Praktikum Pemrograman Berbasis Mobaile
Modul 3 : Membuat Aplikasi dan Menjalankannya

DISUSUN OLEH:

NAMA : NUGROHO WICAKSO
KELAS : TI2
JURUSAN : TEKNIK INFORMATIKA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA
AKAKOM YOGYAKARTA
2019/2020

A. CAPAIANPEMBELAJARAN

1. Mahasiswa mampu membuat aplikasi sederhana dengan desain standard yang disediakan dan menjalankan aplikasi di emulator maupun di perangkat mobile (keluaran)

B. Teori

Programmer yang menggunakan bahasa pemrograman berorientasi objek seperti Java akan terbiasa karena aplikasi Android ditulis di Kotlin, ini masih sangat banyak terjadi. Android, bagaimanapun juga, mengambil konsep yang dapat digunakan kembali komponen ke tingkat yang lebih tinggi.

Aplikasi Android diciptakan dengan menggunakan satu atau lebih komponen bersama, yang dikenal sebagai Activity. Sebuah Activity adalah satu, modul mandiri dari aplikasi yang biasanya berkorelasi langsung ke layar antarmuka pengguna. Activity dimaksudkan sebagai komponen, yang dapat digunakan kembali dan dapat dipertukarkan, dan bisa dibagi di antara aplikasi yang berbeda. Sebuah aplikasi email yang ada, misalnya, mungkin berisi Activity khusus untuk membuat dan mengirim pesan email. Seorang pengembang mungkin menulis sebuah aplikasi yang juga memiliki persyaratan untuk mengirim pesan email. Daripada mengembangkan Activity komposisi email khusus untuk aplikasi baru, pengembang hanya dapat menggunakan Activity dari aplikasi email yang ada.

Activity diciptakan sebagai subclass dari kelas Activity Android dan harus dieksekusi sehingga menjadi terpisah sepenuhnya dari Activity lain dalam aplikasi. Dengan kata lain, Activity bersama tidak bisa dipanggil langsung dalam program (karena aplikasi lain dapat menggunakan Aktivitas) dan satu Activity tidak bisa langsung memanggil metode atau mengakses data Activity lain. Sebagai gantinya, untuk mencapai tujuan ini, dengan menggunakan Intents dan Content Providers. Secara default, suatu Activity tidak dapat memberikan hasil dengan aktivitas yang ia dipanggil. Jika fungsi ini diperlukan, Activity harus secara khusus dimulai sebagai sub-aktivitas.

Aplikasi Manifest

File yang mengatur berbagai elemen dalam aplikasi adalah file Manifest. Berkas Manifest berbasis XML ini, menguraikan Activity, Service, Content Provider dan permisson yang membentuk suatu aplikasi secara lengkap. Selain file Manifest dan file Dex yang berisi kode-kode byte, paket aplikasi Android biasanya berisi kumpulan berkas Resources (sumber daya). Berkas ini mengandung sumber daya seperti string, gambar, huruf dan warna yang muncul dalam antarmuka pengguna secara bersama-sama, dengan representasi XML layout antarmuka pengguna. Secara default, berkas ini disimpan dalam /res, sub-direktori dalam hirarki proyek aplikasi.

Bila aplikasi dikompilasi, kelas bernama R dibuat, yang berisi referensi ke sumber daya aplikasi. File manifest dan sumber daya ini digabungkan untuk membuat apa yang dikenal sebagai Konteks Aplikasi. Konteks ini, diwakili oleh kelas Context Android, dapat digunakan dalam kode aplikasi untuk mendapatkan akses ke sumber daya aplikasi pada saat runtime. Selain itu, berbagai metode dapat dipanggil pada konteks aplikasi untuk mengumpulkan informasi dan membuat perubahan pada lingkungan aplikasi pada saat runtime.

Jelajahi file activity dan layout.

Kita akan fokus pada dua file paling penting yang membentuk aplikasi kita: File MainActivity Kotlin, dan file layout activity_main.xml.

Periksa MainActivity

MainActivity adalah contoh Activity. Suatu Activity adalah kelas inti Android yang menggambar antarmuka pengguna aplikasi Android (UI) dan menerima acara masukan. Saat aplikasi kita diluncurkan, aplikasi meluncurkan aktivitas yang ditentukan dalam file AndroidManifest.xml. Banyak bahasa pemrograman menentukan metode utama yang memulai program. Aplikasi Android tidak memiliki metode utama. Sebaliknya, file AndroidManifest.xml menunjukkan bahwa MainActivity harus diluncurkan ketika pengguna mengetuk ikon peluncur aplikasi. Untuk meluncurkan suatu kegiatan, OS

Android menggunakan informasi dalam manifes untuk mengatur lingkungan aplikasi dan membangun MainActivity. Kemudian MainActivity melakukan beberapa pengaturan secara bergantian. Setiap aktivitas memiliki file layout terkait. Activity dan layout dihubungkan oleh proses yang dikenal sebagai layout inflasi. Saat Activity dimulai, tampilan yang didefinisikan dalam file tata letak XML diubah menjadi (atau "digelembungkan" menjadi) objek tampilan Kotlin di memori. Setelah ini terjadi, Activity dapat menarik objek-objek ini ke layar dan juga secara dinamis memodifikasinya.

Run Emulator

Uji coba aplikasi wajib dilakukan seorang *developer*. Proses *running* atau *debugging* bisa dilakukan dengan dua cara, yaitu *running* dengan emulator atau peranti (*device*). Baik emulator maupun peranti memiliki kelebihan dan kekurangan masing-masing. Kita sebagai *developer* tinggal pilih mana yang sesuai keperluan.

