

LAPORAN PRAKTIKUM PENGEMBANGAN APLIKASI BERGERAK

Android Beginner 3 - Week 5



Disusun oleh:

Nama : Muhammad Ryan Fikri Fakhrezi

Nim : L0122114

Kelas : C

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

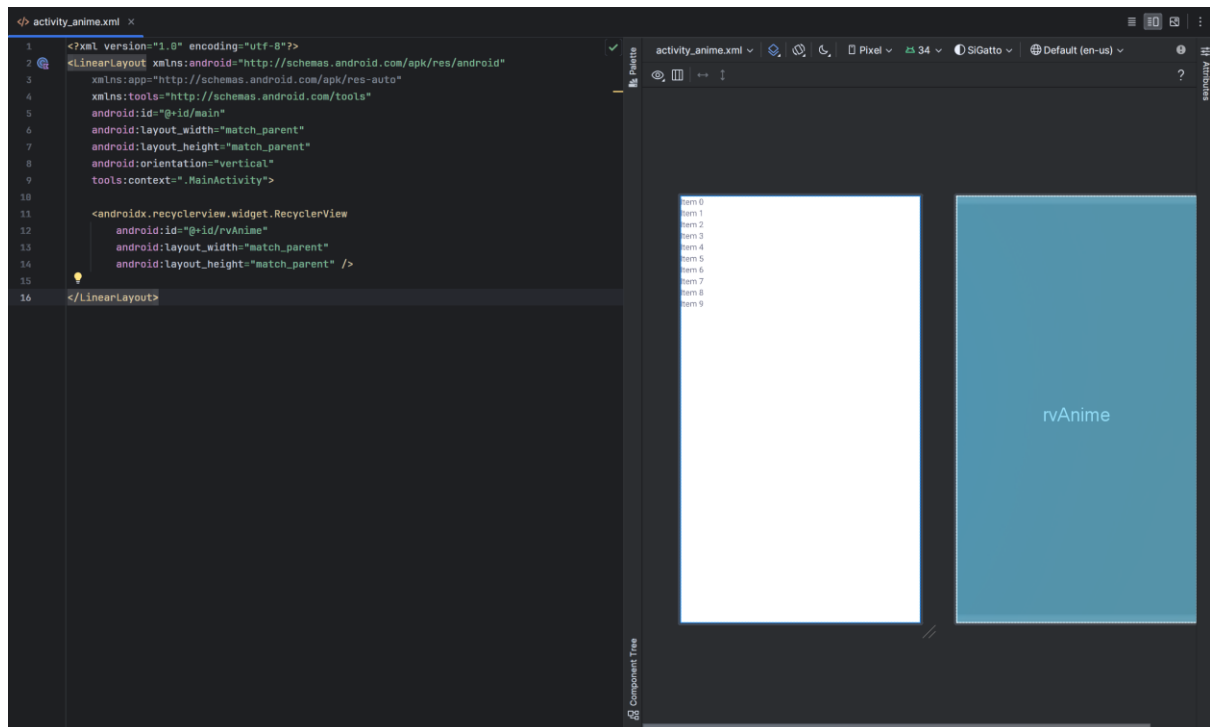
UNIVERSITAS SEBELAS MARET

2024

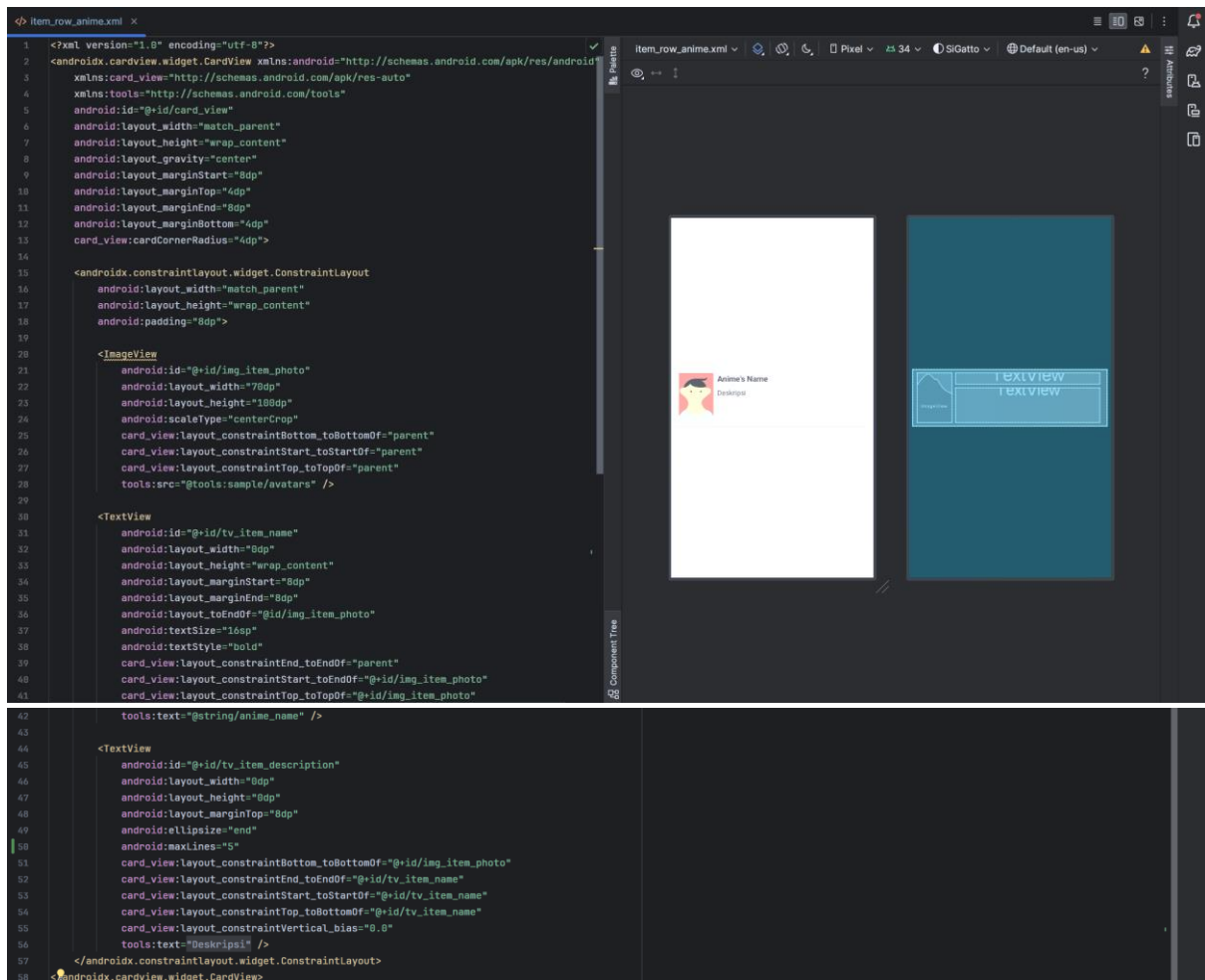
1. Screenshot Source Code

```
1 package com.yanfiq.sigetto
2
3 import android.content.Intent
4 import android.os.Bundle
5 import androidx.activity.enableEdgeToEdge
6 import androidx.appcompat.app.AppCompatActivity
7 import androidx.core.view.ViewCompat
8 import androidx.core.view.WindowInsetsCompat
9 import androidx.recyclerview.widget.LinearLayoutManager
10 import androidx.recyclerview.widget.RecyclerView
11
12 class AnimeActivity : AppCompatActivity() {
13     private lateinit var rvAnime: RecyclerView
14     private val list = ArrayList<Anime>()
15     override fun onCreate(savedInstanceState: Bundle?) {
16         super.onCreate(savedInstanceState)
17         enableEdgeToEdge()
18         setContentView(R.layout.activity_anime)
19         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
20             val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
21             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
22             insets.setOnApplyWindowInsetsListener
23         }
24
25         rvAnime = findViewById(R.id.rvAnime)
26         rvAnime.setHasFixedSize(true)
27
28         list.addAll(getListAnime())
