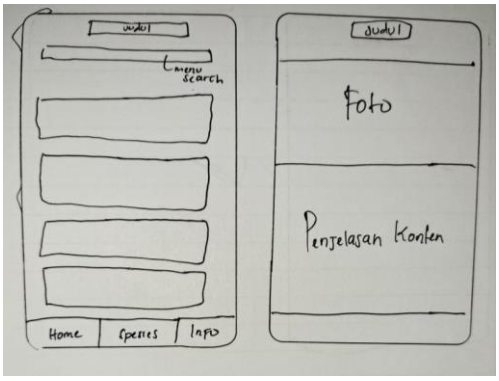
DOKUMENTASI FITUR SPESIES PERLU DILINDUNGI
(ANDROID STUDIO – KOTLIN)

NAMA : RISMA OKTAVIANI

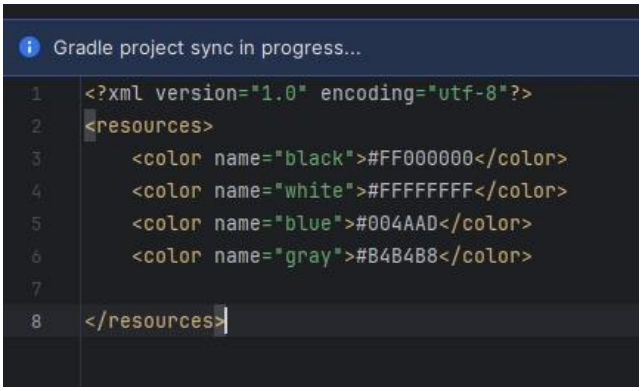
NPM : 22312071

KELAS : IF 22DX

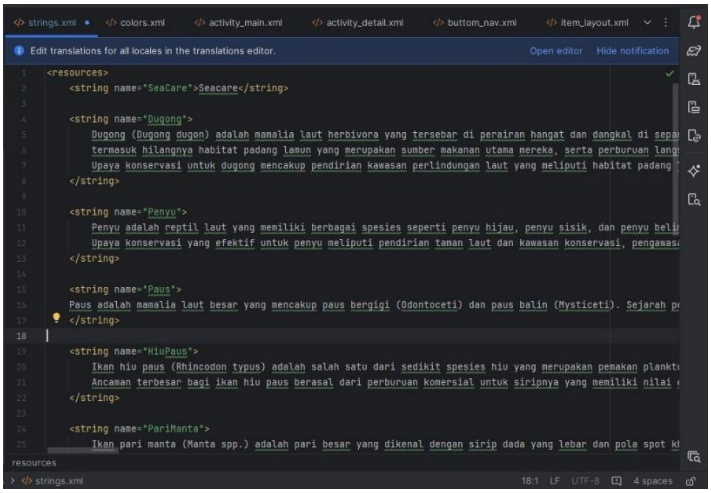
1. Buat sketsa aplikasi yang akan dibuat



2. Langkah pertama buka aplikasi android studionya
3. Lalu buat project baru pada folder program, setah itu atur warna terlebih dahulu agar mempermudah dalam kode, disini saya hanya menggunkan 3 warna yaitu, biru, hitam dan putih



4. atur di bagian string untuk bagian penjelasan konten nya nanti



String digunakan untuk menyimpan teks yang akan digunakan dalam aplikasi dan mengelola menjadi lebih singkat agar mudah dipanggil

5. setelah membuat penjelasan setiap konten, sekarang ke **activity_spesies**

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

<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
```

bar dan tinggi match_parent. XML namespaces digunakan untuk atribut standar (android), atribut khusus (app), dan alat developer (tools). tools:context menunjukkan bahwa layout ini digunakan dalam MainActivity.

```
<androidx.constraintlayout.widget.Guideline
    android:id="@+id/guideline"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    app:layout_constraintGuide_begin="50dp" />
```

Ini adalah Guideline yang orientasinya horizontal, ditempatkan 50dp dari atas layout. Guideline ini digunakan untuk memposisikan elemen lain relatif terhadapnya.