Persiapan Running Menggunakan Emulator

Sebelum menggunakan emulator, pastikan beberapa hal berikut ini:

Virtualization

Untuk menjalankan emulator di dalam Android Studio, pastikan aspek virtualization. Sistem kita harus memenuhi persyaratannya, yakni ketentuan prosesor dan sistem operasi dari laptop / PC yang kita gunakan.

Processor

• Prosesor Intel: Jika laptop/pc kita menggunakan prosesor Intel, maka pastikan ia mendukung Intel VT-x, Intel EM64T (Intel 64), dan Execute Disable (XD) Bit functionality.

• Prosesor AMD: Jika laptop/pc kita menggunakan AMD, maka pastikan bahwa ia support dengan AMD Virtualization (AMD-V) dan Supplemental Streaming SIMD Extensions 3 (SSSE3).

Sistem Operasi

• Intel : Jika menggunakan processor Intel maka kita dapat menjalankannya di sistem operasi Windows, Linux, maupun Mac.

• AMD : Untuk prosesor AMD maka hanya bisa menjalankannya di sistem operasi Linux.

Menginstal Hardware Accelerated Execution Manager (HAXM)

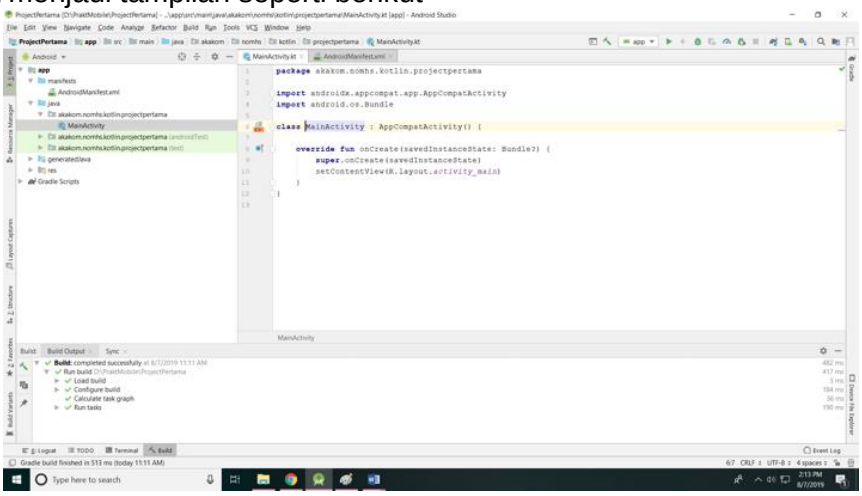
Setelah memenuhi persyaratan di atas, langkah selanjutnya adalah menginstal HAXM. HAXM adalah *hardware-assisted virtualization engine* yang menggunakan teknologi VT dari Intel untuk mempercepat aplikasi Android yang diemulasi di mesin host. HAXM diperlukan untuk menjalankan emulator di Android Studio.

HAXM diperlukan jika sistem operasi yang kita gunakan adalah Windows atau Mac. Untuk menginstalnya, ikuti petunjuk berikut ini.

1. Buka SDK Manager.
2. Pilih SDK Update Sites, kemudian hidupkan Intel HAXM.
3. Tekan OK.
4. Cari berkas installer-nya di directory folder sdk komputer Anda, `~sdk\extras\intel\Hardware_Accelerated_Execution_Manager\intelhaxmandroid.exe`.
5. Jalankan installer dan ikuti petunjuknya sampai selesai.

C. PRAKTIK

1. Memulai membuat project baru, gunakan langkah2 yang sudah dikerjakan pada pertemuan pertama.
2. Buat menjadi tampilan seperti berikut



3. Pada MainActivity, terdapat beberapa koding yang dapat dijelaskan sebagai berikut.
4. Kelas MainActivity extends AppCompatActivity.

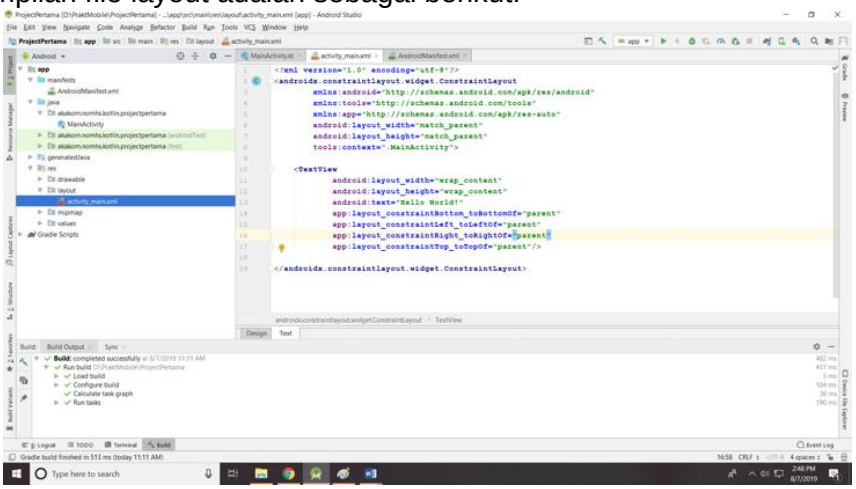
```
class MainActivity : AppCompatActivity() { …
```

5. AppCompatActivity adalah subkelas Kegiatan yang mendukung semua fitur Android modern sambil memberikan kompatibilitas dengan versi Android yang lebih lama. Untuk membuat aplikasi kita tersedia untuk sejumlah besar perangkat dan pengguna mungkin, selalu gunakan AppCompatActivity.
6. Perhatikan metode onCreate(). Activity tidak menggunakan konstruktor untuk menginisialisasi objek. Sebagai gantinya, serangkaian metode yang telah ditentukan (disebut "metode siklus hidup") disebut sebagai bagian dari pengaturan aktivitas. Salah satu metode siklus hidup tersebut adalah onCreate (), yang selalu ditimpa di aplikasi kita sendiri.
7. Di onCreate (), kita menentukan layout mana yang dikaitkan dengan aktivitas, dan Anda mengembang tata letak. Metode setContentView () melakukan kedua hal itu.

```
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)
}
```

