

Pada latihan ini kita akan mencoba menerapkan salah satu **loader** yaitu **AsyncTaskLoader**. Kita akan menggunakan **AsyncTaskLoader** untuk mengambil data dari openweather.org dan kemudian menampilkannya ke dalam **listview**.

1. Silakan buat project baru dengan nama **MyAsyncTaskLoader** dengan **emptyactivity**. Setelah selesai silakan kondisikan **activity_main.xml** menjadi seperti ini.

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:tools="http://schemas.android.com/tools"
4.     android:id="@+id/activity_main"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:paddingBottom="@dimen/activity_vertical_margin"
8.     android:paddingLeft="@dimen/activity_horizontal_margin"
9.     android:paddingRight="@dimen/activity_horizontal_margin"
10.    android:paddingTop="@dimen/activity_vertical_margin"
11.    android:orientation="vertical"
12.    tools:context="com.dicoding.www.asynctaskloader.MainActivity">
13.    <LinearLayout
14.        android:layout_width="match_parent"
15.        android:layout_height="wrap_content"
16.        android:orientation="horizontal"
17.        android:gravity="center"
18.        android:weightSum="1"
19.    >
20.        <EditText
21.            android:id="@+id/edit_kota"
22.            android:layout_width="0dp"
23.            android:layout_height="wrap_content"
24.            android:text="@string/id_jakarta_bandung_semararang"
25.            android:hint="@string/masukkan_nama_kota"
26.            android:layout_weight="0.8"/>
27.        <Button
28.            android:id="@+id/btn_kota"
29.            android:layout_width="wrap_content"
30.            android:layout_height="wrap_content"
31.            android:text="@string/cari"
32.            android:layout_weight="0.2"
33.        />
34.    </LinearLayout>
35.    <ListView
36.        android:layout_width="match_parent"
37.        android:layout_height="match_parent"
38.        android:id="@+id/listView"/>
39. </LinearLayout>
```

Jangan lupa untuk menambahkan file **dimens.xml** secara manual di dalam **res** → **values**. Dan isikan file **dimens.xml** seperti berikut

```
1. <resources>
2.     <!-- Default screen margins, per the Android Design guidelines. -->
3.     <dimen name="activity_horizontal_margin">16dp</dimen>
4.     <dimen name="activity_vertical_margin">16dp</dimen>
5. </resources>
```

Tambahkan 1 lagi pada res / layout dan beri nama **weather_items.xml**

```
40. <?xml version="1.0" encoding="utf-8"?>
41.
42. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
43.     android:orientation="vertical" android:layout_width="match_parent"
44.     android:layout_height="match_parent">
45.     <TextView
46.         android:text="@string/nama_kota"
47.         android:paddingLeft="@dimen/activity_horizontal_margin"
48.         android:paddingRight="@dimen/activity_horizontal_margin"
49.         android:paddingTop="@dimen/activity_vertical_margin"
50.         android:layout_width="wrap_content"
51.         android:layout_height="wrap_content"
52.         android:id="@+id/textKota"/>
53.     <TextView
54.         android:text="@string/temperatur"
55.         android:id="@+id/textTemp"
56.         android:paddingLeft="@dimen/activity_horizontal_margin"
57.         android:paddingRight="@dimen/activity_horizontal_margin"
58.         android:layout_width="wrap_content"
59.         android:layout_height="wrap_content" />
60.     <TextView
61.         android:text="@string/deskripsi"
62.         android:paddingBottom="@dimen/activity_vertical_margin"
63.         android:paddingLeft="@dimen/activity_horizontal_margin"
64.         android:paddingRight="@dimen/activity_horizontal_margin"
65.         android:layout_width="wrap_content"
66.         android:layout_height="wrap_content"
67.         android:id="@+id/textDesc"/>
68. </LinearLayout>
```

2. Akan ada bagian yang dimerahi pada kode layout di atas. Hal ini terjadi karena *resource string* belum ditambahkan.

Tambahkan resource string di dalam res → values → strings.xml.

