MODUL PRAKTIKUM PEMROGRAMAN UNTUK PERANGKAT BERGERAK 1

Indra Azimi, S.T., M.T.

Reza Budiawan, S.T., M.T., OCA

Rizza Indah Mega Mandasari, S.Kom., M.T.

Cahyana S.T., M.Kom.

D3 Rekayasa Perangkat Lunak Aplikasi
Telkom University

Daftar Isi

Modul 02: Build Your First App

1. Overview

Pada modul kali ini, kita akan membuat sebuah aplikasi sederhana bernama “Galeri Hewan”. Aplikasi ini menampilkan sebuah gambar, sebuah teks dan sebuah tombol. Gambar dan teks yang muncul akan berubah jika tombol tersebut ditekan. Aplikasi yang kita buat dapat dilihat versi lengkapnya di repository Github https://github.com/indraazimi/galeri-hewan pada branch first-app.

Tampilan aplikasi yang akan dibuat adalah sebagai berikut:

2. Getting Started

Pertama, jalankan Android Studio, lalu buat project baru bernama “Galeri Hewan”. Tuliskan “org.d3ifxxxx.galerihewan” sebagai nama package. Pilih minimum SDK 21, dengan Empty Activity sebagai template awal. Silahkan baca kembali modul pertama jika diperlukan. Berikutnya, kita akan menambahkan beberapa komponen pada project Android Studio ini. Silahkan ikuti task berikut langkah per langkah.

Dilarang keras untuk copy – paste kode dari modul/sumber lain!

Ngoding pelan-pelan akan membuat kamu lebih jago di masa depan. Selamat ngoding!

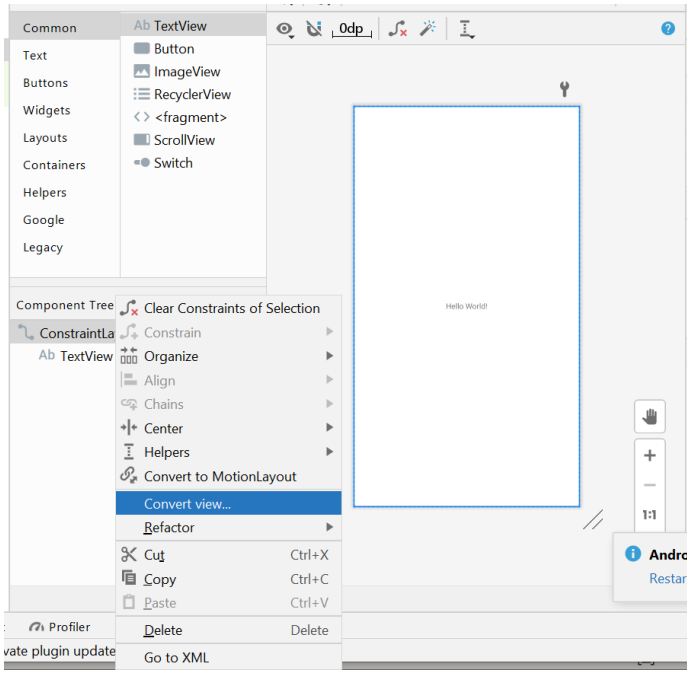
3. Task

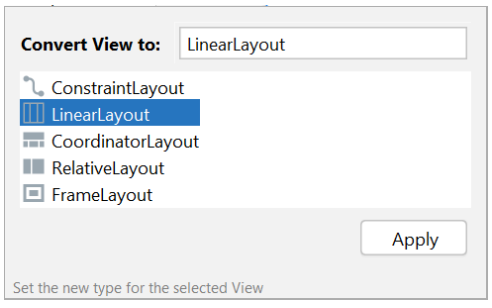
3.1. Menambahkan Tombol Next

Pada task ini, kita akan menambahkan sebuah tombol. Akan tetapi, sebelum melakukan hal tersebut, ubah dulu layout pada activity_main.xml dari ConstraintLayout menjadi LinearLayout. Secara default, saat pertama kali project dibuat, ConstraintLayout merupakan layout awal yang digunakan. Tapi kali ini kita akan menggunakan layout yang lebih simple, yaitu LinearLayout.

