



Android

Security & Recycle View

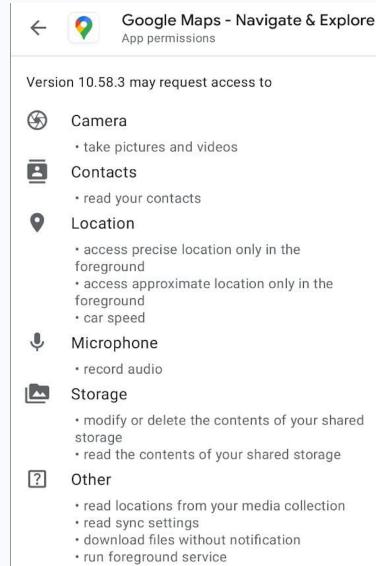
Daftar Isi

A. Permission di Android	3
B. Implementasi Permission pada Project Android	6
C. Recycler View dan Komponennya	8
D. Implementasi Recycler View	9
References	16

A. Permission di Android

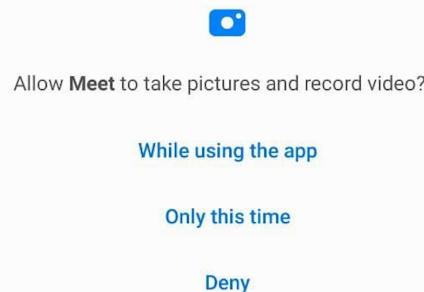
Android secara default tidak memiliki permission. Saat aplikasi perlu menggunakan fitur-fitur khusus yang ada di perangkat (misalnya mengakses kamera, mengirim atau membaca SMS, membaca kontak, dan membaca file storage, dan lain-lain) mereka perlu diberikan permission atau izin dari user agar tujuannya tercapai. Sebelum Marshmallow, permission diminta kepada user pada saat aplikasi pertama di-install. Setelah Marshmallow, permission sekarang harus diminta saat aplikasi dijalankan.

Demi keamanan, kita harus menyatakan atau meminta izin permission sebelum menggunakan fitur tertentu di perangkat Android. Permission melindungi akses ke data terbatas dan tindakan terbatas. Kita mendeklarasikan permission dalam file manifest aplikasi. Jenis permission tertentu juga mengharuskan user untuk secara aktif memberikannya agar aplikasi dapat menggunakannya. Izin yang hanya perlu dideklarasikan dalam manifest aplikasi dan diberikan secara otomatis saat aplikasi diinstal disebut izin waktu penginstalan (install-time permissions). User melihat daftar permission waktu penginstalan yang diperlukan aplikasi Anda di halaman detail aplikasinya di Play Store, tetapi mereka tidak akan mendapatkan petunjuk UI dari aplikasi tentang izin tersebut.



Daftar permission Play Store untuk Google Maps

Permission yang memerlukan otorisasi dari user saat runtime dikenal sebagai izin runtime (runtime permissions), atau izin berbahaya (dangerous permissions). Saat aplikasi kita meminta izin berbahaya dari user, sistem meminta mereka dengan pop up dialog permintaan permission seperti pada gambar di bawah ini.



Untuk meminta salah satu permission agar aplikasi dapat mengakses fitur khusus yang disediakan oleh perangkat, kita cukup menuliskannya ke AndroidManifest.xml kemudian pada kode activity atau fragment tempat diperlukannya permission ditambahkan kode untuk mengecek dan meminta permission dari user. Jika user sudah memberikan izin maka aplikasi dapat menggunakan fitur tersebut. Berikut ini contoh aplikasi yang meminta

permission untuk membaca kontak yang ditentukan di dalam AndroidManifest.xml :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.app.myapp" >

    <uses-permission android:name="android.permission.READ_CONTACTS" />
    ...
</manifest>
```

Berikut ini merupakan beberapa permission yang umum digunakan pada aplikasi Android :

1. **Camera** : Mengambil gambar dan merekam video.
2. **Contacts** : Membaca, membuat atau edit daftar. Termasuk mengakses semua akun dari perangkat
3. **Location** : Mengakses lokasi smartphone secara akurat, data cell phone dan WIFI
4. **Microphone** : Menggunakan microphone untuk merekam suara , umumnya termasuk video, seperti aplikasi kamera, voice recording, telepon, telepon internet atau video call.
5. **Phone** : Mengakses nomor telepon dan jaringan. Termasuk membuat panggilan telepon, menggunakan Voip, Voicemail, dan lainnya termasuk daftar catatan log telepon. Dibutuhkan untuk berbagai kepentingan misal daftar nomor telepon.
6. **SMS** : Membaca, mengirim SMS dan MMS
7. **Storage** : Membaca dan menulis data pada storage

Ada beberapa jenis permission umum yang mungkin perlu diminta oleh aplikasi. Untuk daftar lengkap dan terkini dari semua izin aplikasi Android, lihat referensi API permission berikut ini :

<https://developer.android.com/guide/topics/permissions/overview>.

B. Implementasi Permission pada Project Android

Berikut ini merupakan langkah-langkah implementasi permission untuk menggunakan kamera di aplikasi Android :

1. Menambahkan deklarasi permission kamera ke AndroidManifest aplikasi

