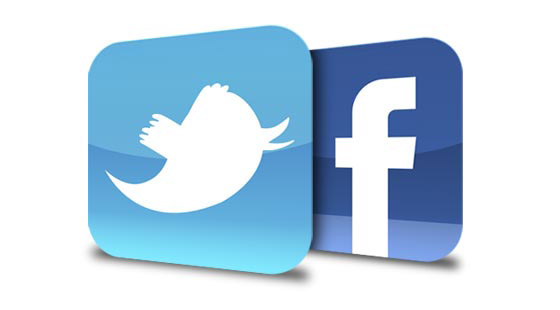
**Introduce Mobile Apps**

**1. Sejarah Mobile Apps**

Perkembangan Mobile Apps secara Exclusive dimulai saat diluncurkanya smartphone BlackBerry. Saat smartphone mulai berkembang, bisnis aplikasi mobile juga ikut berkembang seperti beberapa aplikasi store App World yang diusung Blackberry, App Store yang diusung Apple, Play Store yang diusung Android dan Store yang diusung Windows. Untuk dapat memahami lebih dalam, dapat kita baca lifetime perkembangan Mobile Apps dari tahun 2011 – 2018 berikut :

**1. Tahun 2011**

Pada tahun 2011, permintaan pembuatan aplikasi mobile terus meningkat. Mulai dari aplikasi office, PIM, multimedia. Satu lagi aplikasi yang saat ini lagi nge-trend adalah aplikasi yang menggabungkan teknologi web dengan teknologi mobile. Seperti facebook, twitter, gmaps.



**2. Tahun 2013**

**A. Kuartal Pertama 2013**

Trend pengembangan aplikasi mobile lebih ke arah game seperti angry bird, where’s my water, temple run, flappy bird, dan masih banyak game-game menarik lainnya. IOS telah melebihi jumlah pengeluaran yang terjadi di dalam game mereka dibanding dengan game-game yang dioptimasi untuk handheld.

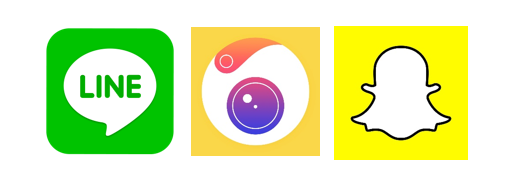


**B. Kuartal Kedua 2013**

Pada kuartal kedua ini, giliran google play yang telah membalap game-game handheld tersebut. Google play mampu menyaingi dan merebut perhatian serta pasar game handheld yang diusung IOS.

**C. Kuartal Ketiga 2013**

Pada kuartal terakhir ini, gabungan antara google play dan iOS telah melampaui game-game yang telah dioptimasi untuk handheld hingga 3x lipatnya.  Selain game, tahun 2013 merupakan era invasi aplikasi bertukar pesan besar-besaran. Selain nama besar yang sudah kita kenal yakni Whatsapp dan Skype, muncul pemain baru yang berhasil merombak peta komunikasi di Indonesia yakni Line, KakaoTalk, dan WeChat. Mereka tidak hanya muncul sebagai aplikasi bertukar pesan, tapi banyak yang sudah berevolusi menjadi sebuah platform dan mereka berhasil meng-generate revenue besar dari sana. Dua aplikasi yang paling booming di 2013 adalah Vine dan Snapchat. Tak hanya itu, aplikasi editing untuk social media berbasis foto dan video juga menjadi sangat viral di 2013 ini. Salah satu yang paling ramai digunakan adalah Camera360.



**3. Tahun 2016**

Pada tahun 2016 aplikasi mobile yang lagi nge-trend yaitu aplikasi yang menerapkan fitur virtual reality (VR) dan augmented reality (AR) seperti aplikasi cardboard, Vrse, Fulldive VR dan VaR’s VR Video Player. Tak hanya itu, Layanan Berbasis Lokasi (LBS – Location Based Services) meningkat hingga 38%. Namun, masih ada peluang besar dalam indoor mapping, navigation services, location-based payments, safety, security dan masih banyak lagi. Tahun 2016 juga marak aplikasi kesehatan. Tak hanya itu, Layanan Berbasis Lokasi (LBS – Location Based Services) meningkat hingga 38%. Namun, masih ada peluang besar dalam indoor mapping, navigation services, location-based payments, safety, security dan masih banyak lagi. Tahun 2016 juga marak aplikasi kesehatan.



**4. Tahun 2017**

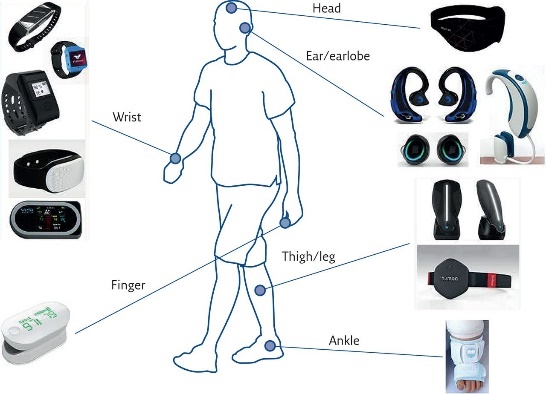
Ditahun 2017, masyarakat lebih suka membagi data personal pada era perkembangan internet yang pesat. Trend aplikasi mobile yang muncul lebih ke arah marketing sepeti aplikasi berbasis teknologi GPS dan real time, aplikasi big data, dan aplikasi yang menyasar segmen atau pasar tertentu (niche market), contohnya di Indonesia seperti Gojek, Tokopedia, Lazada, dll. Aplikasi tersebut mampu menarik minat masyarakat sebab dinilai bermanfaat dan dapat membantu memenuhi kebutuhan dan meringankan pekerjaan masyarakat.

**5. Tahun 2018**

Meningkatnya perkembangan internet telah membawa ke tahap dimana kita mengalami perkembangan aplikasi pada smartphone yang terus meningkat. Teknologi apapun tidak dapat bertahan tanpa perkembangan yang terus – menerus terjadi. Beberapa perkembangan android yang berpengaruh dari segi aplikasi antara lain :

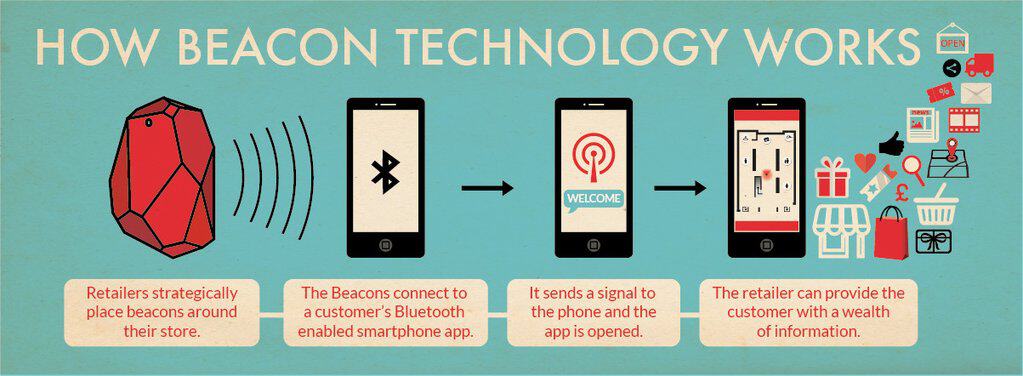
**A. Wearable Device**

Perangkat yang bisa dipakai dibagian tubuh manusia yang berhubungan dengan operasi komputer dan teknologi yang canggih dengan memperhatikan estetika dan fungsi yang bermanfaat dalam kehidupan sehari – hari. Teknologi ini menggunakan prinsip Wearable technology yang bisa dipakai dan diimplementasikan pada kehidupan sehari – hari.



**B. Teknologi Beacon**

Alat sensor yang menggunakan teknologi seperti bluetooth sebagai komunikasi nya. Awalnya teknologi ini tersedia untuk perangkat IOS saja akan tetapi saat ini Teknologi Beacon sudah banyak dibuat oleh perusahaan – perusahaan pengembang Android.



**C. Augmented Reality dan Virtual Reality**

Teknologi ini telah menjadi teknologi landasan yang digunakan untuk saat ini. Teknologi yang memungkinkan seseorang merasakan situasi seperti kehidupan nyata melalui indera pendengaran dan juga indera penglihatan mereka. Teknologi ini juga banyak digunakan untuk pada bidang-bidang seperti industri film, hiburan, simulator penerbangan dan lain sebagainya.



**D. Instant Apps**

Sebuah aplikasi yang dapat digunakan tanpa harus melakukan instalasi terlebih dahulu pada perangkat android. Dengan aplikasi ini pengguna hanya perlu mendonwload satu aplikasi dengan banyak fitur dan kegunaan didalamnya.

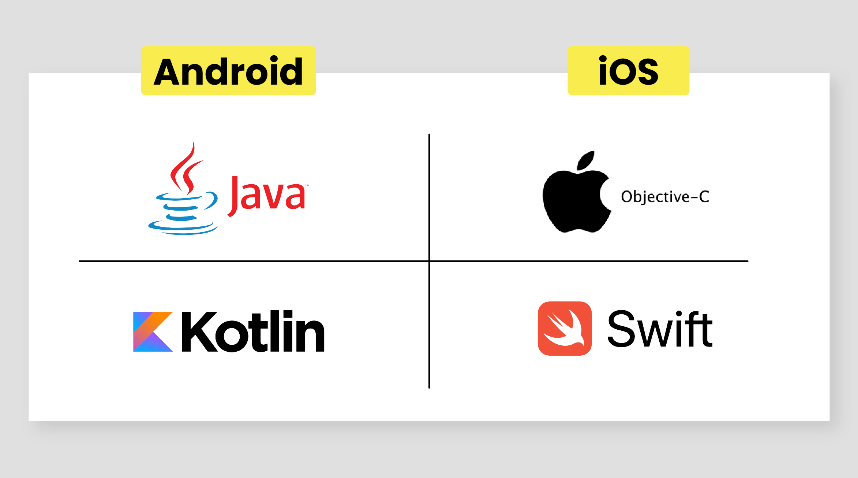
**E. In App Payment**

Sebuah pengembangan android yang digunakan untuk melakukan pembayaran online dengan dilengkapi oleh keamanan yang bisa langsung digunakan pada perangkat android, kebanyakan ini digunakan pada saat melakukan e-commerce.



Pengembang android selalu melakukan inovasi, keunikan dan update yang reguler jadi dengan ini kemajuan perkembangan android telah masuk kedalam level yang lebih tinggi dan telah dapat memuaskan penggunanya di seluruh dunia.

**2. Apa itu Mobile Apps Developer?**



Mobile developer merupakan seorang programmer yang sudah terlatih dan bisa membuat sebuah produk berupa aplikasi. Mereka melakukan pekerjaannya sesuai dengan prinsip-prinsip desain dan juga implementasi rekayasa perangkat lunak. Perlu kamu ketahui, tugas mobile developer berbeda dengan web developer maupun programmer biasa.

Mobile developer bisa dikatakan sebagai orang yang bisa menggunakan banyak sistem dan juga bahasa programming yang berbeda-beda, serta membuatnya saling terhubung satu sama lain. Berbeda dengan programmer, tugas mobile developer tidak hanya menyelesaikan masalah namun juga menciptakan aplikasi sendiri.

Selain itu, seorang mobile developer bertanggung jawab untuk mengembangkan aplikasi untuk perangkat mobile. Baik perangkat dengan sistem operasi Android maupun iOS. Mobile developer dituntut memberi perhatian khusus terhadap kompatibilitas aplikasi dengan beberapa versi sistem operasi dengan jenis perangkatnya.

**A. Tugas Mobile Developer**

*Mobile developer* memiliki beberapa tugas yang membutuhkan kemampuan dan profesionalisme tinggi. Mereka bertugas untuk mendesain dan mengembangkan aplikasi canggih untuk platform Android atau iOS. Jadi, aplikasi yang biasa kita gunakan di *smartphone,*dibuat dan dikembangkan oleh *mobile developer*.

Biasanya, mereka bekerja dengan sumber data eksternal dan API. Mereka harus melakukan tes kode unit untuk ketahanan, mulai dari kegunaan, kehandalan umum, dan lain sebagainya. Selanjutnya, mereka juga harus mengevaluasi dan menyelesaikan *bug* serta meningkatkan kinerja aplikasi yang mereka buat.

Tidak hanya itu, para *mobile developer* juga harus menerapkan teknologi baru dengan tujuan untuk memaksimalkan efisiensi pengembangan aplikasi. Mereka juga bertugas untuk menerjemahkan desain dan juga *wireframes*ke dalam kode yang memiliki kualitas tinggi.

Setelah aplikasi sudah jadi, diluncurkan dan digunakan oleh para pengguna Android atau iOS, mereka harus memastikan aplikasi bekerja dengan baik dan responsif. Terakhir, mereka juga harus menjaga kualitas kode, dan organisasi, serta otomatisasi.

**B. Kualifikasi Mobile Developer**

Untuk menjadi mobile developer profesional, kamu bisa menempuh pendidikan khusus. Posisi mobile developer biasanya terbuka untuk lulusan di bidang ilmu komputer atau apa pun yang berkaitan dengan teknologi komputer dan software.

Sebagian besar perusahaan akan mencari mobile developer yang setidaknya memiliki gelar sarjana yang berfokus pada pemrograman. Menurut[IT Career Finder](https://www.itcareerfinder.com/it-careers/mobile-application-developer.html),  Jika kamu ingin bekerja sebagai mobile developer terutama posisi manajer, kamu membutuhkan portfolio dari contoh aplikasi mobile dan pengembangan proyek terbaik yang telah kamu buat.

Program sarjana dan pascasarjana sangat bisa memenuhi persyaratan ini. Umumnya, dengan menempuh pendidikan formal, kamu diharuskan untuk membuat berbagai aplikasi sebagai tugas kuliah, ujian, maupun persyaratan lulus. Kamu juga bisa memulai karier sebagai mobile developer dengan magang di perusahaan pengembang aplikasi mobile. Banyak di antaranya yang dapat memberikan pelatihan khusus untuk mengembangkan aplikasi.

Keterampilan yang diperlukan dalam dunia mobile development adalah pendidikan dan pengetahuan dalam pengembangan aplikasi iOS (iPhone & iPad), aplikasi Android, Object-Oriented programming (OOP), bahasa pemrograman Java, JavaScript, HTML, CSS, Objective-C, C++, user-interface (UI), desain, game & Simulasi, basis data, dan pemasaran & penyebaran mobile media. Selain menguasai bahasa pemrograman, sebagai mobile developer yang baik penting untuk memiliki keterampilan sebagai berikut:

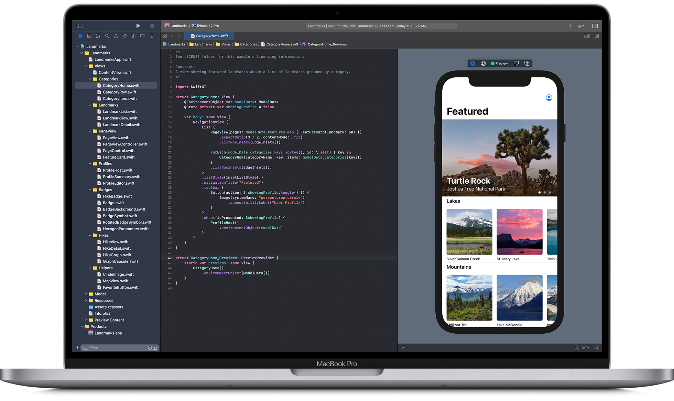
* Keterampilan berorganisasi yang kuat
* Keterampilan dalam bidang matematika
* Mampu untuk terus up to date dengan perubahan dan perkembangan
* Kemampuan untuk belajar dengan cepat (Agile)
* Kemampuan untuk menginterpretasikan serta mengikuti rencana teknis
* Kemampuan menyelesaikan masalah
* Kemampuan berkomunkasi yang baik.

**C. Jenjang Karier Mobile Developer**

Setelah memahami apa itu mobile developer dan kualifikasi yang sebaiknya dimiliki, kamu perlu mengetahui juga jenjang karier profesi ini. Pada praktiknya, pelatihan khusus dan pengalaman akan mempengaruhi posisi kamu di dunia pengembangan program dan teknologi.