```
1. <resources>
2.     <string name="app_name">AsyncTaskLoader</string>
3.     <string name="deskripsi">Deskripsi</string>
4.     <string name="temperatur">Temperatur</string>
5.     <string name="nama_kota">Nama Kota</string>
6.     <string name="masukkan_nama_kota">Masukkan id kota misal Jakarta = 1
    642911</string>
7.     <string name="cari">Cari</string>
8.     <string name="jakarta">Jakarta</string>
9.     <string name="id_jakarta_bandung_semararang">1642911,1650357,1627896
    </string>
10.</resources>
```

Kemudian klik sync now.

3. Pada file **gradle** level module (build.gradle: module) tambahkan library **loopj** dengan menambahkan satu baris ini :

```
1. compile 'com.loopj.android:android-async-http:1.4.9'
```

Kemudian klik sync now.

4. Buat class baru bernama **WeatherAdapter** seperti di bawah ini.

```
1. public class WeatherAdapter extends BaseAdapter {
2. }
```

Akan muncul tanda merah pada kode di atas. Seperti biasa, kita tinggal klik *implements methods*. Kemudian lengkapi weatherAdapter menjadi seperti berikut :

```
3. public class WeatherAdapter extends BaseAdapter {
4.
5.     private ArrayList<WeatherItems> mData = new ArrayList<>();
6.     private LayoutInflater mInflater;
7.     private Context context;
8.
9.     public WeatherAdapter(Context context) {
10.         this.context = context;
11.         mInflater = (LayoutInflater)context.getSystemService(Context.LAY
    OUT_INFLATER_SERVICE);
12.     }
13.
14.     public void setData(ArrayList<WeatherItems> items){
15.         mData = items;
```

```

16.         notifyDataSetChanged();
17.     }
18.     public void addItem(final WeatherItems item) {
19.         mData.add(item);
20.         notifyDataSetChanged();
21.     }
22.
23.     public void clearData(){
24.         mData.clear();
25.     }
26.     @Override
27.     public int getItemViewType(int position) {
28.         return 0;
29.     }
30.
31.     @Override
32.     public int getViewTypeCount() {
33.         return 1;
34.     }
35.
36.     @Override
37.     public int getCount() {
38.         if (mData == null) return 0;
39.         return mData.size();
40.     }
41.
42.     @Override
43.     public WeatherItems getItem(int position) {
44.         return mData.get(position);
45.     }
46.
47.     @Override
48.     public long getItemId(int position) {
49.         return position;
50.     }
51.
52.     @Override
53.     public View getView(int position, View convertView, ViewGroup parent
54.     ) {
55.         ViewHolder holder = null;
56.         if (convertView == null) {
57.             holder = new ViewHolder();
58.             convertView = mInflater.inflate(R.layout.weather_items, null
59.             );
60.             holder.textViewNamaKota= (TextView)convertView.findViewById(
61.             R.id.textKota);
62.             holder.textViewTemperature = (TextView)convertView.findViewById(R.id.textTemp);
63.             holder.textViewDescription = (TextView)convertView.findViewById(R.id.textDesc);
64.             convertView.setTag(holder);
65.         } else {
66.             holder = (ViewHolder) convertView.getTag();
67.         }
68.         holder.textViewNamaKota.setText(mData.get(position).getNama());

```

```

66.     holder.textViewTemperature.setText(mData.get(position).getTemper
ature());
67.     holder.textViewDescription.setText(mData.get(position).getDescri
ption());
68.     return convertView;
69. }
70.
71. private static class ViewHolder {
72.     TextView textViewNamaKota;
73.     TextView textViewTemperature;
74.     TextView textViewDescription;
75. }
76. }

```

5. Kemudian buatlah kelas baru untuk menyimpan *item* dari weather yaitu ***WeatherItems***.