```
<manifest ...>
    <uses-permission android:name="android.permission.CAMERA"/>
    <application ...>
        ...
    </application>
</manifest>
```

2. Memeriksa apakah permission sudah diberikan oleh user

Berikut adalah blok kode yang disarankan untuk memeriksa izin dan memintanya jika diperlukan :

```
when {
    ContextCompat.checkSelfPermission(
        CONTEXT ,
        Manifest.permission.REQUESTED_PERMISSION )
    ) == PackageManager.PERMISSION_GRANTED -> {
    // You can use the API that requires the permission.
}
shouldShowRequestPermissionRationale(...) -> {
    // In an educational UI, explain to the user why your app requires this
    // permission for a specific feature to behave as expected, and what
    // features are disabled if it's declined. In this UI, include a
    // "cancel" or "no thanks" button that lets the user continue
    // using your app without granting the permission.
    showInContextUI(...)
}
else -> {
    // You can directly ask for the permission.
    // The registered ActivityResultCallback gets the result of this request.
    requestPermissionLauncher.launch(
        Manifest.permission.REQUESTED_PERMISSION )
}
```

3. Deklarasi variabel untuk meng-handle respons permission

```
// Register the permissions callback, which handles the user's response to the
// system permissions dialog. Save the return value, an instance of
// ActivityResultLauncher. You can use either a val, as shown in this snippet,
// or a lateinit var in your onAttach() or onCreate() method.
val requestPermissionLauncher =
    registerForActivityResult(RequestPermission())
) { isGranted: Boolean ->
    if (isGranted) {
        // Permission is granted. Continue the action or workflow in your
        // app.
    } else {
        // Explain to the user that the feature is unavailable because the
        // feature requires a permission that the user has denied. At the
        // same time, respect the user's decision. Don't link to system
        // settings in an effort to convince the user to change their
        // decision.
    }
}
```

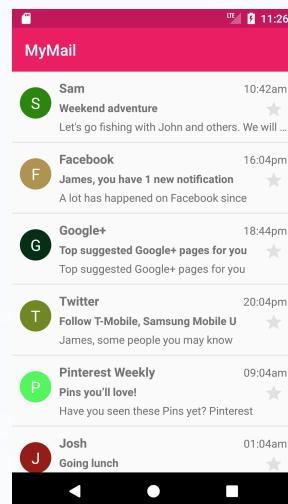
4. Setelah user merespons dialog izin sistem, sistem akan memanggil implementasi `onRequestPermissionsResult()` aplikasi. Sistem meneruskan respons pengguna ke dialog izin serta kode permintaan yang kita tentukan, seperti yang ditunjukkan dalam cuplikan kode berikut:

```
override fun onRequestPermissionsResult(requestCode: Int,
    permissions: Array<String>, grantResults: IntArray) {
    when (requestCode) {
        PERMISSION_REQUEST_CODE -> {
            // If request is cancelled, the result arrays are empty.
            if ((grantResults.isNotEmpty() &&
                    grantResults[0] == PackageManager.PERMISSION_GRANTED)) {
                // Permission is granted. Continue the action or workflow
                // in your app.
            } else {
                // Explain to the user that the feature is unavailable because
                // the feature requires a permission that the user has denied.
                // At the same time, respect the user's decision. Don't link to
                // system settings in an effort to convince the user to change
                // their decision.
            }
            return
        }

