Capaian Pembelajaran :

* Mahasiswa dapat mengetahui dan menjelaskan latar belakang perkembangan komputasi bergerak
* Mahasiswa dapat mengetahui dan menjelaskan pengertian komputasi bergerak dan penerapannya di berbagai bidang

**KOMPUTASI BERGERAK**

* **Apakah yang melatarbelakangi munculnya komputasi bergerak (*mobile computing*) ?**

Kebutuhan terhadap alat (*tools*) pengolah informasi yang mampu **membantu** manusia dalam berkomunikasi kapan, dimana, dan dengan siapa saja telah menciptakan berbagai temuan-temuan baru di bidang teknologi informasi yang selanjutnya akan kita sebut perangkat TI dalam modul ini. Pada awalnya, perangkat TI yang diproduksi masih bersifat statis, yaitu hanya bisa digunakan ditempat dimana perangkat tersebut diletakkan dan sulit untuk dipindahkan karena ukuran yang besar dengan medium kabel (*wire*). Namun seiring perkembangan perangkat TI, maka diciptakanlah perangkat TI yang bersifat dinamis, yaitu perangkat TI yang dapat digunakan kapan dan dimana saja dengan medium tanpa kabel (*wireless*). Perkembangan TI tanpa kabel (*wireless*) pada akhir abad 20 terutama didorong oleh kebutuhan akan perangkat komputasi yang berkinerja memuaskan (*high* perform) sesuai harapan penggunanya.

Apa yang diharapkan terhadap perangkat TI pada masa kini dan akan datang ?

* Komputasi yang lebih *powerfull* pada perangkat yang lebih kecil
* Ringan, ringkas, tampilan menarik, dan konsumsi energinya rendah
* *Interface* yang menyesuaikan dimensi perangkat
* *Bandwidth* yang lebih besar
* Teknik *wireless* yang bermacam-macam
* Kecil, murah, *portable*, *replaceable*
* Terintegrasi dan terpisah

Aspek komputasi bergerak terdiri dari :

* Pengguna : pengguna dapat berkomunikasi kapan saja, dimana saja, dan dengan siapa saja.
* Perangkat portabel : perangkat dapat dihubungkan kapan dan dimana saja melalui jaringan komunikasi
* Metode jaringan : tanpa kabel (*wireless*) dan mampu terintegrasi dengan semua model jaringan

Apa saja yang membutuhkan komputasi bergerak ?

1. Kendaraan
2. Pengguna yang berpindah-pindah (*Nomadic user*)

* Laptop
* Sering terhubung ke jaringan internet

1. Telpon cerdas (*Smart phone*)

* Voice, video call
* Email, sosial media
* Game online/interaktif
* Aplikasi multi media
* Info komersial

1. *Wearable computing*

Perangkat komputasi bergerak yang dapat “ditempelkan” pada tubuh manusia, misalnya arloji, kacamata yang dilengkapi dengan mp3 player

1. *Intelligent office and house*
2. Ruang pertemuan

* Mengirim file
* Mengirim pesan ke ruang lain
* Voting rahasia
* Adhoc network

1. Polisi
2. Penanganan bencana alam

* Satelit
* Ad hoc network

\

1. Militer

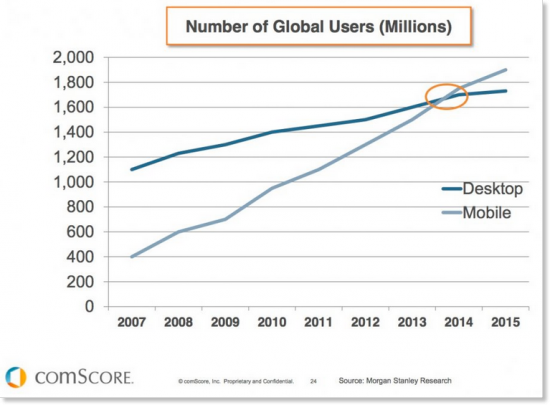
* **Apa yang dimaksud komputasi bergerak ?**

Komputasi bergerak (*mobile computing*) adalah suatu istilah yang digunakan untuk mendeskripsikan aplikasi berbasis komputasi pada piranti berukuran kecil, *portable*, dan *wireless* serta mendukung komunikasi.

Mengapa disebut komputasi bergerak ? karena teknologi yang terpasang pada piranti tersebut dapat digunakan dimanapun dan kapanpun selama dapat terhubung ke jaringan.

* **Perangkat komputasi bergerak**
* **Tren penggunaan teknologi komputasi bergerak**

Penggunaan alat komputasi bergerak dalam mengakses jaringan internet terhitung sejak 2007 hingga 2015 cenderung melonjak signifikan. Hal ini dapat dibuktikan melalui hasil survey oleh Morgan Stanley Research yang dipublikasikan pada <http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/> seperti yang dapat dilihat dibawah ini :



Capaian Pembelajaran :