```

1. public class WeatherItems {
2. }

```

Lengkapi ***WeatherItems*** menjadi di bawah ini :

```

3. public class WeatherItems {
4.     private int id;
5.     private String nama;
6.     private String currentWeather;
7.     private String description;
8.     private String temperature;
9.
10.    public WeatherItems(JSONObject object){
11.
12.        try {
13.            int id = object.getInt("id");
14.            String name = object.getString("name");
15.            String currentWeather = object.getJSONArray("weather").getJS
ONObject(0).getString("main");
16.            String description = object.getJSONArray("weather").getJSO
bject(0).getString("description");
17.            double tempInKelvin = object.getJSONObject("main").getDouble
("temp");
18.
19.            double tempInCelcius = tempInKelvin - 273;
20.            String temperature = new DecimalFormat("##.##").format(tempI
nCelcius);
21.            this.id = id;
22.            this.nama = name;
23.            this.currentWeather = currentWeather;
24.            this.description = description;
25.            this.temperature = temperature;
26.
27.        }catch (Exception e){

```

```

28.
29.         e.printStackTrace();
30.     }
31. }
32.
33. public int getId() {
34.     return id;
35. }
36.
37. public void setId(int id) {
38.     this.id = id;
39. }
40.
41. public String getNama() {
42.     return nama;
43. }
44. public void setNama(String nama) {
45.     this.nama = nama;
46. }
47.
48. public String getCurrentWeather() {
49.     return currentWeather;
50. }
51. public void setCurrentWeather(String currentWeather) {
52.     this.currentWeather = currentWeather;
53. }
54.
55. public String getDescription() {
56.     return description;
57. }
58.
59. public void setDescription(String description) {
60.     this.description = description;
61. }
62.
63. public String getTemperature() {
64.     return temperature;
65. }
66.
67. public void setTemperature(String temperature) {
68.     this.temperature = temperature;
69. }
70. }

```

6. Buatlah kelas yang meng-extend **AsyncTaskLoader** seperti di bawah ini :

```

1. public class MyAsyncTaskLoader extends AsyncTaskLoader<ArrayList<Weather
   Items>> {
2.     private ArrayList<WeatherItems> mData;
3.     private boolean mHasResult = false;
4.
5.     private String mKumpulanKota;
6.

```

```

7.     public MyAsyncTaskLoader(final Context context,String kumpulanKota)
8.     {
9.         super(context);
10.        onContentChanged();
11.        this.mKumpulanKota = kumpulanKota;
12.    }
13.
14.    @Override
15.    protected void onStartLoading() {
16.        if (takeContentChanged())
17.            forceLoad();
18.        else if (mHasResult)
19.            deliverResult(mData);
20.    }
21.
22.    @Override
23.    public void deliverResult(final ArrayList<WeatherItems> data) {
24.        mData = data;
25.        mHasResult = true;
26.        super.deliverResult(data);
27.    }
28.
29.    @Override
30.    protected void onReset() {
31.        super.onReset();
32.        onStopLoading();
33.        if (mHasResult) {
34.            onReleaseResources(mData);
35.            mData = null;
36.            mHasResult = false;
37.        }
38.    }
39.
40.    private static final String API_KEY = "Isikan API KEY anda...";
41.
42.    // Format search kota url JAKARTA = 1642911 ,BANDUNG = 1650357, SEMA
RANG = 1627896
43.    // http://api.openweathermap.org/data/2.5/group?id=1642911,1650357,1
627896&units=metric&appid=API_KEY
44.
45.    @Override
46.    public ArrayList<WeatherItems> loadInBackground() {
47.        SyncHttpClient client = new SyncHttpClient();
48.
49.        final ArrayList<WeatherItems> weatherItemses = new ArrayList<>()
;
50.        String url = "http://api.openweathermap.org/data/2.5/group?id="
+
51.            mKumpulanKota+ "&units=metric&appid=" + API_KEY;
52.
53.        client.get(url, new AsyncHttpResponseHandler() {
54.            @Override
55.            public void onStart() {
56.                super.onStart();

```

