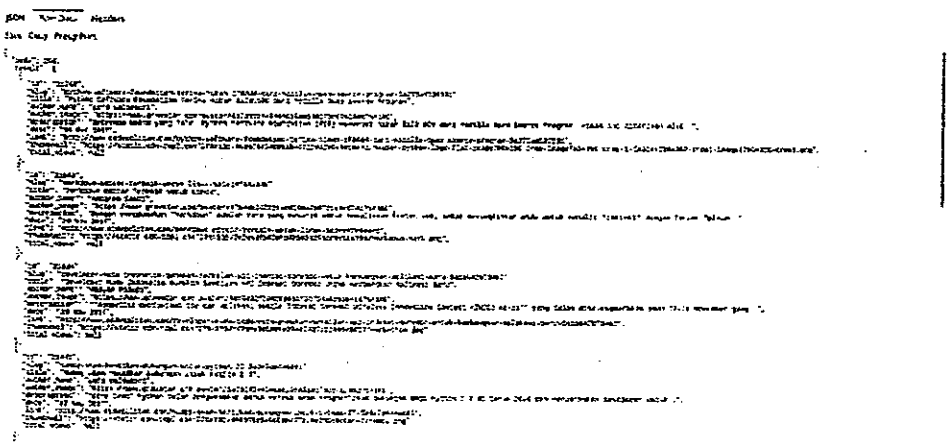
Modul Hari Keempat

Basic Android Application Development

Hanya untuk penggunaan pribadi dan proses pembelajaran di kelas Basic Android Application Development Codepolitan.com

1. Melakukan Konsumsi API

Hari ini kita akan belajar bagaimana mengonsumsi file JSON yang disediakan oleh sebuah API. Kita akan membuat tiruan aplikasi Android Codepolitan yang dapat menampilkan kumpulan data artikel. Buka halaman http://bit.ly/news_cp untuk melihat bagaimana struktur data yang akan kita baca.




```
{
  "code":200,
  "result":[
    {
      "id":"21769",
      "slug":"python-software-foundation-terima-hibah-170000-dari-
mozilla-open-source-program-5a275ad53f531",
      "title":"Python Software Foundation Terima Hibah $170.000 dari
 Mozilla Open Source Program",
      "author_name":"Lord Voldemort",
      "author_image":"https:\/\/www.gravatar.com\/avatar
\/6a7d1f244946a619883181f5b6f0cddd?s=100",
      "description":"Beberapa waktu yang lalu, Python Software
 Foundation (PSF) menerima hibah $170.000 dari Mozilla Open Source
 Program. Hibah ini diberikan oleh..",
```

```
        "date":"06 Dec 2017",
        "link":"http:\/\/www.codepolitan.com\/python-software-foundation-
terima-hibah-170000-dari-mozilla-open-source-program-5a275ad53f531",
        "thumbnail":"https:\/\/static.cdn-cdpl.com
\/270x135\/998b78e349061b4971c0a2b0e8d6be41\/header-python-logo-flat-
image700x350-crop-image700x350-crop-1-image(700x350-crop)-image(700x350-
crop).png",
        "total_views":null
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    },
    {
        // isi disembunyikan agar lebih ringkas
    }
  ]
}
```


1.1. Install Dependensi

Untuk membaca data di atas kita akan menggunakan sebuah *library* bernama Android Asynchronous Http Client. Dari data yang dibaca kita lalu menampilkannya ke RecyclerView. Karena kita akan melanjutkan aplikasi yang

telah kita buat sebelumnya, cukup tambahkan dependensi untuk *library* Android Asynchronous Http Client.

```
dependencies {
    implementation 'com.android.support:recyclerview-v7:26.1.0'
    implementation 'com.android.support:cardview-v7:26.1.0'
    compile 'com.squareup.picasso:picasso:2.5.2'
    compile 'com.loopj.android:android-async-http:1.4.9'
}
```