8. Metode setContentView() mereferensikan layout menggunakan R.layout.activity_main, yang sebenarnya merupakan referensi integer. Kelas R dihasilkan ketika kita membangun aplikasi kita. Kelas R mencakup semua aset aplikasi, termasuk konten direktori res.
9. Dalam kasus ini, R.layout.activity_main merujuk ke kelas R yang dihasilkan, folder layout, dan file layout activity_main.xml. (Sumber daya tidak termasuk ekstensi file.) Kita akan merujuk ke banyak sumber daya aplikasi (termasuk gambar, string, dan elemen dalam file layout) menggunakan referensi serupa di kelas R.
10. Periksa dan jelajahi file layout aplikasi. Semua activity di aplikasi kita memiliki file layout terkait di direktori res/layout aplikasi. File layout adalah file XML yang mengungkapkan seperti apa sebenarnya aktivitas itu. File layout melakukan ini dengan menentukan tampilan dan menentukan di mana tampilan muncul di layar.
11. Tampilan adalah hal-hal seperti teks, gambar, dan tombol yang memperluas kelas tampilan. Ada banyak jenis tampilan, termasuk TextView, Button, ImageView, dan CheckBox.

12. Tampilan file layout adalah sebagai berikut.

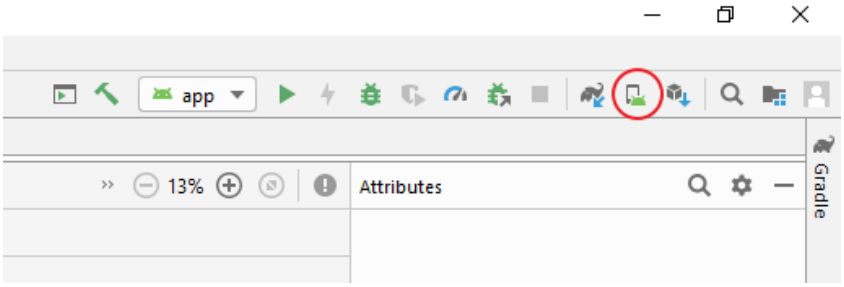


13. Untuk menjalankan aplikasi, dapat dilakukan dengan dua cara, pertama pada emulator, kedua pada perangkat mobile.
14. Menjalankan pada Virtual Device.

Telah dijelaskan sebelumnya tentang Virtual Device yaitu konfigurasi yang mendefinisikan karakteristik ponsel Android, tablet, Wear OS, atau perangkat Android TV yang ingin disimulasikan di Android Emulator.

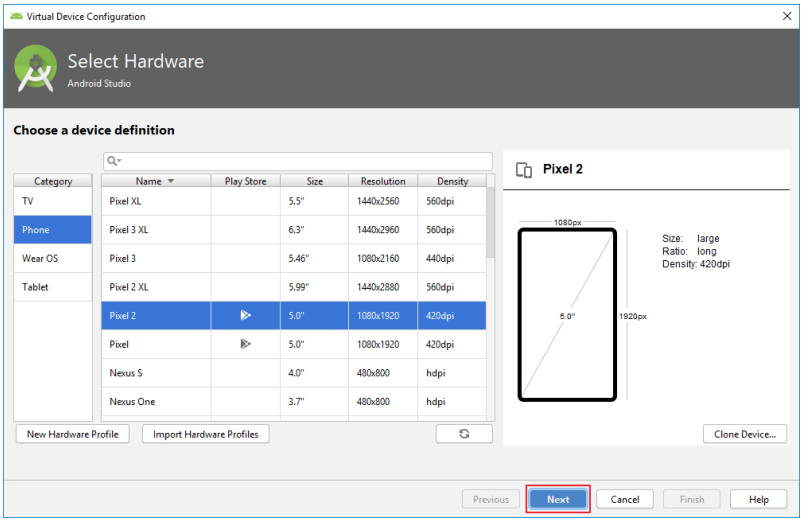
Dalam membuat Virtual Device dapat melalui langkah-langkah berikut :

1. **Klik** pada **AVD Manger** yang terdapat pada toolbar Android Studio.