```

57.         setUseSynchronousMode(true);
58.     }
59.
60.     @Override
61.     public void onSuccess(int statusCode, Header[] headers, byte
    [] responseBody) {
62.         try {
63.             String result = new String(responseBody);
64.             JSONObject responseObject = new JSONObject(result);
65.             JSONArray list = responseObject.getJSONArray("list")
        ;
66.
67.             for (int i = 0 ; i < list.length() ; i++){
68.                 JSONObject weather = list.getJSONObject(i);
69.                 WeatherItems weatherItems = new WeatherItems(wea
    ther);
70.                 weatherItemses.add(weatherItems);
71.             }
72.         } catch (Exception e){
73.             e.printStackTrace();
74.         }
75.     }
76.
77.     @Override
78.     public void onFailure(int statusCode, Header[] headers, byte
    [] responseBody, Throwable error) {
79.         //Jika response gagal maka , do nothing
80.     }
81. });
82.
83.     return weatherItemses;
84. }
85.
86. protected void onReleaseResources(ArrayList<WeatherItems> data) {
87.     //nothing to do.
88. }
89.
90. }

```

Perhatikan pada baris ini :

```

91. "http://api.openweathermap.org/data/2.5/group?id=" +
92.     mKumpulanKota+ "&units=metric&appid=" + API_KEY;

```

Untuk **API_KEY** isikan dengan apikey yang telah kamu buat .

7. Setelah itu pada **MainActivity** kondisikan seperti berikut :

```

1. public class MainActivity extends AppCompatActivity implements LoaderMan
    ager.LoaderCallbacks<ArrayList<WeatherItems>> {
2.

```



```

3.     ListView listView ;
4.     WeatherAdapter adapter;
5.     EditText editKota;
6.     Button buttonCari;
7.
8.     static final String EXTRAS_CITY = "EXTRAS_CITY";
9.     @Override
10.    protected void onCreate(Bundle savedInstanceState) {
11.        super.onCreate(savedInstanceState);
12.        setContentView(R.layout.activity_main);
13.
14.        adapter = new WeatherAdapter(this);
15.        adapter.notifyDataSetChanged();
16.        listView = (ListView)findViewById(R.id.listView);
17.
18.        listView.setAdapter(adapter);
19.
20.        editKota = (EditText)findViewById(R.id.edit_kota);
21.        buttonCari = (Button)findViewById(R.id.btn_kota);
22.
23.        buttonCari.setOnClickListener(myListener);
24.
25.        String kota = editKota.getText().toString();
26.        Bundle bundle = new Bundle();
27.        bundle.putString(EXTRAS_CITY,kota);
28.
29.        getLoaderManager().initLoader(0, bundle, this);
30.    }
31.
32.    //Fungsi ini yang akan menjalankan proses myasynctaskloader
33.    @Override
34.    public Loader<ArrayList<WeatherItems>> onCreateLoader(int id, Bundle
args) {
35.
36.        String kumpulanKota = "";
37.        if (args != null){
38.            kumpulanKota = args.getString(EXTRAS_CITY);
39.        }
40.
41.        return new MyAsyncTaskLoader(this,kumpulanKota);
42.    }
43.
44.    @Override
45.    public void onLoadFinished(Loader<ArrayList<WeatherItems>> loader, A
rrayList<WeatherItems> data) {
46.
47.        adapter.setData(data);
48.    }
49.
50.    @Override
51.    public void onLoaderReset(Loader<ArrayList<WeatherItems>> loader) {
52.        adapter.setData(null);
53.
54.    }
55.

