Pemrograman Aplikasi Mobile (ST088)

AsyncTask

ARIF DWI LAKSITO, M. KOM

S1 INFORMATIKA
UNIVERSITAS AMIKOM YOGYAKARTA

Thread

- Secara umum semua komponen aplikasi di Android berjalan pada proses dan *thread* yang sama. Ini disebut *main thread* atau *ui thread*.
- Secara *default* ini berarti sistem Android tidak secara otomatis menciptakan *thread*lain untuk menjalankan proses secara spesifik.
- Dalam pengembangan aplikasi, terkadang kita membutuhkan proses komputasi yang intensif. Misalnya proses memanipulasi *bitmap* dan proses menghubungi *server*.
- Jika proses-proses tersebut dilakukan pada ui thread atau main thread, maka proses tersebut akan menghambat *rendering* tampilan aplikasi. Hal ini ditandai dengan komponen ui widget tidak bisa diklik. Mirip dengan kondisi *hang*.

ANR

- Sistem Android akan menghitung selama 5 detik apakah aplikasi yang sedang berjalan itu responsif atau tidak.
- Jika tidak, maka sistem Android akan menampilkan dialog Application Not Responding (ANR) Jelas ini akan memberi dampak yang negatif ke pengguna.
- Kesalahan ini dapat berujung ke pengguna mencopot aplikasi kita (*uninstall*).
- Oleh karena itu kita membutuhkan *thread* lain. *Thread* lain ini bernama *worker thread* atau *async task*. Ia akan berjalan pada *thread* yang terpisah dengan *main thread*. Alhasil, aplikasi tetap terasa responsif.

AsyncTask

- Asynctask adalah komponen pada Android yang memiliki kemampuan untuk menjalankan proses secara *asynchronous*.
- Ia tetap bisa berkomunikasi dengan *ui thread* untuk mengirimkan hasil proses yang dilakukannya.
- Selain itu, asynctask sangat mudah dan sederhana untuk diterapkan.

Poin penting AsyncTask

1. Diperuntukan untuk proses *asynchronous* dan mampu berkomunikasi dengan *ui thread* untuk mengirimkan hasil proses.
2. Kelas Java yang dibuat harus inherit kepada AsyncTask.
3. Kasus umum yang biasa diterapkan adalah ketika mengunduh berkas dari Internet dan dengan menampilkan perkembangan pengunduhan di layar.

Params, Progress, dan Result

```
private class DownloadFilesTask extends AsyncTask<URL, Integer, Long>
```

Yang perlu diperhatikan adalah kode <URL, Integer, Long>.

3 *argument* di atas menunjukkan tipe data apa yang digunakan di dalam asynctask. Argument pertama adalah params, kedua adalah progress, dan ketika adalah result.

Params : Tipe parameter yang akan menjadi inputan untuk proses *asynchronous*.

Progress : Tipe satuan unit untuk memberi kabar perkembangan ke *ui thread*.

Result : Tipe hasil dari proses *asynchronous* yang dijalankan.

Metode Utama AsyncTask

1. **onPreExecute()**
   Metode ini akan dijalankan pertama kali sebelum proses *asynchronous* dilakukan. Metode ini masih berada pada *ui thread*. Pada umumnya, ia digunakan untuk menampilkan komponen ui seperti progress bar.

2. **doInBackground()**
   Metode ini akan dijalankan setelah onPreExecute(). Disinilah proses *asynchronous* terjadi. Hasil perkembangan proses asynchronous dikirim melalui metode publishProgress().

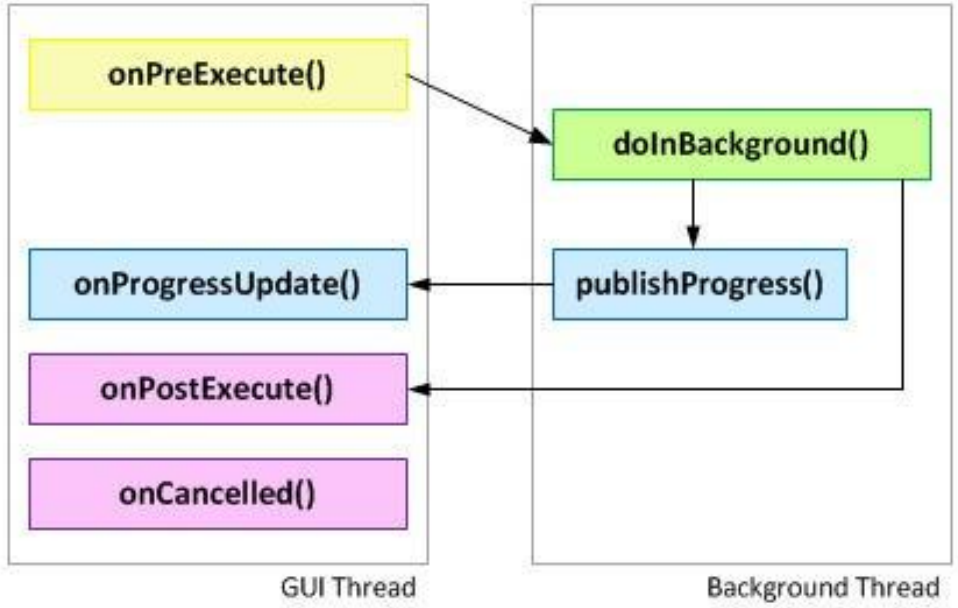
3. **onProgressUpdate()**
   Metode ini yang akan menerima input dari apa yang dilakukan oleh metode publishProgress. Proses umum yang terjadi adalah memperbarui persentasi dari tampilan progress bar.

