

# Buku Saku Android

Bagus Aji Santoso

Revisi 0.0.1

18 Desember 2018

---

---

---

# Daftar Isi

- Tentang penulis ..... v
- 1. Pendahuluan ..... 1
  - 1.1. Tidak berurutan ..... 1
  - 1.2. Lisensi ..... 1
  - 1.3. Kontak ..... 2
  - 1.4. Versi PDF ..... 2
  - 1.5. Versi HTML ..... 2
- 2. RecyclerView ..... 3
  - 2.1. Membuat RecyclerView sederhana ..... 3
    - 2.1.1. Menambahkan dependensi ..... 3
    - 2.1.2. Menambahkan komponen RecyclerView di layout ..... 4
    - 2.1.3. Menambahkan layout item RecyclerView ..... 4
    - 2.1.4. Membuat kelas model (POJO) ..... 6
    - 2.1.5. Membuat Adapter RecyclerView ..... 7
    - 2.1.6. Membuat ViewHolder ..... 7
    - 2.1.7. Membuat list item ..... 8
    - 2.1.8. Implementasi method wajib RecyclerView.Adapter ..... 9
    - 2.1.9. Inisialisasi RecyclerView, Adapter, dan data ..... 13
    - 2.1.10. OnClickListener dari dalam adapter ..... 17
    - 2.1.11. OnClickListener dari luar adapter ..... 18
- 3. Intent ..... 25
  - 3.1. Membuat Splash Screen ..... 25
    - 3.1.1. Mengubah Activity Launcher ..... 26
    - 3.1.2. Membuat layout splash screen ..... 27
    - 3.1.3. Menambahkan theme Splash ke SplashActivity ..... 28
    - 3.1.4. Redirect ke MainActivity ..... 28



---

# Tentang penulis

Bagus Aji Santoso adalah programmer yang baik hati. Alumni Departemen Pendidikan Ilmu Komputer, Universitas Pendidikan Indonesia.

- <https://bagus.me>
- <https://twitter.com/lobothijau>



# Pendahuluan

---

Selamat membaca Buku Saku Android. Buku ini disusun untuk membantu programmer Android pemula untuk mengulang teori penggunaan komponen-komponen Android juga untuk programmer yang sudah tidak lagi pemula agar lebih mudah mencari potongan kode yang bisa di *copy-paste* dalam proses pembuatan aplikasi Android.

## 1.1. Tidak berurutan

Topik-topik yang dibahas dibuku ini saling berdiri sendiri dan dibuat untuk tidak bergantung pada topik-topik sebelumnya. Jika komponen A yang sedang dibahas memakai komponen B, maka penjelasan komponen B hanya sebatas untuk mendukung penjelasan komponen A. Oleh karena itu buku ini tidak mesti dibaca dari halaman 1 sampai halaman terakhir. Silahkan lompat dari bab yang satu ke bab yang lainnya sesuai dengan kebutuhan.

## 1.2. Lisensi

Buku ini dipublikasikan dengan lisensi [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International \(CC BY-NC-SA 4.0\)](http://creativecommons.org/licenses/by-nc-sa/4.0/)<sup>1</sup>. Artinya, buku ini boleh didistribusikan dan dimodifikasi untuk tujuan non komersial selama memberikan atribusi/kredit.

Sementara itu, untuk *source code* contoh yang ada di buku ini boleh dipakai untuk penggunaan pribadi maupun komersial, boleh juga didistribusikan serta dimodifikasi sesuai kebutuhan demi kemaslahatan umat manusia. Akan tetapi,

---

<sup>1</sup> <http://creativecommons.org/licenses/by-nc-sa/4.0/>

tidak ada garansi apapun yang diberikan untuk setiap *source code* yang ditampilkan di dalam buku ini maupun yang disimpan di repositori terpisah (<https://github.com/lobothijau/bukusakuandroid-code>). Segala kerusakan yang terjadi akibat *copy-paste source code* yang tersedia diluar tanggung jawab penulis. Namun, penulis tetap berharap pembaca dapat memberikan umpan balik apabila terdapat kesalahan penjelasan maupun kesalahan kode sehingga bisa diperbaiki dan bisa membantu lebih banyak orang.

### 1.3. Kontak

Jika pembaca menemukan menemukan kesalahan sekecil apapun, memiliki pertanyaan, atau memiliki permintaan topik yang ingin dibahas (meskipun belum tentu langsung dikabulkan), silahkan mention [@lobothijau](https://twitter.com/lobothijau)<sup>2</sup> atau kirimkan email ke [lobothijau\[at\]gmail.com](mailto:lobothijau[at]gmail.com).

### 1.4. Versi PDF

Versi PDF terakhir buku ini bisa di unduh di <https://github.com/lobothijau/bukusakuandroid>.