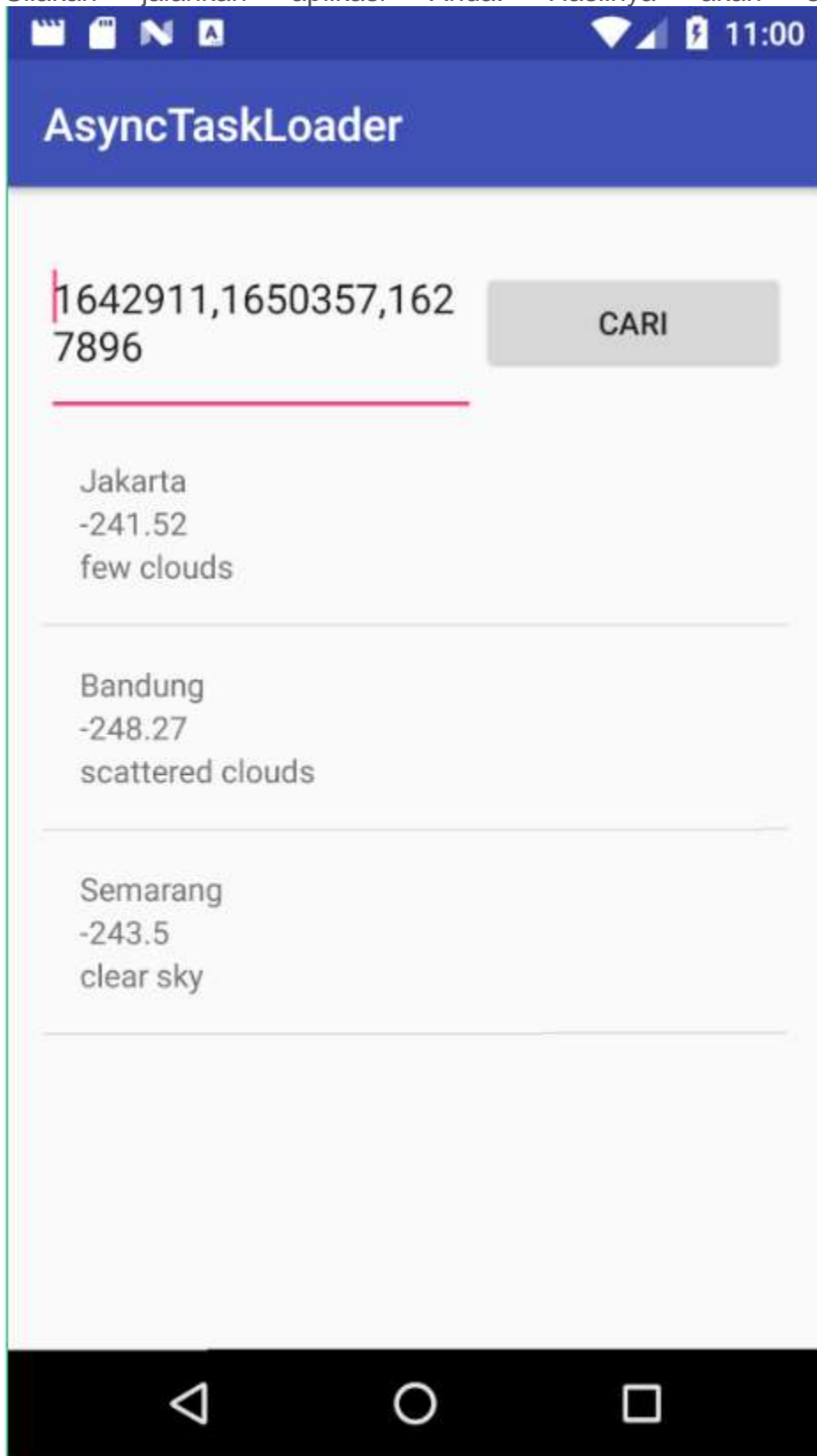
```

```
56.     View.OnClickListener myListener = new View.OnClickListener() {
57.         @Override
58.         public void onClick(View v) {
59.             String kota = editKota.getText().toString();
60.
61.             if (TextUtils.isEmpty(kota))return;
62.
63.             Bundle bundle = new Bundle();
64.             bundle.putString(EXTRAS_CITY,kota);
65.             getLoaderManager().restartLoader(0,bundle,MainActivity.this)
