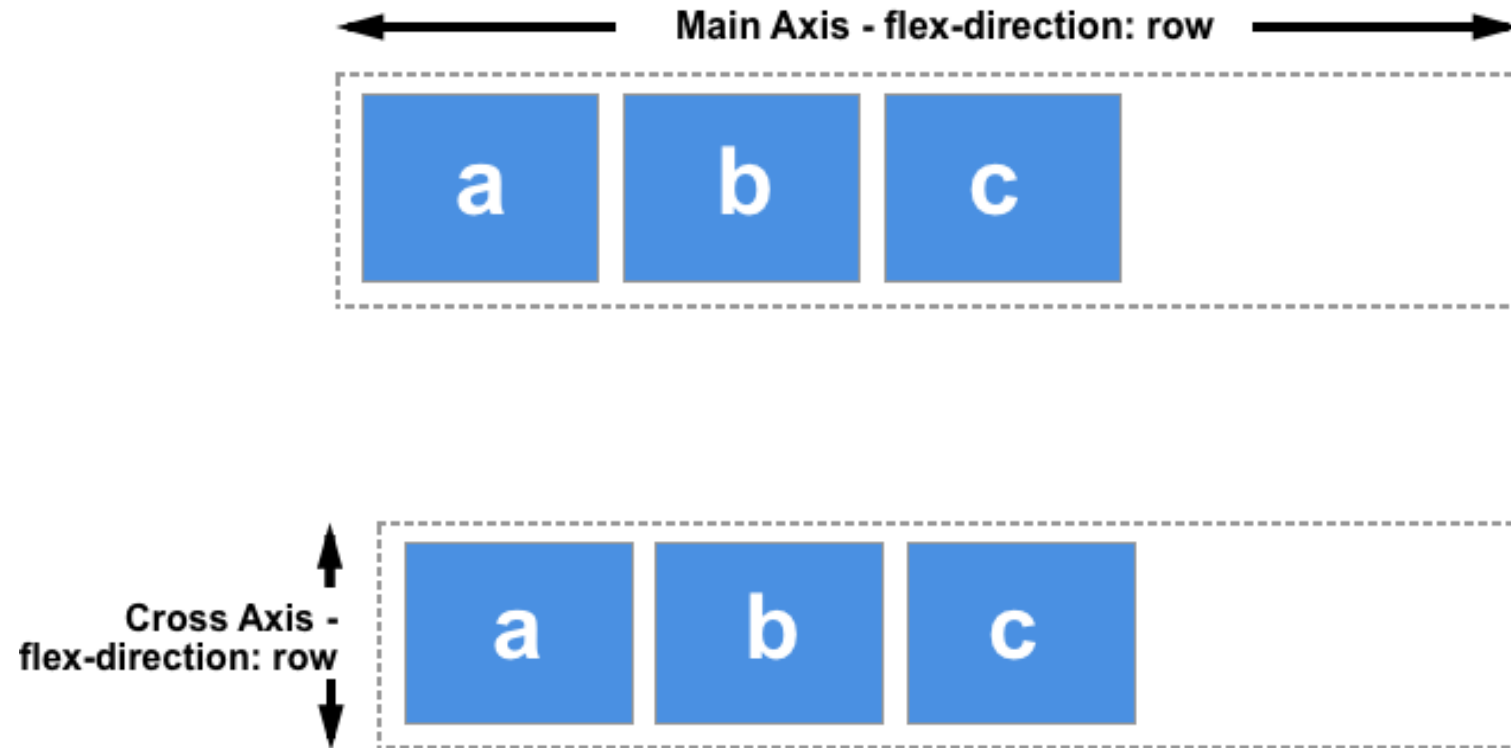
- ListView
- Center
- Padding
- Text
- TextField
- TextFormField
- FloatingActionButton
- Icon
- IconButton



Flex Direction Column

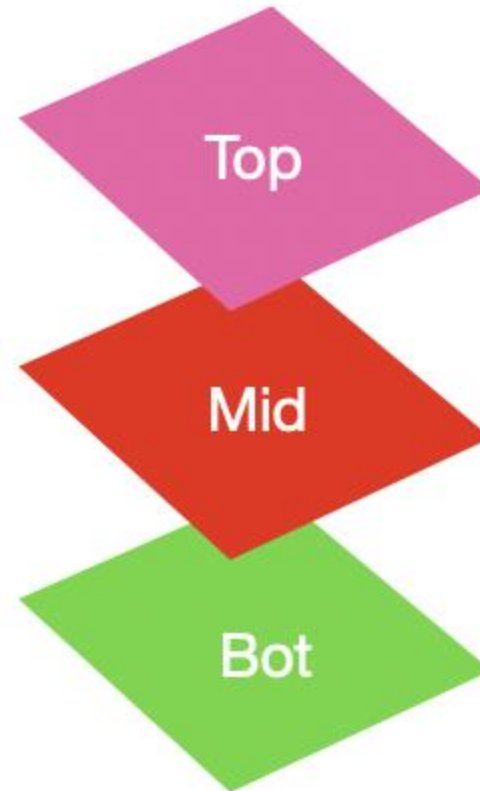


Flex Direction Row



Stack

- Think about rainbow cake



Container & Center



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

class Coba extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text('Mantap Bro!'),
    );
  }
}
```



Container & Center



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

class Coba extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container(
      color: Colors.blue,
      child: Text('Mantap Bro!'),
    );
  }
}
```



Button & handle touches

- FlatButton
- RaisedButton

```
RaisedButton(  
  onPressed: () {  
    print('pressed');  
  },  
  child: const Text('LOGIN'),  
);  
  
FlatButton(  
  onPressed: () {  
    print('pressed');  
  },  
  child: const Text('LOGOUT'),  
);
```



Handle Scroll

- Coba tambahkan teks panjang hingga layar tidak dapat menampilkan semua teks.
- Setelah itu tambahkan ListView sebagai induk dari Teks. Dan cobalah gaya itu.



Handle Scroll

- bit.ly/ScrollWithListView



Challenge



Challenge

- Buat Layar 2 TextInput & 1 Tombol dan terapkan gaya(style)
- Buat Komponen yang disebut FormLabel
- FormLabel hanya menggunakan widget Teks dan menerapkan gaya yang Anda inginkan



Name

Age

Challenge





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

class FormLabel extends StatelessWidget {
  final String data;

  const FormLabel(this.data, {Key key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.only(left: 5.0),
      child: Text(
        data,
        style: TextStyle(fontSize: 12.0, color: Theme.of(context).colorScheme.primary),
        textAlign: TextAlign.start,
      ),
    );
  }
}
```