### 1.5. Versi HTML

Versi HTML terakhir buku ini bisa di baca di <http://droidindonesia.id/bukusakuandroid/>

---

<sup>2</sup> <https://twitter.com/lobothijau>



---

# 2

## RecyclerView

---

RecyclerView merupakan komponen Android yang didesain untuk dapat menampilkan kumpulan data dengan performa yang lebih efisien dibandingkan ListView/GridView.

### 2.1. Membuat RecyclerView sederhana

Contoh aplikasi RecyclerView ini menggunakan template **Basic Activity**.

#### 2.1.1. Menambahkan dependensi

Buka file `build.gradle` modul app lalu tambahkan dependensi RecyclerView di blok `dependencies`.

```
dependencies {  
    implementation fileTree(dir: 'libs', include: ['*.jar'])  
    implementation 'com.android.support:appcompat-v7:28.0.0'  
    implementation 'com.android.support.constraint:constraint-  
layout:1.1.3'  
    implementation 'com.android.support:design:28.0.0'  
    testImplementation 'junit:junit:4.12'  
    androidTestImplementation 'com.android.support.test:runner:1.0.2'  
  
    androidTestImplementation 'com.android.support.test.espresso:espresso-  
core:3.0.2'  
    implementation 'com.android.support:recyclerview-v7:28.0.0' ❶  
    implementation 'com.squareup.picasso:picasso:2.71828' ❷  
}
```

❶ Pastikan nomor versi sama dengan `appcompat-v7`

- ❷ Library Picasso dipakai untuk menampilkan gambar dari sebuah URL

### ***2.1.2. Menambahkan komponen RecyclerView di layout***

Tambahkan komponen RecyclerView ke dalam layout Activity atau Fragment di mana kita ingin menampilkannya. Karena contoh ini menggunakan Basic Activity, maka tambahkan komponen RecyclerView di file `content_main.xml`.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".MainActivity"
    tools:showIn="@layout/activity_main">

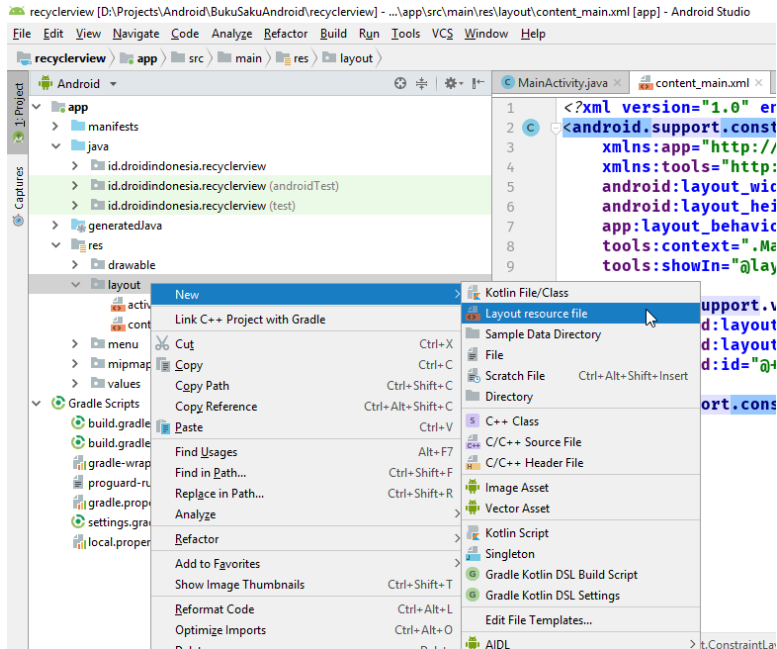
    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/recyclerView" />