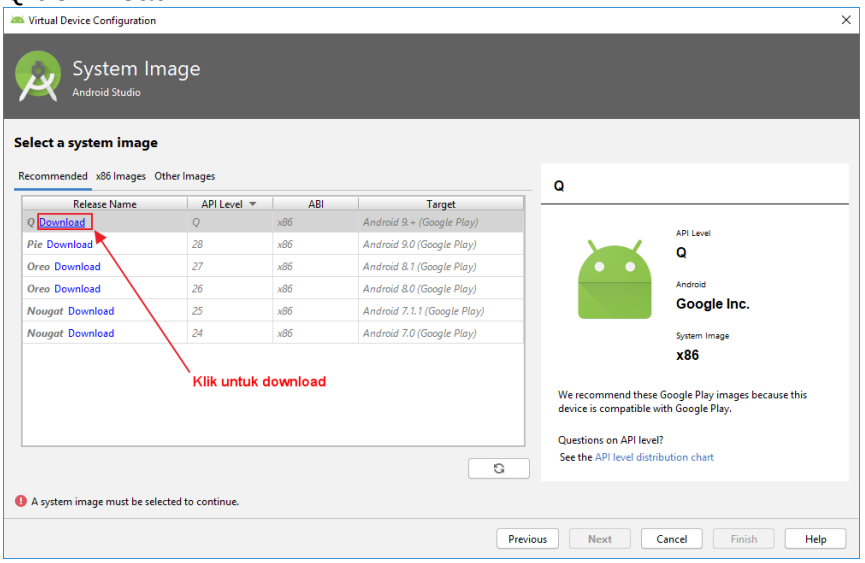


2. Pada jendela Android Virtual Device Manger, **klik** button **+ Create Virtual Device**.

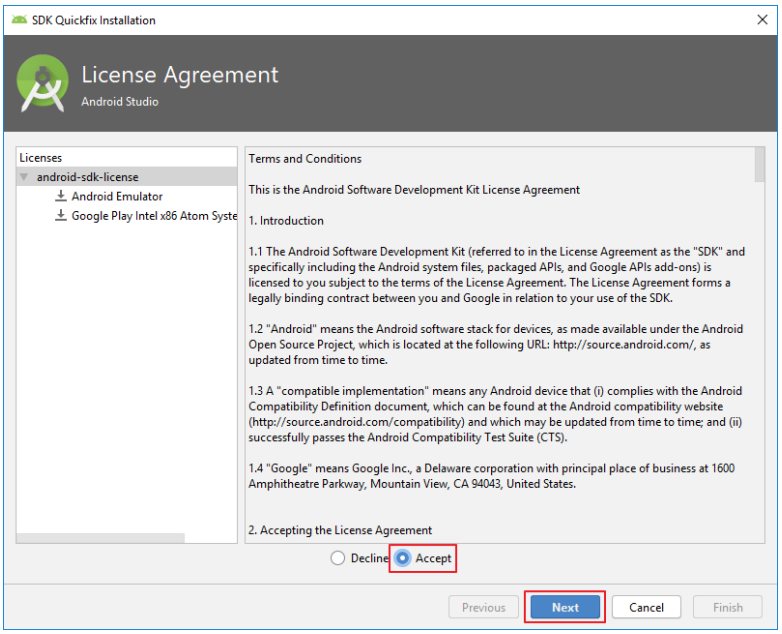
3. Pada jendela Virtual Device Configuration, pilih device yang akan digunakan untuk menampilkan hasil running project Android yang dibuat. Pilih pada **Pixel 2** kemudian **klik** button **Next**



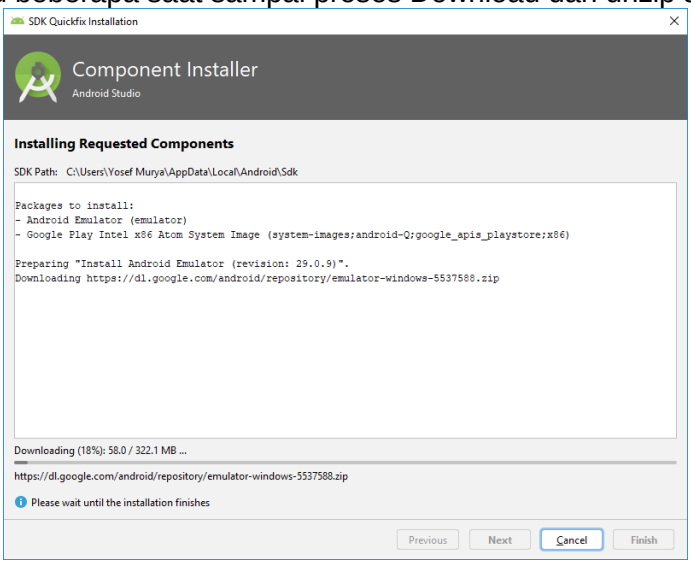
4. Jika pertamakali membuat emulator maka akan diminta untuk mendownload system images atau OS Android yang akan digunakan pada emulator. Silahkan **klik** pada bagian Q **download**



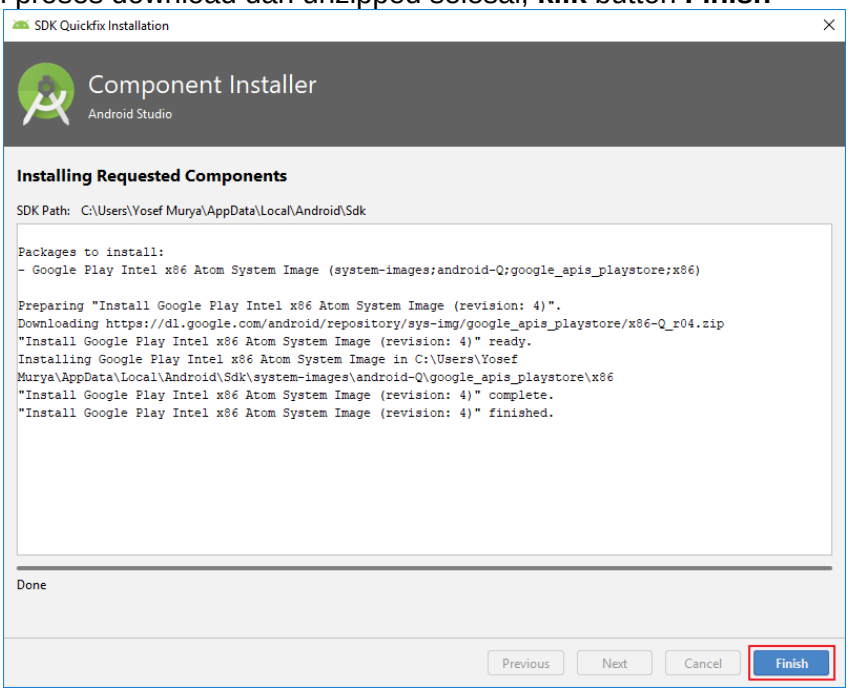
5. Pilih radio button **Accept** pada License Agreement, kemudian **klik** pada button **Next**