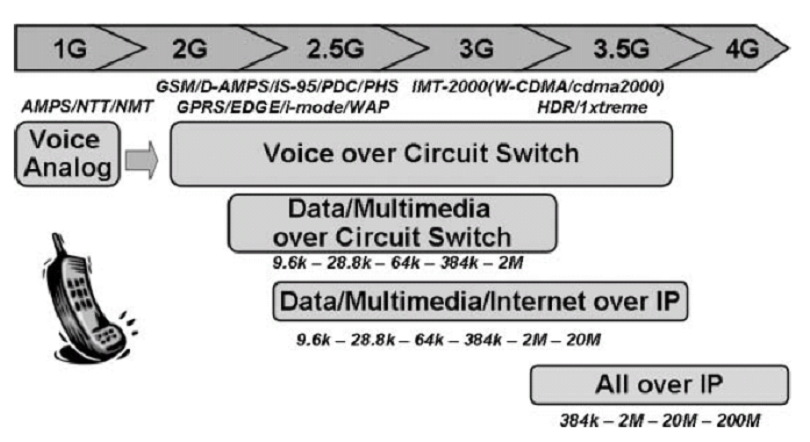
* Mahasiswa dapat mengetahui dan menjelaskan android sebagai *platform* aplikasi *mobile* yang bersifat *open source*
* Mahasiswa dapat mengetahui dan menjelaskan serta mengimplementasikan kebutuhan minimum spesifikasi *hardware* maupun *software* untuk aplikasi Android agar dapat diinstall

**SISTEM KOMPUTASI BERGERAK**

* **Revolusi teknologi telekomunikasi**
* Fleksibilitas pemakaian :
  + Menggeser penggunaan telepon kabel
  + Tidak dibatasi dalam suatu ruang tertentu (selama dalam area hot spot)
  + Bentuk dan ukuran
* Jumlah pemakai yang meningkat
  + 2000 sampai 2005 - kurang lebih 200 juta pelanggan
  + 2006 – kurang lebih 800 juta
* Jumlah network provider yang meningkat
  + Bukan merupakan barang mewah, tetapi menjadi “part of life”.
* Fasilitas layanan yang meningkat : internet dan multimedia
* Kapasitas memori yang memungkinkan :
  + Penyimpanan nomor telepon
  + Pesan singkat (SMS)
  + Gambar
  + Aplikasi
* **Kelebihan teknologi mobile**
* Extreme Personalization
  + Ponsel diantara dompet dan kunci motor
  + Tempat menyimpan segala informasi pribadi
* Pengaksesan informasi setiap saat dan dimanapun
  + Memungkinkan kita untuk bekerja, belanja atau bermain tanpa batasan waktu dan tempat (asal terhubung!)
* Mobilitas tinggi tanpa kerumitan kabel & Instalasi jaringan yang cepat
* Kompatible yang tinggi dengan teknologi lain
* Reduksi biaya : dalam kasus pengembangan, pemindahan maupun perubahan konfigurasi LAN
* **Kekurangan teknologi mobile**
* Keamanan jaringan yang biayanya relative mahal
* Interferensi pada sistem navigasi pesawat terbang
* Sensitive terhadap cuaca
* Keterbatasan daya jangkau sinyal berkisar antara 10 – 100 meter dari sumber daya
* Ijin penggunaan frekuensi masih < 2,4 GHz
* **Telepon bergerak (*Mobile/Cell phone*)**

Telepon bergerak/seluler adalah perangkat elektronik portable yang berfungsi sebagaimana telepon konvensional, namun dapat ‘bergerak’ pada suatu area yang luas dan memiliki kemampuan untuk SMS, MMS, menjalankan aplikasi multi media.

Perkembangan teknologi telepon bergerak/seluler terus berlangsung hingga kini yang dapat digambarkan sebagaimana di bawah ini.



Gambar 10. Perkembangan teknologi telepon bergerak/seluler dari era 1G sampai 4G

Capaian Pembelajaran :

* Mahasiswa dapat mengetahui dan menjelaskan android sebagai *platform* aplikasi *mobile* yang bersifat *open source*
* Mahasiswa dapat mengetahui dan menjelaskan serta mengimplementasikan kebutuhan minimum spesifikasi *hardware* maupun *software* untuk aplikasi Android agar dapat diinstall pada komputer/laptop dan mampu menginstall-nya pada komputer/laptop
* Mahasiswa dapat mengetahui dan menjelaskan *virtual machine, tools API,* dan *plugin* yang terdapat pada aplikasi Android beserta fungsinya.
* Mahasiswa dapat mengetahui dan menjelaskan arsitektur dan komponen aplikasi Android

**PENGENALAN ANDROID**

* **Android sebagai bahasa pemrograman komputasi bergerak**

Android adalah sebuah sistem operasi untuk perangkat *mobile* layar sentuhberbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi. Android bersifat *open source* karena menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi dan karena itu Android sangat pesat perkembangannya. Android awalnya dikembangkan oleh Android.inc yang didirikan di Palo Alto, California pada Oktober 2003 oleh Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Karena Android.inc mulai kekurangan finansial, maka pada 17 Agustus 2005 Android.inc diakuisisi oleh Google. Sejak dirilis pada 5 November 2007 yang diimplementasikan pada *smart phone Nexus One* produksi HTC Corporation, teknologi Android terus mengalami perkembangan dengan merilis berbagai pemutakhiran versi sistem operasinya yang hingga kini mencapai Android Nougat versi 7.0. (2016) dan semakin banyak vendor *smart phone* yang menggunakan Android sebagai sistem operasi mereka. Selain itu, Android kini melebarkan penerapan sistem operasinya pada perangkat Tablet PC dan menyaingi Apple.

Dengan kemajuan yang begitu pesat, maka berbagai perusahaan yang bergerak di bidang perangkat keras, perangkat lunak, dan telekomunikasi membentuk konsorsium untuk berkolaborasi dalam mengembangkan Android. Perusahaan-perusahaan yang membentuk konsorsium tersebut diantaranya, Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, Nvidia, Apache, Asustek Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc.