1.2. Menggunakan Android Asynchronous Http Client

Pertama buka kelas dimana kita ingin melakukan request data ke API Codepolitan. Silahkan buka NewsFragment.java dan tambahkan method berikut ini:

```
public class NewsFragment extends Fragment {

    public void readCodepolitanContent() {
        AsyncHttpClient client = new AsyncHttpClient();
 client.get("https://759b74ce43947f5f4c91aeddc3e5bad3d.codepolitan.com/
api/v2/posts/category/news", new AsyncHttpResponseHandler() {
            @Override
            public void onStart() {
                Log.d("NewsFragment", "onStart: proses pembacaan API
 dimulai");
            }

            @Override
            public void onSuccess(int statusCode, Header[] headers,
 byte[] responseBody) {
                Log.d("NewsFragment", "onSuccess: pembacaan data API
 berhasil");
            }

            @Override
            public void onFailure(int statusCode, Header[] headers,
 byte[] responseBody, Throwable error) {
                Log.d("NewsFragment", "onFailure: pembacaan data gagal
 dengan kode " + statusCode);
            }
```

```
            @Override
            public void onRetry(int retryNo) {
                Log.d("NewsFragment", "onRetry: melakukan pembacaan
 ulang API");
            }
        });
    }
}
```

Hal yang perlu kita lakukan adalah membuat objek AsyncHttpClient terlebih dahulu. Setelah objek dibuat, selanjutnya kita lanjutkan dengan pemanggilan method `get()` yang artinya kita ingin melakukan *request* HTTP GET ke sebuah URL (perhatikan parameter pertama). Sebagai parameter kedua, kita buat sebuah *anonymous class* yang akan meng-handle *callback* dari proses *request* HTTP.

Mari kita lanjutkan dengan melakukan parsing data JSON dan menambahkannya ke kelas `kumpulanArtikel` yang sebelumnya telah kita buat.

1.3. Parsing JSON

Tambahkan kode-kode berikut ke dalam method `onSuccess`:

```
@Override
public void onSuccess(int statusCode, Header[] headers, byte[]
 responseBody) {
    Log.d("NewsFragment", "onSuccess: pembacaan data API berhasil");

    try {
        JSONObject object = new
JSONObject(String.valueOf(responseBody));
        JSONArray result = object.getJSONArray("result");
        kumpulanArtikel.clear();

        for (int i = 0; i < result.length(); i++) {
            JSONObject objectArtikelJSON = result.getJSONObject(i);
            String id = objectArtikelJSON.getString("id");
            String slug = objectArtikelJSON.getString("slug");
            String title = objectArtikelJSON.getString("title");
            String authorName =
objectArtikelJSON.getString("author_name");
```

```
            String authorImage =
objectArtikelJSON.getString("author_image");
            String description =
objectArtikelJSON.getString("description");
            String date = objectArtikelJSON.getString("date");
            String link = objectArtikelJSON.getString("link");
            String thumbnail = objectArtikelJSON.getString("thumbnail");
            String total_views =
objectArtikelJSON.getString("total_views");

            Artikel artikelBaru = new Artikel(id, slug, title,
authorName, authorImage, description, date, link, thumbnail,
total_views);

            kumpulanArtikel.add(artikelBaru);
        }

        adapter.notifyDataSetChanged();

    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

Mari kita bahas kode di atas!

Karena proses parsing JSON dapat menghasilkan error bertipe JSONException, maka kita harus membuat blok try/catch yang membungkus proses parsing-nya.

Data yang dibaca dari URL yang tadi kita berikan disimpan di dalam `responseBody`. Karena responseBody masih dalam bentuk array int, maka kita ubah dulu menjadi String baru kemudian kita masukkan ke dalam sebuah JSONObject.

Sekarang `objectArtikelJSON` sudah memiliki seluruh isi dari data JSON. Karena data artikel ada di dalam sebuah array dengan key result, maka kita perlu membuat objek bertipe JSONArray.

JSONArray ini mirip dengan array biasa (dengan penambahan tentunya) sehingga bisa kita baca menggunakan `for`loop. Sebelum membuat `for` kita `clear()` dulu kumpulan Artikel sehingga data yang kita masukkan di `onCreateView` secara manual dibersihkan terlebih dahulu.

Di dalam for kita baca isi JSONArray result satu-persatu dan memasukkannya ke dalam objectArtikelJSON. Dari objectArtikelJSON kita ambil satu persatu data String-nya dengan method getString. String yang kita dapatkan kita kirimkan sebagai parameter membuat objek artikelBaru.

Dibaris terakhir forloop kita menambahkan artikelBaru kedalam kumpulanArtikel sehingga sekarang kumpulan artikel untuk menyimpan data-data dari API Codepolitan. Namun ada satu lagi yang perlu dilakukan sebelum RecyclerView kita me-refresh datanya.

Dibawah forloop kita panggil method notifyDataSetChanged di objek adapter. Pemanggilan method ini lah yang akan memberitahu adapter bahwa data kita sudah berubah sehingga RecylerView harus memuat ulang datanya.

Agar proses pembacaan API dimulai, kita harus memanggil terlebih dahulu method readCodepolitanContent di dalam onCreateView.

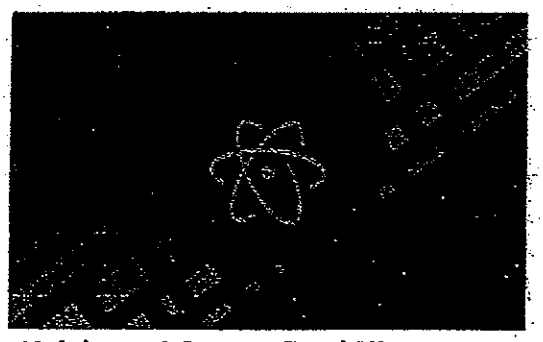
```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {