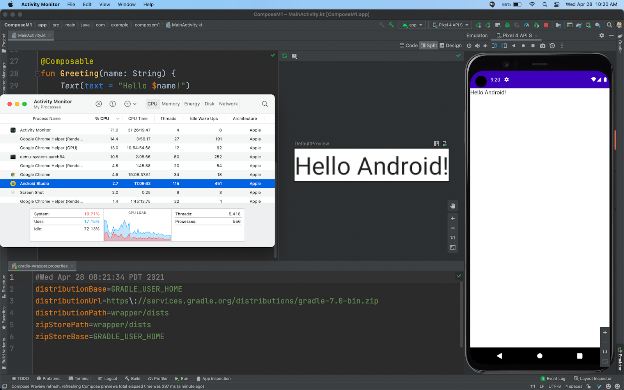
Berikut adalah beberapa contoh jenjang karier yang berkaitan dengan mobile developer:

**1. iOS Developer**



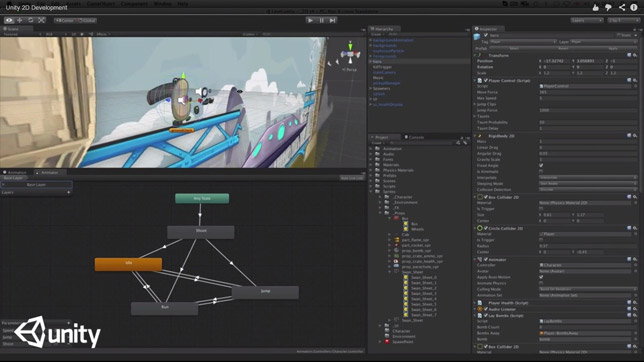
iOS developer akan berkutat pada pengembangan Xcode, memahami Swift dan pustaka mendasar iOS, dan penting untuk memiliki antusiasme juga kemampuan dalam menanangi pustaka dan frameworks baru.

**2. Android Developer**



Sebagai Android developer, kamu diharapkan bisa menerapkan sejumlah layanan yang dimiliki Google, serta memperluas pengalaman dari aplikasi inti agar bisa bertahan dan digunakan secara maksimum dalam berbagai kondisi yang dimiliki pengguna.

**3. Video Game Developer**



Video game developer menulis kode untuk permainan dalam berbagai format, seperti PC, konsol, web browser, dan ponsel. Mereka mengambil ide desainer video game, termasuk gambar dan aturan, untuk mengubahnya menjadi sebuah permainan yang dapat dimainkan dengan visual dan suara melalui penulisan kode.

**D. Gaji Mobile Developer**

Menurut <id.indeed.com> kisaran gaji untuk mobile developer setiap wilayahnya sangat bervariatif, mulai dari 4,5jt sampai 10jt rupiah ataupun lebih, dilihat dari jam terbang atau pengalaman kerja. Untuk setiap range gaji mobile developer dapat dilihat pada gambar dibawah ini :





**3. Basic Dart**

**A. Pengenalan Dart**



Dart 1.0 telah dirilis pada tanggal 14, November 2013 oleh Google dan didirikan oleh Lars Bak dan Kasper Lund. Ini bertujuan untuk membantu pengembang membangun aplikasi web dan mobile modern. Ini mencakup klien, server, dan sekarang seluler dengan Flutter. Hadir dengan berbagai alat termasuk mesin virtual, dan repositori manajemen paket, alat ini memberikan cukup amunisi bagi Anda untuk memulai proyek berikutnya.

Sebagai bahasa yang berorientasi object (object Oriented) dengan sintaksis (Syntax) C-style yang dapat diubah secara opsional menjadi JavaScript. Keunggulan yang sangat terlihat pada Dart yaitu mendukung berbagai macam alat bantu pemrograman seperti antarmuka (interface), class, collection, generics, dan opsional typing.

Alasan mengapa bahasa dart begitu cepat populer yaitu karena kita bisa menggunakan Dart untuk membuat aplikasi Web Android iOS dan juga menjalankan Server. Mudahnya saat kita sedang menggunakan dart yaitu kita dapat membuat UI(User Interface) yang indah serta berkualitas pada setiap device dengan menggunakan:

* Bahasa yang mengoptimalkan client

Dart pertama kali dioptimalkan untuk web apps dan berevolusi untuk membantu pengembangan Mobile App. Dart juga dapat kita gunakan untuk menjalankan Command Line dan Server-Side.

* Kaya akan Framework

Baik untuk pembuatan aplikasi berbasis android maupun iOS secara maksimal fungsi yang tersedia pada framework flutter dapat digunakan dengan baik di 2 device tersebut.

* Tool yang Fleksibel dan menyenangkan

Flutter adalah tool yang sangat direkomendasikan oleh dart karena pada flutter bisa digunakan untuk berbagai tujuan.

**B. Kelebihan Dart**

Beberapa kelebihan bahasa pemrograman Dart menjadikan bahasa ini menjadi salah satu bahasa baru yang dapat dipelajari oleh para developer maupun calon developer. Developers yang bekerja di Google dan perusahaan besar lainnya menggunakan dart untuk membangun aplikasi Android iOS dan Web yang berkualitas. Dart memberikan fitur yang Client Side Development (Pengembangan dari sisi client) yang oleh karena inilah banyak developer yang memilih menggunakan Dart.

1. Mudah dipelajari

* Dart memiliki banyak kemiripan dengan bahasa pemrograman yang banyak digunakan oleh developers(Java, C++, PHP , Java Script...dll).
* Kita bisa jadi tanpa disadari sudah dapat menggunakan dart karena kemiripan nya.
* Dart akan lebih mudah dipelajari jika kita sebelum nya sudah memiliki pengalaman dalam menggunakan bahasa pemrograman yang bersifat Object Oriented seperti Java ataupun C++.

1. CodeBase yang sudah di compile Natively (bawaan)

* Framework lain memberikan kita sedikit akses untuk menggunakan codingan kita pada Platform yang berbeda. Berbeda dengan Dart.
* Dart memberikan kita izin penuh untuk membuat satu aplikasi yang codingannya dapat digunakan di berbagai platform. Aplikasi yang kita buat akan dapat digunakan pada Android juga iOS.
* Dart tidak hanya dapat kita gunakan untuk mobile develop kita juga dapat menggunakan Dart untuk Web Development.

1. Produktif

* Cepat dan mudah dalam Layouting dan menambahkan Feature pada Project.
* Layout juga dapat kita buat dengan menggunakan Codingan.

1. Compile AoT dan JiT.

* Perubahan pada project dapat kita lihat secara Instan, di aplikasi.
* Tidak perlu melakukan Recompile yang memakan banyak waktu.
* Melihat perubahan tidak perlu untuk menunggu project di load.
* Anda hanya perlu untuk save dan perubahan akan terlihat.
* Ini dikarenakan Framework yang dapat meng compile Ahead of Time (AoT) / Lebih Cepat dan Just in Time(JiT) / Tepat waktu.

**C. Kenapa perlu belajar Dart?**

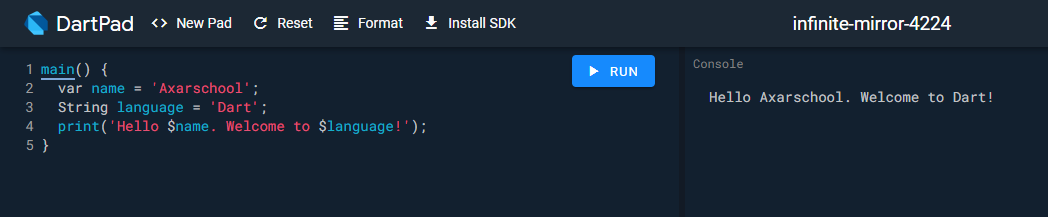
Dalam buku “***The Pragmatic Programmer***” disebutkan bahwa untuk menjadi professional software developer kita perlu belajar setidaknya satu bahasa baru setiap tahunnya. Lalu adakah alasan yang bagus untuk memulai belajar pemrograman dengan Dart?

* Pertama, ***Dart adalah bahasa pemrograman yang fleksibel***. Dart bisa berjalan di mana pun baik itu Android, iOS, maupun web. Sebagai developer, tentunya sebuah keuntungan jika bisa menuliskan kode dan bisa berjalan di mana saja.
* ***Dart adalah project open-source***. Dart dibuat oleh Google, lalu bersama dengan komunitas developer Dart mengembangkan teknologi dan fitur-fitur menarik yang bisa ditambahkan pada Dart. Jika Anda menemukan bug atau masalah pada Dart, Anda dapat melaporkannya atau bahkan memperbaikinya sendiri. Selain itu Anda tidak perlu khawatir dalam masalah lisensi ketika menggunakan bahasa Dart. Anda dapat ikut berkontribusi pada bahasa Dart pada repositori berikut: <https://github.com/dart-lang>.
* ***Dart digunakan oleh Flutter***. Sejak kemunculan Flutter, Dart kembali menjadi perhatian. Saat ini ada banyak perusahaan yang menggunakan Flutter pada aplikasinya. Flutter bisa dibilang merupakan proyek yang revolusioner dari Google untuk mengembangkan aplikasi multiplatform dengan tampilan UI yang menarik. Untuk itu, jika Anda tertarik mengembangkan aplikasi dengan Flutter, maka menguasai Dart adalah hal yang fundamental.
* ***Dart memiliki dukungan tools yang lengkap***. Hampir setiap teks editor atau IDE memiliki dukungan besar untuk Dart. Anda dapat menggunakan IDE seperti IntelliJ IDEA, Webstorm, Android Studio maupun editor sederhana seperti VS Code, Sublime text, Atom, atau yang lainnya sesuai kenyamanan Anda.
* ***Dart mudah dipelajari dan bagus sebagai first language***. Anda akan bisa memahami Dart dengan cepat khususnya jika sudah familiar dengan bahasa pemrograman populer lain seperti Java, Python, JavaScript, dll. Bahkan jika Anda baru memulai pemrograman, Dart adalah bahasa yang bagus. Anda tidak perlu menginstal apapun, cukup memanfaatkan online compiler dari Dart, Anda sudah bisa menulis dan menjalankan aplikasi Dart. Selain itu, dokumentasi dan tutorial Dart yang disediakan Google cukup mudah untuk diikuti, ditambah dengan sintaks yang sederhana, dan komunitas yang bersahabat dalam membantu kita mempelajari Dart.

**D. Karakteristik Dart**

Dart merupakan bahasa modern dan berfitur lengkap. Dart juga memiliki banyak kemiripan dengan bahasa lain yang sudah banyak dikenal seperti Java, C#, Javascript, Swift, dan Kotlin. Salah satu rancangan utama dari Dart adalah supaya bahasa ini familiar bagi developer Javascript dan Java/C#. Artinya,  yang telah familiar dengan kedua bahasa tersebut dapat memulai belajar bahasa Dart dengan lebih mudah. Namun, jika Anda adalah calon developer yang baru memulai belajar pemrograman dan memutuskan Dart sebagai first language, tenang saja. Dart adalah bahasa yang nyaman dan mudah dipelajari untuk memulai pemrograman.

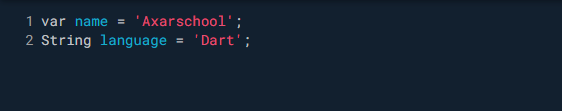
Kita ambil contoh potongan kode Dart berikut:



Bahasa pemprograman dart juga termasuk dalam Bahasa yang memiliki beberapa karakteristik, antara lain :

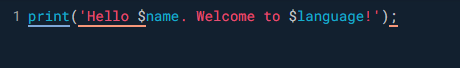
* *Type Inference*
* *String Expressions*
* *Statically Typed*
* *Multi-Paradigm : OOP & Functional.*

Bahasa pemrprograman dart merupakan Bahasa yang ***Statically Typed,*** yang berarti kita perlu mendefinisikan variabel sebelum bisa menggunakannya. Berikut contoh kode yang mendeklarasikan variabel pada Bahasa dart.



Dapat dilihat pada kode diatas bahwa dart tidak memerlukan definisi tipe data variabel secara eksplisit. Ini dikarenakan dart mendukung ***Type Inference,*** yang mana tipe data akan secara otomatis mendeteksi ketika ada variabel diinisialisasi. Seperti variabel ***name***  akan mendeteksi sebagai ***String.***

Pada Bahasa dart juga memiliki fitur ***String Interpolation***. Yang dimana fitur tesebut bisa menyisipkan variabel ke dalam sebuah objek ***String*** tanpa ***Concatenantion*** (penggabungan objek  ***String*** menggunakan **+)**. Dengan fitur tesebut, tentu dapat menjadi mempermudah dalam membuat objek ***String*** yang dinamis. Sebagai contoh dapat dilihat sebagai berikut :



**E. Instalasi Tools**

Sebelum kita mulai mencoba bahasa pemrograman Dart, Beritkut ini beberapa tools yang harus di-install. Seperti meng-install Dart SDK, Dart Tool (DartPad, IDEs, Command-line tools) dan lainnya.

**Instalasi Dart**

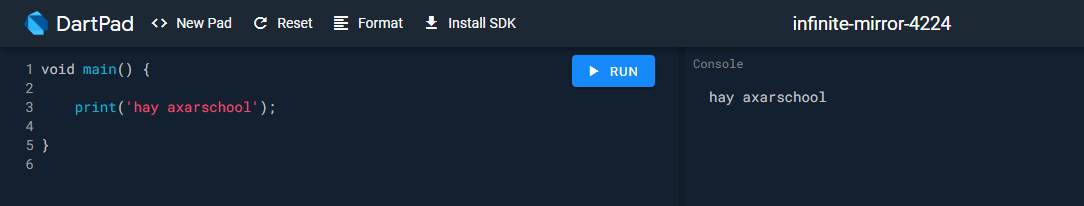
* Click Link Berikut ini Ini : https://gekorm.com/dart-windows/



* Download fully tested version
* Jalankan file yang telah didownload, tekan tombol “next” seterusnya hingga “finish”

**First Code on Dart**

* Dart menyediakan editor online di <https://dartpad.dev/?null_safety=true>. Editor Dart dapat menjalankan skrip dan menampilkan HTML serta Console Output.



* **Fungsi Main()** Setiap aplikasi terdapat point dalam programnya yang berfungsi untuk masuk ke aplikasi. Ketika sebuah aplikasi dijalankan, dimulai dari point masuk yang ditentukan itu. Di dart, point untuk masuknya adalah fungsi main ().
* **print( )** adalah fungsi yang telah ditetapkan untuk mencetak string atau nilai tertentu ke output.

Berikut ini beberapa penjelasan tentang **Dart Fundamental** yang perlu diketahui sebelum memulai Coding menggunakan Dart

**4. Dart Fundamental**

Sebelum mempelajari lebih dalam tentang Dart dan Mobile Apps, terlebih dahulu kita mempelajari hal yang paling dasar pada pemprograman, yaitu ‘**Fundamental**’. Menurut Paul Barry dan David Griffiths “Banyak orang tahu cara menggunakan komputer, tapi hanya sedikit saja yang melanjutkan untuk belajar mengendalikannya”.

Apakah belajar mengoperasikan komputer adalah sesuatu yang membanggakan? Tidak, sampai kita tahu ad acara untuk membuat komputer mengikuti semua perintah kita. Gimana caranya? Membuat program komputer atau **programming**!

**Kenapa harus belajar Fundamental Pemprograman?**

Umumnya, fundamental pemprograman yaitu antara Bahasa pemprograman yang satu dengan Bahasa pemprograman yang lain adalah sama. Perbedaan terletak pada bentuk syntax pada Bahasa pemprograman tersebut

Kita tidak perlu rakus belajar semua Bahasa pemprograman, karena pasti bakal ketemu konsep-konsep yang itu-itu saja. Makanya orang-orang yang belajar pemprograman secara otodidak bakal sering kali mudah megikuti tutorial yang ada, tapi kerepotan ketika harus membuat variasi program yang lain.

Itu karena kebanyakan tutorial memberikan contoh kasus yang sifatnya umum dan kurang tepat guna. Ya, walaupun juga banyak tutorial yang sudah bagus dan layak untuk diikuti. Sehingga bagi kita yang baru belajar (apalagi otodidak) butuh bertahun-tahun untuk mengerti tentang Fundamental Pemprograman ini.