4. **onPostExecute()**
   Setelah proses di doInBackground() selesai, maka hasilnya akan dikirimkan ke metode onPostExecute(). Kemudian prosesnya akan dikembalikan lagi ke *ui thread*. Disinilah proses penampilan atau proses lain yang menunjukkan bahwa AsyncTask telah selesai dijalankan.

Running AsyncTask

Untuk menjalankan AsyncTask, caranya adalah dengan menuliskan kode baris :

```
new DownloadFilesTask().execute(url1,url2,url3);
```



Exercise 5

AsyncTask Sederhana

AsyncTask Sederhana

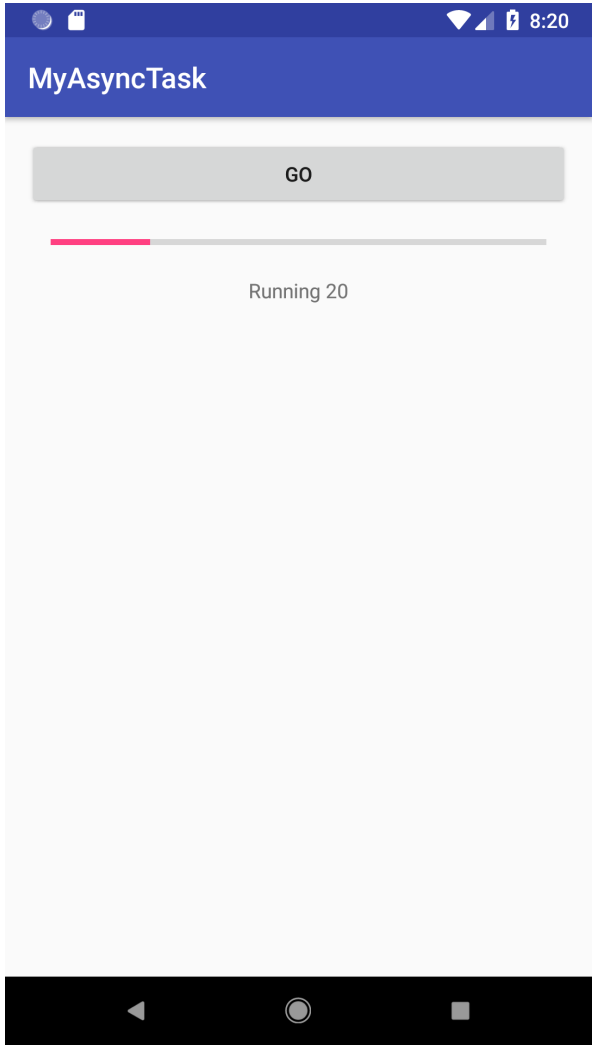
1. Buat Project baru dengan nama MyAsyncTask, pilih EmptyActivity.
2. Tambahkan 3 komponen View yaitu: Button, ProgressBar dan textView
3. Buat Class yang inherit dari AsyncTask

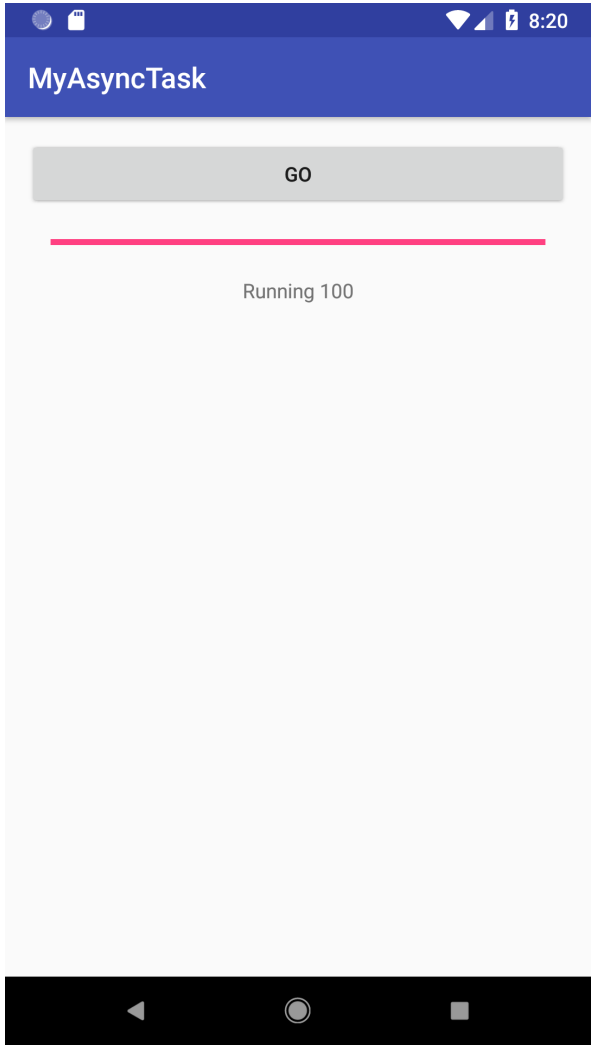
```
private class DemoAsync extends AsyncTask<Integer, Integer, String>
```

4. Lakukan perulangan yang berisi Thread pada method doInBackground( )

```
int count = params[0];
for (int i = 0; i < count; i++) {
    Thread.sleep(1500);
    publishProgress((int) (((i + 1) / (float) count) * 100));
}
```


8:20
MyAsyncTask
GO
TextView


8:20
MyAsyncTask
GO
Running 20


8:20
MyAsyncTask
GO
Running 100

Referensi

1. Kelas menjadi Android Developer Expert, Dikoding Indonesia, https://www.dicoding.com/academies/14/tutorials/159
2. Android AsyncTask Example with ProgressBar, https://www.concretepage.com/android/android-asynctask-example-with-progress-bar