Android sebagai calon *platform mobile computation* masa depan yang diunggulkan mempunyai kelebihan, yaitu **terbuka**, **lengkap**, dan **bebas** sehingga membuatnya terus berkembang dari segi teknologi maupun dari segi jumlah *device* yang menggunakannya. Disebut **Terbuka** karena disediakan melalui lisensi *open source*, pengembang dapat dengan bebas untuk mengembangkan aplikasi yang didesain. Android sendiri menggunakan Linux Kernel 2.6. Disebut **Lengkap** karena para desainer dapat melakukan pendekatan yang komprehensif ketika sedang mengembangkan *platform* Android. Android merupakan sistem operasi yang aman dan banyak menyediakan *tools* dalam membangun *software* dan memungkinkan untuk mengembangkan aplikasi. Dikatakan **Bebas** karena Android sebagai sistem operasi/aplikasi yang bebas untuk dikembangkan (*develope*). Tidak ada lisensi atau biaya royalti yang harus dibayarkan saat ingin mengembangkan aplikasi atau *platform* Android. Namun, aplikasi untuk Android dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

* **Spesifikasi *hardware & software* PC untuk Android**

Untuk dapat menggunakan Android dalam mengembangkan aplikasi *smart phone* dan *Tablet PC*, maka spesifikasi hardware dan software harus memenuhi standar minimal berikut :

Perangkat Keras :

* Processor CPU berkecepatan 1,9 GHz (arsitektur 32 atau 64 bit)
* RAM berkapasitas 1 GB
* HDD berkapasitas 500 GB

Perangkat Lunak :

* Sistem operasi : Windows (XP, Vista, 7,8,10); Mac OS X 10; Linux
* Sun Java SE versi 1.6 atau versi diatasnya
* Android SDK 4.2.2 dan Eclipse
* **Dalvik Virtual Machine**

Salah satu elemen kunci dari Android adalah *Dalvik Virtual Machine* (selanjutnya disingkat dengan sebutan DVM). Android berjalan di dalam DVM bukan di dalam *Java Virtual Machine* (JVM). Walaupun memiliki banyak persamaan dengan JVM seperti *Java Mobile Edition* (Java ME), tetapi Android menggunakan *virtual machine* sendiri yang dirancang untuk menjalankan beberapa fitur secara lebih efisien pada perangkat *mobile*. DVM umumnya dirancang dengan bahasa pemrograman Java dan dikompile dengan kodebit, kemudian dikonversi menjadi berkas *dex* (Dalvik Executable) sebelum dipasang pada piranti kerasnya.

Semua *hardware* yang berbasis Android dijalankan dengan menggunakan *virtual machine* untuk mengeksekusi aplikasi. DVM mengeksekusi *file executable*, sebuah format yang dioptimalkan untuk memastikan memori yang digunakan sangat kecil. File exe tersebut diciptakan dengan mengubah kelas bahasa Java dan dikompilasi menggunakan *tools* yang disediakan dalam Android SDK.

* **Android SDK (Software Development Kit)**

Android SDK adalah *tools* API (*Application Programming Interface*) yang menggunakan bahasa pemrograman Java dan berfungsi untuk mengembangkan aplikasi pada *platform* Android. Android SDK memberikan kesempatan bagi kita untuk membuat aplikasi yang bukan merupakan aplikasi bawaan *handphone/smartphone*. Beberapa fitur Android yang penting adalah :

* *Framework* aplikasi yang mendukung penggantian komponen dan dapat digunakan kembali pada proyek aplikasi lainnya
* *Dalvik Virtual Machine* dapat dioptimalkan untuk perangkat *mobile*
* *Integrated browser* berdasarkan webkit yang bersifat *open source*
* Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, 3D berdasarkan spesifikasi openGL ES 1.0 (opsional akselerasi hardware)
* SQLite untuk penyimpanan data
* Mendukung multi media, baik audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung hardware)
* Bluetooth, EDGE, 3G, dan WiFi (tergantung spesifikasi hardware)
* Kamera, GPS, kompas, dan accelerometer (tergantung spesifikasi hardware)
* Fasilitas pengembangan aplikasi yang lengkap dan kaya, diantaranya perangkat emulator, *tools debugging*, profil dan kinerja memori, dan *plugin* untuk IDE Eclipse
* **Android Development Tools (ADT)**

ADT adalah *plugin* yang didesain untuk IDE Eclipse yang memberikan kemudahan dalam merancang proyek aplikasi Android, membuat GUI aplikasi, dan menambahkan komponen-komponen yang lainnya serta melakukan *package* aplikasi android (.apk) yang digunakan untuk mendistribusikan aplikasi yang dirancang.

* **Arsitektur Android**

Secara garis besar arsitektur Android terdiri atas :

* Applications dan Widgets

Komponen android yang berhubungan dengan aplikasi, di lapisan ini biasanya tempat mengunduh aplikasi yang kemudian dipasang (install) dan dijalankan (running). Pada komponen ini terdapat aplikasi inti, seperti klien email, program sms, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis dengan bahasa pemrograman Java.

* Applications Frameworks

Lapisan dari android yang berfungsi sebagai tempat untuk melakukan perancangan atau pembuatan aplikasi yang akan dijalankan pada sistem operasi android, seperti aplikasi *content-providers* yang berupa sms dan panggilan telepon.