```
<TextView
    android:id="@+id/title"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_margin="12dp"
    android:text="Spesies yang perlu dilindungi"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/search"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="@id/guideline" />
```

TextView ini menampilkan teks "Spesies yang perlu dilindungi" dengan ukuran teks 24sp dan gaya bold. Lebar layout diatur ke 0dp karena menggunakan constraint dari ConstraintLayout untuk menyesuaikan lebarnya. Teks ini diatur agar berada di tengah secara horizontal dan berada di bawah Guideline.

```
<androidx.appcompat.widget.SearchView
    android:id="@+id/search"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_margin="12dp"
    android:layout_marginTop="12dp"
    android:background="@drawable/search_bkg"
    android:focusable="false"
    app:closeIcon="@drawable/baseline_close_24"
    app:iconifiedByDefault="false"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/title"
    app:queryHint="Search..."
    app:searchHintIcon="@null"
    app:searchIcon="@drawable/ic_search" />
```

SearchView ini digunakan untuk input pencarian. Ia memiliki background custom, ikon pencarian custom, dan ikon tutup custom. Query hint adalah "Search...". Lebar layout diatur ke 0dp dan menggunakan constraint untuk menyesuaikan lebarnya dengan parent.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_marginTop="12dp"
    app:layout_constraintTop_toBottomOf="@id/search"
    app:layout_constraintBottom_toTopOf="@+id/bottom_nav"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"/>
```

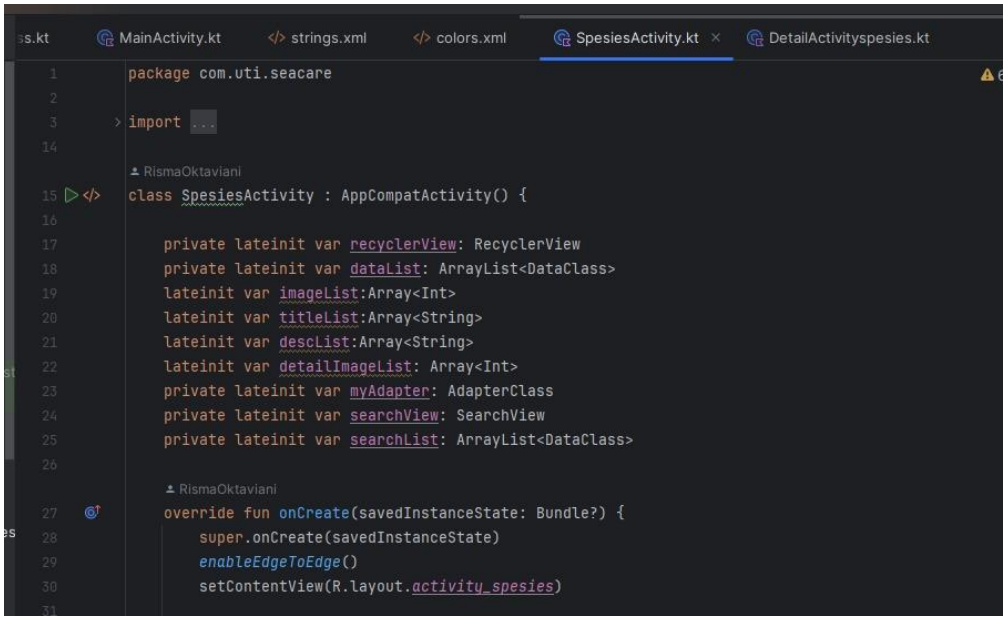
RecyclerView ini digunakan untuk menampilkan daftar data secara efisien. Lebar diatur match_parent dan tinggi 0dp karena menggunakan constraint dari ConstraintLayout untuk menyesuaikan tingginya. RecyclerView ditempatkan di antara SearchView dan BottomNavigationView.

```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:id="@+id/bottom_nav"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:background="@color/blue"
    app:itemIconSize="35dp"
    app:itemIconTint="@color/white"
    app:itemTextColor="#FFFFFF"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:menu="@menu/button_nav" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

BottomNavigationView ini digunakan untuk navigasi bawah dengan ikon yang berukuran 35dp dan berwarna putih. Menu yang digunakan adalah button_nav yang didefinisikan di folder res/menu. Layout ini diatur untuk menempel pada bagian bawah layout utama (parent).

6. SpesiesAcitiviy



Bagian ini mendefinisikan kelas SpesiesActivity yang mewarisi dari AppCompatActivity. Dalam onCreate, aktivitas diinisialisasi dengan memanggil setContentView untuk mengatur layout menggunakan R.layout.activity_spesies.

Bagian ini menginisialisasi data dengan daftar gambar, judul, deskripsi, dan gambar detail untuk berbagai spesies.