Tapi, dengan memahami tentang Fundamental Pemprograman ini, kita akan lebih mudah untuk belajar hal-hal baru terkait pemprograman dalam Bahasa pemprograman apa saja. Menarik kan? Ya, karena Fundamental Pemprograman bersifat umum dan tidak terikat dengan Bahasa pemprogramannya. Jadi, kita bisa gunakan Java, PHP, Javascript, Switft atau bahkan Python.

**Apakah setelah belajar Fundamental Pemprograman sudah bisa membuat program?**

Hmmmm, ketawa batin.

Mempelajari Fundamental Pemprograman **tidak sama** dengan membuat program. Sama halnya seperti mempelajari cara menggunakan kamera digital **tidak sama** dengan membuat film. Sama halnya seperti mempelajari tentang cara menggunakan gergaji **tidak sama** dengan membuat rumah.

Tapi, ketika kita sudah bisa membuat rumah nantinya, kita bisa bekreasi sesuka hati ketika ingin membuat rangka atap rumah kita, karena sebelumnya kita belajar teknik memotong kayu yang benar menggunakan gergaji. Dan, **kita tidak harus** mengikuti gaya rangka atap pada umumnya. Sampai disini paham, kan?

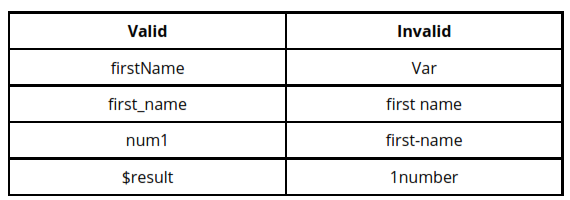
**Apa saja yang dipelajari dalam Fundamental Pemprograman?**

Pada Fundamental Dart akan mempelajari seperti syntax sederhana identifier, comment, Data Type dan sebagainya. Kita mulai dahulu dengan identifier pada Dart.

**Identifier pada Dart**

Pengidentifikasi (identifier) adalah nama yang diberikan kepada elemendalam program seperti variabel, fungsi, dll. Rules (Aturan) untuk identifier adalah

* + Identifier dapat menyertakan karakter, dan digit(angka). Namun, identifier tidak dapat dimulai dengan digit.
  + Identifier tidak dapat berisikan simbol-simbol spesial selain underscore (\_) dan tanda dolar ( $ ).
  + Identifier tidak dapat berisikan Keyword \*
  + Identifier harus unik - Identifier tidak dapat berisikan spasi(space)
  + Berikut beberapa contoh identifier yang valid(dapat digunakan) dan invalid identifier (yang tidak dapat digunakan).



**Whitespace dan Line Breaks**

Dart mengabaikan spasi, tab, dan baris baru yang muncul dalam program. Kita dapat menggunakan spasi, tab, dan baris baru secara bebas dalam programkita dan kita bebas untuk memformat dan memasukkan program kita dengancara yang rapi dan konsisten yang membuat kode mudah dibaca dan dipahami.

**Dart adalah case-sensitive**

Ini berarti bahwa Dart membedakan antara huruf besar dan huruf kecil. Kesalahan penggunaan huruf besar dan huruf kecil sering terjadi pada pembuatan identifier dan class. Untuk itu harap lebih diperhatikan lagi dalam penamaan pada Dart!.

**Pernyataan (statement) diakhiri dengan Semicolon**

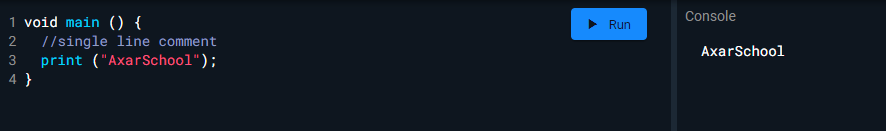
Setiap baris instruksi disebut pernyataan. Setiap pernyataan Dart harus diakhiri dengan titik koma (;). Satu baris dapat berisi beberapa pernyataan. Namun, pernyataan ini harus dipisahkan dengan titik koma.

**Comment di Dart**

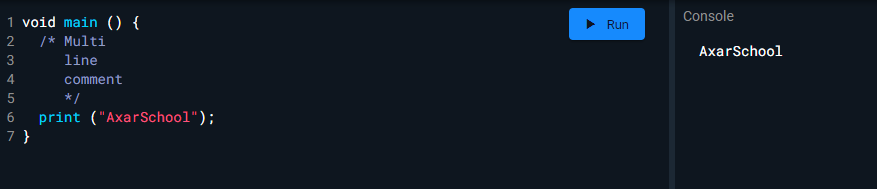
Komentar (Comment) adalah cara untuk meningkatkan keterbacaan suatu program. Comment dapat digunakan untuk memasukkan informasi tambahan tentang program seperti penulis kode, petunjuk tentang fungsi / konstruksi dll. Comment akan diabaikan oleh compiler. Dart dapat berisikan jenis komentar berikut

* **Single Line Comment ( // )** ⇒ Teks apapun yang berada setelah " // " dalam satu line diperlakukan sebagai komentar
* **Multi Line Comments ( /\* \*/ )** ⇒ Teks apapun yang berada diantara “/\* \*/” diperlakukan sebagai komentar, Komentar ini dapat mencakup banyak line.

**Single Line Comment**



**Multi Line Comment**



**Syntax pada Dart**

Syntax mendefinisikan seperangkat aturan untuk menulis program. Setiap spesifikasi bahasa mendefinisikan sintaksnya sendiri. Sintaks Dart terdiri dari:

* Variable
* Operators
* Class
* Function
* Loop
* Comments
* Library
* Package
* Typedefs
* Decision making
* Expression dan Programming Construct (konstruksi pemrograman)
* Data Structure yang direpresentasikan sebagai Generics/Collections

**Tipe Data Pada Dart**

Salah satu karakteristik paling mendasar dari bahasa pemrograman adalah himpunan tipe data yang dimiliki oleh bahasa pemrograman. Ini adalah jenis nilai yang dapat direpresentasikan dan dimanipulasi dalam bahasa pemrograman. Dart memiliki tipe-tipe data berikut :

* Numbers
* Strings
* Booleans
* Lists
* Maps
* **Numbers**

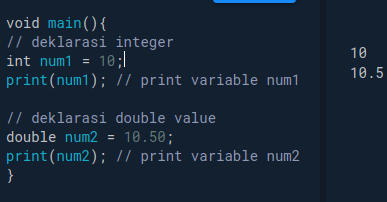
Numbers (Angka) dalam Dart digunakan untuk mewakili literal numerik. Number pada Dart memiliki dua tipe :

* **Integer** sebuah tipe data dari Number yang nilainya berbentuk bilangan bulat. Nilainya harus bersifat angka dan tidak boleh ada koma.
* **Double** digunakan untuk merepresentasikan angka float / decimal.

Sintaks untuk menyatakan number adalah seperti yang diberikan di bawah ini :

int nama\_var; // mendeklarasikan variable integer

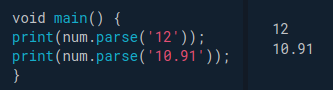
double nama\_var; // mendeklarasikan variable variable



*NOTE: Dart akan throw exception (memberikan error) jika nilai pecahan diletakkan pada variabel integer.*

**Parsing Number**

Fungsi statis parse() memungkinkan penguraian string yang berisi numerik literal menjadi angka. contoh berikut menunjukkan hal yang sama :



**Property dari Number**

Tabel berikut mencantumkan properti yang dimiliki oleh Number pada Dart

|  |  |  |
| --- | --- | --- |
| **Property** | **Deskripsi** | **Contoh** |
| **hashcode** | Me-return kode hash untuk  nilai numerik | *void main() {*  *int n = 5000;*  *print(n.hashCode);*  *}* |
| **isFinite** | **true**, jika jumlahnya terbatas.  **false**, jika jumlahnya tidak terbatas. | *void main() { int n = 5000;*  *print(n.isFinite);}* |
| **isInfinite** | **true**, jika jumlahnya adalah infinity (tidak terbatas) positif atau tidak terbatasnya negatif.  **false**, jika tidak infinity. | *void main() { int n = 5000;*  *print(n.isInfinite);}* |
| **isNegative** | **true**, jika jumlahnya negatif.  **false**, jika sebaliknya. | *void main() { int posNum = 10;*  *int posNeg = -10;*  *print(posNum.isNegative);*  *print(posNeg.isNegative); }* |
| **sign** | Return minus satu, nol atau plus satu tergantung pada tanda dan nilai numerik dari angka tersebut. | *void main() { int posNum = 10;*  *int posNeg = -12; int valZero = 0;*  *print(posNum.sign); print(posNeg.sign);*  *print(valZero.sign); }* |
| **isEven** | Me-return nilai true jika nomornya adalah bilangan genap. | *void main() {int posNum = 10; print(posNum.isEven); }* |
| **isOdd** | Me-return nilai true jika nomor tersebut adalah angka ganjil. | *void main() { int posNum = 10;*  *print(posNum.isOdd);}* |

**Method dari Number**

Dibawah ini adalah List Method yang dapat kita gunakan pada Numbers

|  |  |  |
| --- | --- | --- |
| **Method** | **Deskripsi** | **Contoh** |
| **abs** | Return nilai absolute dari sebuah number. | *void main() {*  *var a = -2;*  *print(a.abs());* *}* |
| **ceil** | Return bilangan bulat terkecil yang tidak lebih kecil dari angka. | *void main() {*  *var a = 2.4;*  *print(“ceiling value dari 2.4 = + ${a.ceil()}”);* *}* |
| **compare** | Membandingkan number ini dengan  number lain. | *void main() {*  *var a = 2.4;*  *print(a.compareTo(12));*  *print(a.compareTo(2.4));*  *print(a.compareTo(0));* *}* |
| **Floor** | Return bilangan bulat terbesar tidak lebih besar dari angka saat ini. | *void main() {*  *var a = 2.9;*  *print(“floor value 2.9 =*  *${a.floor()}”);* *}* |
| **remainder** | Return sisa terpotong setelah  membagi dua angka. | *void main() {*  *var a = 10;*  *var b = 17;*  *print(a.remainder(2));*  *print(b.remainder(2));* *}* |
| **Round** | Return integer terdekat dari number  saat ini. | *void main() {*  *double n1 = 12.023;*  *double n2 =*  *12.89;*  *var value = n1.round();*  *print(value);*  *value = n2.round();*  *print(value);* *}* |
| **toDouble** | Return double yang setara dengan number saat ini. | *void main() {*  *int n1 = 2;*  *var value =*  *n1.toDouble();*  *print(“Output = ${value}”);* *}* |
| **toInt** | Return int yang setara dengan number  saat ini. | *void main() {*  *int n1 = 2.0;*  *var value = n1.toInt();*  *print(“Output = ${value}”);* *}* |
| **toString** | Return string yang setara dengan  number saat ini. | *void main() {*  *int n1 = 2;*  *var value = n1.toString();*  *print(value is String);* *}* |
| **Truncate** | Return integer setelah membuang digit  pecahan apapun | *void main() {*  *double n1 = 2.123;*  *var value =*  *n1.truncate();*  *print(“truncate value dari 2.123 =*  *${value}”);* *}* |

**Strings**

Tipe data String mewakili urutan karakter. String Dart adalah urutan unit kode UTF 16 1 . Nilai string di Dart dapat direpresentasikan menggunakan kutip tunggal (‘ ’) atau ganda ( “ ” ) atau tiga (‘’’ ‘’’). String baris tunggal direpresentasikan menggunakan tanda kutip tunggal atau ganda. Kutipan rangkap(kutip tiga) digunakan untuk mewakili string multi-line.

//kutip satu dan dua untuk string satu line (tunggal)

String variable\_name = 'value';

String variable\_name = ''value'';

//kutip rangkap (tiga) untuk string multi-line

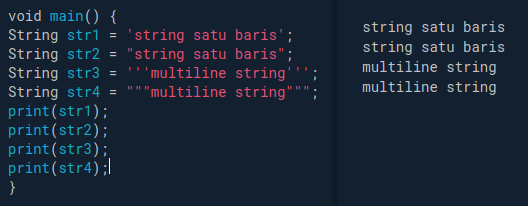
String variable\_name = '''line1

line2''';

String variable\_name= ''''''line1

Line2'''''';

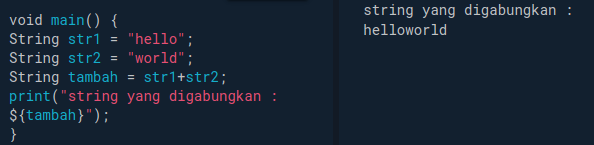
**Contoh**



**String Interpolation (interpolasi)**

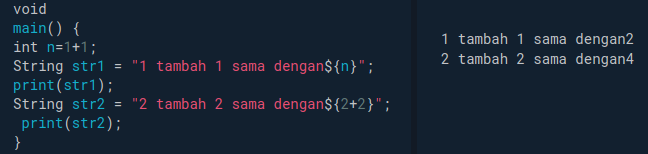
Proses membuat string baru dengan menambahkan nilai ke string statis disebut sebagai penggabungan atau interpolasi. Dengan kata lain, itu adalah proses menambahkan string ke string lain.

Operator plus ( + ) adalah mekanisme yang biasa digunakan untuk menyatukan / menyisipkan string.



**Contoh 2**

Kita dapat menggunakan " $ { } ", dapat digunakan untuk menginterpolasi nilai ekspresi Dart dalam string. Contoh berikut menggambarkan hal yang sama.



**String Properties**

Properti string pada tabel berikut hanya dapat dibaca(read-only properties).

|  |  |  |
| --- | --- | --- |
| **Properti** | **Deskripsinya** | **Contoh** |
| **codeUnits** | Me-return daftar unit kode UTF-16 yang tidak dapat dimodifikasi dari  string ini. | *void main() {*  *String str = “Hello”;*  *print(str.codeUnits); }* |
| **isEmpty** | Return true jika string tidak memiliki value atau empty (kosong). | *void main() {*  *String str = “Hello”;*  *print(str.isEmpty); }* |
| **Length** | Return panjang (jumlah karakter) pada string termasuk spasi, tab dan karakter  baris baru (enter). | *void main() {*  *String str = “Hello All”;*  *print(“Length dari string*  *adalah ${str.length}”); }* |

**5. Perkenalan Flutter**



**A. Sejarah Flutter**

Flutter adalah framework untuk pengembangan aplikasi mobile open source yang dikembangkan oleh Google, versi pertama dari flutter dinamai dengan codename Sky dan berjalan pada sistem operasi Android, lalu pada Desember 2018 flutter stabil versi 1 dirilis.

**B. Kelebihan Flutter**

memiliki beberapa kelebihan yaitu :

1. **Hot Reload** merupakan sebuah fitur ketika developer mengembangkan sebuah aplikasi, lalu melakukan perubahan pada kode program, program akan otomatis mengupdate perubahan, maka dengan adanya hot reload kita tidak perlu melakukan build ulang, cukup save lalu akan otomatis mengupdate sesuai dengan apa yang telah kita ubah, sehingga sangat mempercepat proses pengembangan program.
2. **Multiplatform** artinya kode yang telah kita buat maka akan bisa dijalankan baik di ios, android, web, maupun desktop tanpa perlu melakukan porting secara manual.
3. **UI yang Cantik** UI flutter telah didesain dengan sangat baik, juga untuk menggunakannya sangat mudah sekali.