Komponen-komponen yang tergolong dalam *Applications Frameworks* adalah:

* Views
* Content Provider
* Resources Manager
* Notification Manager
* Activity Manager
* Libraries

Lapisan dimana fitur-fitur android berada, diantaranya :

* Libraries media untuk menjalankan audio dan video
* Libraries untuk manajemen tampilan
* Libraries grafik mencakup SGL dan Open GL untuk grafis 2D dan 3D
* Libraries SQLite untuk dukungan basis data
* Libraries SSL dan Webkit untuk terintegrasi dengan web browser dan pengamanan
* Libraries LiveWebcore mencakup web browser modern dengan *engine embedded web view*
* Libraries 3D yang mencakup implementasi openGL ES 1.0
* Android Run Time

Lapisan yang membuat aplikasi dapat dijalankan dengan menggunakan model implementasi Linux. Pada lapisan ini terdapat 2 bagian yang berperan penting, yaitu :

1. *Core Libraries* : aplikasi android dibangun dalam bahasa java, sementara *Dalvik* sebagai mesin virtualnya menggunakan kernel Linux sehingga diperlukan *library* yang berfungsi untuk menginterpretasikan bahasa java/c yang ditangani pada bagian *Core Libraries*.
2. *Dalvik Virtual Machine* (DVM) : mesin virtual berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi pada kernel Linux secara efisien, seperti melakukan *threading* dan manajemen tingkat rendah.

* Linux Kernel

Letak dari lapisan inti sistem operasi Android. Berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem operasi Android lainnya. Linux Kernel yang digunakan Android adalah linux kernel versi 2.6

* **Komponen Aplikasi Android**

Aplikasi Android ditulis dalam bahasa pemrograman Java. Kode Java dikompilasi bersama dengan file *resource* yang dibutuhkan oleh aplikasi, dimana prosesnya dipaket (*package*) oleh *tools* yang disebut “apt tools” ke dalam paket Android sehingga menghasilkan file dengan ekstensi apk. File apk itulah yang disebut dengan aplikasi yang nantinya dapat diinstall di perangkat mobile.

Ada 4 macam komponen pada aplikasi Android yaitu :

1. ***Activities***

Suatu *activity* akan menyajikan *user interface* (UI) kepada pengguna sehingga pengguna dapat berinteraksi. Sebuah aplikasi Android dapat saja hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity* tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu *activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai *user interface* (UI) saat aplikasi ditampilkan kepada pengguna. Untuk pindah dari satu *activity* ke *activity* lainnya dapat dilakukan dengan satu *even*, misalkan *click* tombol, memilih opsi atau menggunakan *trigger* (pemicu) tertentu. Secara hirarki sebuah *windows activity* dinyatakan dengan *method Activity.setContentView*(). *ContentView* adalah objek yang berada pada root hirarki.

1. ***Service***

Service tidak memiliki *Graphic User Interface* (GUI), tetapi service berjalan secara *background*, sebagai contoh dalam memainkan musik, service mungkin memainkan musik atau mengambil data dari jaringan, tetapi setiap service harus berada dalam kelas induknya. Misalnya, media player sedang memutar lagu dari daftar lagu yang tersedia, maka aplikasi ini akan memiliki 2 atau lebih *activity* yang memungkinkan pengguna untuk memilih lagu atau menulis sms sambil player musik tetap berjalan. Untuk menjaga musik tetap dijalankan, *activity* player dapat menjalankan service. Service dijalankan pada *thread* utama dari proses aplikasi.

1. ***Broadcast Receiver***

*Broadcast Receiver* berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contohnya seperti notifikasi perubahan zona waktu, baterai low, gambar telah selesai diambil oleh kamera, atau pengubahan referensi bahasa yang digunakan. Aplikasi juga dapat menginisiasi *broadcast* misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah diunduh ke perangkat dan siap untuk digunakan.

*Broadcast Receiver* tidak memiliki *user interface* (UI), tetapi memiliki sebuah *activity* untuki merespon informasi yang diterima atau mungkin menggunakan *Notification Manager* untuk memberi tahu kepada pengguna, seperti lampu latar atau *vibrating* (getaran) perangkat, nada dering telpon atau sms dan lain sebagainya.

1. ***Content Provider***

*Content Provider* membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti basis data SQLite. *Content Provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity*, misalnya penggunaan aplikasi yang membutuhkan peta atau aplikasi untuk mengakses data kontak dan navigasi.

Capaian Pembelajaran :

* Mahasiswa dapat mengetahui dan menjelaskan komponen-komponen sintaks dan letak file *User Interface* pada suatu project aplikasi android
* Mahasiswa dapat mengetahui dan menjelaskan elemen-elemen, letak dan fungsi file manifest.xml pada suatu project aplikasi android
* Mahasiswa dapat mengetahui dan menjelaskan sintaks coding contoh aplikasi *user interface* untuk menampilkan teks dan tombol pada layar handphone

**USER INTERFACE (UI)**

* **Arsitektur *User Interface***

Secara umum arsitektur UI pada aplikasi Android meliputi 2 bagian yaitu UI *activity* dan UI yang terdiri dari komponen. Semua yang berhubungan dengan UI pada aplikasi Android biasanya berada pada lokasi res/layout/filename.xml.