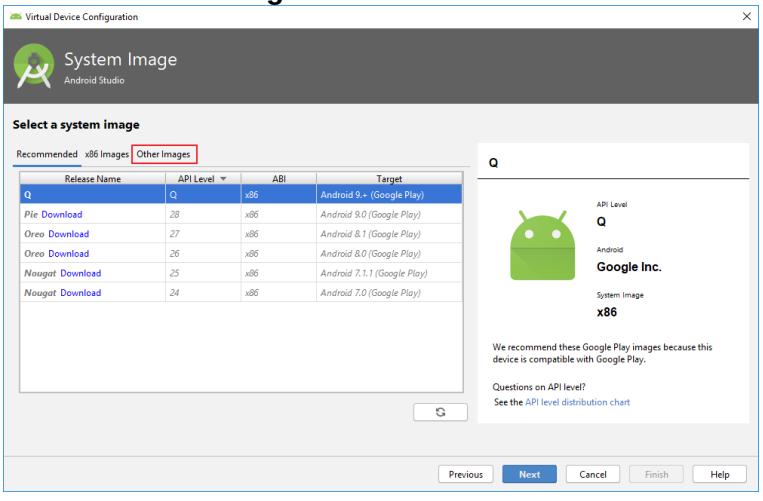
6. Tunggu beberapa saat sampai proses Download dan unzip selesai.



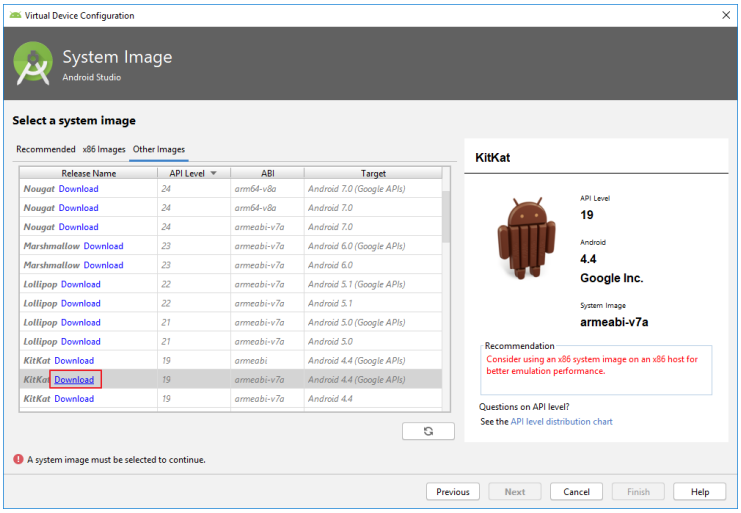
7. Setelah proses download dan unzipped selesai, **klik** button **Finish**



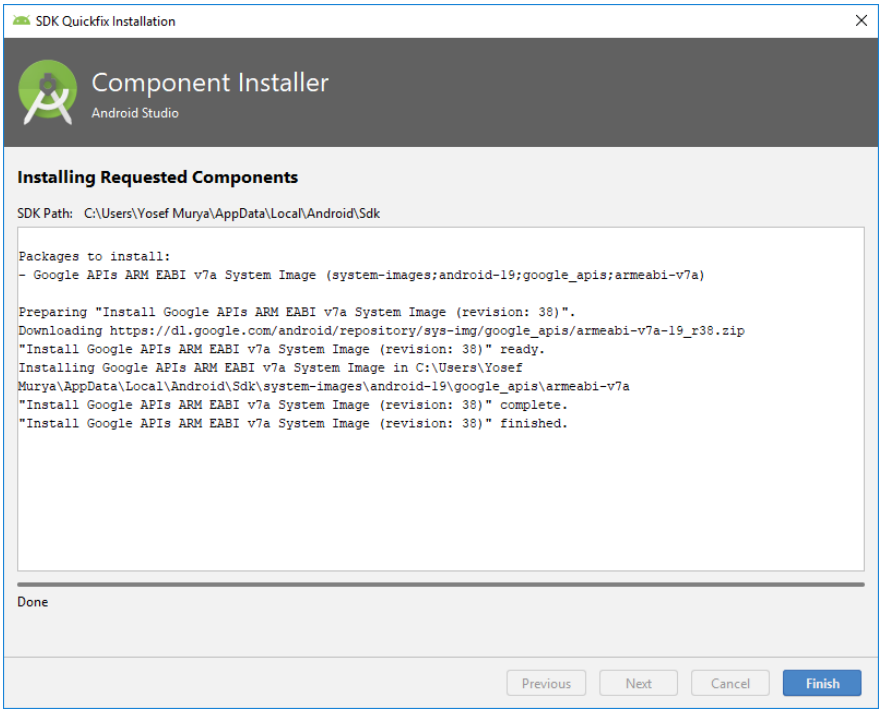
8. Langkah selanjutnya, **pilih** pada system image **Q** yang telah diinstall kemudian **klik** menu tab **other images**.