```
recyclerView = findViewById(R.id.recyclerView)
searchView = findViewById(R.id.search)
recyclerView.layoutManager = LinearLayoutManager( context: this)
recyclerView.setHasFixedSize(true)

dataList = arrayListOf<DataClass>()
searchList = arrayListOf<DataClass>()
getData()

searchView.clearFocus()
searchView.setOnQueryTextListener(object : SearchView.OnQueryTextListener {
    override fun onQueryTextSubmit(query: String?): Boolean {
        searchView.clearFocus()
        return true
    }

    override fun onQueryTextChange(newText: String?): Boolean {
        searchList.clear()
        val searchText = newText!!.toLowerCase(Locale.getDefault())
        if (searchText.isNotEmpty()) {
            dataList.forEach { it: DataClass
                if (it.dataTitle.toLowerCase(Locale.getDefault()).contains(searchText)) {
                    searchList.add(it)
                }
            }
            recyclerView.adapter!!.notifyDataSetChanged()
        } else {
```

Di sini, RecyclerView dan SearchView diinisialisasi dan disetup. RecyclerView diatur dengan LinearLayoutManager dan flag setHasFixedSize(true). dataList dan searchList diinisialisasi sebagai ArrayList<DataClass>. getData() dipanggil untuk mengisi dataList. SearchView diatur untuk memfilter daftar berdasarkan teks yang diinput.

```
        } else {
            searchList.clear()
            searchList.addAll(dataList)
            recyclerView.adapter!!.notifyDataSetChanged()
        }
        return false
    }

})

val bottomNav = findViewById<BottomNavigationView>(R.id.bottom_nav)
bottomNav.setOnItemSelectedListener { menuItem ->
    when (menuItem.itemId) {
        R.id.home -> {
            val intent = Intent( packageContext: this, MainActivity::class.java)
            startActivity(intent)
            finish() // Optional: Close current activity
        }
        R.id.spesies -> {
            // Already in this activity
        }
        R.id.info -> {
            val intent = Intent( packageContext: this, MainActivity::class.java)
            intent.putExtra( name: "fragment",  value: "info")
            startActivity(intent)
            finish() // Optional: Close current activity
        }
        else -> {
```

BottomNavigationView diinisialisasi dan listener ditetapkan untuk menangani navigasi antar aktivitas saat item menu dipilih.

```
        myAdapter  = AdapterClass(searchList)
        recyclerView.adapter = myAdapter

        myAdapter.onItemClick = { it: DataClass
            val intent = Intent( packageContext: this, DetailActivityspesies::class.java)
            intent.putExtra( name: "android",it)
            startActivity(intent)
        }
    }
```

AdapterClass diinisialisasi dengan searchList dan diatur sebagai adapter untuk RecyclerView. Item click listener ditetapkan untuk menangani navigasi ke DetailActivityspesies saat item di klik.