Umumnya sintaks file res/layout memiliki struktur sebagai berikut :

<?xml version=”1.0” encoding=“utf-8” ?>

<ViewGroup

xmlns:android=<http://schemas.android.com/apk/res/android>

android:id=”@[+][*package*:]id/*resource\_name*”

android:layout\_height=[“*dimension*”|”*fill\_parent*”|

“wrap\_content”]

android:layout\_width=[“*dimension*”|”*fill\_parent*”|

“wrap\_content”]

[*ViewGroup-specific attributes*]>

<*View*

android:id=”@[+][*package*:]id/*resource\_name*”

android:layout\_height=[“*dimension*”|”*fill\_parent*”|

“wrap\_content”]

android:layout\_width=[“*dimension*”|”*fill\_parent*”|

“wrap\_content”]

[*View-specific attributes*]>

<requestFocus/>

</*View*>

<*ViewGroup>*

*<View* />

</*ViewGroup*>

<include layout=”@layout/layout\_resource”/>

</*ViewGroup*>

Komponen-komponen dari sintaks file res/layout diatas terdiri dari :

**<ViewGroup>**

Kumpulan *view* yang dapat ditentukan tata letaknya secara berbeda, seperti *LinearLayout, RelativeLayout, FrameLayout,* dan Tabulasi. Atribut dari ViewGroup ini biasanya terdiri dari :

* android:id

resource id, yang berisi variabel unik dari elemen tersebut

* android:layout\_height

dimensi valuenya height yang diikuti dengan opsi “fill parent” atau “wrap content”

* android:layout\_width

dimensi valuenya width yang diikuti dengan opsi “fill parent” atau “wrap content”

**<View>**

Mirip <ViewGroup> tetapi <View> lebih dikenal dengan “*individual UI component*”, atributnya juga minimal terdiri dari tiga bagian yang sama persisi dengan atribut yang dimiliki oleh <ViewGroup>

**<requestFocus>**

Elemen kosong yang bias didefenisikan di dalam <View>

**<include>**

Memasukkan file layout ke dalam layout. Atributnya sama dengan <ViewGroup> dan <View> tetapi ada tambahan atribut <resource> yang berfungsi untuk menentukan file layoutnya.

* **AndroidManifest.xml**

File Manifest.xml diperlukan oleh setiap aplikasi Android, letak file ini berada dalam folder *root* aplikasi. File manifest.xml mendeskripsikan variabel global yang terdapat pada aplikasi, file ini juga berfungsi untuk mendeskripsikan resource apa saja yang akan digunakan oleh proyek seperti koneksi internet, sms, mengakses GPS, dan lain sebagainya. Berikut ini contoh sintaks file manifest.xml :

<?xml version=”1.0”encoding=”utf-8”?>

<manifest

xmlns:android=[*http://schemas.android.com/apk/res/android*](http://schemas.android.com/apk/res/android)

package=”com.wilis.uicoding1”

android:versionCode=”1”

android:versionName=”1.0”>

<application android:icon=*”@drawable/icon”*

android:label=*”@string/app\_name”*>

<activity android:name=*”.uicoding1”*

android:label=*”@string/app\_name”*>

<intent-filter>

<action android:name=*”android.intent.action.MAIN”*/>

<category android:name=*”android.intent.category.LAUNCHER”*/>

</intent-filter>

</activity>

</application>

<uses-sdk android:minSdkVersion=”9”/>

</manifest>

Dalam file AndroidManifest.xml inilah berisi semua pendefinisian hal-hal yang dibutuhkan oleh aplikasi android, beberapa elemen yang terdapat di dalam *coding* file AndroidManifest.xml adalah :

**<manifest>**

Titik *root* utama dari file AndroidManifest.xml yang berisi atribut *package* aplikasi serta paket *activity*. Dari contoh sintaks diatas atribut *package* adalah *com.wilis.uicoding1* beserta *versionCode* yang digunakan, sedangkan atribut *activity*-nya adalah *uicoding1*.

**<uses-permission>**

Menjelaskan tentang *user permission/security permission* yang harus diberikan agar aplikasi dapat berjalan sebagaimana mestinya, misalnya apabila mengirim sms.

<uses-permission android.name=”android.permission.RECEIVE\_SMS”/>

**<permission>**

Menjelaskan batasan tentang user permission/security permission.

**<instrumentation>**

Mendeklarasikan komponen instrument yang tersedia untuk menguji fungsionalitas dari paket aplikasi yang digunakan dalam aplikasi android.

**<application>**

Elemen root yang berisi deklarasi aplikasi android. Contoh :

<application android:icon=*”@drawable/icon”>*

**<intent filter>**

Mendeklarasikan *intent* yang dibutuhkan oleh aplikasi android, atribut-atribut bisa diberikan disini untuk memasok label, icon, dan data yang akan digunakan dalam aplikasi android yang dirancang.

**<action>**

Berisi tentang tipe *action* yang didukung oleh komponen-komponen yang terdapat dalam aplikasi android yang dirancang.

**<category>**

Mendeklarasikan kategori-kategori yang didukung oleh aplikasi android.

**<data>**

Mendeklarasikan tipe MIME, URL, authority penggunaan URL, dan penentuan *path* yang digunakan dalam URL.

**<meta-data>**

Mendeklarasikan meta data yang dibutuhkan sebagai tambahan data yang akan digunakan dalam aplikasi android yang dirancang.

**<receiver>**

Mendeklarasikan suatu perubahan atau aksi yang terjadi pada aplikasi dalam bentuk informasi, misalnya menerima sms.

**<service>**

Mendeklarasikan komponen yang dapat berjalan sebagai service (berjalan di bagian *background*).

**<provider>**

Mendeklarasikan komponen-komponen yang mengelola data dan mempublikasikannya untuk dikelola/dipakai oleh aplikasi lain.

**<uses-sdk android:minSdkVersion = ?>**

Mendeklarasikan versi sdk Android yang digunakan, bisa juga menentukan versi minimum dari sdk Android yang digunakan oleh aplikasi tersebut.

Berikut ini contoh aplikasi UIcoding1 untuk menampilkan teks “Hello, saya TextView” dan “Hello, saya Button” pada *layout* handphone.