66.         ;
67.     }
68. }
```

8. Jangan lupa tambahkan permission

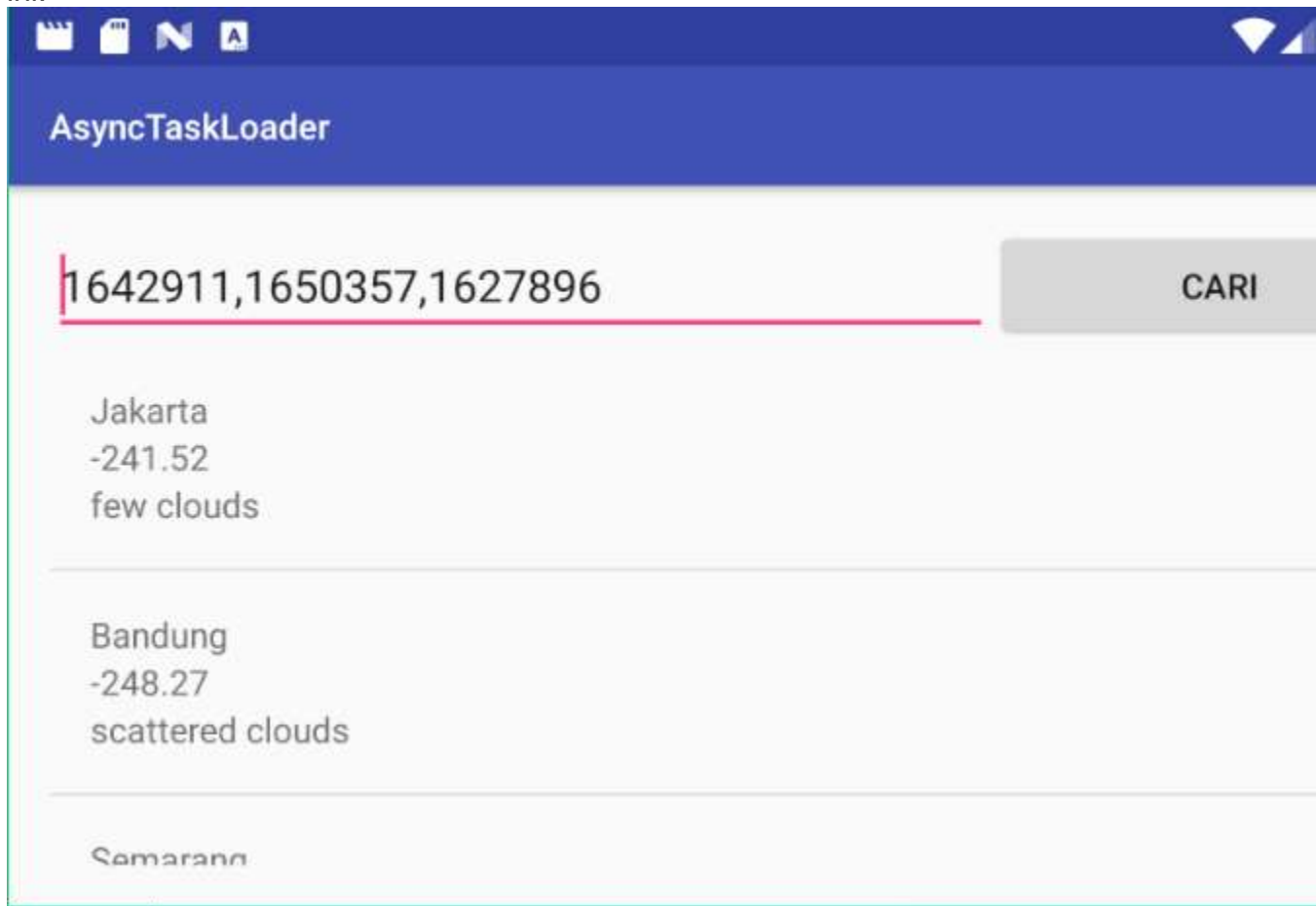
```
1. <uses-permission android:name="android.permission.INTERNET"></uses-permission>
```

9. Silakan jalankan aplikasi Anda. Hasilnya akan seperti ini ::



Ketika orientasi perangkat Android diubah, maka tampilannya kurang lebih akan menjadi seperti

ini:



Bedah Kode

Pertama, mari kita perhatikan kode berikut ini. Kode ini berada di dalam **MainActivity.java**.