**C. Instalasi**

Untuk menggunakan flutter, maka ada 2 IDE yang direkomendasikan , yaitu Visual Studio Code dan Android Studio, untuk mendapatkannya download melalui link berikut :

1. Android Studio : <https://developer.android.com/studio>

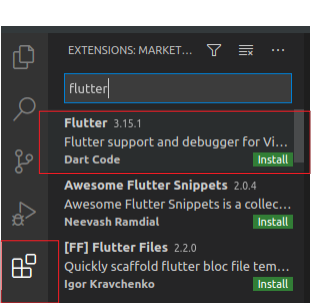
2. Visual Studio Code : <https://code.visualstudio.com/>

Setelah menginstall IDE maka diperlukan Flutter SDK, untuk mendapatkannya bisa didownload melalui link <https://flutter.dev/docs/get-started/install>

Setelah itu install plugin Flutter :

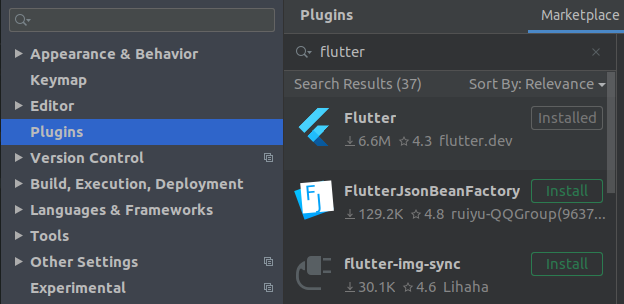
**1. Visual Studio Code :**

Install plugin melalui menu extension, lalu search Flutter dan install



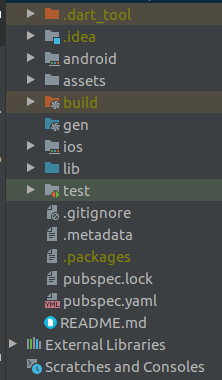
**2. Android Studio**

Buka Settings > Plugins > Marketplace, lalu search Flutter dan install



**D. Hal yang Perlu diketahui**

Hal yang perlu diketahui ketika menggunakan flutter yaitu :



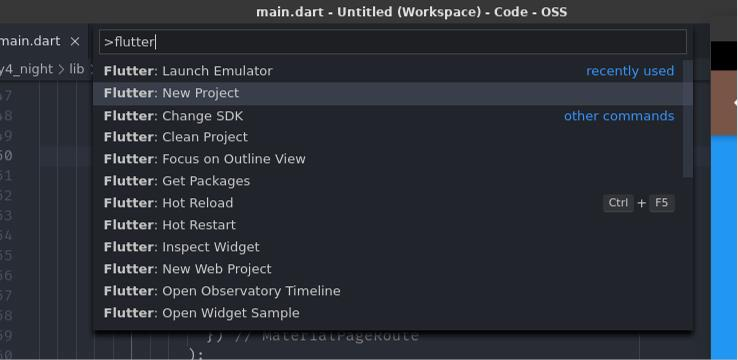
1. **folder lib** pada folder lib inilah kode kita akan dibuat, dan biasanya di folder ini bisa berisi beberapa folder agar file kodingan kita bisa lebih rapi.
2. **pubspec.yaml** pada flutter semua gambar atau assets yang ditambahkan ke project maka dia wajib didaftarkan di file pubspec.yaml, file ini juga berfungsi apabila kita ingin menambah dependensi pada project kita, maka kita perlu menambahkannya disini lalu setelah itu kita tinggal menjalankan perintah flutter pub get pada terminal.
3. **build.gradle** pada umumnya fungsinya untuk menambah dependensi seperti pada project android.
4. **Info.plist** Pada file ini berisi konfigurasi permission untuk ios.

**E. Hot Reload**

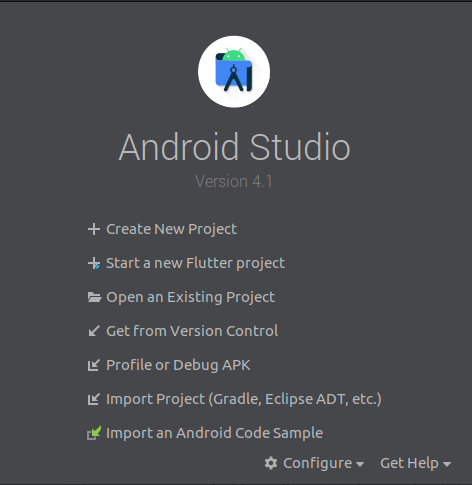
Hot reload flutter mempunyai fitur pengembangan cepat / hot reload, dimana saat kita memperbarui kode kita tidak perlu membuild aplikasinya. Sama halnya saat kita koding di web, hanya save dan lihat hasilnya, Untuk melakukan build project flutter, kita dapat menggunakan sistem operasi apapun seperti Windows, Linux, dan juga MacOS. Namun apabila kita ingin build project untuk iOS, maka diperlukan perangkat berupa MacBook, dan menggunakan XCode sebagai emulatornya.

**6. Begin Create a Project!**

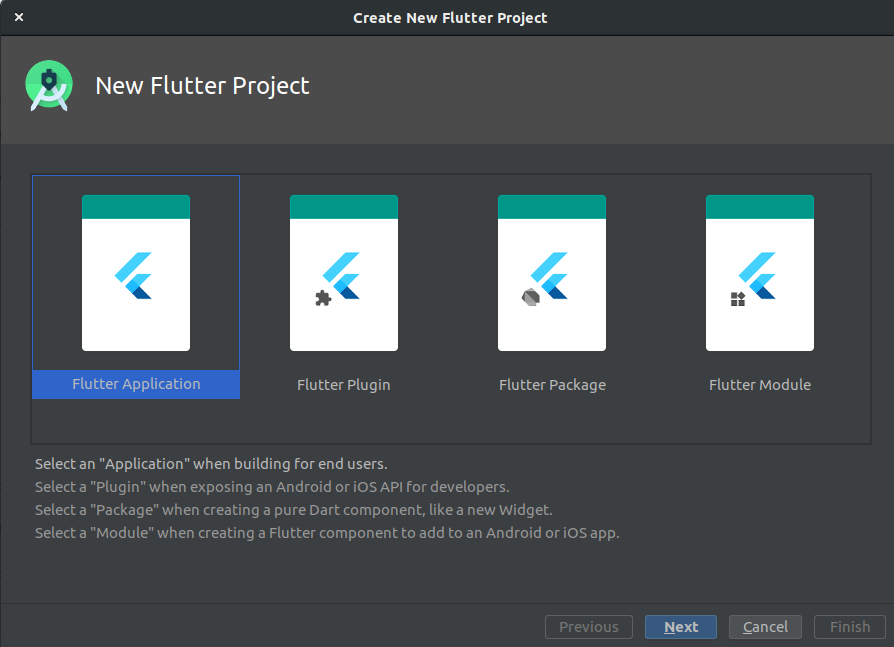
1. **Visual Studio Code** Untuk membuat flutter dapat menggunakan command melalui shortcut **Ctrl+Shift+P**, lalu pilih Flutter : New Project dan tunggu hingga proses selesai.



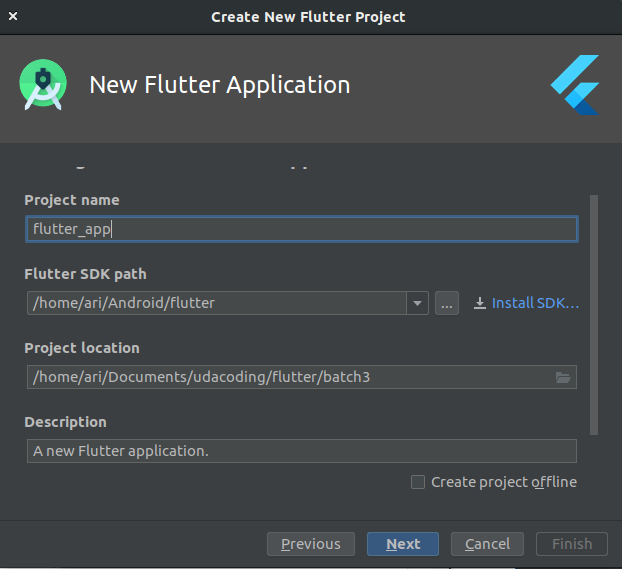
2. **Android Studio** ada beberapa langkah untuk membuat project flutter .



Pilih Start new Flutter project



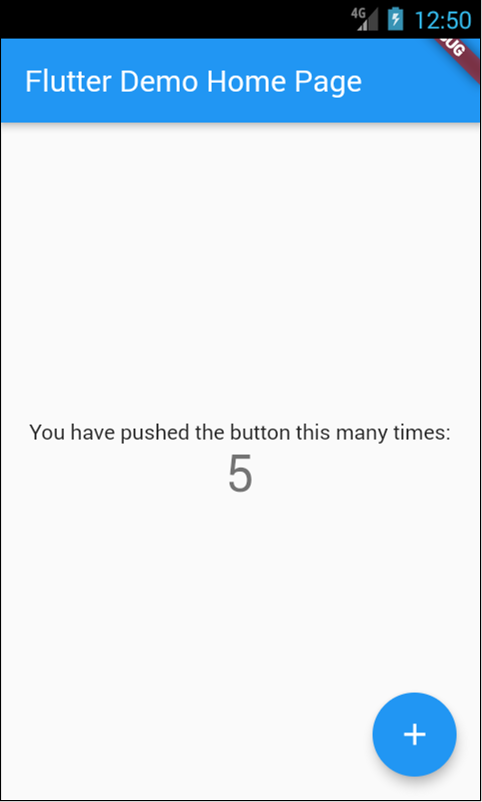
Klik Next pada Flutter Application



Buat nama project anda di Project name

Tentukan SDK anda pada Flutter SDK path

Pilih lokasi penyimpanan project anda pada Project Location



**A. Penjelasan Singkat**

import 'package:flutter/material.dart';

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

class **MyApp** extends **StatelessWidget** {

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Demo',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: MyHomePage(title: 'Flutter Demo Home Page'),

);

}

}

Pada kode tersebut kita melakukan import material.dart fungsinya agar bisa menggunakan widget-widget yang telah disediakan.

Lalu pada fungsi main() kita memanggil class MyApp dan melakukan extend StatelessWidget

class **MyHomePage** extends **StatefulWidget** {

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

final String title;

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

class **\_MyHomePageState** extends **State**<**MyHomePage**> {

int \_counter = 0;

void \_incrementCounter() {

setState(() {

\_counter++;

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(widget.title),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'You have pushed the button this many times:',

),

Text(

'$\_counter',

style: Theme.of(context).textTheme.display1,

),

], ), ),

floatingActionButton: FloatingActionButton(

onPressed: \_incrementCounter,

tooltip: 'Increment',

child: Icon(Icons.add),

), // This trailing comma makes auto-formatting nicer for build

methods.

);

}

}

Pada class MyApp kita mengembalikan/return MaterialApp(), MaterialApp merupakan pondasi dari aplikasi kita berjalan nantinya. Lalu didalamnya terdapat property seperti title, theme, dan home, pada title kita isikan dengan nama aplikasi kita, pada theme kita bisa mengatur tema dari aplikasi kita, lalu pada home inilah kita mengisikan halaman awal pada saat aplikasi dijalankan, disini kita memanggil MyHomePage sebagai halaman awal kita. Pada MyHomePage kita memanggil \_MyHomePageState dan mererturn sebuah Scaffold, scaffold adalah Widget yang bisa kita isi dengan tampilantampilan umum pada aplikasi android seperti dukungan AppBar, bagian body, ataupun floating action button.

**B. Perbedaan Stateless dan Stateful**

Pada flutter ada dua widget yang akan sangat sering digunakan yaitu Stateless Widget dan StatefulWidget, lalu apa beda keduanya ?

* **StatefullWidget** adalah widget yang digunakan untuk menampilkan datadata yang dinamis atau datanya mengalami perubahan, seperti data dari database atau semisalnya.
* **StalessWidget** adalah widget yang berfungsi untuk menampilkan hal-hal yang sifatnya statis.

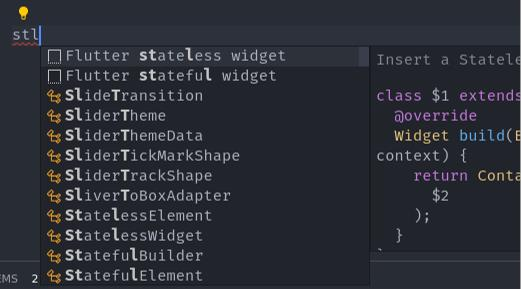
Sebagai contoh pada saat membuat project baru, maka sudah ada kode default yang apabila di run maka akan tampil seperti pada gambar sebelumnya

Pada aplikasi flutter tersebut terdapat 1 buah tombol dan terdapat text yang akan berubah dan melakukan penambahan angkanya pada saat mengklik tombolnya, hal ini bisa dilakukan apabila kita menggunakan Stateful widget, pada stateful widget terdapat fungsi setState yang artinya kita mengupdate sebuah nilai dari suatu variabel dan widget yang menggunakan atau menampilkan nilai tersebut akan di update tampilannya.

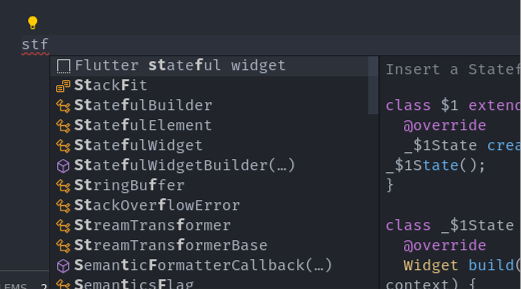
Sedangkan pada stateless tidak terdapat setState artinya widget-widget yang tampil akan statis.

Ada cara yang mudah untuk membuat stateless dan statefull, caranya cukup mudah yaitu dengan menggunakan autocomplete yang telah disediakan yaitu dengan mengetikkan:

**stl** = untuk stateless lalu enter



**stf** = untuk statefull lalu enter



**C. Property Child dan Children**

**1. Child**

Sesuai dengan namanya child, dalam bahasa indonesia artinya adalah anak berarti widget tersebut hanya bisa memiliki 1 widget di bawahnya.

Sebagai contoh :

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Property Child"),

),

body: Container(

child: Text("Halo !!!"),

color: Colors.yellow,

padding: EdgeInsets.all(16.0),

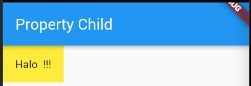
),

);

}

}

Hasilnya seperti berikut



Container memiliki properti child, sehingga hanya bisa menampung satu buah widget.

**2. Children**

Children atau anak-anak (jamak) berarti widget tersebut bisa berisi dengan banyak widget.

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Property Children"),

),

body: Column(

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

],

));

}

Hasilnya seperti berikut



**D. Widget Layout**

Untuk mengatur tata letak Terdapat dua widget yaitu Row dan Column

**1. Row**

Pada Row widget-widget akan tampil dengan arah horizontal atau sebaris, widget Row menggunakan property children artinya widget ini bisa diisi oleh banyak widget.

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Row"),

),

body: Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

],

));

}

}

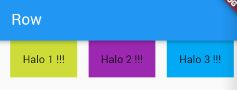
Hasilnya seperti berikut



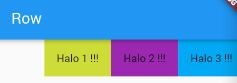
Kita juga bisa menambahkan pengaturan posisi menggunakan mainAxisAlignment, ada beberapa opsi diantaranya :

1. MainAxisAlignment.spaceBetween = akan tampil seperti diatas.

2. MainAxisAlignment.spaceEvenly = akan seperti berikut :



3. mainAxisAlignment: MainAxisAlignment.end = akanseperti berikut :



4. dan masih ada yang lain seperti center, start, end

**2. Column**

Sedangkan pada Column berlaku sebaliknya widget akan mengarah secara vertikal atau lurus kebawah.

Sebagai contoh :

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Column"),

),

body: Column(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

],

));

}

}

Hasilnya seperti berikut



Kita juga bisa mengatur mainAxisAlignment, bedanya hanya pada arahnya saja.

**7. Widget Umum**

**A. Center**

Center berfungsi untuk memposisikan widget yang kita buat berada ditengah.

...

body: Center(

child: Text("Text di tengah",

style: TextStyle(

fontSize: 30.0,

fontWeight: FontWeight.bold),

),

)

...

Hasilnya seperti berikut



**B. Text**

Text berfungsi untuk menampilkan sebuah teks, atau bisa kita berikan style dengan menambahkan property style.

body: Text('Ini Text', style: TextStyle(

color: Colors.blue,

backgroundColor: Colors.pink,

fontSize: 20.0,

fontStyle: FontStyle.italic,

fontWeight: FontWeight.bold

),)

Hasilnya seperti berikut



**C. Icon**

Widget ini untuk menggunakan icon yang telah disediakan, berikut adalah contohnya.

body: Container(

padding: EdgeInsets.all(16.0),

child: Row(

mainAxisAlignment: MainAxisAlignment.spaceEvenly,

children: <Widget>[

Column(

children: <Widget>[

Icon(Icons.access\_alarm),

Text('Alarm')

],

),

Column(

children: <Widget>[

Icon(Icons.phone),

Text('Phone')

],

),

Column(

children: <Widget>[

Icon(Icons.book),

Text('Book')

],

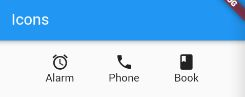
),

],

),

)

Hasilnya seperti berikut



**D. Container**

Container merupakan widget yang fungsinya untuk membungkus widget lain sehingga dapat diberikan nilai seperti margin, padding, warna background, atau dekorasi.