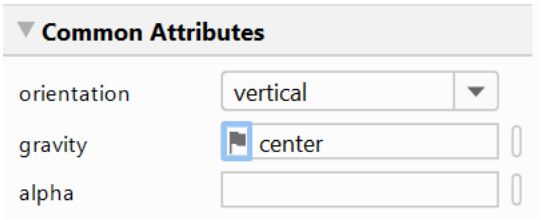
LinearLayout adalah layout yang memungkinkan pengaturan komponen tersusun urut secara linear, baik itu vertikal (dari atas-ke-bawah) atau horizontal (dari kiri-ke-kanan). Karena sifatnya yang linear, LinearLayout tidak bisa digunakan untuk membuat layout yang bertumpuk (misal, ada teks di atas gambar). Secara default orientasi dari LinearLayout adalah horizontal.

Buka activity_main.xml pada Android Studio project, pada bagian “Component Tree” klik kanan pada ConstraintLayout, dan pilih “Convert view…” Berikutnya pada pop-up yang muncul, pilih LinearLayout dan klik tombol Apply.



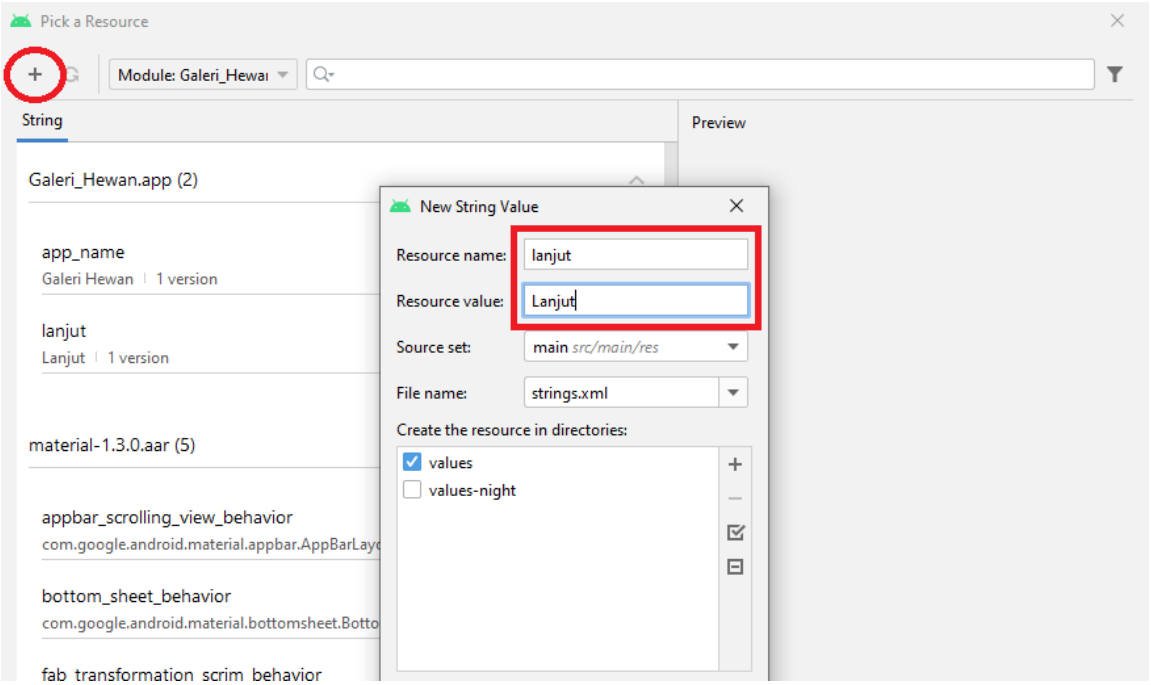


Tambahkan pengaturan “gravity” dan “orientation” pada LinearLayout ini. Di bagian properties, dengan LinearLayout terpilih pada Component Tree, atur gravity menjadi center, dan orientation menjadi vertical.

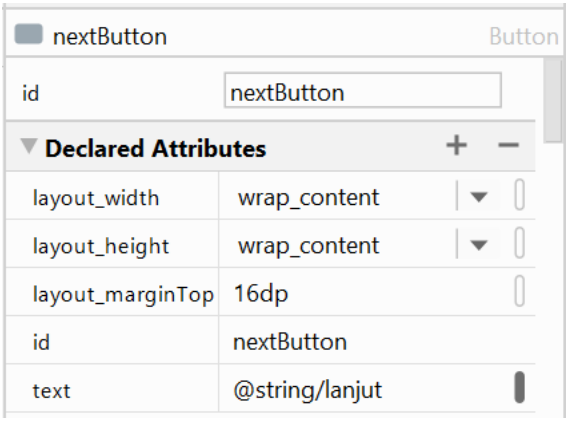


Selanjutnya, drag-n-drop komponen Button pada bagian Pallete ke Canvas. Atur properties pada Button dengan konfigurasi berikut:

- Berikan id “nextButton”.
- Atur lebar (layout_width) dan tinggi (layout_height) menjadi wrap_content. Ini agar ukuran tombol menyesuaikan dengan ukuran teksnya.
- Isi layout_marginTop dengan nilai 16dp agar ada jarak antara button dan text. dp adalah satuan untuk menyatakan dimensi/jarak di Android.
- Berikan teks “Lanjut”. Caranya klik tombol kecil di bagian kanan isian text. Di dialog yang muncul, klik tombol tambah, pilih menu “String value”, isikan resource name dan value-nya, lalu klik OK. Jadi teks **tidak boleh** ditulis langsung pada properties, tapi harus diambil dari strings.xml



Berikut adalah hasil akhir pengaturan tombol.



Sekarang cek bagian kode dari activity_main.xml. Apakah sudah sama dengan kode berikut?

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />

    <Button
        android:id="@+id/nextButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="@string/lanjut" />

</LinearLayout>
```

Pastikan juga kode dari strings.xml sudah sama seperti ini:

```
<resources>
    <string name="app_name">Galeri Hewan</string>
    <string name="lanjut">Lanjut</string>
</resources>
```

Setelah mengatur tampilan, tambahkan aksi saat pengguna menekan Button. Hal ini dilakukan pada MainActivity.kt. Ubah kode dari file tersebut dengan kode berikut. Pastikan untuk menggunakan auto-complete yang disediakan Android Studio untuk menghindari salah ketik. Perhatikan juga penggunaan huruf besar/kecil, karena Kotlin bersifat case-sensitive. Kode berwarna abu-abu adalah kode default saat MainActivity.kt dibuat pertama kali.

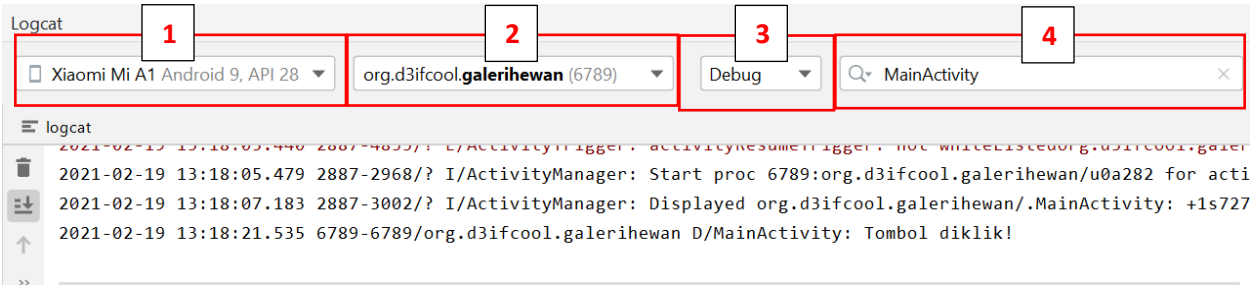
```
import android.os.Bundle
import android.util.Log
import android.widget.Button
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val nextButton: Button = findViewById(R.id.nextButton)
        nextButton.setOnClickListener { showNext() }
    }

    private fun showNext() {
        Log.d("MainActivity", "Tombol diklik!")
    }
}
```

Pada kode di atas, kita menambahkan sebuah method berupa showNext(). Method ini akan menampilkan tulisan “Tombol diklik!” di bagian Logcat dari Android Studio. Method showNext() ini dipanggil pada kode