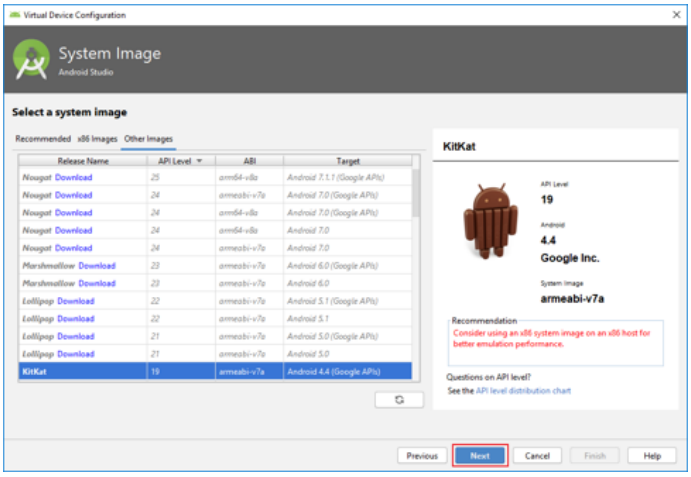
9. Kemudian sesuaikan dengan rekomendasi (Recommendation), agar emulator dapat dijalankan dengan mudah. Dikarenakan hardware yang digunakan untuk menjalankan emulator Android yaitu Intel Dual Code dengan RAM 4GB maka penulis memilih system image Kitkat dengan API 19, ABI armeabi-v7a, selanjutnya **klik Download**.



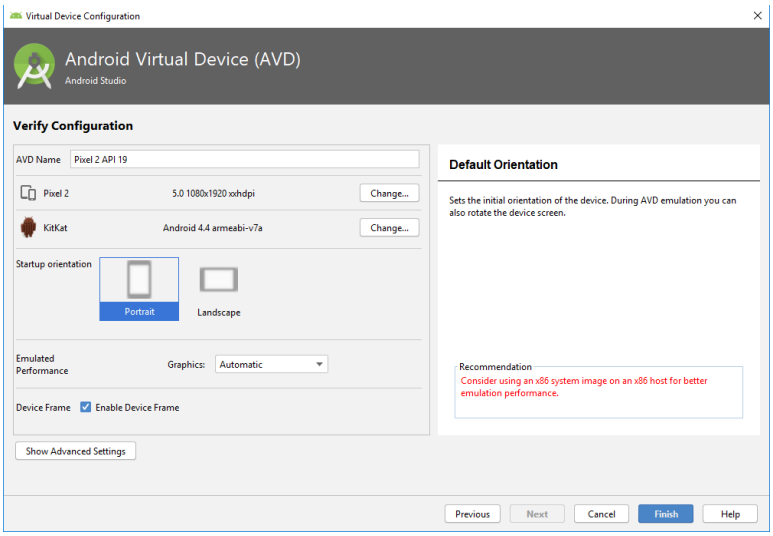
10. Proses download memerlukan waktu (pastikan anda terkoneksi internet), tunggu sampai proses download dan unzipped selesai, kemudian **klik** button **Finish**.



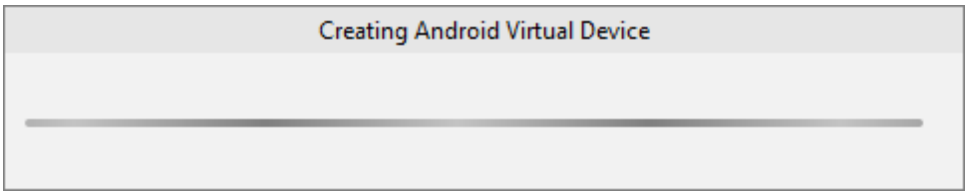
11. Maka akan terlihat system image KitKat telah berhasil didownload, **klik** button **Next** untuk melanjutkan ke proses selanjutnya.



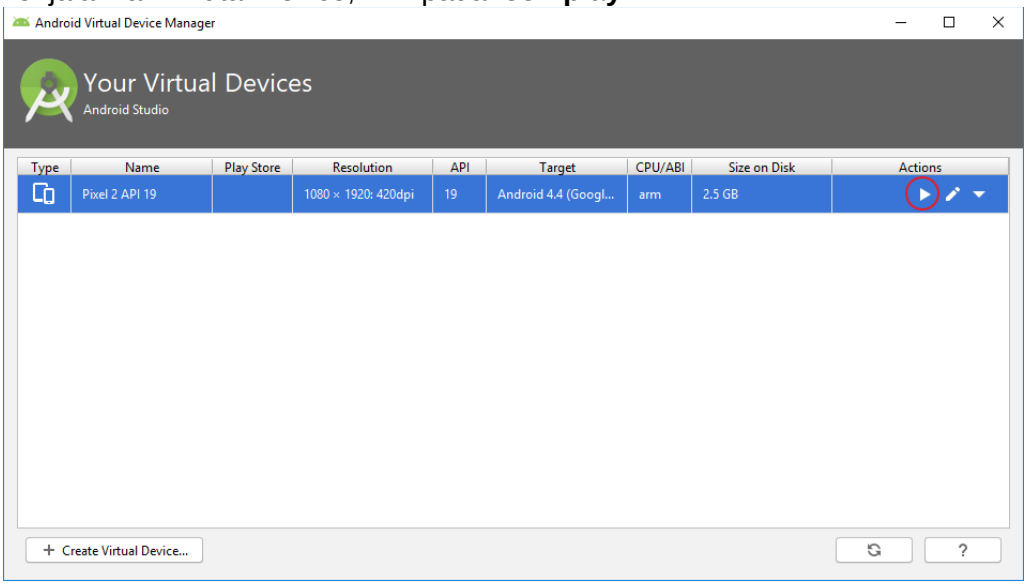
12. Pada bagian AVD Name akan terlihat secara default dengan nama Pixel 2 API 19 (dapat diganti dengan nama lain sesuai dengan kebutuhan), lalu pada bagian Startup orientation terdapat pilihan antara Potrait dan Landscape, secara default akan terpilih Potrait. **Klik** button **Finish**.