Project Name : UIcoding1

Built Target : Android 4.2.2

Application Name : uicoding1

Package Name : com.wilis.uicoding1

Activity : uicoding1

Min SDK : 8

***uicoding1.java***

**package** com.wilis.uicoding1;

**import** android.app.Activity;

**import** android.os.Bundle;

**public class** uicoding1 **extends** Activity {

/\*\* Called when the activity is first created. \*/

@override

**public void** onCreate(Bundle savedInstanceState){

**super**.onCreate(savedInstanceState);

setContentView(R.layout.main);

}

}

Sintaks uicoding1.java diatas berarti hanya menampilkan isi dari main.xml, hal ini terlihat pada sintaks coding setContentView(R.layout.main), juga dapat diartikan bahwa *user interface* aplikasi dikontrol sepenuhnya di main.xml

***main.xml***

<?xml version=”1.0” encoding=”utf-8”?>

<LinearLayout

xmlns:android=<http://schemas.android.com/apk/res/android>

android:layout\_width=”*fill\_parent*”

android:layout\_height=”*fill\_parent*”

android:orientation=”*vertical*”>

<TextView android:id=”@+id/text”

android:layout\_width=”*wrap\_content*”

android:layout\_height=”*wrap\_content*”

android:text=”*Hello, saya TextView*” />

<Button android:id=”@+id/button”

android:layout\_width=”*wrap\_content*”

android:layout\_height=”*wrap\_content*”

android:text=”*Hello, saya Button*” />

</LinearLayout>

Sintaks main.xml diatas terdiri dari 2 komponen yaitu TextView dan Button. TextView berupa tulisan “Hello, saya TextView” dengan nama variabel *id/text* serta Button dengan tulisan “Hello, saya Button” dengan variabel *id/button*. Kedua komponen itu juga ditampilkan dengan layout *“wrap\_content”* yang akan membuat tampilan elemen tersebut selebar dan setinggi tulisan/teks dari elemen tersebut, sedangkan layout *“fill\_parent”* akan membuat tulisan/teks mengisi layar dari ruang elemen tersebut secara penuh.

Pemberian nama variabel dari setiap komponen di dalam main.xml diawali dengan tanda @+, mengapa ? dan apakah maksudnya ?

***string.xml***

<?xml version=”1.0” encoding=”utf-8”?>

<resources>

<string name=”*app\_name*”>uicoding1</string>

</resources>

String.xml cukup berisi satu variabel saja, yaitu variabel *app\_name*. Variabel tersebut kemudian dipanggil di file AndroidManifest.xml dengan perintah coding android:label=”@string/app\_name” untuk menampilkan teks uicoding1 sebagai tampilan nama aplikasi. Namun, dapat pula file string.xml ini dihapus atau tak perlu digunakan karena nama aplikasi dapat dituliskan langsung pada coding file AndroidManifest.xml, misalkan android:label=”uicoding1”. Dengan demikian file AndroidManifest.xml tidak membutuhkan string.xml lagi untuk mengisi android label-nya.

***Android.Manifest.xml***

<?xml version=”1.0” encoding=”utf-8”?>

<manifest

xmlns:android=”*http://schemas.android.com/apk/res/android”*

package=”com.wilis.coding1”

android:versionCode=”1”

android:versionName=”1.0”>

<application android:icon=*”@drawable/icon”*

android:label=”*@string/app\_name*”>

<activity android:name=”*.uicoding1*”

android:label=”*@string/app\_name*”>

<intent-filter>

<action android:name=”*android.intent.action.MAIN*” />

<category android:name=”*android.intent.category.LAUNCHER*” />

</intent-filter>

</activity>

</application>

</manifest>

Pada coding file Android.Manifest.xml diatas terdapat 2 intent yang umumnya wajib dimiliki oleh aplikasi android yaitu :

action android:name=”*android.intent.action.MAIN*” dan

category android:name=”*android.intent.category.LAUNCHER*”.

Perubahan atau penambahan komponen di dalam *layout* (main.xml) tidak harus dilakukan dengan *coding manual*, namun bisa juga dilakukan dengan memindahkan (*click and drag*) komponen-komponen apa saja yang dibutuhkan di dalam *layout* (man.xml) pada editor eclipse.

Capaian Pembelajaran :

* Mahasiswa dapat mengetahui dan menjelaskan *activity* dan *activity stack*
* Mahasiswa dapat mengetahui dan menjelaskan metode-metode dan fungsi *activity*
* Mahasiswa dapat mengetahui dan menjelaskan contoh penggunaan *activity* pada aplikasi android

**ACTIVITY**

* **Konsep activity**

*Activity* merupakan *public class* dalam aplikasi Android yang bertujuan untuk menangani berbagai macam hal yang bisa dilakukan oleh *user*. Umumnya *activity* berhubungan dengan apa yang akan dilakukan oleh *user* sehingga *activity* dirancang untuk menampilkan *windows* atau *user interface* (UI) dengan menggunakan sintaks *setContentView*. Ada 2 metode yang dimiliki oleh setiap *activity*, yaitu :

1. onCreate, untuk menginisiasi suatu activity. Biasanya dipanggil dengan perintah *setContentView(int)* untuk resource yang didefinisikan pada layout UI, dan perintah *findViewById(int)* untuk memanggil *widget* yang dibutuhkan UI agar dapat berinteraksi dengan aplikasi.
2. onPause, untuk menyatakan ketika *user* meninggalkan suatu *activity*.