</android.support.constraint.ConstraintLayout>
```

### ***2.1.3. Menambahkan layout item RecyclerView***

Selanjutnya buat file baru sebagai layout item-item yang akan ditampilkan di dalam RecyclerView. Klik kanan di folder **res>layout>New>Layout resource file**. Beri nama `country.xml`.

## Menambahkan layout item RecyclerView



Kemudian isi dengan kode berikut:

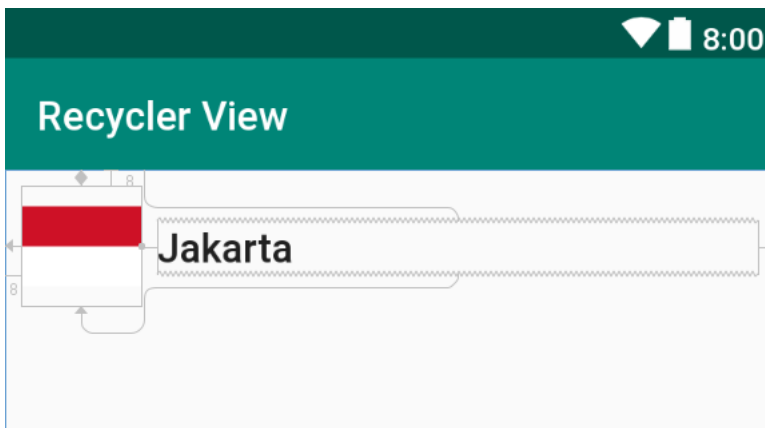
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <ImageView
        android:id="@+id/country_image"
        android:layout_width="60dp"
        android:layout_height="60dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:src="@drawable/indonesia"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/country_capital_text"
        android:layout_width="0dp"
```

```
android:layout_height="wrap_content"
android:layout_marginEnd="8dp"
android:layout_marginStart="8dp"
tools:text="Jakarta"
style="@style/TextAppearance.AppCompat.Title"
app:layout_constraintBottom_toBottomOf="@+id/country_image"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toEndOf="@+id/country_image"
app:layout_constraintTop_toTopOf="@+id/country_image" />
```

```
</android.support.constraint.ConstraintLayout>
```



Bendera Indonesia bisa kamu unduh di <http://flags.fmcdn.net/data/flags/w580/id.png>. Simpan di folder **res>drawable** dengan nama indonesia.png.

#### 2.1.4. Membuat kelas model (POJO)

Setelah kita mendesain layout untuk item di dalam RecyclerView, kita perlu membuat sebuah kelas yang merepresentasikan sebuah data. Kumpulan data inilah yang akan ditampilkan oleh si RecyclerView.

Buat file Java baru dan beri nama Country.java.

```
public class Country {
    String name, image;

    public Country() {
```

```
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getImage() {
    return image;
}

public void setImage(String image) {
    this.image = image;
}
}
```



## Tentang pembuatan kelas model (POJO)

Setiap membuat kelas model, cukup tuliskan field-nya saja (dalam model di atas adalah `String name`, `image`). Untuk kode sisanya lakukan secara otomatis lewat menu **Generate>Constructor>Select None** dan **Generate>Getter and Setter**. Menu **Generate** bisa dilihat dengan klik kanan atau ALT+INSERT.

### 2.1.5. Membuat Adapter RecyclerView

Selanjutnya, buat sebuah kelas Java baru dan beri nama `CountryListAdapter` yang meng-extends `RecyclerView.Adapter`.

```
public class CountryListAdapter extends
    RecyclerView.Adapter<CountryListAdapter.CountryViewHolder> {
}
```

### 2.1.6. Membuat ViewHolder

Setelah itu, buat kelas `CountryViewHolder` di dalam kelas `CountryListAdapter` (`CountryListAdapter.CountryViewHolder` maksudnya kita memanggil kelas `CountryViewHolder` yang berada di dalam `CountryListAdapter`).

```

public class CountryListAdapter extends
RecyclerView.Adapter<CountryListAdapter.CountryViewHolder> {

    class CountryViewHolder extends RecyclerView.ViewHolder {
        ImageView countryImageView; ❶
        TextView capitalTextView; ❷

        public CountryViewHolder(@NonNull View itemView) {
            super(itemView);

            countryImageView =
itemview.findViewById(R.id.country_image); ❸
            capitalTextView =
itemview.findViewById(R.id.country_capital_text); ❹
        }
    }
}

```

❸ Deklarasi objek `ImageView` yang ada di `country.xml`

❹ Deklarasi objek `TextView` yang ada di `country.xml`



## Selesaikan dulu ViewHolder

Untuk mempermudah proses pembuatan adapter `RecyclerView`, biasakan untuk menyelesaikan `ViewHolder` sebelum melanjutkan ke tahap berikutnya. Menyelesaikan `ViewHolder` akan sangat membantu proses sesudahnya.

### 2.1.7. Membuat list item

Sebuah adapter perlu memiliki list yang berisi kumpulan data yang ingin ditampilkan. Mari kita deklarasi objek list tersebut.

```

public class CountryListAdapter extends
RecyclerView.Adapter<CountryListAdapter.CountryViewHolder> {

    List<Country> countries = new ArrayList<>(); ❶

    public void setCountries(List<Country> countries) { ❷
        this.countries = countries;
    }
}

```

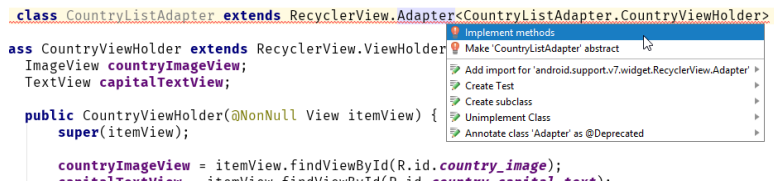
## Implementasi method wajib RecyclerView.Adapter

```
}  
  
// ... kode lain disembunyikan  
}
```