        // Add other 'when' lines to check for other
        // permissions this app might request.
        else -> {
            // Ignore all other requests.
        }
    }
}
```

C. Recycler View dan Komponennya

RecyclerView merupakan salah satu tampilan yang umum dan banyak sekali digunakan untuk menampilkan informasi yang ingin disampaikan lewat aplikasi Android, khususnya saat informasi yang ingin disampaikan tersebut banyak dan relatif seragam.

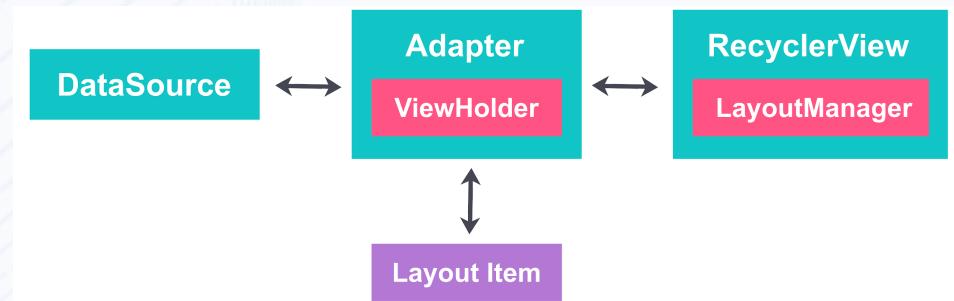


Contoh list email yang ditampilkan dalam sebuah RecyclerView

RecyclerView mempunyai beberapa komponen yang membentuknya yaitu sebagai berikut :

1. **DataSource** : ArrayList atau List yang berisi kelas data
2. **Adapter** : Berfungsi untuk menghubungkan data ke RecyclerView
3. **Layout Item** : File XML (UI) untuk satu baris item untuk setiap data pada RecyclerView
4. **ViewHolder** : Memiliki informasi tampilan untuk menampilkan satu item
5. **LayoutManager** : Menangani organisasi komponen UI dalam tampilan

Hubungan antara komponen-komponen tersebut dapat dilihat pada gambar di bawah ini.



Berikut ini merupakan jenis-jenis LayoutManager yang terdapat pada RecyclerView.



D. Implementasi Recycler View

Secara umum untuk mengimplementasikan RecyclerView, langkah-langkah yang perlu dilakukan dapat dirangkum sebagai berikut:

1. Menambahkan komponen RecyclerView di layout activity atau fragment
2. Membuat layout untuk satu item RecyclerView
3. Membuat model untuk item data (optional)
4. Membuat adapter dan view holder
5. Setup RecyclerView di activity atau fragment

Sekarang, kita akan mencoba mengimplementasikan secara mendetail setiap langkah tersebut.

1. Menambahkan RecyclerView di layout

Untuk menambahkan komponen view RecyclerView di layout activity atau fragment, cukup menambahkan tag XML RecyclerView di lokasi yang diinginkan. Sebagai contoh, pada gambar berikut kita menambahkan RecyclerView di layout MainActivity.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/email_recycler_view"
        android:layout_width="match_parent"
        android:orientation="vertical"
        android:layout_height="match_parent"/>

</LinearLayout>
```

2. Membuat layout item

Layout item merepresentasikan satu view untuk item data yang ingin ditampilkan. Misalkan pada contoh daftar inbox di aplikasi email di atas, maka layout item itu digunakan untuk menampilkan satu item email yang ada. Untuk membuat layout item, kita perlu membuat satu file xml. Agar mudah untuk membedakannya dengan file xml lainnya, sebaiknya kita gunakan prefix item untuk nama file. Sebagai contoh, blok kode berikut merupakan layout item.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="4dp"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/email_image"
        android:layout_width="@dimen/item_height"
        android:layout_height="@dimen/item_height"
        android:layout_margin="4dp"
        android:src="@mipmap/ic_launcher_round" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="@dimen/item_height"
        android:layout_margin="4dp"
        android:layout_weight="2"
        android:orientation="vertical">

        <TextView
            android:id="@+id/email_sender"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Andy Brown"
            android:textColor="@android:color/black"
            android:textSize="20sp" />

        <TextView
            android:id="@+id/email_subject"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Bring Your Parents to Work Day!"
            android:textColor="#222222" />

        <TextView
            android:id="@+id/email_message"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hey what do you think about...?"
            android:textColor="#222222" />

    </LinearLayout>

</LinearLayout>
```

3. Membuat model item data

Model item merupakan class dari objek data yang ingin ditampilkan di layout item nantinya. Untuk membuat class dari model item ini kita perlu

kreatif untuk melakukan abstraksi agar model yang dibuat efektif dan efisien. Model ini kita implementasikan dalam file Java. Sebagai contoh, model data untuk item email di atas yaitu sebagai berikut.

```
data class Email(  
    val sender: String,  
    val email: String,  
    val message: String,  
    val image: Int  
)
```

4. Membuat adapter dan view holder

Langkah selanjutnya yaitu membuat adapter dan view holder. View Holder berguna untuk binding data dari model ke item layoutnya. Sedangkan adapter berguna untuk mengatur beberapa view holder untuk ditampilkan dan mengatur interaksinya. Berikut ini merupakan contoh bagaimana view holder diimplementasikan, fungsi bind berguna untuk binding data dari model item ke layout item.