Hasilnya seperti berikut

...

body: Container(

padding: EdgeInsets.all(32.0),

margin: EdgeInsets.fromLTRB(20.0, 10.0, 20.0, 0),

decoration: BoxDecoration(

borderRadius: BorderRadius.circular(20.0),

color: Colors.purple),

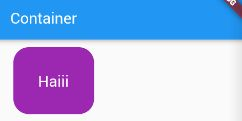
// color: Colors.purple,

child: Text('Haiii', style: TextStyle(color:

Colors.white,fontSize: 20.0),)

)

...



**E. SizedBox**

fungsi ini yaitu untuk menambahkan jarak baik secara vertikal atau horizontal tergantung nilai yang kita tentukan.

...

body: Column(

children: <Widget>[

Text("A", style: TextStyle(fontSize: 30.0),),

SizedBox(height: 20.0,),

Text("B", style: TextStyle(fontSize: 30.0),)

],

)

...

Hasilnya seperti berikut



**F. Button**

Terdapat 3 widget Button yang umumnya dipakai yaitu RaisedButton, MaterialButton, dan FlatButton

* Pada raised button dan Material tombol akan sedikit menonjol.
* Pada flat button tombolnya akan lebih datar tanpa adanya efek-efek seperti bayangan dan lain-lain.

...

body: Column(

children: <Widget>[

RaisedButton(

color: Colors.amber,

child: Text("Raised Button"),

onPressed: () {},

),

MaterialButton(

color: Colors.lime,

child: Text("Material Button"),

onPressed: () {},

),

FlatButton(

color: Colors.lightGreenAccent,

child: Text("FlatButton Button"),

onPressed: () {},

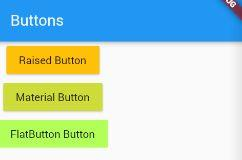
),

],

)

...

Hasilnya seperti berikut



**G. TextFormField**

TextFormField merupakan widget yang berguna untuk membuat form untuk diisi user.

...

body: Padding(

padding: const EdgeInsets.all(8.0),

child: Form(

child: Column(

children: <Widget>[

TextFormField(

decoration: InputDecoration(hintText: "Username"),

),

TextFormField(

obscureText: true,

decoration: InputDecoration(hintText: "Password"),

),

RaisedButton(

child: Text("Login"),

onPressed: () {},

)

],

),

),

)

...

Hasilnya seperti berikut



**H. InkWell**

Inkwell berguna untuk menambahkan action pada widget seperti action on Tap misalnya :



...

body: Padding(

padding: const EdgeInsets.all(8.0),

child: InkWell(

onTap: () {

print("Ditekan");

},

child: Container(

width: 100.0,

height: 100.0,

color: Colors.pink,

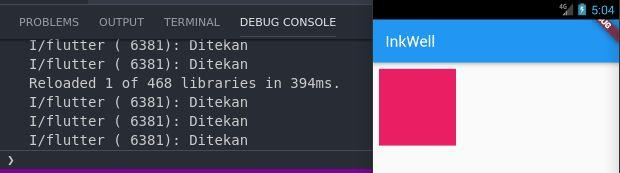
),

)

)

...

Hasilnya seperti berikut



**8. Navigator**

Untuk berpindah-pindah halaman maka kita menggunakan Navigator.

**A. Navigator Push**

Untuk pindah halaman, kita bisa menggunakan Navigator.push, karena tampilan yang di push akan berada di lapisan teratas, sehingga akan tampil.

sebagai contoh : edit ​**main.dart​**  lalu isikan seperti berikut :

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

import​ ​ 'package​ :module\_play\_ground/ui\_view/​ PageTwo​ .dart';

void main() => runApp(​ MyApp​ ());

class​ ​ MyApp​ ​ extends​ ​ StatelessWidget​ {

​ *// This widget is the root of your application.*

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ MaterialApp​ (

title: ​ 'Flutter​ ​Demo​ ',

theme: ​ ThemeData​ (

primarySwatch: ​ Colors​ .blue,

),

home: ​ MyHomePage​ (),

);

}

}

class​ ​ MyHomePage​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​"Navigator"​ ),

),

body: ​ Row​ (

mainAxisAlignment: ​ MainAxisAlignment​.center,

children: <​ Widget​>[

​ MaterialButton​ (

color: ​ Colors​.yellow,

child: ​ Text​ (​"Page 2"​ ),

onPressed: () {

​ *// dibuat berikutnya*

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) => ​ PageTwo​ ()

)

);

},

),

],

)

);

}

}

**B. Navigator Pop**

Sedangkan pop, kita menghilangkan halaman lapisan teratas, sehingga akan kembali ke halaman sebelumnya .

buat folder dengan nama ui\_view lalu buat file PageTwo.dart dan isikan seperti berikut :

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

class​ ​ PageTwo​ ​ extends​ ​StatelessWidget​ {

​ @override

Widget build(BuildContext context) {

​ return​ Scaffold(

appBar: AppBar(

title: Text(​ "Page Two"​ ),

),

body: Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

MaterialButton(

color: Colors.lightBlue,

child: Text(​ "Kembali menggunakan Pop"​ ),

onPressed: () {

Navigator.pop(context);

},

),

],

)

);

}

}

hasil akhir dia akan bisa buka halaman baru dan kembali.



**C. Navigator Pushnamed**

Navigator menggunakan pushNamed artinya kita menggunakan nama yang telah didefinisikan diawal.

caranya yaitu dengan mengisi properti routes:

class​ ​ MyApp​ ​ extends​ ​ StatelessWidget​ {

​ // This widget is the root of your application.

​ @override

Widget build(BuildContext context) {

​ return​ MaterialApp(

title: ​ 'Flutter Demo'​ ,

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: MyHomePage(),

routes: <String, WidgetBuilder>{

​ '/page2'​ : (BuildContext context) => PageTwo(),

},

);

}

}

dan untuk menjalankannya cukup dengan :

...

onPressed: () {

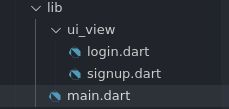
Navigator.of(context).pushNamed(​ '/page2'​ );

},

...

**D. Latihan membuat halaman Login dan Register**

Berikut adalah strukturnya



1. Pada main.dart, isikan seperti berikut :

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

import​ ​ 'package:login\_flutter/ui\_view/login.dart'​ ;

import​ ​ 'package:login\_flutter/ui\_view/signup.dart'​ ;

void​ main() => runApp(MyApp());

class​ ​ MyApp​ ​ extends​ ​ StatelessWidget​ {

​ // This widget is the root of your application.

​ @override

Widget build(BuildContext context) {

​ return​ MaterialApp(

title: ​ 'Login Register'​ ,

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: HomePage(),

);

}

}

class​ ​ HomePage​ ​ extends​ ​ StatelessWidget​ {

​ @override

Widget build(BuildContext context) {

​ return​ Scaffold(

backgroundColor: Colors.lightBlue,

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Icon(Icons.android, color: Colors.white,size: 45,),

SizedBox(height: 200,),

Text(​ "Welcome to Flutter"​ , style: TextStyle(color:

Colors.white, fontSize: 22)),

SizedBox(height: 10,),

Text(​ "Get real-time updates about what"​ , style:

TextStyle(color: Colors.white, fontSize: 18)),

Text(​ "maters to you"​ , style: TextStyle(color:

Colors.white, fontSize: 18)),

SizedBox(height: 20,),

MaterialButton(

minWidth: 210,

color: Colors.white,

textColor: Colors.lightBlue,

child: Text(​ "Sign Up"​ , style: TextStyle(fontWeight:

FontWeight.bold, fontSize: 18),),

onPressed: (){

Navigator.push(context, MaterialPageRoute(builder:

(context) => SignUp()));

},

),

FlatButton(

child: Text(​ "Log in"​ , style: TextStyle(color:

Colors.white, fontWeight: FontWeight.bold,fontSize: 18),),

onPressed: () {

Navigator.push(context, MaterialPageRoute(builder:

(context) => Login()));

},

)

],

),

),

);

}

}

2. Pada login.dart, isikan seperti berikut :

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

import​ ​ 'package:login\_flutter/ui\_view/signup.dart'​ ;

class​ ​ Login​ ​ extends​ ​ StatelessWidget​ {

​ @override

Widget build(BuildContext context) {

​ return​ Scaffold(

backgroundColor: Colors.lime,

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Container(

padding: EdgeInsets.all(10),

width: 250,

color: Colors.white,

child: TextFormField(

decoration: InputDecoration(

hintText: ​ "Email"​ ,

border: InputBorder.none

),

),

),

SizedBox(height: 10,),

Container(

padding: EdgeInsets.all(10),

width: 250,

color: Colors.white,

child: TextFormField(

decoration: InputDecoration(

hintText: ​ "Password"​ ,

border: InputBorder.none

),

),

),

SizedBox(height: 20,),

MaterialButton(

padding: EdgeInsets.all(20),

minWidth: 250,

color: Colors.white,

child: Text(​ "Login"​ , style: TextStyle(color:

Colors.lightBlue),),

onPressed: () {},

),

FlatButton(

child: Text(​ "Not a member? signup now"​ , style:

TextStyle(color: Colors.white),),

onPressed: () {

Navigator.push(context, MaterialPageRoute(builder:

(context) => SignUp()));

},

)

],

),

),

);

}

}

3. pada signup.dart, isikan seperti berikut :

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

import​ ​ 'package:login\_flutter/ui\_view/login.dart'​ ;

class​ ​ SignUp​ ​ extends​ ​ StatelessWidget​ {

​ @override

Widget build(BuildContext context) {

​ return​ Scaffold(

backgroundColor: Colors.amber,

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Container(

padding: EdgeInsets.all(10),

color: Colors.black87,

width: 250,

child: TextFormField(

decoration: InputDecoration(

hintText: ​ "Fullname"​ ,

hintStyle: TextStyle(color: Colors.white),

border: InputBorder.none

),

),

),

SizedBox(height: 10,),

Container(

padding: EdgeInsets.all(10),

color: Colors.black87,

width: 250,

child: TextFormField(

decoration: InputDecoration(

hintText: ​ "Email"​ ,

hintStyle: TextStyle(color: Colors.white),

border: InputBorder.none

),

),

),

SizedBox(height: 10,),

Container(

padding: EdgeInsets.all(10),

color: Colors.black87,

Hasilnya seperti berikut :

width: 250,

child: TextFormField(

decoration: InputDecoration(

hintText: ​ "Password"​ ,

hintStyle: TextStyle(color: Colors.white),

border: InputBorder.none

),

),

),

SizedBox(height: 20,),

MaterialButton(

padding: EdgeInsets.all(20),

minWidth: 250,

child: Text(​ "REGISTER"​ , style: TextStyle(color:

Colors.white),),

color: Colors.black,

onPressed: () {},

),

FlatButton(

child: Text(​ "Already registered! Login me"​ , style:

TextStyle(color: Colors.white),),

onPressed: () {

Navigator.push(context, MaterialPageRoute(builder:

(context) => Login()));

},

)

],

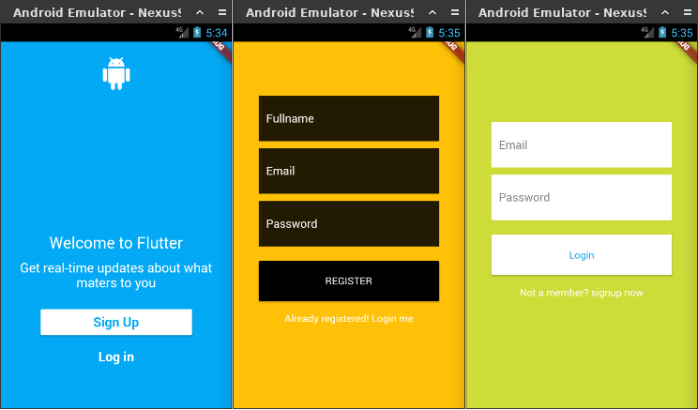
),

),

);

}

}



**9. Notification Widget**

Kali ini kita akan membuat notification widget, yaitu Toast, AlertDialog, dan SnackBar, untuk memakai nya kita perlu menginstall library yang bisa dilihat melalui website ​https://pub.dev/

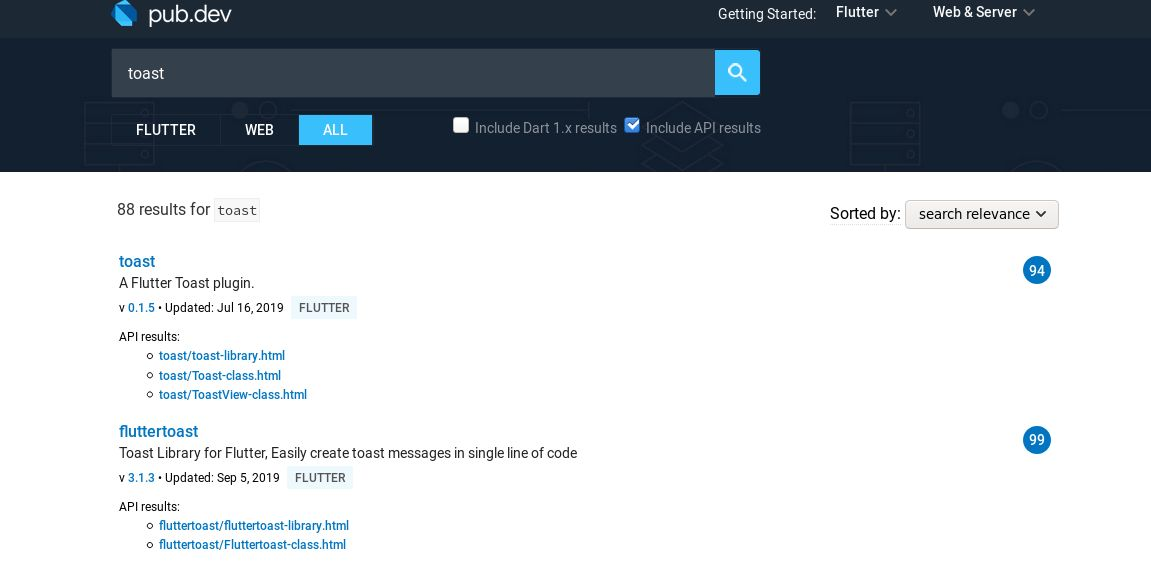


**A. Toast**

Untuk menggunakan toast, ada dua package yaitu toast, dan fluttertoast, bedanya toast lebih

sederhana sedangkan fluttertoast ada konfigurasi yang kita bisa atur:

1. Cari flutter :



2. Coba kita gunakan yang fluttertoast, cara installnya terdapat pada tab Installing



3. taruh ​ **fluttertoast: ^3.1.3​** , pada pubspec.yaml seperti berikut :

Pastikan posisinya sejajar dengan flutter.



lalu jalankan ​**flutter pub get**

4. import package-nya



lalu kita isikan codenya sesuai dengan panduannya.

body:​ Container(

color:​ Colors.red,

child:​ MaterialButton(

child:​ Text(​ "Toast"​ ),

textColor:​ Colors.white,

onPressed:​ () {

Fluttertoast.showToast(

msg:​ ​ "This is Center Short Toast"​ ,

toastLength:​ Toast.LENGTH\_SHORT,

gravity:​ ToastGravity.CENTER,

timeInSecForIos:​ ​ 1​ ,

backgroundColor:​ Colors.green,

textColor:​ Colors.white,

fontSize:​ ​ 16.0

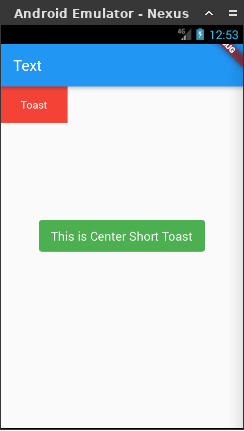
);

},

),

)

Hasilnya :

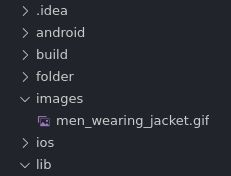


**B. Alert Dialog**

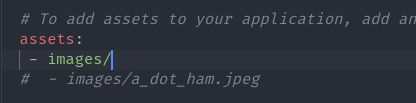
Untuk alert dialog kita gunakan giffy\_dialog, caranya :