- ❶ Deklarasi objek list item Country
- ❷ Method ini berfungsi untuk mengisi list countries dari luar kelas CountryListAdapter

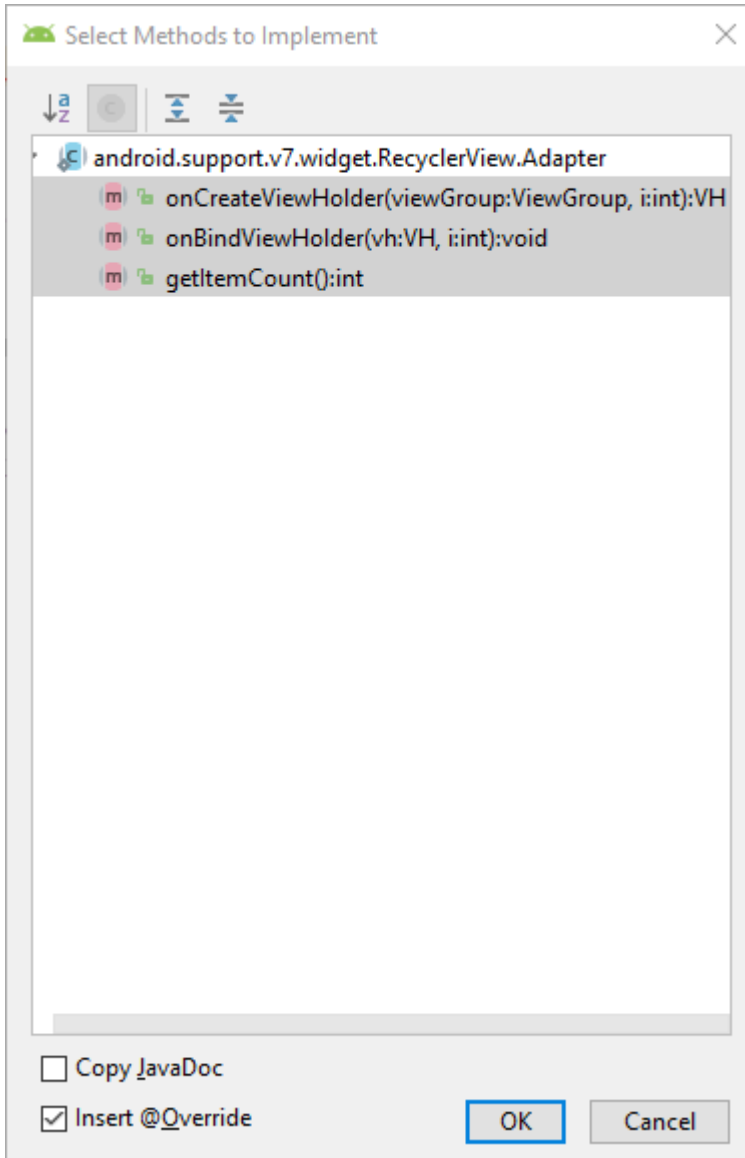
### 2.1.8. Implementasi method wajib RecyclerView.Adapter

Sebuah kelas yang meng-extends RecyclerView.Adapter wajib memiliki tiga method getItemCount, onCreateViewHolder, dan onBindViewHolder. Tiga method ini bisa ditulis secara otomatis menggunakan auto complete milik Android Studio. Klik di baris deklarasi kelas CountryListAdapter lalu tekan tombol ALT +ENTER di keyboard untuk menampilkan kotak dialog berikut.



```
class CountryListAdapter extends RecyclerView.Adapter<CountryListAdapter.CountryViewHolder>  
{  
    CountryViewHolder onCreateViewHolder(RecyclerView.ViewHolder itemView)  
    {  
        ImageView countryImageView;  
        TextView capitalTextView;  
  
        public CountryViewHolder(@NonNull View itemView) {  
            super(itemView);  
  
            countryImageView = itemView.findViewById(R.id.country_image);  
            capitalTextView = itemView.findViewById(R.id.country_capital_textview);  
        }  
    }  
}
```

Pilih **Implement methods**. Akan muncul sebuah kotak dialog baru. Pilih ketiga method yang ada dan klik **OK**.