    // ... kode lain dihilangkan agar lebih ringkas
    adapter = new ArtikelListAdapter(kumpulanArtikel, getContext());
    recyclerView.setAdapter(adapter);

    readCodepolitanContent();

    return v;
}
```





Kolaborasi Secara Real Time dengan Teletype for Atom

Lord Voldemort



Dapatkah kamu selesaikan aplikasi ini dengan membuat tab INFO dan TUTORIAL menampilkan data-data dari dua URL berikut?

- http://bit.ly/info_cpcp
- http://bit.ly/tutorial_cp

1.4. Detail Activity

Pada umumnya aplikasi hanya menampilkan ringkasan data di dalam list dan menampilkan keseluruhan datanya ke halaman baru saat item tersebut di klik. Mari kita perbarui aplikasi ini dengan menambah sebuah `OnClickListener` pada RecyclerView lalu menampilkan halaman web Codepolitan menggunakan WebView dan mengubah teks judul di Toolbar.

Buat Activity Baru

Buat dulu sebuah Activity baru dengan nama DetailActivity.java lalu ubah isi content_detail.xml menjadi:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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="com.codepolitan.viewpager.DetailActivity"
    tools:showIn="@layout/activity_detail">

    <WebView
        android:id="@+id/browser"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</RelativeLayout>
```

Komponen <WebView> merupakan komponen yang dapat menampilkan sebuah halaman web. Mirip seperti Google Chrome tapi dalam bentuk yang lebih sederhana.

Buat OnClickListener di Adapter

Buka ArtikelListAdapter.java, di dalam method onBindViewHolder tambahkan kode yang mendeteksi klik pada salah satu item.

```
@Override
public void onBindViewHolder(ArtikelViewHolder holder, int position) {
    Artikel artikel = data.get(position);

    holder.tvPenulis.setText(artikel.getAuthor_name());
    holder.tvJudul.setText(artikel.getTitle());


Picasso.with(context).load(artikel.getThumbnail()).into(holder.ivThumbnail);

    holder.itemView.setOnClickListener(new View.OnClickListener() {
        @Override
```

```
        public void onClick(View v) {
            // operasi di sini
        }
    });
}
```

Seperti yang bisa kita lihat, objek untuk mendeteksi klik adalah `holder.itemView`. Objek `holder.itemView` merupakan objek yang membaca satu layout `artikel.xml`. Dengan memberikan `setOnClickListener` pada objek ini kita bisa mendeteksi klik dibagian manampun baik itu di gambar thumbnail, teks judul maupun teks penulis. Apabila hanya ingin memberikan deteksi klik dibagian tertentu misalnya hanya saat pengguna mengklik gambar thumbnail maka ganti dari `holder.itemView.setOnClickListener()` menjadi `holder.ivThumbnail.setOnClickListener()`.

Pindah Activity

Karena `OnClickListener` sudah siap, sekarang kita lakuan perpindahan Activity dengan membuat objek intent. Dari objek intent yang baru dibuat, kita juga ingin mengirimkan data-data tiap artikel sehingga perlu memanggil `putExtra`.

```
holder.itemView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent pindahIntent = new Intent(context, DetailActivity.class);
        pindahIntent.putExtra("TITLE", artikel.getTitle());
        pindahIntent.putExtra("LINK", artikel.getLink());
        context.startActivity(pindahIntent);
    }
});
```

Contructor Intent selalu meminta objek context sebagai parameter pertamanya. Sekali lagi, karena bukan berada di dalam sebuah Activity kita tidak bisa menulis NamaActivity.this. Namun, karena kita juga sudah memiliki objek `context` yang dikirim dari `getContext` NewsFragment, kita bisa mendapatkan hasil yang sama.