```
RismaOktaviani
private fun getData() {
    for (i in imageList.indices) {
        val dataClass = DataClass(imageList[i], titleList[i], descList[i], detailImageList[i])
        dataList.add(dataClass)
}
    searchList.addAll(dataList)
    recyclerView.adapter = AdapterClass(searchList)
}
}
```

Fungsi getData mengisi dataList dengan objek DataClass yang berisi gambar, judul, deskripsi, dan gambar detail. searchList diisi dengan semua data dari dataList, dan adapter diatur dengan searchList.

7. Spesies_detail_acitivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".DetailActivityspesies">
```

Bagian ini mendefinisikan root layout sebagai ConstraintLayout dengan lebar dan tinggi match_parent. XML namespaces digunakan untuk atribut standar (android), atribut khusus (app), dan alat developer (tools). tools:context menunjukkan bahwa layout ini digunakan dalam DetailActivityspesies.

```
<androidx.cardview.widget.CardView
    android:id="@+id/cardView"
    android:layout_width="0dp"
    android:layout_height="200dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="160dp"
    android:layout_marginEnd="8dp"
    app:cardCornerRadius="10dp"
    app:cardElevation="10dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <ImageView
        android:id="@+id/_imageDetail"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        tools:srcCompat="@drawable/dugongframe" />

</androidx.cardview.widget.CardView>
```

CardView ini berisi sebuah ImageView. CardView diatur agar memiliki lebar 0dp (diatur oleh constraint) dan tinggi 200dp, dengan margin di kiri, atas, dan kanan. CardView memiliki sudut melengkung (10dp) dan elevasi (10dp) untuk efek bayangan. ImageView di dalamnya menampilkan gambar dengan scaleType "centerCrop" untuk memastikan gambar memenuhi seluruh ImageView.

```
<TextView
    android:id="@+id/_detailTitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Dugong"
    android:textColor="@color/blue"
    android:textSize="30sp"
    android:textStyle="bold"
    android:layout_marginTop="100dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

TextView ini menampilkan judul "Dugong" dengan ukuran teks 30sp, warna teks biru, dan gaya teks tebal (bold). TextView ini diatur agar berada di tengah secara horizontal dan 100dp dari atas parent.

```
<TextView
    android:id="@+id/detailDesc"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="20dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="20dp"
    android:padding="15dp"
    android:text="Dugong (Dugong dugon) adalah mamalia laut herbivora yan..."
    android:textAlignment="viewStart"
    android:textColor="@color/blue"
    android:textSize="15sp"
    app:layout_constraintBottom_toTopOf="@id/scrollView"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/_detailTitle" />
```

TextView ini menampilkan deskripsi yang diambil dari resource string dengan ID @string/Dugong. Ukuran teks adalah 15sp dengan warna biru. TextView ini diatur agar berada di antara judul (@id/_detailTitle) dan ScrollView.

```
<ScrollView
    android:id="@+id/scrollView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginTop="10dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/cardView"
    app:layout_constraintVertical_bias="1.0">

    <TextView
        android:id="@+id/scrollTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Dugong (Dugong dugon) adalah mamalia laut herbivora yan..."
        android:textColor="@color/blue"
        android:textSize="15sp"
        android:padding="15dp"
        android:textAlignment="viewStart"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</ScrollView>
```

ScrollView ini memungkinkan konten yang dapat digulir jika konten melebihi ukuran layar. TextView di dalam ScrollView menampilkan deskripsi yang panjang dari resource string dengan ID @string/Dugong. Ukuran teks adalah 15sp dengan warna biru.

8. DetilAcitivitySpesies

```
class DetailActivityspesies : AppCompatActivity() {
    RismaOktaviani
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_detail_activityspesies)
```

setContentView(R.layout.activity_detail_activityspesies): Mengatur layout XML yang akan digunakan untuk aktivitas ini.

```
        val getData = intent.getParcelableExtra<DataClass>( name: "android")
```

val getData = intent.getParcelableExtra<DataClass>("android")

Bagian ini mengambil data yang dikirimkan melalui intent. Data ini diambil sebagai objek DataClass dengan key "android".