Sekarang kelas countryListAdapter menjadi:

```
public class CountryListAdapter extends  
    RecyclerView.Adapter<CountryListAdapter.CountryViewHolder> {  
  
    List<Country> countries = new ArrayList<>();
```



```
public void setCountries(List<Country> countries) {
    this.countries = countries;
}

@NonNull
@Override
public CountryViewHolder onCreateViewHolder(@NonNull ViewGroup
viewGroup, int i) {
    return null;
}

@Override
public void onBindViewHolder(@NonNull CountryViewHolder
countryViewHolder, int i) {

}

@Override
public int getItemCount() {
    return 0;
}

class CountryViewHolder extends RecyclerView.ViewHolder {
    ImageView countryImageView;
    TextView capitalTextView;

    public CountryViewHolder(@NonNull View itemView) {
        super(itemView);

        countryImageView =
itemView.findViewById(R.id.country_image);
        capitalTextView =
itemView.findViewById(R.id.country_capital_text);
    }
}
}
```

## ***Implementasi getItemCount***

Method getItemCount akan mengembalikan jumlah data yang akan ditampilkan di RecyclerView. Untuk lebih amannya, kita akan mengambil jumlah data yang terdapat di dalam objek countries secara otomatis. Setial objek List memiliki method size yang bisa menghitung jumlah datanya secara otomatis.

```
@Override
public int getItemCount() {
    return countries.size();
}
```

## ***Implementasi onCreateViewHolder***

Method onCreateViewHolder berfungsi untuk membuat objek view dengan membaca layout country.xml yang kemudian bisa dipakai dalam proses inisiasi (findViewById) yang ada di dalam CountryViewHolder.

```
@NonNull
@Override
public CountryViewHolder onCreateViewHolder(@NonNull ViewGroup
viewGroup, int i) {
    View view =
    LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.country,
    viewGroup, false);
    return new CountryViewHolder(view);
}
```

## ***Implementasi onBindViewHolder***

Method onBindViewHolder akan dipanggil sebanyak jumlah data yang terdapat di dalam list countries. Method ini memiliki parameter bernama i (di beberapa versi Android Studio diberi nama position dan sesungguhnya boleh memiliki nama apapun, bahkan bisa diganti). Parameter i akan bernilai 0 hingga jumlah yang dikembalikan oleh method getItemCount yang dalam kasus ini merupakan jumlah list countries. Parameter tersebut bisa dimanfaatkan untuk mengambil isi list countries satu persatu berdasarkan nomor elemennya.

Setiap objek yang diambil dari dalam list kemudian bisa dipakai untuk mengisi layout country.xml berdasarkan isi masing-masing data.

```
@Override
public void onBindViewHolder(@NonNull CountryViewHolder
countryViewHolder, int i) {
    Country country = countries.get(i); ❶

    countryViewHolder.capitalTextView.setText(country.getName()); ❷
}
```

```
Picasso.get().load(country.getImage()).into(countryViewHolder.countryImageView);
}
```

- ❶ Ambil salah satu dari dalam list berdasarkan posisi
- ❷ Mengisi nama salah satu data ke item layout
- ❸ Menampilkan gambar dari URL menggunakan Picasso.

Pengisian data negara akan kita lakukan di tahap selanjutnya.

### 2.1.9. Inisialisasi RecyclerView, Adapter, dan data

Setelah selesai membuat adapter, sekarang mari lakukan inisialisasi data, Adapter, dan RecyclerView-nya. Deklarasikan tiga objek untuk RecyclerView, CountryListAdapter dan List<Country> seperti berikut ini.

```
public class MainActivity extends AppCompatActivity {

    RecyclerView recyclerView;
    CountryListAdapter adapter;

    List<Country> countryList = new ArrayList<>();

    // ... kode lain di sembunyikan
}
```

Setelah melakukan deklarasi tiga objek di atas, buat sebuah method baru bernama insertDummyData untuk menambahkan data ke countryList.

```
private void insertDummyData() {
    Country indonesia = new Country();
    indonesia.setName("Indonesia");
    indonesia.setImage("http://flags.fmcdn.net/data/flags/w580/id.png");
    countryList.add(indonesia);

    Country italy = new Country();
    italy.setName("Italy");
    italy.setImage("http://flags.fmcdn.net/data/flags/w580/it.png");
    countryList.add(italy);

    Country japan = new Country();
    japan.setName("Japan");
}
```

```
japan.setImage("http://flags.fmcdn.net/data/flags/w580/jp.png");
countryList.add(japan);
}
```

Berikutnya, lakukan inisialisasi RecyclerView dan Adapter.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);