```
class EmailViewHolder(val binding: ItemEmailBinding) : RecyclerView.ViewHolder(binding.root) {  
  
    fun bind(email: Email) {  
        binding.emailImage.setImageResource(email.image)  
        binding.emailSender.text = email.sender  
        binding.emailSubject.text = email.email  
        binding.emailMessage.text = email.message  
    }  
}
```

Selanjutnya adalah membuat adapter. Ada 3 method penting yang perlu diperhatikan dalam membuat RecyclerView adapter ini, yaitu onCreateViewHolder, onBindViewHolder, dan getItemCount. Pada

onCreateViewHolder, kita inflate layout item yang sudah kita buat. Pada onBindViewHolder, kita bind data dari model ke layout item menggunakan fungsi bind dari view holder. Sedangkan pada getItemCount kita menentukan berapa banyak data yang ingin kita tampilkan di RecyclerView.

```
class EmailAdapter(private val emails: List<Email>) :  
    RecyclerView.Adapter<EmailViewHolder>() {  
  
    override fun onCreateViewHolder(parent: ViewGroup, i: Int): EmailViewHolder {  
        val inflater = LayoutInflater.from(parent.context)  
        val binding = ItemEmailBinding.inflate(inflater)  
        return EmailViewHolder(binding)  
    }  
  
    override fun onBindViewHolder(viewHolder: EmailViewHolder, position: Int) {  
        viewHolder.bind(emails[position])  
    }  
  
    override fun getItemCount(): Int {  
        return emails.size  
    }  
}
```

Umumnya adapter dan viewholder dibuat dalam satu kelas dengan class viewholder menjadi nested class-nya.

5. Setup RecyclerView

Langkah yang terakhir adalah setup RecyclerView di activity, fragment, atau dialog yang kita inginkan. Ada 3 hal penting yang perlu diperhatikan saat setup RecyclerView ini, yaitu: setLayoutManager, setAdapter, dan mengisi data. setLayoutManager berkaitan dengan bagaimana kita ingin menampilkan RecyclerView tersebut, apakah dalam bentuk list (LinearLayoutManager), grid (GridLayoutManager), dsb. setAdapter berguna untuk menghubungkan RecyclerView dengan adapternya yang

sudah diimplementasikan sebelumnya. Bagian mengisi data berkaitan dengan bagaimana cara kita mengambil data dan menambahkannya ke RecyclerView. Berikut contoh bagaimana ketiga hal tersebut diimplementasikan.

```
class MainActivity : AppCompatActivity() {

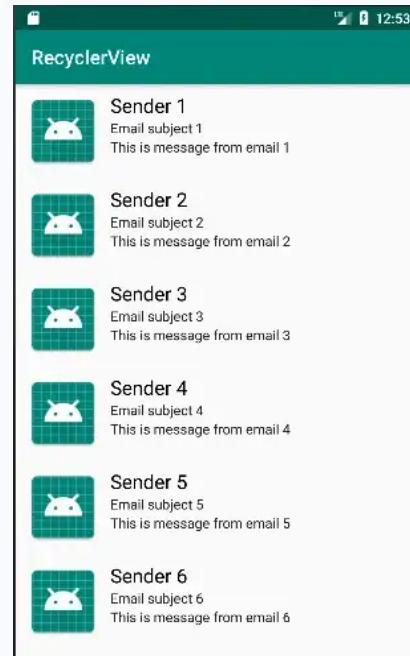
    private lateinit var binding: ActivityMainBinding
    private lateinit var emailAdapter: EmailAdapter
    private lateinit var emails: MutableList<Email>

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
        setupRecyclerView()
    }

    private fun setupRecyclerView() {
        val layoutManager = LinearLayoutManager(context)
        binding.emailRecyclerView.layoutManager = layoutManager
        emails = mutableListOf()
        populateEmails()
        emailAdapter = EmailAdapter(emails)
        binding.emailRecyclerView.adapter = emailAdapter
    }

    private fun populateEmails() {
        for (i in 0 .. 9) {
            val sender = "Sender " + (i + 1).toString()
            val subject = "Email subject " + (i + 1).toString()
            val message = "This is message from email " + (i + 1).toString()
            emails.add(Email(sender, subject, message, R.drawable.ic_dialog_email))
        }
    }
}
```

Dan berikut ini tampilan dari RecyclerView yang telah kita buat.



References

- <https://codepolitan.com/blog/memahami-app-permission-di-android-598adf12a57ab>
- <http://www.obengplus.com/articles/7864/1/Permission-di-Android-yang-perlu-kita-perhatikan.html>
- <https://www.kodeco.com/books/android-app-distribution/v1.0/chapters/5-permissions>
- <https://developer.android.com/training/permissions/declaring?hl=id>
- <https://medium.com/ristex/android-recyclerview-basic-dan-templating-fa17f8637ce5>
- <https://developer.android.com/guide/topics/ui/layout/recyclerview>