```
        if (getData != null) {
            val detailTitle: TextView = findViewById(R.id._detailTitle)
            val detailDesc: TextView = findViewById(R.id.detailDesc)
            val detailImage: ImageView = findViewById(R.id._imageDetail)
```

Memastikan Data Tidak Null dan Mengatur View

Memeriksa Null: Memastikan bahwa data yang diambil dari intent tidak null.

```
            detailTitle.text = getData.dataTitle
            detailDesc.text = getData.dataDesc
            detailImage.setImageResource(getData.dataDetailImage)
        }
    }
}
```

Mengatur View:

detailTitle: TextView untuk judul spesies, diambil dari layout menggunakan findViewById.

detailDesc: TextView untuk deskripsi spesies, diambil dari layout menggunakan findViewById.

detailImage: ImageView untuk gambar detail spesies, diambil dari layout menggunakan findViewById.

Mengatur Konten View:

detailTitle.text = getData.dataTitle: Mengatur teks judul berdasarkan data yang diterima.

detailDesc.text = getData.dataDesc: Mengatur teks deskripsi berdasarkan data yang diterima.

detailImage.setImageResource(getData.dataDetailImage): Mengatur gambar berdasarkan data yang diterima.

9. DataClass

DataClass Menyimpan informasi spesies dengan empat property

```
package com.uti.seacare

import android.os.Parcel
import android.os.Parcelable

RismaOktaviani
data class DataClass(
    var dataImage: Int,
    var dataTitle: String,
    var dataDesc: String,
    var dataDetailImage: Int
): Parcelable {
```

dataImage: ID resource gambar (tipe Int).

dataTitle: Judul spesies (tipe String).

dataDesc: Deskripsi spesies (tipe String).

dataDetailImage: ID resource gambar detail (tipe Int).

Implementasi Parcelable

Untuk memungkinkan objek DataClass dipaketkan dan dikirim antar aktivitas, DataClass mengimplementasikan Parcelable.

Constructor dari Parcel