* 1. buka ​ https://pub.dev/packages/giffy\_dialog​ , lalu taruh di pubspec.yaml dan import.
  2. Setelah itu coba download gambar ini

https://github.com/xsahil03x/giffy\_dialog/blob/master/example/assets/men\_wearing\_jacket.gif​ , lalu buat folder images dan taruh gambarnya seperti berikut :



* 1. Daftarkan asset images di pubspec.yaml



4. lalu masukan kode berikut :

Container(

​ color​ : Colors.red,

child: MaterialButton(

​ child​ : Text(​ "Alert Dialog"​ ),

textColor: Colors.white,

onPressed: () {

showDialog(

​ context​ : context,builder: (​ \_​ ) =>

NetworkGiffyDialog(

​ image​ :

Image.asset('assets/men\_wearing\_jacket.gif'),

title: Text('Men Wearing Jackets',

style: TextStyle(

​ fontSize​ : ​ 22.0​ , fontWeight:

FontWeight.w600),

),

description: Text('This is a men wearing

jackets',

textAlign: TextAlign.center,

style: TextStyle(),

),

onOkButtonPressed: () {},

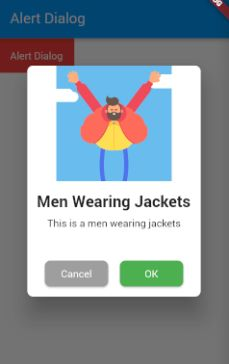
) )​ *;*

},

),

)

hasilnya :



**C. SnackBar**

Untuk membuat snackbar kita perlu menambahkan Globalkey, caranya seperti berikut :

class​ ​ \_MyHomePageState​ ​ extends​ ​ State<MyHomePage>​ {

​ GlobalKey​ <​ ScaffoldState​ > \_key = ​ GlobalKey​ <​ ScaffoldState​ >();

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

key: \_key,

appBar: ​ AppBar​ (title: ​ Text​ (​ "SnackBar"​ ),),

body: ​ Container​ (

color: ​ Colors​ .red,

child: ​ MaterialButton​ (

child: ​ Text​ (​ "Snack Bar"​ ),

textColor: ​ Colors​ .white,

onPressed: () {

\_key.currentState.showSnackBar(​ SnackBar​ (content:

Text​ (​ 'Snack​ bar'),));

},

),

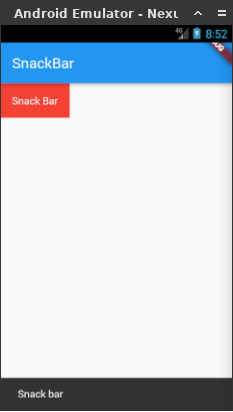
),

);

}

}

Lalu hasilnya :



**10. Retrieve Data**

**A. Passing dengan TextEditingController**

Untuk mengambil isi data pada sebuah form text kita membutuhkan TextEditingController  dan kita lakukan setState, lalu kita tampilkan dibawah seperti berikut :

class​ ​ SimpleLogin​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_SimpleLoginState createState() => \_SimpleLoginState();

}

class​ ​ \_SimpleLoginState​ ​ extends​ ​ State<SimpleLogin>​ {

​ final​ ​ GlobalKey​ <​ ScaffoldState​ > \_key = ​ GlobalKey​ <​ ScaffoldState​ >();

​ TextEditingController​ usernameController = ​ TextEditingController​ ();

​ TextEditingController​ passwordController = ​ TextEditingController​ ();

​ String​ username = ​ ""​ , password = ​ ""​ ;

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

key: \_key,

appBar: ​ AppBar​ (

backgroundColor: ​ Colors​ .green,

title: ​ Text​ (​ 'Logi​ n'),

),

body: ​ Padding​ (

padding: const ​ EdgeInsets​ .all(​ 8.0​ ),

child: ​ Center​ (

child: ​ Column​ (

children: <​ Widget​ >[

​ TextFormField​ (

controller: usernameController,

decoration: ​ InputDecoration​ (

fillColor: ​ Colors​ .greenAccent,

hintText: ​ 'usernam​ e'

),

),

​ SizedBox​ (height: ​ 8.0​ ,),

​ TextFormField​ (

controller: passwordController,

Hasilnya :

obscureText: ​ true​ ,

decoration: ​ InputDecoration​ (

fillColor: ​ Colors​ .greenAccent,

hintText: ​ 'passwor​ d'

),

),

​ SizedBox​ (height: ​ 8.0​ ,),

​ Container​ (

color: ​ Colors​ .green,

child: ​ MaterialButton​ (

child: ​ Text​ (​ "submit"​ ),

textColor: ​ Colors​ .white,

onPressed: () {

setState(() {

username = usernameController.text;

password = passwordController.text;

});

},

),

),

​ SizedBox​ (height: ​ 16.0​ ,),

​ Text​ (username),

​ Text​ (password),

],

)

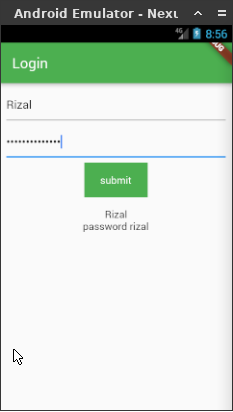
),

)

);

}

}



**B. Passing dengan Constructor**

Untuk membuat constructor dengan parameter, kita, cukup membuatnya seperti ini :

class​ ​ PageTwo​ ​ extends​ ​ StatelessWidget​ {

​ String​ msg;

// wajib diisi

​ PageTwo​ (​ this​ .msg);

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ "Page Two"​ ),

),

body: ​ Column​ (

children: <​ Widget​ >[

​ Text​ (msg)

],

)

);

}

}

atau seperti ini artinya opsional

class​ ​ PageTwo​ ​ extends​ ​ StatelessWidget​ {

​ String​ msg = ​ ""​ ;

​ PageTwo​ (​ {this​ .msg​ }​ );

...

Cara memanggilnya seperti ini

PageTwo​ (​ "Pesan terkirim"​ )

untuk yang opsional seperti ini

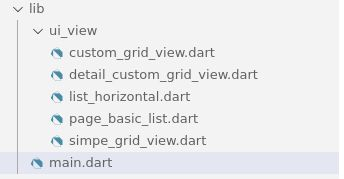
PageTwo​ (msg: ​ "Pesan terkirim"​ )

hasilnya apabila dipanggil :

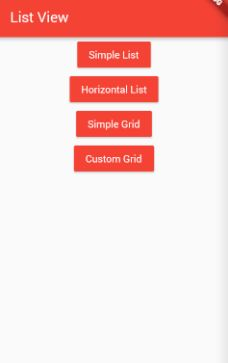


**11. List View**

Kali ini kita akan membuat project yang menampilkan daftar list, pertama buat dulu strukturnya seperti berikut :



halaman-halaman tersebut nanti kita buka melalui tombol-tombol di halaman main.dart :



**A. Simple List**

edit page\_basic\_list.dart :

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

class​ ​ PageBasicList​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Basic​ ​ List​ '),

backgroundColor: ​ Colors​ .red,

),

body: ​ ListView​ (

children: <​ Widget​ >[

​ Padding​ (

padding: ​ EdgeInsets​ .all(​ 8.0​ ),

child: ​ ListTile​ (

leading: ​ Icon​ (​ Icons​ .access\_alarm),

title: ​ Text​ (​ 'Alar​ m'),

),

),

​ ListTile​ (

leading: ​ Icon​ (​ Icons​ .phone),

title: ​ Text​ (​ 'Phon​ e'),

),

​ ListTile​ (

leading: ​ Icon​ (​ Icons​ .camera),

title: ​ Text​ (​ 'Camer​ a'),

),

​ ListTile​ (

leading: ​ Icon​ (​ Icons​ .message),

title: ​ Text​ (​ 'Messag​ e'),

),

],

),

);

}

}

**B. Horizontal List**

pada list\_horizontal.dart kita isikan seperti berikut :

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

class ListHorizontal extends StatelessWidget {

@override

Widget build(BuildContext context) {

​ return​ Scaffold(

appBar: AppBar(

title: Text(​ 'List Horizontal'​ ),

backgroundColor: Colors.amberAccent,

),

body: Container(

​ *//margin : jarak antar widget*

​ *// padding : jarak di dalam objek*

margin: EdgeInsets.symmetric(vertical: ​ 20.0​ ),

​ height​ : ​ 200.0​ ,

child: ListView(

​ *// vertical kebawah*

scrollDirection: Axis.horizontal,

children: <Widget>[

Container(

​ width​ : ​ 160.0​ ,

​ height​ : ​ 200.0​ ,

​ color​ : Colors.​ red​ ,

),

Container(

​ width​ : ​ 160.0​ ,

​ height​ : ​ 200.0​ ,

​ color​ : Colors.​ blue​ ,

),

Container(

​ width​ : ​ 160.0​ ,

​ height​ : ​ 200.0​ ,

​ color​ : Colors.yellow,

),

Container(

​ width​ : ​ 160.0​ ,

​ height​ : ​ 200.0​ ,

​ color​ : Colors.orange,

),

Container(

​ width​ : ​ 160.0​ ,

​ height​ : ​ 200.0​ ,

​ color​ : Colors.​ green​ ,

)

],

),

),

);

}

}

**C. Simple Grid**

untuk simple grid kita isikan seperti berikut :

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

class​ ​ SimpleGridView​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Simple​ ​ Grid​ ​ Veiw​ '),

backgroundColor: ​ Colors​ .amber,

),

body: ​ GridView​ .count(

crossAxisCount: ​ 3​ ,

children: ​ List​ .generate(​ 12​ , (index) {

int nIndex = index + ​ 1​ ;

​ String​ dataIndex = ​ "$nIndex"​ ;

​ return​ ​ Center​ (

child: ​ Container​ (

margin: ​ EdgeInsets​ .all(​ 10​ ),

color: ​ Colors​ .amber,

height: ​ 100.0​ ,

width: ​ 100.0​ ,

child: ​ Text​ (

​ 'Data​ ​ Ke​ - $dataIndex', style:

Theme​ .of(context).textTheme.headline,

),

)

);

}),

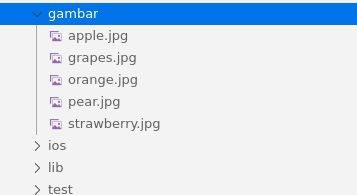
),

);

}

}

**D. Custom Grid List**

Sebelum itu kita tambahkan dulu gambar-gambarnya, dan daftarkan di pubspec.yaml :

lalu untuk custom grid list kita isikan seperti berikut :

**import ​'package:flutter/cupertino.dart'​**;

**import ​'package:flutter/material.dart'​**;

**import ​'package:flutterweek3/screen/DetailCustumGridview.dart'​**;

**class ​**PageCustomGridview ​**extends ​**StatefulWidget {

​@override

​\_PageCustomGridviewState createState() => ​\_PageCustomGridviewState​();

}

**class ​**\_PageCustomGridviewState ​**extends ​**State<PageCustomGridview> {

​*//deklarasi data list*

*​*List<Container> ​**daftarMakananPadang ​**= ​**new ​**List​();

 ​**var ​itemMakanan ​**= [

{​**"nama" ​**: ​**"Gulai Banak"​**, ​**"gambar" ​**: ​**"gulai-banak.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan Gulai**

**Banak"​**},

{​**"nama" ​**: ​**"Kalio Daging"​**, ​**"gambar" ​**: ​**"kalio-dagiang.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan Kalio**

**Dagiang"​**},

{​**"nama" ​**: ​**"Kalio Jariang"​**, ​**"gambar" ​**: ​**"kalio-jariang.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan Kalio**

**Jariang"​**},

{​**"nama" ​**: ​**"Lompong Sagu"​**, ​**"gambar" ​**: ​**"lompong-sagu.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan**

**Lompong Sagu"​**},

{​**"nama" ​**: ​**"Sala Lauk"​**, ​**"gambar" ​**: ​**"sala-lauak.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan Sala Lauk"​**},

{​**"nama" ​**: ​**"Soto Padang"​**, ​**"gambar" ​**: ​**"soto-padang.jpg"​**, ​**"Keterangan" ​**: ​**"Ini Makanan Soto**

**Padang"​**},

];

​*//method*

*​*\_buatDataListMakanan() ​**async​**{

​**for​**(​**var ​**i=​0​; i<​**itemMakanan​**.​**length​**; i++){

​**final ​**dataMakanan = ​**itemMakanan​**[i]; ​*//variable untuk data makanan*

*​***final ​**String gambarMakanan = dataMakanan[​**"gambar"​**];

​**daftarMakananPadang​**.add(​**new ​**Container​(

padding: ​EdgeInsets​.​all​(​10​),

child: ​Card​(

child: ​InkWell​(

onTap: (){

​*//pindah ke detail*

*​*Navigator.​*push​*(​**context​**, ​MaterialPageRoute​(builder: (context)=>

PageDetailCustomGridview​(

nama: dataMakanan[​**"nama"​**],

gambar: gambarMakanan,

keterangan: dataMakanan[​**"Keterangan"​**],

)));

},

child: ​Column​(

children: [

​Hero​(

tag: dataMakanan[​**"nama"​**],

child: ​Image​.​asset​(​**'gambar/​**$gambarMakanan​**'​**, height: ​85.0​,width: ​125​, fit:

BoxFit.​**contain​**,),

),

​Padding​(

padding: ​EdgeInsets​.​all​(​10​),

),

​Text​(dataMakanan[​**'nama'​**], style: ​TextStyle​(fontSize: ​14.0​, fontWeight:

FontWeight.​*bold​*, color: Colors.​*deepOrange​*),)

],

),

),

),

));

}

}

​*//proses background*

@override

​**void ​**initState() {

​*// ​****TODO: implement initState***

***​*super​**.initState();

\_buatDataListMakanan();

}

​*//menampilkan ke view*

*​*@override

​Widget build(BuildContext context) {

​**return ​**Scaffold​(

appBar: ​AppBar​(

title: ​Text​(​**'Apps Makanan Padang'​**),

backgroundColor: Colors.​*deepOrange​*,

),

body: ​GridView​.​count​(crossAxisCount: ​2​, children: ​**daftarMakananPadang​**,),

);

}

}

Lalu kita tambahkan halaman detailnya :

**import** ​ **'package:flutter/material.dart'**​ ;

**class** ​ PageDetailCustomGridview ​ **extends** ​ StatefulWidget {

String ​ **nama**​ , ​ **gambar**​ , ​ **keterangan**​ ;

PageDetailCustomGridview({​ **this**​ .​ **nama**​ , ​ **this**​ .​ **gambar**​ , ​ **this**​ .​ **keterangan**​ });

​ @override

​ \_PageDetailCustomGridviewState createState() =>

​ \_PageDetailCustomGridviewState​ ();

}

**class** ​ \_PageDetailCustomGridviewState ​ **extends** ​ State<PageDetailCustomGridview>

{

​ @override

​ Widget build(BuildContext context) {

​ **return** ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ **'**​ ${​ **widget**​ .​ **nama**​ }​ **'**​ ),

backgroundColor: Colors.​ *deepOrange*​,

),

body: ​ ListView​ (

children: [

Container​ (

height: ​ 240.0​ ,

child: ​ Hero​ (

tag: ​ **widget**​ .​ **nama**​ ,

child: ​ Material​ (

child: ​ InkWell​ (

child: ​ Image​ .​ asset​ (

​ **'gambar/**​ ${​ **widget**​ .​ **gambar**​ }​ **'**​ ,

fit: BoxFit.​ **contain**​ ,

),

),

),

),

),

​ BagianNama​ (

nama: ​ **widget**​ .​ **nama**​ ,

),

​ BagianKeterangan​ (

keterangan: ​ **widget**​ .​ **keterangan**​ ,

)

],

),

);

}

}

**class** ​ BagianNama ​ **extends** ​ StatelessWidget {

BagianNama({​ **this**​ .​ **nama**​ });

String ​ **nama**​ ;