*Activity* aplikasi android dikelola dengan sistem yang disebut dengan *activity stack*. Ketika suatu *activity* dalam posisi start, maka *activity* diletakkan pada *stack* yang teratas dan *activity-activity* yang telah berjalan sebelumnya berada di posisi bawahnya dan akan terus berada pada posisi tersebut hingga muncul *activity* yang baru. Pada dasarnya *activity* memiliki 4 kondisi, yaitu :

1. *Active/running,* jika *activity* berada pada posisi atas *stack*.
2. *Pause,* jika *activity* tidak dipakai atau dibutuhkan pada suatu saat tertentu, tetapi *activity* itu masih ada. Ketika *activity* baru yang ditangani oleh system, maka *activity* yang alam disebut pause dan masih berada di memory. Namun, bisa jadi suatu *activity* yang sudah keadaan pause tidak ada di memory yang kemungkinan disebabkan oleh keterbatasan kapasitas memory.
3. *Stopped*, jika suatu *activity* sudah tidak dipakai dan digantikan oleh *activity* lain. Suatu *activity* yang dalam kondisi stopped tidak akan pernah dipanggil lagi dan secara permanen info mengenai *activity* ini tidak disimpan di memory.
4. *Restart,* jika suatu *activity* dalam kondisi pause atau stopped, namun dibutuhkan kembali, maka sistem dapat memanggil kembali *activity* tersebut dari sistem memory dalam kondisi awal *activity*. Keadaan ini disebut *activity* mengalami proses restart.

*Activity* merupakan bagian penting dari keseluruhan siklus aplikasi, bagaimana *activity* dijalankan merupakan bagian penting dari model aplikasi. Berikut ini siklus *activity* yang terdapat dalam aplikasi android :

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Description** | **Killable** | **Next** |
| **onCreate()** | Ketika suatu activity dibuat, pada metode inilah dilakukan inisialisasi seperti create view, list data, dll. Metode **onCreate**()dapat diikuti oleh **onStart**() | N | **onStart()** |
| **onRestart()** | Ketika suatu activity dihentikan dan merupakan prioritas untuk memanggil activity itu kembali. Metode ini dapat juga diikuti oleh metode **onStart**() | N | **onStart()** |
| **onStart()** | Ketika suatu activity dipanggil sebelum diperlihatkan ke user, metode ini diikuti oleh **onResume**() atau **onStop**() | N | **onResume()**  **or**  **onStop()** |
| **onResume()** | Ketika suatu activity mulai melakukan interaksi dengan user, maka activity itu berada pada posisi teratas dari activity stack dimana saat user melakukan input. Metode ini diikuti oleh **onPause**() | N | **onPause()** |
| **onPause()** | Ketika suatu activity lainnya dipanggil atau dimulai. Metode ini digunakan jika data tidak perlu disimpan ke ssstem secara permanen, metode ini diikuti oleh **onResume**() atau **onStop**() | Y | **onResume()**  **or**  **onStop()** |
| **onStop()** | Ketika suatu activity tidak lagi dibutuhkan atau tidak terlihat lagi oleh user, metode ini diikuti oleh **onRestart**() atau **onDestroy**() | Y | **onRestart()**  **or**  **onDestroy()** |
| **onDestroy()** | Ketika suatu activity secara permanen tidak lagi dibutuhkan (activity dihapus), dapat juga dilakukan dengan fungsi *finish()* atau *isFinishing()* | Y |  |

Berikut contoh implementasi *activity* dalam suatu aplikasi android berupa program aplikasi jam :

Project Name : Jam

Built Target : Android 2.3

Application Name : jam

Package Name : com.wilis.jam

Activity : jam

Min SDK : 9

***jam.java***

**package** com.wilis.jam;

**import** android.app.Activity;

**import** android.os.Bundle;

**import** android.view.View;

**import** android.widget.Button;

**import** java.util.Date;

**public class** jam **extends** activity **implements** View.onClickListener

{

Button btn;

@Override

**public void** onCreate(Bundle icicle){

**super**.onCreate(icicle);

setContentView(R.layout.*main*);

btn=(Button)findViewById(R.id.*button*);

btn.setOnClickListener(**this**);

updateTime();

setContentView(btn);

}

**public void** onClick(View view) {

updateTime();

}

**private void** updateTime() {

btn.setText(**new** Date().toString());

}

}

Package deklarasi adalah *package* yang wajib dideklarasikan saat pertama melakukan *create project*. Seperti pemrograman java, biasanya dibutuhkan import beberapa *class* yang memiliki fungsi yang dibutuhkan dalam aplikasi. Biasanya *class* inilah yang dikenal dengan android package.

Activity yang merupakan *public class* jam diwariskan dari android app.Activity, perlu diingat bahwa tidak semua *class* yang bisa didefinisikan di java dapat didefinisikan di android.

Kemudian terdapat 2 public void dan 1 private void. Metode untuk memulai activity menggunakan onCreate(), dimana activity telah didefinisikan di dalam superclass. Dalam aplikasi juga membuat (create) 1 buah objek Button (btn) yang atributnya telah diatur pada file *main.xml* dan hanya dipanggil saja untuk ditampilkan melalui perintah *setContentView(R.layout.main)* dan *btn=(Button)findViewById(R.id.button)*. Apabila button click beraksi, maka akan dikirm ke activity sesuai dengan sintaks *btn.setOnClickListener(this)* yang selanjutnya akan memanggil fungsi *updateTime()*. Fungsi *onClick()* akan membuat setiap kali di-*click* maka akan melakukan fungsi *updateTime()* yang akan ditampilkan ke layar. Dan jika melakukan *onCreate()* atau user melakukan *onClick()* akan terjadi pembaharuan waktu yang diambil dari hasil proses pada fungsi *Date().toString()*.