```
1. getLoaderManager().initLoader(0, bundle, this);
```

Ketika kode di atas dijalankan, metode **onCreateLoader()** akan dijalankan. Di dalam metode **onCreateLoader()** kita membuat obyek **MyAsyncTaskLoader** baru.

```
1. @Override
2. public Loader<ArrayList<WeatherItems>> onCreateLoader(int id, Bundle args) {
3.     String kumpulanKota = "";
4.     if (args != null){
5.         kumpulanKota = args.getString(EXTRAS_CITY);
6.     }
```

```

7.
8.     return new MyAsyncTaskLoader(this, kumpulanKota);
9. }
10.
11. @Override
12. public void onLoadFinished(Loader<ArrayList<WeatherItems>> loader, ArrayList<W
    eatherItems> data) {
13.     adapter.setData(data);
14. }
15.
16. @Override
17. public void onLoaderReset(Loader<ArrayList<WeatherItems>> loader) {
18.     adapter.setData(null);
19. }
20. }

```

Mari kita bedah kode di atas. Metode **OnLoadFinished()** akan dijalankan ketika proses loading background telah selesai. Sedangkan metode **OnLoadReset()** akan dijalankan ketika **loader** direset.

Perhatikan parameter **Bundle** pada metode **onCreateLoader**. Dengan menggunakan **Bundle**, kita bisa mengirim parameter ke **loadermanager**. Parameter ini akan digunakan pada constructor kelas **Loader** di dalam **onCreateLoader**. **Bundle** ini juga berguna ketika kita ingin mengambil data ketika **onClick** dijalankan:

```

1. View.OnClickListener myListener = new View.OnClickListener() {
2.     @Override
3.     public void onClick(View v) {
4.         String kota = editKota.getText().toString();
5.
6.         if (TextUtils.isEmpty(kota))return;
7.
8.         Bundle bundle = new Bundle();
9.         bundle.putString(EXTRAS_CITY, kota);
10.        getLoaderManager().restartLoader(0, bundle, MainActivity.this);
11.    }
12. };

```

restartLoader() berarti **loader** akan direstart kembali. Dengan memanggil **restartLoader()** maka **onCreateLoader** akan kembali dijalankan. Secara otomatis loader baru akan diciptakan. Cara ini lah yang digunakan untuk mengambil data kembali dan kemudian menampilkannya di dalam **onLoadFinished**.

```

1. public void setData(ArrayList<WeatherItems> items){
2.     mData = items;
3.     notifyDataSetChanged();
4. }

```

Dalam **adapter.setData(data)** terjadi proses pengisian item dari **WeatherItems** ke dalam **adapter**. Kemudian metode **notifyDataSetChanged()** akan dijalankan.

notifyDataSetChanged() berfungsi untuk mengabari **adapter** bahwa ada data baru yang telah diterima. Ketika fungsi ini dijalankan, maka **listview** yang didaftarkan pada **adapter** akan menampilkan data tersebut.

Setelah itu kita perhatikan kelas **MyAsyncTaskLoader**. Ada beberapa metode penting untuk dicermati :

```
1. @Override
2. protected void onStartLoading() {
3.     Log.d("Content Changed", "1");
4.     if (takeContentChanged())
5.         forceLoad();
6.     else if (hasResult)
7.         deliverResult(mData);
8. }
9.
10. @Override
11. public void deliverResult(final ArrayList<WeatherItems> data) {
12.     mData = data;
13.     hasResult = true;
14.     super.deliverResult(data);
15. }
16.
17. @Override
18. protected void onReset() {
19.     super.onReset();
20.     onStopLoading();
21.     if (hasResult) {
22.         onReleaseResources(mData);
23.         mData = null;
24.         hasResult = false;
25.     }
26. }
```

OnStartLoading() dipanggil setelah proses load berjalan. Terdapat percabangan **if else** yang menuju pada proses **forceLoad** dan **deliverResult**.

Metode **forceLoad()** akan dijalankan ketika data belum tersedia. Maka data tersebut akan diambil terlebih dahulu. Load data dapat terjadi ketika **loader** pertama kali dipanggil atau ketika terjadi perubahan pada data.

Metode **deliverResult()** digunakan untuk menampilkan result data yang sudah ada. Metode ini akan dijalankan juga ketika terjadi reset pada **loader**.

Ketika proses reset terjadi, maka **hasResult** akan diperiksa. Bila hasilnya tersedia, maka **onReleaseResources** akan dijalankan. Metode ini akan melepas memori resources data.