    FloatingActionButton fab = (FloatingActionButton)
    findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Snackbar.make(view, "Replace with your own action",
            Snackbar.LENGTH_LONG)
                .setAction("Action", null).show();
        }
    });

    recyclerView = findViewById(R.id.recyclerView); ❶
    adapter = new CountryListAdapter(); ❷
    recyclerView.setAdapter(adapter); ❸
    recyclerView.setLayoutManager(new LinearLayoutManager(this)); ❹

    insertDummyData(); ❺

    adapter.setCountries(countryList); ❻
    adapter.notifyDataSetChanged(); ❼
}
```

- ❶ Inisialisasi RecyclerView, R.id.recyclerView adalah id komponen RecyclerView di content\_main.xml
- ❷ Inisialisasi objek adapter
- ❸ Memasangkan RecyclerView dengan sebuah Adapter. RecyclerView sesungguhnya adalah *dumb view*. Ia tidak tahu apa-apa atas data yang ia tampilkan, semua itu diatur oleh sebuah Adapter.

- ④ Mengatur bagaimana item-item di dalam RecyclerView, apakah berbentuk list, grid, atau staggered grid. LayoutManager akan di bahas di subbab mendatang.
- ⑤ Mengisi objek `countryList`
- ⑥ Menyalin isi `countryList` di `MainActivity` ke dalam `countries` di `CountryListAdapter`.
- ⑦ Memberitahu adapter bahwa ada perubahan data sehingga ia harus merefresh datanya.

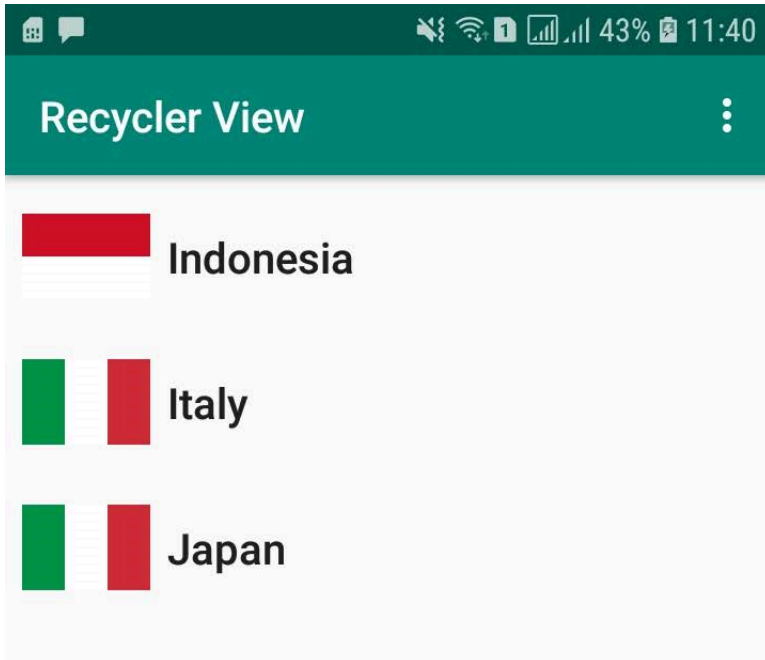
Pada contoh di atas, kita memanggil `notifyDataSetChanged` dari luar Adapter. Kita juga bisa memanggil `notifyDataSetChanged` dari dalam Adapter. Perhatikan kode berikut:

```
// di dalam CountryListAdapter
public void setCountries(List<Country> countries) {
    this.countries = countries;
    notifyDataSetChanged();
}

// di dalam MainActivity tidak perlu memanggil
// adapter.notifyDataSetChanged
// karena sudah dipanggil langsung saat mengirimkan data lewat
// setCountries
adapter.setCountries(countryList);
```

Kamu bebas untuk menggunakan cara yang pertama atau cara yang kedua, hasilnya sama saja. Selalu pilih cara yang menurutmu paling mudah dan bisa menyelesaikan permasalahan.

Jalankan aplikasi di emulator atau perangkat asli untuk melihat hasilnya.



Jika gambar tidak muncul, tambahkan permission INTERNET di AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.droidindonesia.recyclerview">

    // tambahkan baris berikut
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
```

```

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

        <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>

</manifest>

```

### 2.1.10. *OnClickListener dari dalam adapter*

Seringkali ketika pengguna menyentuh salah satu item kita ingin melakukan suatu operasi. Misalnya membuat halaman baru untuk menampilkan detail, mengubah warna latar belakang, dsb. Cara paling mudah dan paling cepat untuk mengimplementasi ialah dengan membuat OnClickListener dari dalam adapter.

```

@Override
public void onBindViewHolder(@NonNull CountryViewHolder
    countryViewHolder, int i) {
    Country country = countries.get(i);

    countryViewHolder.capitalTextView.setText(country.getName());

    Picasso.get().load(country.getImage()).into(countryViewHolder.countryImageView);

    ❶
    countryViewHolder.itemView.setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // lakukan sesuatu di sini
        }
    });
    ❷
    countryViewHolder.capitalTextView.setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // lakukan sesuatu di sini
        }
    });
    ❸

