Materi VIII

Nama : Roy Agus Martin Marbun

Kelas : XII – RPL

Mata Pelajaran : Pemrograman Berorientasi Objek

BAB II

Operasi *File* dan *Input Output* (IO)

A. Dasar-Dasar *Thread*

Sebuah *thread* merupakan sebuah pengontrol aliran program. *Thread* adalah suatu bagian program yang tidak tergantung pada bagian lain dan dapat dijalankan secara bersama-sama. Hal ini berarti suatu *thread* dapat diberhentikan atau diistirahatkan tanpa harus menghentikan yang lainnya. Pada Java setiap *thread* dikontrol oleh suatu objek unik turunan *Thread*, yang didefinisikan di dalam paket *java.lang*.

B. Pengertian *Thread*

Thread adalah rangkaian eksekusi dari sebuah aplikasi *java* dan setiap program *java* minimal satu buah *thread*. Sebuah *thread* bisa berada di salah satu dari 4 status, yaitu *New*, *Runnabel*, *Blocked* dan *Dead*.

a. *New*

Thread yang berada di status ini adalah objek dari kelas *Thread* yang baru dibuat, yaitu saat instansiasi objek dengan statement *new*. Saat thread berada di status *new*, belum ada sumber daya yang dialokasikan, sehingga *thread* belum bisa menjalankan perintah apapun.

b. *Runnable*

Agar *thread* bisa menjalankan tugasnya, method *start()* dari kelas *Thread* harus dipanggil. Ada dua hal yang terjadi saat pemnggilan method *start()*, yaitu alokasi memori untuk thread yang dibuat dan pemanggilan method *run()*. Saat method *run()* dipanggil, status thread berubah menjadi *runnable*, artinya *thread* tersebut sudah memenuhi syarat untuk dijalankan oleh *JVM*. *Thread* yang sedang berjalan juga berada di status *runnable*.

c. *Blocked*

Sebuah *thread* dikatakan berstatus *blocked* atau terhalang jika terjadi *blocking statement*, misalnya pemanggilan method *sleep()*. Method *sleep()* adalah method yang menerima argumen bertipe *integer* dalam bentuk milisekon. Argumen tersebut menunjukkan seberapa lama *thread* akan "tidak aktif". Selain *sleep()*, dulunya dikenal method suspend(), tetapi sudah disarankan untuk tidak digunakan lagi karena mengakibatkan terjdinya deadlock. *Thread* akan menjadi *runnable* kembali jika interval method *sleep()* berakhir atau pemanggilan method *resume()* jika untuk menghalangi thread tadi digunakan method *suspend()*.

d. *Dead*

Sebuah *thread* berada di status *dead* bila telah keluar dari method *run()*. Hal ini bisa terjadi karena *thread* tersebut memang telah menyelesaikan pekerjaannya di method *run()*, maupun karena adanya pembatalan thread. Status jelas dari sebuah *thread* tidak dapat diketahui, tetapi method *isAlive()* mengembalikan nilai *Boolean* untuk mengetahui apakah thread tersebut *dead* atau tidak.

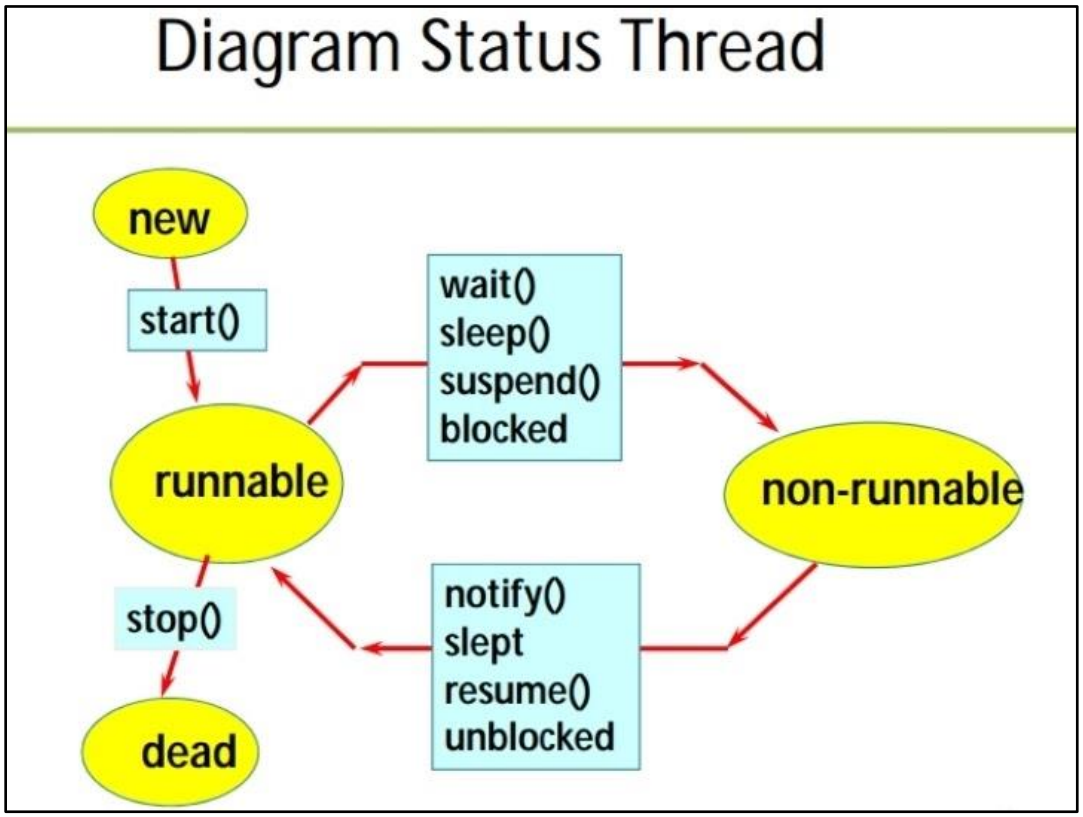

Diagram Status *Thread*

C. Pembentukan *Thread dalam* Java

Untuk membuat *thread* dalam *Java* terdapat 2 cara yaitu:

a. *Extends class Thread*

Untuk menjalankan *thread*, dapat dilakukan dengan memanggil method *start()*. Saat *start()* dijalankan, maka sebenarnya method *run()* dari class akan dijalankan. Jadi untuk membuat *thread*, harus mendefinisikan method *run()* pada definisi *class*.
Berikut adalah konstruktor dari cara ini.

```
SubThread namaObject = new SubThread();
namaObject.start();
```

b. Implements interface *Runnable*

Cara ini merupakan cara yang paling sederhana dalam membuat *thread*. *Runnable* merupakan unit abstrak, yaitu kelas yang mengimplementasikan *interface* ini hanya

cukup mengimplementasikan fungsi *run()*. Dalam mengimplementasikan fungsi *run()*, haruslah mendefinisikan instruksi yang membangun sebuah thread.
Berikut adalah konstruktor dari cara ini.

```
MyThread myObject = new MyThread();
Thread namaObject = new Thread(myObject);
```

Atau dengan cara seperti ini.

```
New Thread(new ObjekRunnable());
```