```
val nextButton: Button = findViewById(R.id.nextButton)
nextButton.setOnClickListener { showNext() }
```

Maksud dari kode di atas adalah, kita mengambil komponen button pada layout XML dengan id “nextButton”, dan menjadikannya sebagai objek Kotlin. Objek Kotlin ini disimpan di variabel nextButton lalu ditambahkan aksi berupa tekan tombol (setOnClickListener). Aksi yang dilakukan adalah method showNext().

Jika sudah selesai menuliskan kode di atas, jalankan project di smartphone/emulator masing-masing. Tekan tombol klik, tidak akan muncul apa-apa di layar smartphone. Tulisan “Tombol diklik!” akan muncul di bagian Logcat (cari di bagian bawah Android Studio).



Pada bagian Logcat terdapat 4 hal yang perlu diperhatikan:

1. Pada area 1, pastikan memilih smartphone/emulator tempat aplikasi dijalankan
2. Pada area 2, pilih aplikasi yang sedang berjalan untuk di-debug
3. Pada area 3, pilih “Debug”. Pemilihan ini bergantung pada mode Log yang dituliskan. Pilih “Debug” karena kode yang dituliskan adalah Log.**d**
4. Pada area 4, tuliskan MainActivity, karena itu adalah parameter pertama kita pada kode Log.d

3.2. Menampilkan Nama Hewan Sesuai Urutan

Task berikutnya adalah menampilkan nama hewan sesuai urutan. TextView akan menampilkan nama hewan sesuai list yang telah ditentukan ketika tombol diklik. Langkah pertama pada task ini adalah mengubah tampilan. Berikan id “nameTextView” dan teks “Ayam” sebagai nama hewan default yang ditampilkan ketika aplikasi dijalankan.

Untuk melakukan hal ini, kita bisa menggunakan design view seperti ketika memberi teks ke tombol di Task 3.1. Atau langsung mengedit kode activity_main.xml dan strings.xml menjadi seperti berikut, agar lebih terasa programmernya 😊

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    ...
    <TextView
        android:id="@+id/nameTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hewan_default" />
    ...
</LinearLayout>
```

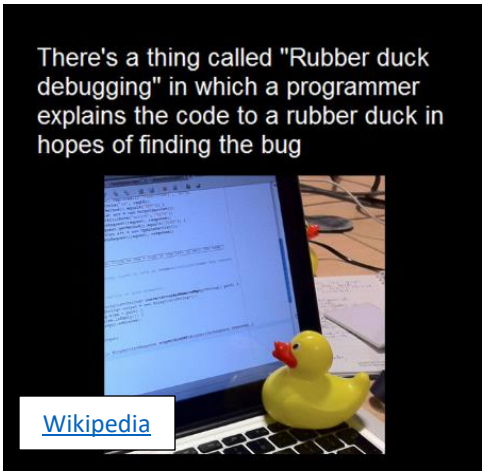
```
<resources>
    ...
    <string name="hewan_default">Ayam</string>
</resources>
```

Berikutnya, ubah kode MainActivity.kt sebagai berikut. Kode yang dicoret berarti harus dihapus.

```
import android.os.Bundle
import android.util.Log
import android.widget.Button
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
    private val hewan = listOf("Ayam", "Bebek", "Domba", "Kambing", "Sapi")
    private var index = 0

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

        val nextButton: Button = findViewById(R.id.nextButton)
        nextButton.setOnClickListener { showNext() }
    }

    private fun showNext() {
        Log.d("MainActivity", "Tombol diklik!")
        index++
        val textView: TextView = findViewById(R.id.nameTextView)
        textView.text = hewan[index]
    }
}
```

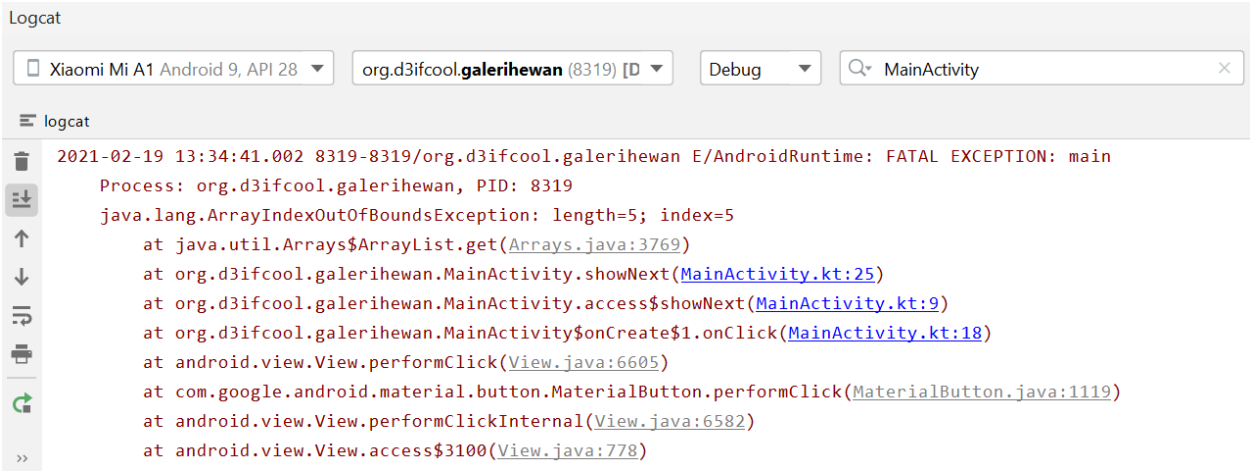
Pada MainActivity.kt, kita membuat sebuah list yang berisi daftar nama hewan yang akan ditampilkan ke aplikasi. Pada method showNext(), kita tidak lagi menampilkan tulisan ke Logcat, tapi menampilkan list tersebut pada TextView secara terurut dimulai dari index 0, dan bergerak sebanyak 1, hingga seterusnya.. Perhatikan penggunaan val dan var. Saat mendeklarasikan variabel, kita menggunakan var jika variabel tersebut akan kita ubah nilainya ketika aplikasi dijalankan.