```
1. protected void onReleaseResources(ArrayList<WeatherItems> data) {
2.     //nothing to do.
3. }
```

Metode **onReleaseResources()** di atas tidak berisikan kode apapun, kita tidak perlu melepaskan memori. Berbeda bila kita menggunakan **cursor** yang membutuhkan proses pelepasan memori. Bila memori tidak dibebaskan, maka bisa terjadi kebocoran memori (*memory leak*).

Yang terakhir perhatikan beberapa baris berikut ini :

```
1. @Override
2. public ArrayList<WeatherItems> loadInBackground() {
3.     SyncHttpClient client = new SyncHttpClient();
4.
5.     final ArrayList<WeatherItems> weatherItemses = new ArrayList<>();
6.     String url = "http://api.openweathermap.org/data/2.5/group?id=" +
7.         mKumpulanKota + "&units=metric&appid=" + API_KEY;
8.
9.     client.get(url, new AsyncHttpResponseHandler() {
10.         @Override
11.         public void onStart() {
12.             super.onStart();
13.             //Menggunakan synchronous karena pada dasarnya thread yang digunak
14.             //loadInBackground mengembalikan nilai balikan
15.             setUseSynchronousMode(true);
16.         }
17.
18.         @Override
19.         public void onSuccess(int statusCode, Header[] headers, byte[] respons
20.             eBody) {
21.             try {
22.                 String result = new String(responseBody);
23.                 JSONObject responseObject = new JSONObject(result);
24.                 JSONArray list = responseObject.getJSONArray("list");
25.
26.                 for (int i = 0; i < list.length(); i++) {
27.                     JSONObject weather = list.getJSONObject(i);
28.                     WeatherItems weatherItems = new WeatherItems(weather);
29.                     weatherItemses.add(weatherItems);
30.                 }
31.             } catch (Exception e) {
32.                 //Jika terjadi error pada saat parsing maka akan masuk ke catc
33.                 h()
34.                 e.printStackTrace();
35.             }
36.         }
37.     })
38. }
```

```

37.     public void onFailure(int statusCode, Header[] headers, byte[] respons
    eBody, Throwable error) {
38.         //Jika response gagal maka , do nothing
39.     }
40. });
41.
42.     return weatherItemses;
43. }

```

Metode di atas akan menjalankan proses pengambilan data secara *synchronous*. Hal ini dapat kita lihat pada inisiasi obyeknya seperti pada potongan kode di bawah ini :

```

1. SyncHttpClient client = new SyncHttpClient();

```

Meskipun ketika menjalankan **client.get()** kita memasukkan handler **AsyncHttpResponseHandler**, namun prosesnya tetap berjalan secara *synchronous*. Hal ini karena kita menjalankan **setUseSynchronousMode** dengan nilai **true** seperti terlihat pada kode di bawah ini:

```

1. public void onStart() {
2.     super.onStart();
3.     setUseSynchronousMode(true);
4. }

```

Dengan menggunakan **setUseSynchronousMode(true)**, kita dapat memanfaatkan **handlerloopj** yang semula **async** menjadi **synchronous** sesuai dengan bagaimana **client** diinisiasi.

Kenapa *synchronous*? Ketika kita meng-override metode **loadInBackground**, kita perlu mengembalikan sebuah obyek. Untuk mengembalikan sebuah obyek, kita perlu menunggu proses pengambilan data hingga selesai. Sehingga obyek yang telah diterima dapat dikembalikan.

Perlu diingat bahwa metode **loadInBackground** akan berjalan di thread background. Jadi kita tidak perlu khawatir dengan proses *synchronous* di atas. Performa aplikasi tidak akan menurun karenanya.