29         showRecyclerView()
30     }
31
32     fun getListAnime(): ArrayList<Anime> {
33         val dataName = resources.getStringArray(R.array.data_anime_name)
34         val dataImg = resources.obtainTypedArray(R.array.data_anime_img)
35         val dataShortDesc = resources.getStringArray(R.array.data_anime_shortDesc)
36         val dataLongDesc = resources.getStringArray(R.array.data_anime_longDesc)
37         val listHero = ArrayList<Anime>()
38         for (i in dataName.indices) {
39             val hero = Anime(dataName[i], dataImg.getResourceId(i, R.drawable.default), dataShortDesc[i], dataLongDesc[i])
40             listHero.add(hero)
41         }
42         return listHero
43     }
44
45     private fun showRecyclerView() {
46         rvAnime.layoutManager = LinearLayoutManager(this)
47         val listAnimeAdapter = ListAnimeAdapter(list)
48         rvAnime.adapter = listAnimeAdapter
49         listAnimeAdapter.setOnItemClickListener(object : ListAnimeAdapter.OnItemClickListener {
50             override fun onItemClick(data: Anime) {
51                 seeDetails(data)
52             }
53         })
54     }
55
56     private fun seeDetails(data: Anime) {
57         val intent = Intent(this, AnimeDetailsActivity::class.java)
58         intent.putExtra("animeTitle", data.name)
59         intent.putExtra("animePict", data.img)
60         intent.putExtra("animeDesc", data.longDesc)
61         startActivity(intent)
62     }
63 }
```