Jalankan aplikasi ini, dan ketika pertama kali dijalankan, pada tulisan tampil nama hewan “Ayam”. Ketika tombol ditekan, maka nama hewan akan berubah sesuai urutan dari list. Dan setelah menampilkan “Sapi” di layar, jika tombol ditekan lagi, maka aplikasi akan terhenti. Hmm.. Kenapa ya?



3.3. Debugging Error & Perbaikannya

Ketika aplikasi berhenti atau crash, hal pertama yang harus dilakukan adalah jangan panik. Buka kembali Logcat di Android Studio, lalu cari tau error apa yang menyebabkan aplikasi terhenti. Skill ini sangat penting karena aplikasi error mungkin akan sering terjadi ketika masih belajar Android. Carilah pesan berikut di Logcat:



Perhatikan baris ke-3. Dari pesan Logcat tersebut kita menjadi tahu bahwa telah terjadi error "ArrayIndexOutOfBoundsException", atau dengan kata lain, kita mengakses list of hewan dengan indeks yang melebihi kapasitasnya. Hal ini dikarenakan setelah "Sapi", tidak ada nama hewan lagi pada list. Error ini terjadi di MainActivity.kt sekitar baris 25.

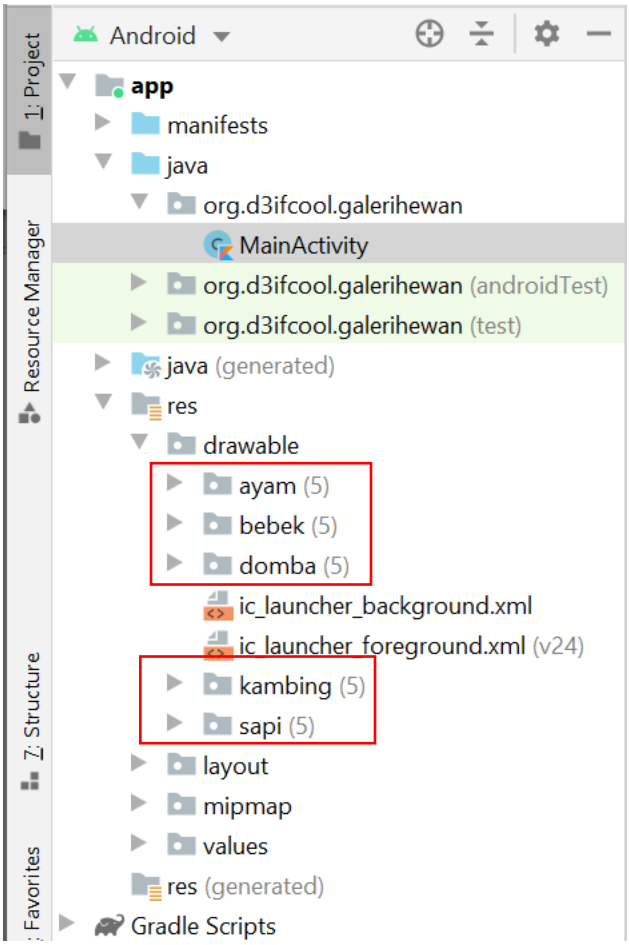
Error di atas dapat kita tangani dengan cara menggunakan pengkondisian. Ketika indeks yang diakses telah mencapai indeks maksimal (size dari list dikurangi 1), maka kembalikan indeks ke nilai 0. Jika tidak, lakukan penambahan seperti biasa. Penanganan ini kita lakukan dengan mengubah kode MainActivity.kt pada method showNext().

```
private fun showNext() {
    index++
    index = if (index == hewan.size-1) 0 else index + 1
    val textView: TextView = findViewById(R.id.nameTextView)
    textView.text = hewan[index]
}
```

Jalankan kembali aplikasi di smartphone/emulator. Pastikan aplikasi tidak error setelah tombol ditekan, dan setelah "Sapi" dimunculkan, maka hewan yang ditampilkan berikutnya adalah "Ayam".