```

```
countryViewHolder.countryImageView.setOnClickListener(new
view.OnClickListener() {
    @Override
    public void onClick(View v) {
        // lakukan sesuatu di sini
    }
});
}
```

- ❶ Deteksi klik dibagian manapun di dalam layout country.xml
- ❷ Deteksi klik hanya di teks nama negara
- ❸ Deteksi klik hanya di bagian gambar

### ***2.1.11. OnClickListener dari luar adapter***

Ada kalanya kita ingin mendeteksi klik dari luar adapter. Jika ingin mendeteksi klik di luar adapter, maka pertama buat dulu sebuah interface sebagai berikut.

```
interface OnClickListener {
    public void onItemClick(int position, Country country);
}
```

Interface ini boleh disimpan di file terpisah atau disimpan di dalam kelas adapter. Penulis lebih sering menyimpannya di dalam kelas adapter langsung kecuali ia akan dipakai di lebih dari satu adapter. Saat dipakai oleh dari satu adapter baru akan penulis keluarkan ke file terpisah.

Berikutnya buat sebuah objek baru dari `OnClickListener` dan sebuah constructor untuk menginisialisasinya.

```
public class CountryListAdapter extends
RecyclerView.Adapter<CountryListAdapter.CountryViewHolder> {

    List<Country> countries = new ArrayList<>();
    OnClickListener listener;

    public CountryListAdapter(OnClickListener listener) {
        this.listener = listener;
    }
}
```



```
interface OnClickListener {
    void onItemClick(int position, Country country);
    void onCapitalTextClick(int position, Country country);
    void onImageClick(int position, Country country);
}

// ... kode lain disembunyikan
}
```

Berikutnya kita perbarui method `onBindViewHolder` menjadi:

```
@Override
public void onBindViewHolder(@NonNull CountryViewHolder
    countryViewHolder, final int i) {
    final Country country = countries.get(i);

    countryViewHolder.capitalTextView.setText(country.getName());

    Picasso.get().load(country.getImage()).into(countryViewHolder.countryImageView);

    countryViewHolder.itemView.setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            listener.onItemClick(i, country); ❶
        }
    });

    countryViewHolder.capitalTextView.setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            listener.onCapitalTextClick(i, country); ❷
        }
    });

    countryViewHolder.countryImageView.setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            listener.onImageClick(i, country); ❸
        }
    });
}
```

Sekarang kembali ke Activity atau Fragment dimana kita memanggil CountryListAdapter. Kelas Activity atau Fragment tersebut harus mengimplement interface OnClickListener yang ada di CountryListAdapter sebagai berikut:

```
public class MainActivity extends AppCompatActivity implements
    CountryListAdapter.OnClickListener {
}
```

Karena kita menambahkan baris implements CountryListAdapter.OnClickListener maka di dalam kelas ini kita harus mengimplementasi method-method yang didefinisikan di dalam OnClickListener tadi. Implementasi method yang ada di dalam interface OnClickListener bisa dilakukan dengan ALT+ENTER lalu **Implement methods** atau menuliskannya langsung seperti berikut:

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is
    // present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        return true;
    }

    return super.onOptionsItemSelected(item);
}

❶
@Override
public void onItemClick(int position, Country country) {
```

```
// lakukan sesuatu saat item di klik
}
❷
@Override
public void onCapitalTextClick(int position, Country country) {
    // lakukan sesuatu saat teks nama di klik
}
❸
@Override
public void onImageClick(int position, Country country) {
    // lakukan sesuatu saat gambar di klik
}
```

Berikutnya, karena tadi kita membuat sebuah constructor yang menerima *instance* kelas yang meng-implement `CountryListAdapter.OnClickListener`, maka kita perlu mengirimkan instance tersebut saat inisiasi objek adapter.

```
recyclerView = findViewById(R.id.recyclerView);
adapter = new CountryListAdapter(this); ❶
recyclerView.setAdapter(adapter);
recyclerView.setLayoutManager(new LinearLayoutManager(this));
```

Hasil akhir kelas MainActivity menjadi:

```
public class MainActivity extends AppCompatActivity implements
    CountryListAdapter.OnClickListener {

    private static final String TAG =
        MainActivity.class.getSimpleName();
    RecyclerView recyclerView;
    CountryListAdapter adapter;

    List<Country> countryList = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = (FloatingActionButton)
            findViewById(R.id.fab);
```