- Pada metode onCreate(), layout ditetapkan menggunakan setContentView(), dan RecyclerView diinisialisasi dari layout.
- Data anime diambil dari resources menggunakan metode getListAnime().
- Metode showRecyclerView() menetapkan layout manager dan adapter RecyclerView.
- Ketika pengguna mengklik salah satu item di RecyclerView, metode seeDetails() dipanggil untuk membuka halaman detail anime.
- Metode enableEdgeToEdge() digunakan untuk mengaktifkan mode edge-to-edge, memungkinkan tampilan aplikasi menggunakan lebih banyak ruang layar.
- Dalam metode onCreate(), ViewCompat.setOnApplyWindowInsetsListener() digunakan untuk menyesuaikan padding tampilan utama sesuai dengan area sistem, seperti status bar dan navigation bar.

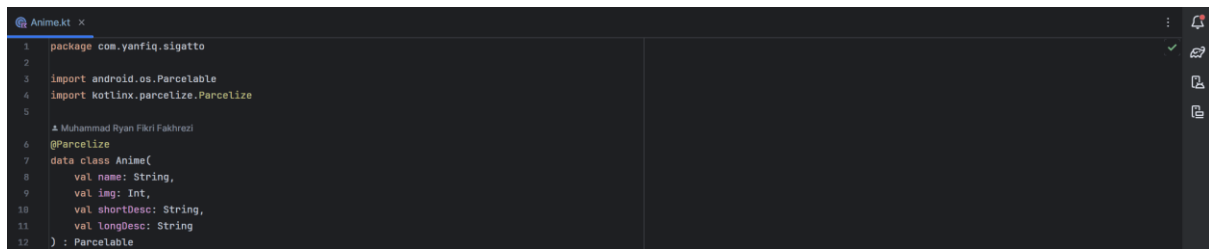


Ini adalah file layout XML untuk tata letak halaman AnimeActivity. Halaman ini menggunakan LinearLayout sebagai root view, dengan orientasi vertikal. Di dalamnya, terdapat satu RecyclerView dengan ID "rvAnime" yang akan digunakan untuk menampilkan daftar anime.