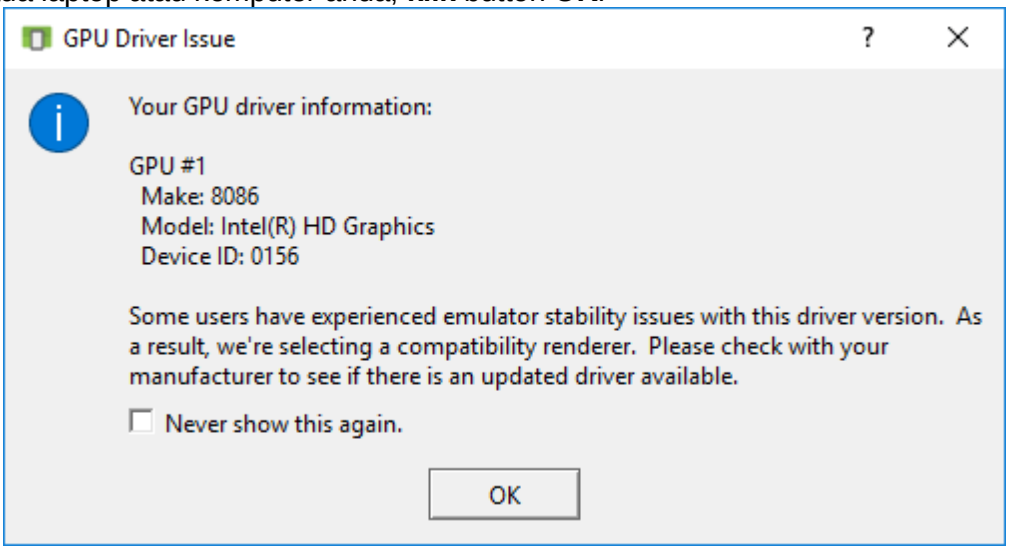
13. Proses pembuatan Android Virtual Device membutuhkan waktu beberapa detik, silahkan tunggu sampai proses pembuatan Android Virtual Device selesai.



14. Jika pembuatan Android Virtual Device telah selesai dilakukan maka akan muncul Android Virtual Device yang telah dibuat seperti terlihat pada gambar dibawah ini. Untuk menjalankan Virutal Device, **klik** pada **icon play**.



15. Pada bagian GPU Driver Issue akan terlihat informasi tentang Graphic Processing Unit pada laptop atau komputer anda, **klik** button **OK**.



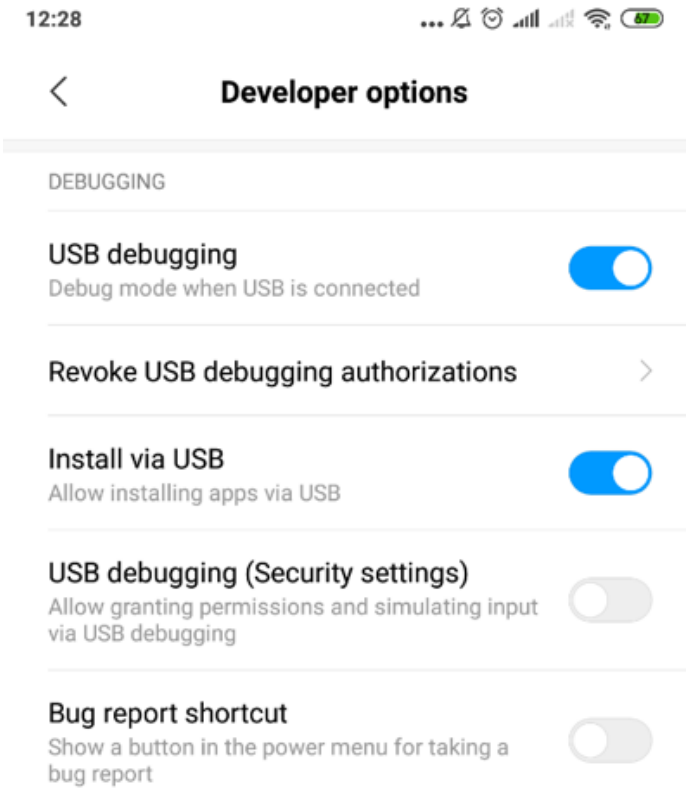
16. Maka akan muncul virtual device yang berfungsi untuk menampilkan hasil coding Android nantinya.

17. Jalankan aplikasi pada virtual device yang baru saja dibuat.

Physical Device.

Selain menggunakan virtual device dalam melakukan debug, anda dapat menggunakan physical deivce.
Bila Anda hendak melakukan *run* atau *debugging*, lebih baik Anda menjalankannya pada peranti *smartphone* asli. *Running* dengan menggunakan peranti memiliki beberapa kelebihan jika dibandingkan dengan emulator yaitu :
• Lebih cepat;
• Fitur seperti geo-location, push notif bisa digunakan;
• Bisa mengetahui daya serap baterai terhadap aplikasi;
• Lebih mudah.
Dengan menggunakan peranti smartphone asli, kita dapat memastikan bahwa aplikasi kita berjalan dengan wajar ketika sudah sampai di tangan pengguna.
Kendala dari pendekatan ini adalah beragamnya model peranti yang ada di pasaran. Namun, pembahasan mengenai hal tersebut tidak tercakup dalam kelas ini.
Mari ikuti langkah-langkah untuk menjalankan proses *run* atau *debugging*. Tampilan dari langkah berikut bisa dipastikan akan berbeda dengan peranti yang Anda pakai. Akan tetapi secara garis besar langkahnya akan sama.
1. Pastikan peranti yang akan dipakai sesuai dengan target SDK atau paling tidak mendukung versi SDK terendah yang digunakan aplikasi.
2. Buka *setting* dan masuk ke dalam menu **About**. Pada halaman menu ini, Anda perlu menemukan informasi tentang **Build number***.*
Berikut persiapan yang harus dilakukan :
1. Siapkan smartphone dan kabel data. Smartphone dengan OS Mobile Android, sedangkan kabel data adalah kabel yang dapat membaca data dari smartphone ke laptop (karena beberapa kasus terdapat kabel data yang hanya dapat digunakan untuk mengisi power atau batrei).