​ @override

​ Widget build(BuildContext context) {

​ **return** ​ Container​ (

padding: ​ EdgeInsets​ .​ all​ (​ 10​ ),

child: ​ Row​ (

children: [

​ Expanded​ (

child: ​ Column​ (

crossAxisAlignment: CrossAxisAlignment.​ **start**​ ,

children: [

​ Text​ (

​ **nama**​ ,

style: ​ TextStyle​ (fontSize: ​ 20​ , color: Colors.​ *brown*​ ),

),

​ Text​ (

**'**​ $​ **nama**​ **\@**​ **gmail.com'**​ ,

style: ​ TextStyle​ (

fontSize: ​ 17.0​ ,

color: Colors.​ *green*​ ,

fontWeight: FontWeight.​ *bold*​),

)

],

),

),

​ Row​ (

children: [

​ Icon​ (

Icons.​ *star*​ ,

size: ​ 40.0​ ,

color: Colors.​ *deepOrange*​,

),

​ Text​ (

​ **"14"**​ ,

style: ​ TextStyle​ (fontSize: ​ 18.0​ ),

)

],

)

],

),

);

}

}

**class** ​ BagianKeterangan ​ **extends** ​ StatelessWidget {

String ​ **keterangan**​ ;

BagianKeterangan({​ **this**​ .​ **keterangan**​ });

​ @override

​ Widget build(BuildContext context) {

​ **return** ​ Container​ (

padding: ​ EdgeInsets​ .​ all​ (​ 10​ ),

child: ​ Card​ (

child: ​ Padding​ (

padding: ​ EdgeInsets​ .​ all​ (​ 6​ ),

child: ​ Text​ (

​ **'**​ $​ **keterangan**​ **'**​ , style: ​ TextStyle​ (fontSize: ​ 14.0​ ) ,textAlign:

TextAlign.​ **justify**​ ,

),

),

),

);

}

}

**E. Halaman Main**

pada halaman main kita buat tombol-tombol untuk membuat keempat list tersebut :

import​ ​ 'package​ :day3\_night/ui\_view/simpe\_grid\_view.dart';

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

import​ ​ 'ui\_view​ /page\_basic\_list.dart';

import​ ​ 'ui\_view​ /list\_horizontal.dart';

import​ ​ 'ui\_view​ /custom\_grid\_view.dart';

void main() => runApp(​ MyApp​ ());

class​ ​ MyApp​ ​ extends​ ​ StatelessWidget​ {

​ *// This widget is the root of your application.*

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ MaterialApp​ (

title: ​ 'Flutter​ ​ Demo​ ',

theme: ​ ThemeData​ (

primarySwatch: ​ Colors​ .blue,

),

home: ​ PageHome​ (),

);

}

}

class​ ​ PageHome​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'List​ ​ View​ '),

backgroundColor: ​ Colors​ .red,

),

body: ​ Center​ (

child: ​ Column​ (

children: <​ Widget​ >[

Container​ (

child: ​ MaterialButton​ (

onPressed: () {

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) => ​ PageBasicList​ ()

));

},

color: ​ Colors​ .red,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Simple​ ​ List​ '),

),

),

​ Container​ (

child: ​ MaterialButton​ (

onPressed: () {

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) => ​ ListHorizontal​ ()

));

},

color: ​ Colors​ .red,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Horizontal​ ​ List​ '),

),

),

​ Container​ (

child: ​ MaterialButton​ (

onPressed: () {

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) => ​ SimpleGridView​ ()

));

},

color: ​ Colors​ .red,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Simple​ ​ Grid​ '),

),

),

​ Container​ (

child: ​ MaterialButton​ (

onPressed: () {

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) => ​ CustomGridView​ ()

));

},

color: ​ Colors​ .red,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Custom​ ​ Grid​ '),

),

)

],

),

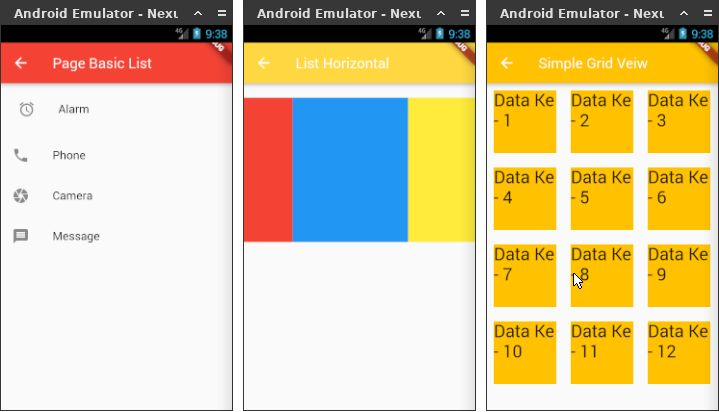
),

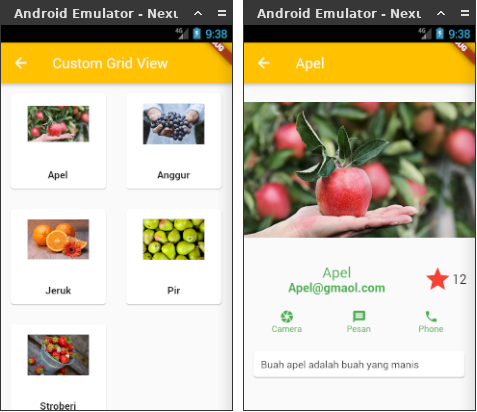
);

}

}

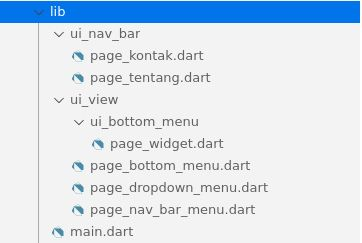
Dan hasilnya seperti berikut :





**12. Navigation Menu**

Untuk navigation menu kita buat struktur file seperti berikut :



**A. Navigation Drawer**

pada page\_nav\_bar\_menu.dart kita isikan seperti berikut :

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

class​ ​ PageNavBarMenu​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_PageNavBarMenuState createState() => \_PageNavBarMenuState();

}

class​ ​ \_PageNavBarMenuState​ ​ extends​ ​ State<PageNavBarMenu>​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Nav​ ​ Bar​ '),

backgroundColor: ​ Colors​ .brown,

),

drawer: ​ Drawer​ (

child: ​ ListView​ (children: <​ Widget​ >[

​ ListTile​ (

title: ​ Text​ (​ 'Welcom​ e'),

),

​ Divider​ (),

​ ListTile​ (

title: ​ Text​ (​ 'Tentan​ g'),

trailing: ​ Icon​ (​ Icons​ .info),

onTap: () {

​ Navigator​ .of(context).pop();

​ Navigator​ .of(context).pushNamed('/tentang');

},

),

ListTile​ (

title: ​ Text​ (​ 'Konta​ k'),

trailing: ​ Icon​ (​ Icons​ .phone),

onTap: () {

​ Navigator​ .of(context).pop();

​ Navigator​ .of(context).pushNamed('/kontak');

},

),

],),

),

body: ​ Center​ (

child: ​ Text​ (​ 'Home​ ​ Page​ ',

style: ​ TextStyle​ (fontWeight: ​ FontWeight​ .bold, fontSize:

30.0​ ))),

);

}

}

Akan ada dua halaman untuk kontak dan tentang, untuk halaman tentang kita isikan seperti

berikut :

page\_tentang.dart

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

class​ ​ PageTentang​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Tentan​ g'),

backgroundColor: ​ Colors​ .brown,

),

body: ​ Center​ (

child: ​ Text​ (​ 'Page​ ​ Halaman​ ​ Tentang​ ', style: ​ TextStyle​ (

fontSize: ​ 30.0

),),

),

);

}

}

page\_kontak.dart :

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

class​ ​ PageKontak​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Konta​ k'),

backgroundColor: ​ Colors​ .brown,

),

body: ​ Center​ (

child: ​ Text​ (​ 'Page​ ​ Halaman​ ​ Kontak​ ', style: ​ TextStyle​ (

fontSize: ​ 30.0

),),

),

);

}

}

**B. Bottom Bar Menu**

Selanjutnya kita buat halaman yang berisi Bottom bar, edit file page\_bottom\_menu.dart, lalu isikan seperti berikut :

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

import​ ​ 'ui\_bottom\_menu​ /page\_widget.dart';

class​ ​ PageHomeBottomMenu​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_PageHomeBottomMenuState createState() => \_PageHomeBottomMenuState();

}

class​ ​ \_PageHomeBottomMenuState​ ​ extends​ ​ State<PageHomeBottomMenu>​ {

int currentIndex = ​ 0​ ;

​ final​ ​ List​ <​ Widget​ > \_listColorMenu = [

​ PageWidget​ (​ Colors​ .blue),

​ PageWidget​ (​ Colors​ .deepOrange),

​ PageWidget​ (​ Colors​ .green)

];

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Bottom​ ​ Bar​ '),

backgroundColor: ​ Colors​ .brown,

),

body: \_listColorMenu[currentIndex],

bottomNavigationBar: ​ BottomNavigationBar​ (

onTap: onTabTapped,

currentIndex: currentIndex,

items: [

​ BottomNavigationBarItem​ (

icon: ​ Icon​ (​ Icons​ .home),

title: ​ Text​ (​ 'Hom​ e')

),

​ BottomNavigationBarItem​ (

icon: ​ Icon​ (​ Icons​ .message),

title: ​ Text​ (​ 'Messag​ e')

),

​ BottomNavigationBarItem​ (

icon: ​ Icon​ (​ Icons​ .person),

title: ​ Text​ (​ 'Perso​ n')

)

],

),

);

}

void onTabTapped(int index) {

setState(() {

currentIndex = index;

});

}

}

lalu pada page\_widget.dart ini kita menampilkan halaman berwarna, isikan seperti berikut :

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

class​ ​ PageWidget​ ​ extends​ ​ StatelessWidget​ {

​ final​ ​ Color​ color;

​ PageWidget​ (​ this​ .color);

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Container​ (

color: color,

);

}

}

**C. Dropdown Menu**

pada page\_dropdown\_menu.dart, kita isikan seperti berikut :

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

class​ ​ PageDropDownMenu​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_PageDropDownMenuState createState() => \_PageDropDownMenuState();

}

class​ ​ \_PageDropDownMenuState​ ​ extends​ ​ State<PageDropDownMenu>​ {

​ List​ <​ String​ > listKota = [​ "DKI Jakarta"​ , ​ "Tangerang"​ , ​"Bekasi"​ ,

"Bogor"​ , ​ "Bandung"​ ];

​ String​ nKota = ​ "DKI Jakarta"​ ;

int nilaiKota;

void pilihKota (​ String​ value) {

setState(() {

nKota = value;

});

}

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Drop​ ​ Down​ ​ Menu​ '),

backgroundColor: ​ Colors​ .brown,

),

body: ​ Form​ (

child: ​ ListView​ (

children: <​ Widget​ >[

​ Container​ (

padding: ​ EdgeInsets​ .only(left: ​ 16.0​ ),

child: ​ Row​ (

children: <​ Widget​ >[

​ Text​ (​ 'Lokasi​ ', style: ​ TextStyle​ (fontSize: ​ 18.0​ ,

color: ​ Colors​ .brown),),

​ DropdownButton​ (

onChanged: (​ String​ value) {

pilihKota(value);

nilaiKota = listKota.indexOf(value);

},

**D. Halaman Main**

value: nKota,

items: listKota.map(

(​ String​ value) {

​ return​ ​ DropdownMenuItem​ (

child: ​ Text​ (value),

value: value

);

}).toList(),

)

],

),

),

​ Container​ (

child: ​ MaterialButton​ (

child: ​ Text​ (​ 'Submi​ t'),

color: ​ Colors​ .brown,

textColor: ​ Colors​ .white,

onPressed: () {

print(​ "Kota yang dipilih: "​ +​ " $nilaiKota "​ + nKota);

},

),

)

],

),

),

);

}

}

Pada halaman main kita isikan seperti berikut :

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

import​ ​ 'ui\_view​ /page\_bottom\_menu.dart';

import​ ​ 'ui\_view​ /page\_nav\_bar\_menu.dart';

import​ ​ 'ui\_nav\_bar​ /page\_kontak.dart';

import​ ​ 'ui\_nav\_bar​ /page\_tentang.dart';

import​ ​ 'ui\_view​ /page\_dropdown\_menu.dart';

void main() => runApp(​ MyApp​ ());

class​ ​ MyApp​ ​ extends​ ​ StatelessWidget​ {

​ *// This widget is the root of your application.*

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ MaterialApp​ (

title: ​ 'Flutter​ ​ Demo​ ',

theme: ​ ThemeData​ (

primarySwatch: ​ Colors​ .blue,

),

home: ​ PageHome​ (),

*// deklarasi file yang akan dipanggil*

routes: <​ String​ , ​ WidgetBuilder​ >{

'/home' : (​ BuildContext​ context) => ​ PageNavBarMenu​ (),

'/tentang' : (​ BuildContext​ context) => ​ PageTentang​ (),

'/kontak' : (​ BuildContext​ context) => ​ PageKontak​ ()

},

debugShowCheckedModeBanner: ​ false​ ,

);

}

}

class​ ​ PageHome​ ​ extends​ ​ StatelessWidget​ {

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Home​ '),

backgroundColor: ​ Colors​ .green,

),

body: ​ Center​ (

child: ​ Column​ (

children: <​ Widget​ >[

​ MaterialButton​ (

color: ​ Colors​ .brown,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Navigation​ ​ Drawer​ '),

onPressed: () {

​ Navigator​ .push(context,

​ MaterialPageRoute​ (builder: (context) {

​ return​ ​ PageNavBarMenu​ ();

})

);

},

),

​ MaterialButton​ (

color: ​ Colors​ .brown,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Bottom​ ​ Bar​ ​ Menu​ '),

onPressed: () {

​ Navigator​ .push(context,

MaterialPageRoute​ (builder: (context) {

​ return​ ​ PageHomeBottomMenu​ ();

})

);

},

),

MaterialButton​ (

color: ​ Colors​ .brown,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ 'Dropdown​ ​ Menu​ '),

onPressed: () {

​ Navigator​ .push(context, ​ MaterialPageRoute​ (

builder: (context) {

​ return​ ​ PageDropDownMenu​ ();

}));

},

),

],

),

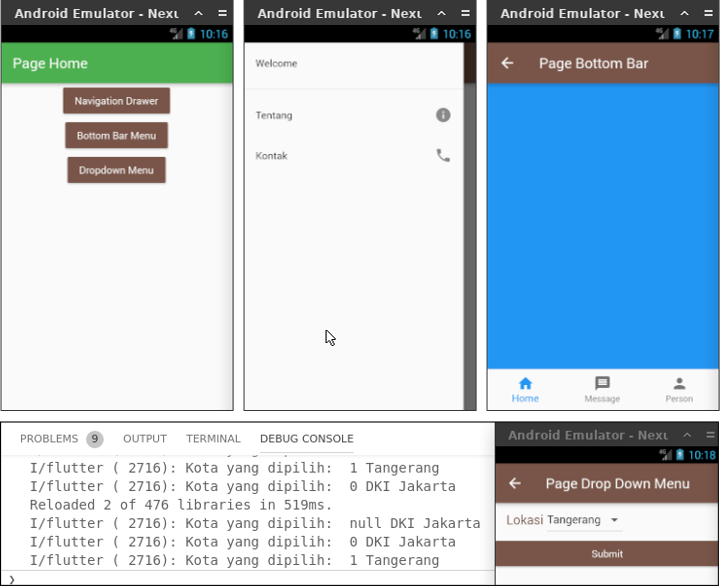
),

);

}

}

hasilnya :



**E. Latihan Form**

Buat lah halaman RegisterForm seperti berikut, lalu ketikkan seperti berikut :

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

import​ ​ 'package​ :fluttertoast/fluttertoast.dart';

class​ ​ RegisterForm​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_RegisterFormState createState() => \_RegisterFormState();

}

class​ ​ \_RegisterFormState​ ​ extends​ ​ State<RegisterForm>​ {

​ List​ <​ String​ > \_agama = [​ 'Isla​ m', ​ 'Kriste​ n', ​ 'Budh​ a', ​ 'Hindh​ u',

'Konghuc​ u'];

​ String​ \_nAgama = ​ "Islam"​ ;

int \_groupValue1 = ​ 0​ ;

​ TextEditingController​ nama = ​ TextEditingController​ ();

​ TextEditingController​ pass = ​ TextEditingController​ ();

​ TextEditingController​ moto = ​ TextEditingController​ ();