1. CardView (androidx.cardview.widget.CardView):
 - CardView digunakan sebagai kontainer untuk item.
 - cardCornerRadius digunakan untuk menentukan radius sudut card.
2. ConstraintLayout (androidx.constraintlayout.widget.ConstraintLayout):
 - ConstraintLayout digunakan sebagai parent layout untuk mengatur posisi dan tata letak elemen dalam item.
3. ImageView (android.widget.ImageView):
 - ImageView menampilkan gambar anime.
 - layout_width dan layout_height menentukan ukuran ImageView.
 - scaleType digunakan untuk menentukan bagaimana gambar akan diatur dalam ImageView.
4. TextView (android.widget.TextView):
 - tv_item_name adalah TextView untuk menampilkan nama anime.
 - tv_item_description adalah TextView untuk menampilkan deskripsi singkat anime.
 - layout_width dan layout_height menentukan ukuran TextView.
 - layout_toEndOf dan layout_toBottomOf digunakan untuk menentukan posisi relatif terhadap elemen lain dalam ConstraintLayout.
 - textSize menentukan ukuran teks.
 - textStyle menentukan gaya teks (bold).
 - ellipsize digunakan untuk menetapkan cara penyingkatan teks jika terlalu panjang.

Layout ini akan digunakan sebagai template untuk setiap item dalam RecyclerView, di mana setiap item akan menampilkan gambar anime beserta nama dan deskripsi singkatnya.



```

1 package com.yanfiq.sigatto
2
3 import android.os.Parcelable
4 import kotlinx.parcelize.Parcelize
5
6 // Muhammad Ryan Fikri Fakhrezi
7 @Parcelize
8 data class Anime(
9     val name: String,
10    val img: Int,
11    val shortDesc: String,
12    val longDesc: String
13 ) : Parcelable

```

Data class ‘Anime’ adalah model data yang digunakan untuk merepresentasikan sebuah anime dalam aplikasi.

1. name (String):
 - Properti ini menyimpan nama anime.
2. img (Int):
 - Properti ini menyimpan referensi gambar anime dalam bentuk resource ID (Integer).
 - Resource ID digunakan untuk mengambil gambar dari sumber daya aplikasi.
3. shortDesc (String):
 - Properti ini menyimpan deskripsi singkat anime.
4. longDesc (String):
 - Properti ini menyimpan deskripsi panjang anime.

Selain itu, kelas ‘Anime’ diimplementasikan sebagai Parcelable dengan menggunakan anotasi ‘@Parcelize’. Ini memungkinkan objek ‘Anime’ untuk dikirimkan melalui Intent antar komponen Android, seperti dari satu aktivitas ke aktivitas lainnya, dengan mudah.

Dengan class ‘Anime’, aplikasi dapat menyimpan dan menampilkan informasi tentang anime dalam daftar, termasuk nama, gambar, deskripsi singkat, dan deskripsi panjangnya.

```

1 package com.yanfiq.sigetto
2
3 import android.view.LayoutInflater
4 import android.view.View
5 import android.view.ViewGroup
6 import android.widget.ImageView
7 import android.widget.TextView
8 import androidx.recyclerview.widget.RecyclerView
9
10 class ListAnimeAdapter(private val listAnime: ArrayList<Anime>) : RecyclerView.Adapter<ListAnimeAdapter.ListViewHolder>() {
11
12     private lateinit var onItemClickCallback: OnItemClickCallback
13
14     class ListViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
15         val imgPhoto: ImageView = itemView.findViewById(R.id.img_item_photo)
16         val tvName: TextView = itemView.findViewById(R.id.tv_item_name)
17         val tvDescription: TextView = itemView.findViewById(R.id.tv_item_description)
18     }
19
20     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ListViewHolder {
21         val view: View = LayoutInflater.from(parent.context).inflate(R.layout.item_row_anime, parent, attachToRoot = false)
22         return ListViewHolder(view)
23     }
24
25     override fun getItemCount(): Int = listAnime.size
26
27     override fun onBindViewHolder(holder: ListViewHolder, position: Int) {
28         val (name, img, desc) = listAnime[position]
29         holder.imgPhoto.setImageResource(img)
30         holder.tvName.text = name
31         holder.tvDescription.text = desc
32         holder.itemView.setOnClickListener { onItemClickCallback.onItemClicked(listAnime[holder.adapterPosition]) }
33     }
34
35     fun setOnItemClickCallback(onItemClickCallback: OnItemClickCallback) {
36         this.onItemClickCallback = onItemClickCallback
37     }
38 }
39
40 interface OnItemClickCallback {
41     fun onItemClicked(data: Anime)
42 }