2. Penggunaan physical device dalam melakukan debug memerlukan beberapa pengaturan pada smartphone yaitu mengaktifkan developer option pada smartphone. Silakan mencari referensi untuk mengaktifkan developer option dari handphone yang anda pakai.
3. Kemudian aktifkan USB Debugging sampai pada bagian USB debugging terlihat

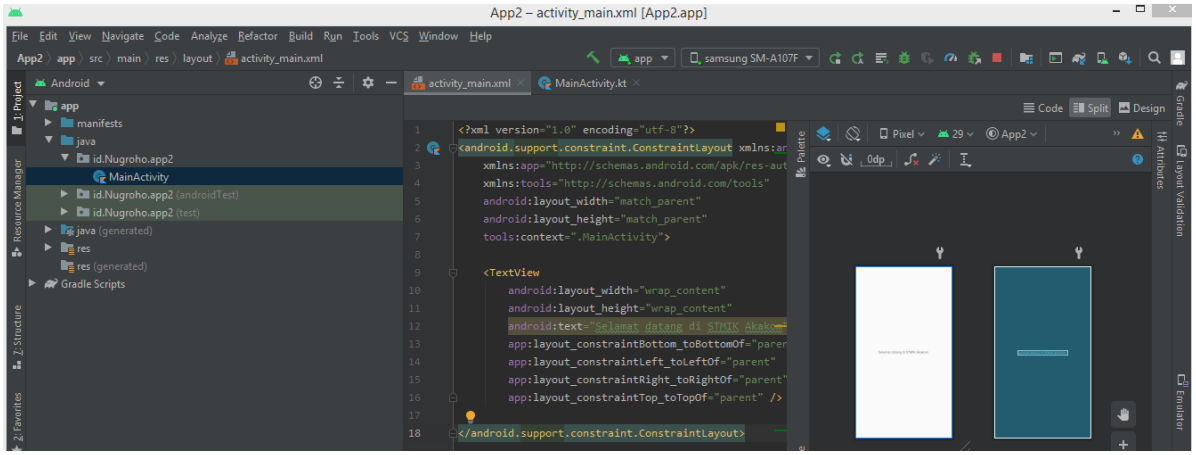


4. Jalankan aplikasi Anda pada perangkat handphone.

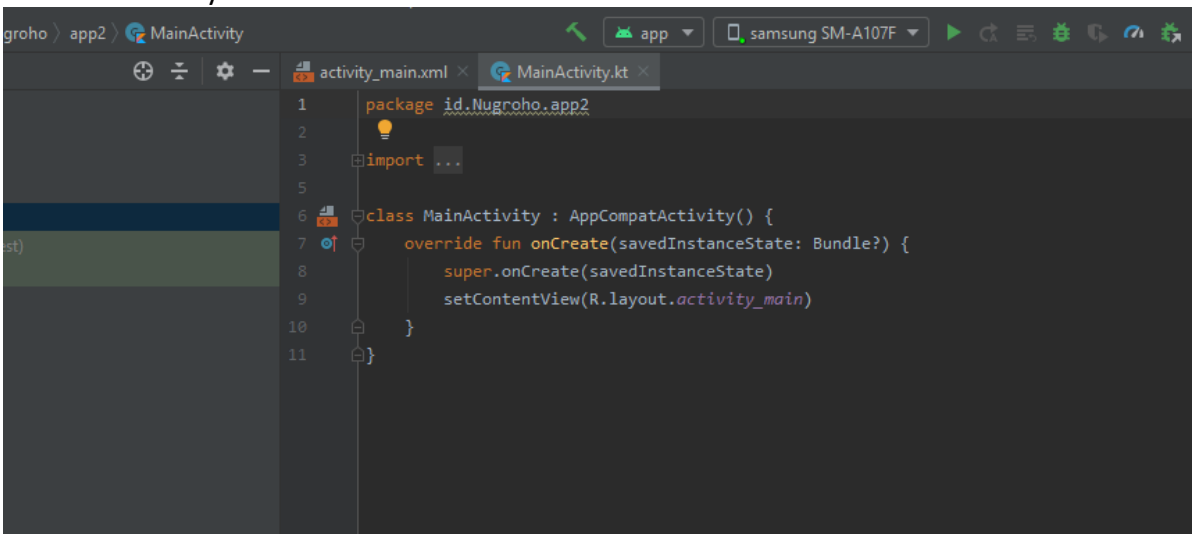
Latihan

1. Ganti tulisan Hello wold dengan tulisan “Selamat datang di STMIK AKAKOM
2. Jalankan.
3. Tambahkan tulisan lain dengan memodifikasi koding pada layout. TUGAS

Membuka file Activity_main.xml memastikan teks yang di hasilkan “selamat datang di STMIK akakom”



File MainActivity.kt



Mengunakan emulator untuk menampilkan hasil

Tugas

1. Buat aplikasi android dengan perangkat komputer anda di rumah dan jalankan pada perangkat anda.

2. Jelaskan tentang koding yang ada di file activity_main.xml

di file activity_main.xml berfungsi untuk menampilkan pesan saat pengguna mengetuk tombol Send.

Penjelasn

Mahasiswa diharapkan mampu membuat aplikasi sederhana dengan desain standard yang disediakan dan menjalankan aplikasi di emulator maupun di perangkat mobile