​ @override

​ Widget​ build(​ BuildContext​ context) {

return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ "Register Form"​ ),

),

body: ​ Padding​ (

padding: ​ EdgeInsets​ .all(​ 16.0​ ),

child: ​ Form​ (

child: ​ Column​ (

children: <​ Widget​ >[

​ TextFormField​ (

controller: nama,

decoration: ​ InputDecoration​ (

hintText: ​ 'Nama​ ​ Lengkap​ ',

border: ​ OutlineInputBorder​ (borderRadius:

BorderRadius​ .circular(​ 8.0​ ))

),

),

extFormField​ (

controller: pass,

obscureText: ​ true​ ,

decoration: ​ InputDecoration​ (

hintText: ​ 'Passwor​ d',

border: ​ OutlineInputBorder​ (borderRadius:

BorderRadius​ .circular(​ 8.0​ ))

),

),

TextFormField​ (

controller: moto,

minLines: ​ 3​ ,

maxLines: ​ 10​ ,

decoration: ​ InputDecoration​ (

hintText: ​ 'Moto​ ​ Hidup​ ',

border: ​ OutlineInputBorder​ (borderRadius:

BorderRadius​ .circular(​ 8.0​ ))

),

),

​ Row​ (children: <​ Widget​ >[

​ Radio​ (

onChanged: (int i) {

setState(() {

\_groupValue1 = i;

});

},

value: ​ 0​ ,

groupValue: \_groupValue1,),

​ Text​ (​ 'Lak​ -laki'),

​ Radio​ (

onChanged: (int i) {

setState(() {

\_groupValue1 = i;

});

},

value: ​ 1​ ,

groupValue: \_groupValue1,),

​ Text​ (​ 'Perempua​ n'),

],),

​ Text​ (​ "Agama : "​ , style : ​ TextStyle​ (color: ​ Colors​ .brown,

fontSize: ​ 16.0​ )),

​ DropdownButtonFormField​ (

onChanged: (​ String​ value) {

setState(() {

\_nAgama = value;

});

},

value: \_nAgama,

items: \_agama.map((​ String​ value) {

​ return​ ​ DropdownMenuItem​ (child: ​ Text​ (value), value:

value,);

}).toList(),

),

​ MaterialButton​ (

color: ​ Colors​ .red,

textColor: ​ Colors​ .white,

child: ​ Text​ (​ "Submit"​ ),

onPressed: () {

​ Fluttertoast​ .showToast(

msg: ​ "Nama Lengkap : "​ +nama.text.toString()+​ "\n"​ +

​ "Password : "​ +pass.text.toString()+​ "\n"​ +

​ "Moto hidup : "​ +moto.text.toString()+​ "\n"​ +

​ "Jenis Kelamin :"​ +((\_groupValue1 == ​ 0​) ? ​ "Laki"

: ​ "Wanita"​ )+​ "\n"​ +

"Agama : "​ +\_nAgama,

toastLength: ​ Toast​ .​ LENGTH\_SHORT​ ,

gravity: ​ ToastGravity​ .​ CENTER​ ,

timeInSecForIos: ​ 1​ ,

backgroundColor: ​ Colors​ .red,

textColor: ​ Colors​ .white,

fontSize: ​ 16.0

);

},

)

],

),

)

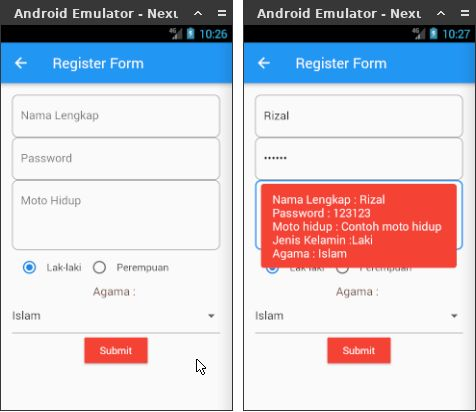
),

);

}

}

hasilnya :



**13. Number Picker**

Untuk membuat number picker kita perlu dependency

https://pub.dev/packages/numberpicker​, install seperti biasa, jika sudah kita ketikkan kode berikut :

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

import​ ​ 'dart​ :async';

import​ ​ 'package​ :numberpicker/numberpicker.dart';

class​ ​ PageNumberPicker​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_PageNumberPickerState createState() => \_PageNumberPickerState();

}

class​ ​ \_PageNumberPickerState​ ​ extends​ ​ State<PageNumberPicker>​ {

int currentIntValue = ​ 10​ ;

double currentDoubleValue = ​ 3.0​ ;

​ NumberPicker​ intNumberPicker, decNumberPicker;

​ *// method untuk menghandle value ketika berubah*

handleValueChanged(num value) {

​ if​ (value != ​ null​ ) {

​ if​ (value is int) {

setState(() {

currentIntValue = value;

});

} ​ else​ {

setState(() {

currentDoubleValue = value;

});

}

}

}

​ *// method untuk menghandle value ketika berubah dari luar*

handleValueChangedExternally(num value) {

​ if​ (value != ​ null​ ) {

​ if​ (value is int) {

setState(() {

currentIntValue = value;

intNumberPicker.animateInt(value);

});

} ​ else​ {

setState(() {

currentDoubleValue = value;

decNumberPicker.animateDecimal(currentDoubleValue.toInt());

});

}

}

}

​ @override

​ Widget​ build(​ BuildContext​ context) {

intNumberPicker = ​ new​ ​ NumberPicker​ .integer(

initialValue: currentIntValue,

minValue: ​ 0​ ,

maxValue: ​ 100​ ,

step: ​ 10​ ,

onChanged: handleValueChanged);

decNumberPicker = ​ new​ ​ NumberPicker​ .decimal(

initialValue: currentDoubleValue,

minValue: ​ 1​ ,

maxValue: ​ 5​ ,

decimalPlaces: ​ 2​ , ​ *// mengatur 2 angka di belakang koma*

onChanged: handleValueChanged);

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Number​ ​ Picker​ '),

backgroundColor: ​ Colors​ .brown,

),

body: ​ Center​ (

child: ​ Column​ (

mainAxisAlignment: ​ MainAxisAlignment​ .spaceAround,

children: <​ Widget​ >[

intNumberPicker,

​ RaisedButton​ (

child: ​ Text​ (​ 'Current​ int value : $currentIntValue'),

color: ​ Colors​ .green,

onPressed: () {

showDialogInteger();

},

),

decNumberPicker,

​ RaisedButton​ (

child: ​ Text​ (​ 'Current​ ​ Decimal​ ​ Value​ :

$currentDoubleValue'),

onPressed: () {

showDialogDouble();

},

)

],

),

));

}

*// saat menekan tombol Current int value*

​ Future​ showDialogInteger() async {

await showDialog<int>(

context: context,

builder: (​ BuildContext​ context) {

​ return​ ​ NumberPickerDialog​ .integer(

minValue: ​ 0​ ,

maxValue: ​ 100​ ,

initialIntegerValue: currentIntValue,

);

}).then(handleValueChangedExternally);

}

​ *// saat menekan tombol Current dec value*

​ Future​ showDialogDouble() async {

await showDialog<double>(

context: context,

builder: (​ BuildContext​ context) {

​ return​ ​ NumberPickerDialog​ .decimal(

minValue: ​ 1​ ,

maxValue: ​ 5​ ,

initialDoubleValue: currentDoubleValue,

decimalPlaces: ​ 2​ ,

title: ​ Text​ (​ "Silahkan pilih bilangan desimal"​ ),

);

});

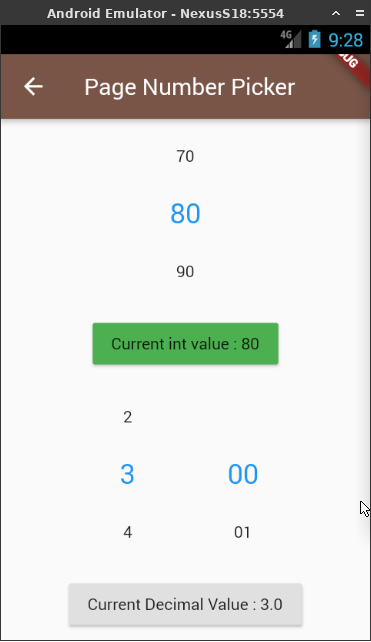
}

}

Penjelasan :

1. handleValueChanged(num value), yaitu fungsi yang dijalankan saat kita menggeser nilai angkanya
2. handleValueChangedExternally(num value), yaitu fungsi yang dijalankan saat kita menggeser nilai dari dialog yang muncul
3. showDialogInteger() = menampilkan dialog numberpicker integer
4. showDialogDouble() = menampilkan dialog numberpicker decimal

hasilnya :



**14. Membuat Search**

Untuk membuat halaman search kita ketikkan kode berikut :

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

class​ ​ SearchListPage​ ​ extends​ ​ StatefulWidget​ {

​ @override

\_SearchListPageState createState() => \_SearchListPageState();

}

class​ ​ \_SearchListPageState​ ​ extends​ ​ State<SearchListPage>​ {

​ var​ etSearce = ​ new​ ​ TextEditingController​ ();

bool isSearch = ​ true​ ;

​ String​ query = ​ ""​ ;

​ List​ <​ String​ > dataList;

​ List​ <​ String​ > filterList;

​ @override

void initState() {

​ *//* ​ *TODO:*​ *implement initState*

​ super​ .initState();

dataList = ​ new​ ​ List​ <​ String​ >();

dataList = [​ "Snake"​ , ​ "Elephant"​ , ​ "cats"​ , ​ "dog"​ , ​ "orion"​ ,

"boomerang"​ ,

​ "pelican"​ , ​ "ghost"​ , ​ "eagle"​ , ​ "horse head"​ , ​ "elephant trunk"​ ,

"butterfly"​ ];

dataList.sort();

}

\_SearchListPageState() {

etSearce.addListener((){

​ if​ (etSearce.text.isEmpty) {

setState(() {

isSearch = ​ false​ ;

query = ​ ""​ ;

});

} ​ else​ {

setState(() {

isSearch = ​ true​ ;

query = etSearce.text;

});

}

});

}

​ @override

​ Widget​ build(​ BuildContext​ context) {

​ return​ ​ Scaffold​ (

appBar: ​ AppBar​ (

title: ​ Text​ (​ 'Page​ ​ Search​ ​ Listview​ '),

backgroundColor: ​ Colors​ .brown,

),

body: ​ Container​ (

margin: ​ EdgeInsets​ .all(​ 10.0​ ),

child: ​ Column​ (

children: <​ Widget​ >[

\_createSearchView(),

isSearch ? \_performSearch() : \_createSearchView(),

],

),

),

);

}

​ *// membuat form search*

​ Widget​ \_createSearchView () {

​ return​ ​ Container​ (

decoration: ​ BoxDecoration​ (

border: ​ Border​ .all(width: ​ 1.0​ )

),

child: ​ TextField​ (

controller: etSearce,

decoration: ​ InputDecoration​ (

hintText: ​ "Search"​ ,

hintStyle: ​ TextStyle​ (color: ​ Colors​ .green)

),

textAlign: ​ TextAlign​ .center,

),

);

}

​ *// membuat form*

​ Widget​ \_createListView() {

​ return​ ​ Flexible​ (child: ​ ListView​ .builder(

itemCount: dataList.length,

itemBuilder: (​ BuildContext​ context, int index){

​ return​ ​ Card​ (

child: ​ Container​ (margin: ​ EdgeInsets​ .all(​ 10.0​ )),

color: ​ Colors​ .white,

elevation: ​ 5.0​ ,);

},

),);

}

​ *// lakukan search dengan fungsi contains*

​ Widget​ \_performSearch() {

filterList = ​ new​ ​ List​ <​ String​ >();

​ for​ (int i = ​ 0​ ; i < dataList.length; i++ ) {

var​ item = dataList[i];

​ if​ (item.toLowerCase().contains(query.toLowerCase())){

filterList.add(item);

}

}

​ return​ \_createFilteredListView();

}

*// tampilkan daftar hasil pencarian*

​ Widget​ \_createFilteredListView() {

​ return​ ​ Flexible​ (

child: ​ ListView​ .builder(

itemCount: filterList.length,

itemBuilder: (​ BuildContext​ context, int index) {

​ return​ ​ Card​ (

color: ​ Colors​ .white,

elevation: ​ 5.0​ ,

child: ​ Container​ (

margin: ​ EdgeInsets​ .all(​ 10.0​ ),

child: ​ Text​ ('${filterList[index]}'),

),

);

},),

);

}

}

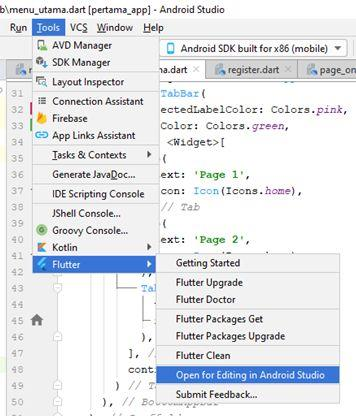
Hasilnya :



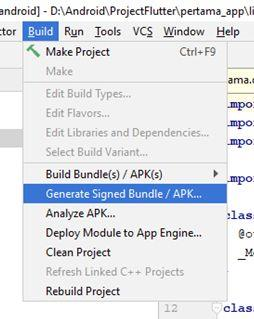
**15. MEMBUAT APK DARI PROJECT FLUTTER**

Adapun langkah kerja yang dilakukan untuk membuat apk dari project flutter, yaitu sebagai berikut:

1. Buka project flutter di android studio, klik **Tools** kemudian pilih **flutter** dan pilih **Open For Editing In Android Studio.**



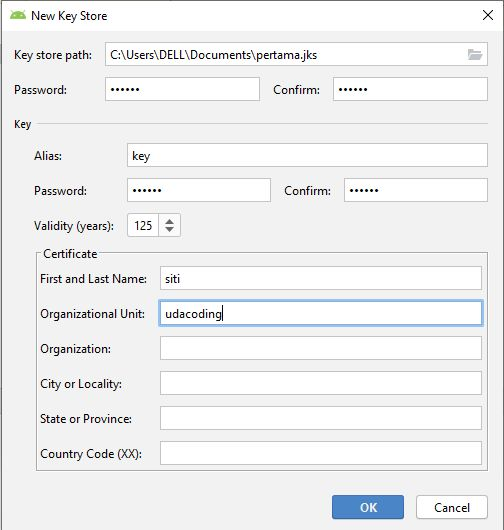
2. Pilih Build dan pilih **Generate Signed Bundle/APK**



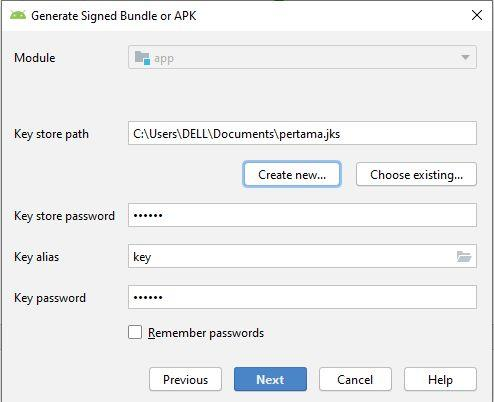
3. Selanjutnya pilih **APK**



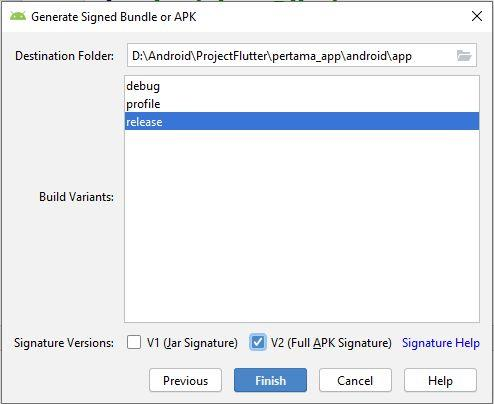
4. Isi form dengan benar



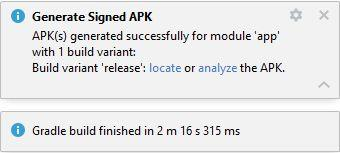
5. Selanjutnya klik OK maka akan muncul tampil seperti berikut ini:



6. Klik Next. Selanjutnya pilih **Release**, hal ini bertujuan untuk release APK yang akan diupload pada playstore. dan klik salah satu Signature Versionnya.



7. Selanjutnya pilih locate untuk melihat APK release



8. Berikut adalah APK sudah release