```

1. Constructor:
 - Constructor menerima ArrayList dari objek Anime sebagai parameter.
2. ListViewHolder:
 - Kelas inner yang mewakili ViewHolder untuk setiap item dalam RecyclerView.
 - ViewHolder ini memegang referensi ke elemen-elemen UI yang akan ditampilkan dalam setiap item.
3. onCreateViewHolder():
 - Metode ini dipanggil ketika RecyclerView membutuhkan ViewHolder baru untuk menampilkan item.
 - Melalui LayoutInflater, layout dari item (item_row_anime.xml) di-inflate ke dalam tampilan baru.
4. onBindViewHolder():
 - Metode ini dipanggil ketika RecyclerView perlu menampilkan data di posisi tertentu.
 - ViewHolder diikat (bind) dengan data dari listAnime pada posisi tertentu.
 - Data anime (nama, gambar, deskripsi) ditetapkan ke elemen-elemen UI dalam ViewHolder.
5. getItemCount():
 - Metode ini mengembalikan jumlah total item dalam RecyclerView, dalam hal ini, jumlah elemen dalam listAnime.
6. setOnItemClickCallback():
 - Metode ini digunakan untuk menetapkan callback yang akan dipanggil ketika salah satu item dalam RecyclerView diklik.
 - Callback ini akan memberitahu aktivitas atau fragmen yang menangani klik item tersebut.
7. OnItemClickCallback:
 - Interface yang didefinisikan di dalam adapter untuk menangani klik pada item RecyclerView.
 - Terdiri dari satu metode 'onItemClicked()' yang akan dipanggil ketika item diklik.