Disini kita bisa melakukan pengiriman data seperti yang kita pelajari di hari kedua. Hanya saja, saat data yang akan dikirimkan cukup banyak tentu kita akan kerepotan menuliskannya secara manual. Solusi lainnya ialah dengan mengirimkan objek `artikel` langsung.

```
holder.itemView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent pindahIntent = new Intent(context, DetailActivity.class);
        pindahIntent.putExtra("ARTIKEL", artikel);
        context.startActivity(pindahIntent);
    }
});
```

Kode di atas masih diberi warna merah oleh Android studio karena ia tidak mengenal objek artikel. Agar bisa mengirimkan satu objek secara keseluruhan, kita harus membuat kelas Artikel.java untuk mengimplement `Parcelable`.

```
public class Artikel implements Parcelable {

    // ...  kode lain disembunyikan agar lebih singkat.

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

    @Override
    public void writeToParcel(Parcel dest, int flags) {
        dest.writeString(this.id);
        dest.writeString(this.slug);
        dest.writeString(this.title);
        dest.writeString(this.author_name);
        dest.writeString(this.author_image);
        dest.writeString(this.description);
        dest.writeString(this.date);
        dest.writeString(this.link);
        dest.writeString(this.thumbnail);
        dest.writeString(this.total_views);
    }

    protected Artikel(Parcel in) {
        this.id = in.readString();
        this.slug = in.readString();
        this.title = in.readString();
        this.author_name = in.readString();
        this.author_image = in.readString();
        this.description = in.readString();
        this.date = in.readString();
```

```
        this.link = in.readString();
        this.thumbnail = in.readString();
        this.total_views = in.readString();
    }

    public static final Parcelable.Creator<Artikel> CREATOR = new
Parcelable.Creator<Artikel>() {
        @Override
        public Artikel createFromParcel(Parcel source) {
            return new Artikel(source);
        }

        @Override
        public Artikel[] newArray(int size) {
            return new Artikel[size];
        }
    };
}
```



Untuk mempermudah penulisan Parcelable gunakan plugin **Android Parcelable Generator** untuk Android Studio.

Baca Data Extra di Detail Activity

Proses pembacaan data yang dikirimkan oleh Adapter di DetailActivity tidak jauh berbeda dengan yang kita pelajari sebelumnya. Perbedaannya hanya apda tipe data yang kita terima dan method yang digunakan untuk mengambil data Artikel.

```
public class DetailActivity extends AppCompatActivity {

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

        getSupportActionBar().setDisplayHomeAsUpEnabled(true);

        Artikel artikel = getIntent().getParcelableExtra("ARTIKEL");
    }
```

```
}
```

Karena kita sudah memiliki data yang diperlukan dari Adapter, sekarang mari kita muat halaman web dari link yang dimiliki oleh objek Artikel lalu mengubah judul Activity menjadi judul Artikel.

```
public class DetailActivity extends AppCompatActivity {

    WebView webView;

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

        getSupportActionBar().setDisplayHomeAsUpEnabled(true);

        Artikel artikel = getIntent().getParcelableExtra("ARTIKEL");

        webView = findViewById(R.id.browser);
        webView.loadUrl(artikel.getLink());

        getSupportActionBar().setTitle(artikel.getTitle());
    }

}
```

1.5. Penutup

Proses pembacaan data dari API dapat memakan waktu hingga beberapa detik. Apabila terdeteksi ada sebuah proses yang memakan waktu dan mem-blok Thread UI, sistem Android akan menampilkan jendela yang dikenal dengan istilah ANR (Android Not Responding). Oleh karena itu semua proses yang memakan waktu harus dilakukan di Thread yang berbeda. Android Asynchronous Http Client sudah melakukan pekerjaannya di belakang layar sehingga kita tidak perlu menuliskannya secara manual.

Apabila kita ingin melakukan proses lain dibelakang layar maka kita perlu menggunakan kelas AsyncTask [1].

[1] https://developer.android.com/reference/android/os/AsyncTask.html