```
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
            .setAction("Action", null).show();
    }
});

recyclerView = findViewById(R.id.recyclerView);
adapter = new CountryListAdapter(this);
recyclerView.setAdapter(adapter);
recyclerView.setLayoutManager(new LinearLayoutManager(this));

insertDummyData();

adapter.setCountries(countryList);
adapter.notifyDataSetChanged();
}

private void insertDummyData() {
    Country indonesia = new Country();
    indonesia.setName("Indonesia");
    indonesia.setImage("http://flags.fmcdn.net/data/flags/w580/
id.png");
    countryList.add(indonesia);

    Country italy = new Country();
    italy.setName("Italy");
    italy.setImage("http://flags.fmcdn.net/data/flags/w580/it.png");
    countryList.add(italy);

    Country japan = new Country();
    japan.setName("Japan");
    japan.setImage("http://flags.fmcdn.net/data/flags/w580/jp.png");
    countryList.add(japan);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is
present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}
```

```

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        return true;
    }

    return super.onOptionsItemSelected(item);
}

@Override
public void onItemClick(int position, Country country) {
    // lakukan sesuatu saat item di klik
    Log.d(TAG, "onItemClick");
}

@Override
public void onCapitalTextClick(int position, Country country) {
    // lakukan sesuatu saat teks nama di klik
}

@Override
public void onImageClick(int position, Country country) {
    // lakukan sesuatu saat gambar di klik
}
}

```



---

# 3

## Intent

---

Intent adalah suatu mekanisme di dalam sistem Android untuk melakukan perpindahan antar Activity. Perpindahan tersebut bisa dilakukan dari dalam aplikasi yang sama (explicit intent) maupun dari aplikasi yang berbeda (implicit intent).

### 3.1. Membuat Splash Screen

Unduh logo untuk splash screen [di sini](#)<sup>1</sup> atau kamu bisa menggunakan logo sendiri. Simpan logo tadi di folder drawable.

Selanjutnya buatlah sebuah file Java baru, beri nama SplashActivity dan isikan menjadi:

```
public class SplashActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
    }  
}
```

Karena Activity ini kita buat secara manual, maka kita juga perlu mendaftarkannya secara manual di AndroidManifest.xml.

```
<activity  
    android:name=".MainActivity"
```

---

<sup>1</sup> <https://bit.ly/2GBwK17>

```

        android:label="@string/app_name"
        android:theme="@style/AppTheme.NoActionBar">
</activity>
<activity android:name=".SplashActivity">
</activity>

```

### 3.1.1. Mengubah Activity Launcher

By default, setiap aplikasi Android baru akan membuat MainActivity sebagai launcher (kecuali jika namanya diganti saat awal membuat *project*). Karena sebuah splash screen akan selalu tampil pertama sebelum Activity yang lain, maka kita harus mengubah SplashActivity sebagai launcher.

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.droidindonesia.intent">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
        </activity>
        <activity android:name=".SplashActivity">
            ❶
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>

```

- ❶ Pindahkan blok <intent-filter> dari MainActivity ke Splash Activity.



### 3.1.2. Membuat layout splash screen

Saat aplikasi pertama kali dijalankan, ada sedikit delay sebelum Activity launcher sukses memanggil onCreate. Saat delay ini terjadi (meskipun hanya beberapa milisecond, namun tetap terlihat mata), biasanya Android akan menampilkan windowBackground yang umumnya berwarna putih atau hitam. Atribut windowBackground ini, bisa diganti dengan sebuah drawable sebagai layar splash screen.

Buka file **res/values/styles.xml** lalu tambahkan deklarasi theme berikut:

```
<resources>

    // ... deklarasi theme lain disembunyikan

    <style name="AppTheme.Splash" parent="AppTheme.NoActionBar">
        <item name="android:windowBackground">@drawable/launch_screen</
item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    </style>
</resources>
```

Kita belum memiliki file launch\_screen. Buat file ini di dalam folder drawable, lalu isikan dengan:

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.android.com/apk/res/android"
    android:opacity="opaque">

    <item android:drawable="@color/colorPrimary"/>

    <item android:width="100dp" android:height="100dp"
        android:gravity="center">
        <bitmap
            android:src="@drawable/logo"
        />
    </item>

</layer-list>
```

### 3.1.3. Menambahkan theme Splash ke SplashActivity

Buka file AndroidManifest.xml lalu tambahkan atribut theme ke tag <activity> milik SplashActivity.

```
<activity android:name=".SplashActivity" android:theme="@style/
AppTheme.Splash">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

### 3.1.4. Redirect ke MainActivity

Buka SplashActivity lalu modifikasi method onCreate.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_splash);

    Intent intent = new Intent(this, MainActivity.class);
    startActivity(intent);
    finish();
}
```

Di sini, begitu masuk ke dalam SplashActivity, kita akan langsung berpindah ke MainActivity.

Run aplikasi untuk melihat hasilnya.