```
constructor(parcel: Parcel) : this(
    parcel.readInt(),
    parcel.readString()!!,
    parcel.readString()!!,
    parcel.readInt()
) {
}
```

Constructor ini digunakan untuk membuat objek DataClass dari Parcel. Ini membaca data dari Parcel dalam urutan yang sama seperti saat ditulis:

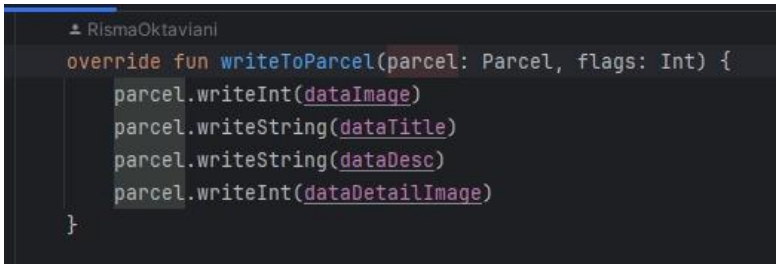
dataImage: Dibaca sebagai Int.

dataTitle: Dibaca sebagai String.

dataDesc: Dibaca sebagai String.

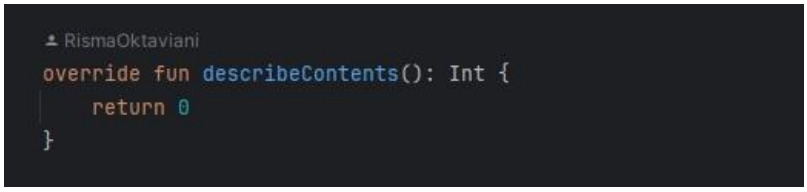
dataDetailImage: Dibaca sebagai Int.

Metode writeToParcel



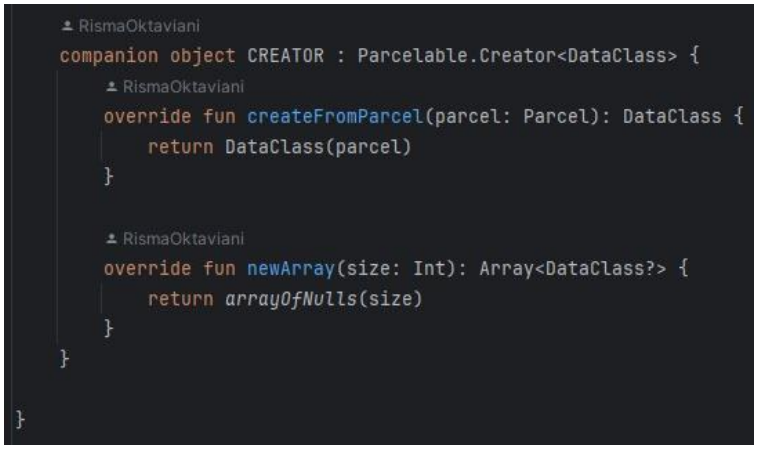
Metode ini digunakan untuk menulis objek DataClass ke dalam Parcel dalam urutan yang sama seperti yang akan dibaca

Metode describeContents



Metode ini biasanya mengembalikan 0, dan digunakan untuk mendeskripsikan konten objek. Dapat diabaikan untuk sebagian besar kasus.

Companion Object CREATOR



Companion object ini diperlukan untuk mengimplementasikan Parcelable. Ini menyediakan dua metode:

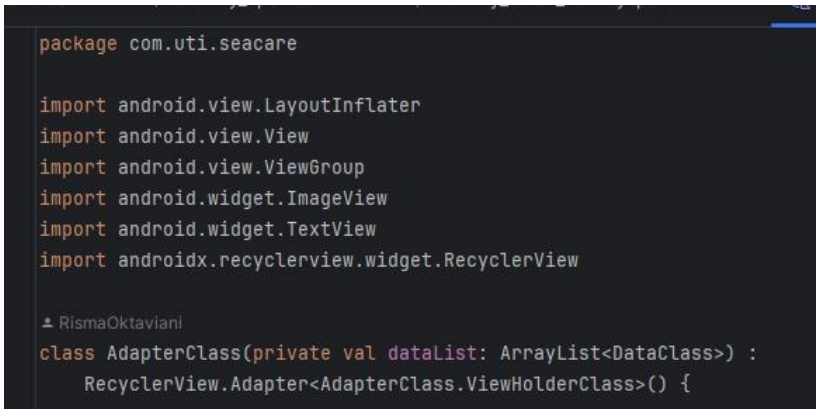
createFromParcel: Membuat objek DataClass dari Parcel.

newArray: Membuat array dari objek DataClass dengan ukuran tertentu.

10. AdapterClass

AdapterClass adalah sebuah adapter untuk RecyclerView yang bertanggung jawab untuk menampilkan data spesies dalam bentuk daftar.

Deklarasi AdapterClass



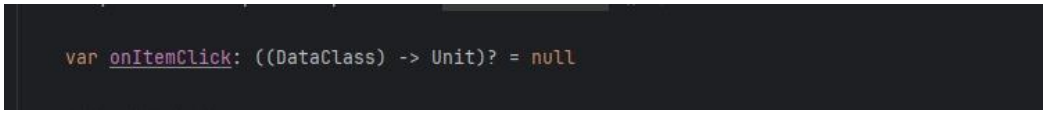
```
package com.uti.seacare

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

RismaOktaviani
class AdapterClass(private val dataList: ArrayList<DataClass>) :
    RecyclerView.Adapter<AdapterClass.ViewHolderClass>() {
```

AdapterClass meng-extend RecyclerView.Adapter dengan ViewHolderClass sebagai ViewHolder-nya. dataList adalah daftar objek DataClass yang akan ditampilkan.

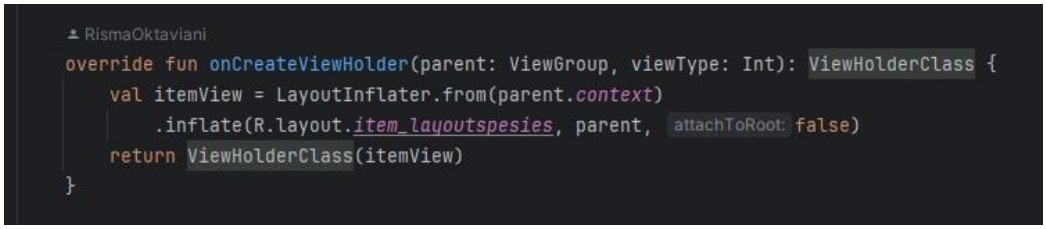
Properti dan Metode