3.4. Menambahkan Gambar Hewan

Pada task ini kita akan menambahkan gambar pada aplikasi. Gambar pada aplikasi Android ditampilkan menggunakan komponen ImageView. Sebelum menambahkan komponen ImageView, tambahkan dulu gambar yang ingin ditampilkan (file Zip terlampir). Simpan ke dalam folder res, sehingga pada drawable di Android Studio akan terlihat file sebagai berikut:



Smartphone Android tersedia dalam berbagai ukuran layar dan kerapatan (density) piksel. Jadi agar gambar yang tampil di layar tidak pecah, kita perlu menyiapkan beberapa versi gambar. Mulai dari density medium (mdpi) sampai ke yang tertinggi (xxxhdpi). Itulah sebabnya mengapa setiap gambar hewan disediakan dalam 5 versi. Lebih lanjut, silahkan baca Provide alternative bitmaps di web resmi dokumentasi Android.

Berikutnya, tambahkan komponen ImageView sebelum komponen TextView pada activity_main.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    ...

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="132dp"
        android:layout_height="132dp"
        android:layout_marginBottom="16dp"
        android:contentDescription="@string/gambar_hewan"
        android:src="@drawable/ayam" />

    <TextView
        ...
</LinearLayout>
```

Atribut contentDescription pada ImageView di atas berfungsi untuk mendukung pengguna yang memiliki keterbatasan penglihatan. Seperti biasa, jangan tulis teks langsung di layout, tapi gunakan string dari strings.xml. Tambahkan kode berikut pada strings.xml.

```
<resources>
    ...
    <string name="gambar_hewan">Gambar hewan</string>
</resources>
```

Setelah selesai mengatur tampilan, berikutnya, tampilkan gambar di ImageView sesuai list. Hal ini dilakukan dengan mengubah kode MainActivity.kt sebagai berikut.

```
import android.widget.ImageView

class MainActivity : AppCompatActivity() {
    ...
    private fun showNext() {
        index = if (index == hewan.size-1) 0 else index + 1

        val imageView: ImageView = findViewById(R.id.imageView)
        val resourceId = when(index) {
            1 -> R.drawable.bebek
            2 -> R.drawable.domba
            3 -> R.drawable.kambing
            4 -> R.drawable.sapi
            else -> R.drawable.ayam
        }
        imageView.setImageResource(resourceId)
```

```
        val textView: TextView = findViewById(R.id.nameTextView)
        textView.text = hewan[index]
    }
}
```

Pada kode di atas, terlihat bahwa kita telah mengambil komponen ImageView pada XML dan menjadikannya sebagai objek dari Kotlin. Pada ImageView, gambar yang ditampilkan berubah-ubah tergantung indeksnya. Hal ini dilakukan dengan menggunakan kode “when”. Kode ini mirip seperti “switch-case” di bahasa pemrograman Java. Pada kode when, kita memilih resourceId yang akan ditampilkan, dan setelah dipilih, akan dimunculkan pada ImageView menggunakan perintah “setImageResource”.

Jalankan kembali aplikasi di smartphone/emulator, dan perhatikan saat menekan tombol, akan muncul gambar dan teks sesuai dengan yang sudah dituliskan logikanya di kode Kotlin. Jika semua sudah benar, maka selamat! Kamu telah berhasil menyelesaikan modul ini dengan baik.

4. Summary

Pada modul kali ini kita telah membuat sebuah aplikasi Android menggunakan beberapa komponen yaitu: LinearLayout, ImageView, TextView, dan juga Button. Selain itu, kita juga telah melakukan pemberian aksi pada tombol yang berpengaruh pada gambar dan teks yang dimunculkan pada aplikasi.

5. Challenge

Pada kode yang telah dibangun sebelumnya, kita telah membuat button “Lanjut”. Sebagai tantangan, buatlah button “Kembali” yang memperlihatkan gambar yang ditampilkan sebelumnya.