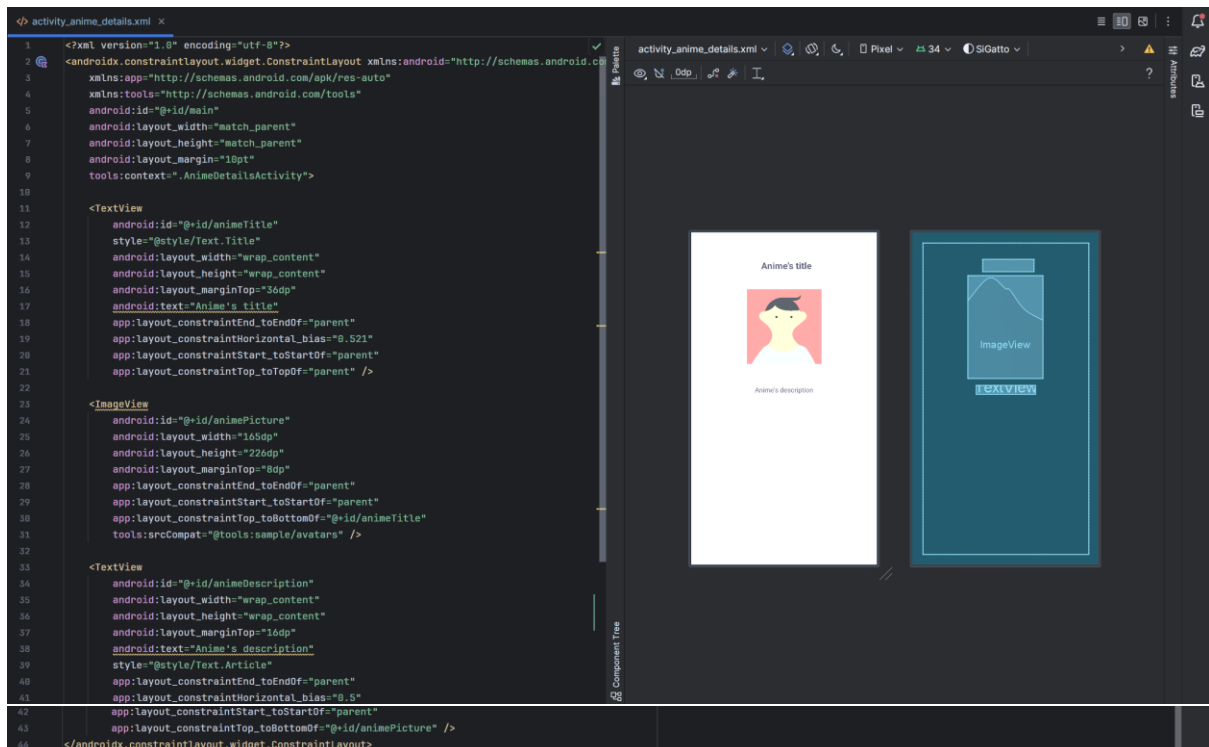
Dengan menggunakan adapter ini, data anime akan ditampilkan dalam RecyclerView dengan menggunakan layout yang telah ditentukan sebelumnya, dan pengguna dapat berinteraksi dengan item tersebut melalui callback yang ditetapkan.

```
1 package com.yanfiq.sigetto
2
3 import android.os.Bundle
4 import android.widget.ImageView
5 import android.widget.TextView
6 import androidx.activity.enableEdgeToEdge
7 import androidx.appcompat.app.AppCompatActivity
8 import androidx.core.view.ViewCompat
9 import androidx.core.view.WindowInsetsCompat
10
11 class AnimeDetailsActivity : AppCompatActivity() {
12     private lateinit var animeTitle : TextView
13     private lateinit var animePicture : ImageView
14     private lateinit var animeDesc : TextView
15
16     override fun onCreate(savedInstanceState: Bundle?) {
17         super.onCreate(savedInstanceState)
18         enableEdgeToEdge()
19         setContentView(R.layout.activity_anime_details)
20         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
21             val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
22             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
23             insets
24         }
25
26         animeTitle = findViewById(R.id.animeTitle)
27         animePicture = findViewById(R.id.animePicture)
28         animeDesc = findViewById(R.id.animeDescription)
29
30         animeTitle.text = intent.getStringExtra("animeTitle")
31         animePicture.setImageResource(intent.getIntExtra("animePict", -1))
32         animeDesc.text = intent.getStringExtra("animeDesc")
33     }
34 }
```

File Kotlin `AnimeDetailsActivity.kt` berisi kode untuk mengatur tampilan detail anime.

1. onCreate():
 - Metode ini dipanggil saat aktivitas dibuat.
 - Mengaktifkan mode edge-to-edge menggunakan 'enableEdgeToEdge()'.
 - Mengatur tampilan menggunakan layout yang telah ditetapkan sebelumnya dengan 'setContentView()'.
 - Mengatur padding tampilan utama menggunakan 'ViewCompat.setOnApplyWindowInsetsListener()'.
2. Menginisialisasi Elemen UI:
 - Inisialisasi TextView 'animeTitle', 'animeDesc', dan ImageView 'animePicture'.
3. Mengambil Data dari Intent:
 - Mengambil data judul anime, gambar anime, dan deskripsi anime dari Intent yang dikirimkan dari aktivitas sebelumnya.
 - Menetapkan data yang diambil ke elemen-elemen UI yang sesuai.
 -

Dengan demikian, 'AnimeDetailsActivity' akan menampilkan detail anime yang dipilih oleh pengguna, termasuk judul, gambar, dan deskripsi.



File XML `activity_anime_details.xml` mendefinisikan tata letak untuk tampilan detail anime dalam `AnimeDetailsActivity`.

1. ConstraintLayout (androidx.constraintlayout.widget.ConstraintLayout):
 - ConstraintLayout digunakan sebagai parent layout untuk mengatur posisi dan tata letak elemen-elemen UI.
2. TextView (android.widget.TextView):
 - `animeTitle`: TextView untuk menampilkan judul anime.
 - `animeDescription`: TextView untuk menampilkan deskripsi anime.
3. ImageView (android.widget.ImageView):
 - `animePicture`: ImageView untuk menampilkan gambar anime.

5. Screenshot Aplikasi