```
var onItemClick: ((DataClass) -> Unit)? = null
```

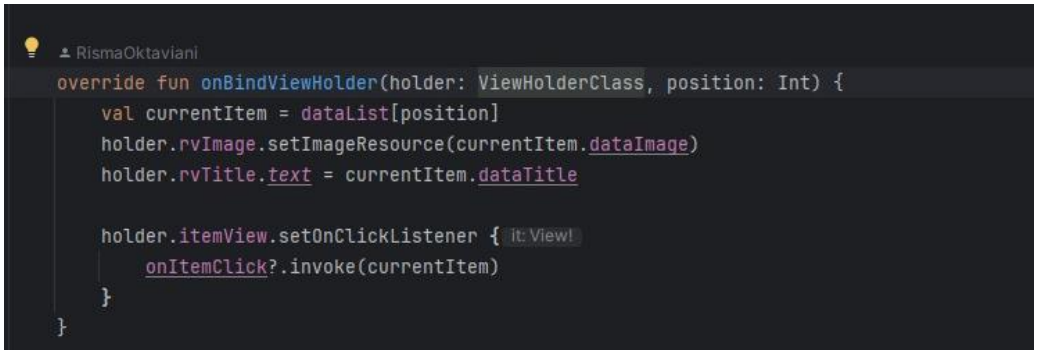
Properti ini adalah sebuah lambda function yang akan dipanggil saat item dalam RecyclerView diklik. Ini memungkinkan kita untuk menentukan aksi yang akan dilakukan ketika item diklik.

OnCreateViewHolder



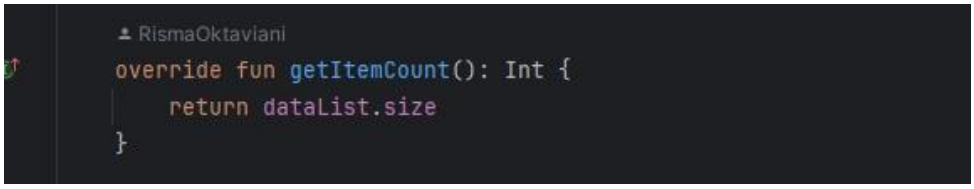
Metode ini bertanggung jawab untuk membuat ViewHolder baru. Layout item_layoutspesies di-inflate untuk setiap item dalam RecyclerView.

onBindViewHolder



Metode ini bertanggung jawab untuk mengikat data ke ViewHolder.
currentItem: Objek DataClass saat ini berdasarkan posisi.
holder.rvImage.setImageResource(currentItem.dataImage): Mengatur gambar dari currentItem.
holder.rvTitle.text = currentItem.dataTitle: Mengatur judul dari currentItem.
Setiap item juga memiliki listener untuk menangani klik, yang memanggil lambda onItemClick dengan currentItem.

getItemCount



Metode ini mengembalikan jumlah item dalam dataList.

ViewHolderClass

```
    RismaOktaviani
    class ViewHolderClass(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val rvImage: ImageView = itemView.findViewById(R.id.image)
        val rvTitle: TextView = itemView.findViewById(R.id.title)
    }
}
```

ViewHolderClass adalah sebuah ViewHolder untuk RecyclerView. Ia menyimpan referensi ke ImageView dan TextView dalam setiap item layout (item_layoutspesies).

12. HASIL / OUTPUT

Menu Utamanya

Menu Search

Menu Penjelasam