***main.xml***

<?xml version=”1.0” encoding=”utf-8”?>

<Button xmlns:android=<http://schemas.android.com/apk/res/android>

android:id=”*@+id/button*”

android.text=””

android:layout\_width=”*fill\_parent*”

android:layout\_height=”*fill\_parent*”

/>

Pada coding file main.xml diatas, nama varibel untuk objek Button (btn) yang telah dikirimkan pada file jam.java diberikan id *value* = @+id/button. Selain itu, id/button adalah *identifier* untuk button widget *setContentView(R.layout.main)* akan membuat R.java akan diperbaharui (update) ketika aplikasi dalam keadaan *run* dengan referensi layout file main.xml yang berada pada res/layout.

***Android.Manifest.xml***

<manifest

xmlns:android=<http://schemas.android.com/apk/res/android>

package=”com.wilis.jam”>

<application>

<activity android:name=”.jam” android:label=”jam”>

<intent-filter>

<action android:name=”android.intent.action.MAIN” />

<category android:name=”android.intent.category.LAUNCHER” />

</intent-filter>

</activity>

</application>

</manifest>

Pada coding file Android.Manifest.xml diatas nama activity = .jam dan terdapat 2 intent yang dimiliki oleh aplikasi jam diatas yaitu action.MAIN dan category LAUNCHER.

Capaian Pembelajaran :

* Mahasiswa dapat mengetahui dan menjelaskan konsep penggunaan *widget* pada aplikasi android
* Mahasiswa dapat mengetahui dan menjelaskan *layout management*, *containers*, beberapa *classes widget* beserta fungsinya

**ANDROID WIDGET**

* **Konsep Widget**

Paket widget pada dasarnya merupakan visualisasi dari elemen user interface (UI) yang digunakan pada layar aplikasi android dan dapat dirancang sendiri sesuai kebutuhan aplikasi yang akan dibuat. Widget di dalam android ditampilkan dengan konsep *View*. Dimana aplikasi android pada umumnya menggunakan widget sebagai *layout xml*. Untuk mengimplementasikan widget, maka dibutuhkan file-file berikut :

1. File Java

Merupakan file yang mengimplementasikan aksi dari widget. Bila mendefinisikan suatu widget beserta letak posisinya di layar melalui file xml, maka harus melakukan coding secara manual di file java.

1. File xml

File xml yang berada di res/values/ yang mendefinisikan komponen elemen-elemen xml yang digunakan untuk inisialisasi widget serta atribut yang mendukungnya

1. Layout xml (optional)

File ini tidak harus ada, file ini berada dalam res/layout/ yang menggambarkan atau penambahan keterangan pada layout widget. File ini dikenal dengan string.xml, dimana fungsinya dapat diambil alih oleh file java-nya.

* **Jenis Android Widget**

Terdapat berbagai jenis (*Classes*) android widget dengan fungsi yang berbeda-beda, diantaranya widget TextView berfungsi untuk menampilkan teks ke layar, widget EditText berfungsi untuk mengkustomisasi TextView sehingga dapat diedit, widget RadioButton/RadioGroup berfungsi untuk menampilkan pilihan (opsi) yang hanya dapat dipilih sebanyak satu pilihan, widget ImageView berfungsi untuk menampilkan gambar seperti *icon*, widget ImageButton berfungsi menampilkan tombol bergambar yang bisa diklik oleh *user*, dan masih banyak lagi jenis widget lainnya.

Kumpulan widget yang memiliki struktur tersendiri-sendiri disebut dengan *contrainers*. Jika anda mendesain sebuah aplikasi yang memakai beberapa widget, maka anda harus menentukan *layout* yang digunakan. Seperti kebanyakan GUI yang memiliki *layout management* untuk menampilkan *containers*, seperti layout BoxLayout di Java.

Android juga memiliki manajemen tampilan pada layar handphone (*layout* *management*) yaitu :

1. Linear Layout

Untuk menampilkan *widgets* atau *containers* dalam posisi baris atau kolom satu persatu. Konfigurasi Linear Layout mempunyai 5 konten yaitu :

1. Orientation

Konten ini mengindikasikan layout ditampilkan berdasarkan baris ataukah kolom, sintaksnya adalah android.orientation=”value”

Jika berdasarkan baris, maka valuenya adalah horizontal. Namun jika berdasarkan kolom, maka valuenya adalah vertikal

1. Fill Model

Konten ini adalah bentuk tampilan elemen/widgets terhadap dimensi ruang (*space*) yang tersedia. Terdiri atas 3 pilihan, yaitu :

* Spesific dimension, misalkan 125px untuk mengisi tampilan widgets berukuran 125 pixel
* wrap\_content, widgets akan mengisi posisi dengan konsep *natural space*
* fill\_parent, widgets akan mengisi posisi yang tersedia

1. Weight

Konten ini mengatur android.layout\_width (untuk baris) dan android.layout\_height (untuk kolom)

1. Gravity

Konten ini secara default pada tampilan dengan Linear Layout adalah “Left and top aligned”, namun jika menginginkan tampilan yang berbeda maka *gravity* bisa diatur sesuai kebutuhan. Value *gravity* yang lainnya adalah left, center\_horizontal, right for left-aligned, dan right-aligned.

1. Padding

Konten ini memiliki 4 value standar, yaitu paddingLeft, paddingRight, paddingTop, dan paddingBottom.

1. Relative Layout

Untuk menempatkan tampilan salah satu widget di posisi bawah sebelah kiri atau kanan dari widget lainnya atau ditempatkan di posisi lainnya.