2020

DASAR-DASAR THREAD

Pande Made Mahendri Pramadewi

SMK Negeri 1 Denpasar

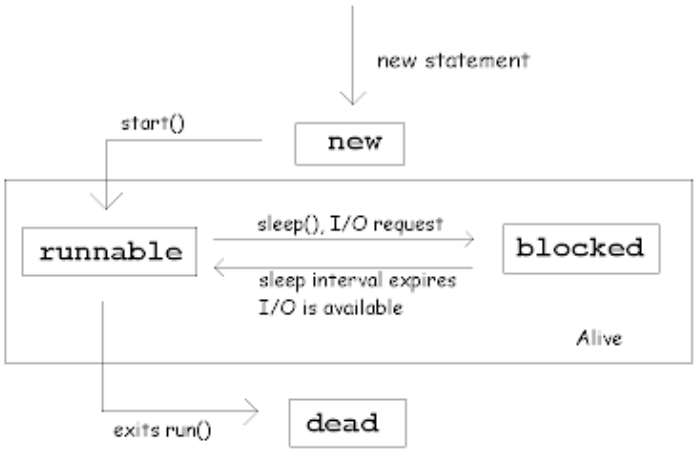
A. THREAD DALAM JAVA

Thread merupakan potongan program dengan ukuran kecil dalam suatu proses yang dapat dijadwalkan oleh sistem operasi. Setiap program java minimal memiliki satu buah thread yang dikenal dengan *main thread*, yang dibuat oleh *Java Virtual Machine* (JVM) ketika *main method* pertama kali dipanggil.

Dengan menggukan thread, kita daapat mengeksekusi proses dalam sebuah program tertentu secara paralel. Eksekusi dilakukan dengan membagi proses atau operasi tertentu ke dalam beberapa thread.

B. THREAD LIFE CYCLE/ SIKLUS HIDUP THREAD

Sebuah thread bisa berada di salah satu dari 4 status, yaitu new runnable, blocked dan dead.



- 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.
- Runnable, saat method run() dipanggil melalui method start(), status thread berubah menjadi runnable, artinya thread tersebut sudah memenuhi syarat untuk dijalankan oleh JVM. Thread yang sedang berjalan juga dikategorikan dalam status runnable.
- Blocked, sebuah thread dikatakan berstatus blocked jika terjadi blocking statement, misalnya pemanggilan method sleep(). Sleep adalah suatu method yang menerima argument bertipe integer dalam bentuk milisekon. Argument tersebut menunjukkan seberapa lama thread akan tidur.

- 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.

C. PEMBUATAN THREAD DALAM JAVA

Dua cara pembuatan thread dalam Java, yaitu:

1. Melalui 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. Contoh :

```
1  package thread;
2
3  class Count extends Thread{
4      public void run() {
5          try{
6              System.out.println("Thread " + Thread.currentThread().getId() +" Sedang Berjalan");
7          } catch(Exception e) {
8              System.out.println("Proses Terganggu");
9          }
10     }
11 }
12
13 public class Exthread {
14
15     public static void main(String[] args) {
16         for(int i = 1; i <= 10; i++) {
17             Count satu = new Count();
18             satu.start();
19         }
20
21     }
22
23 }
24
```

Output :

```
Console
<terminated> Exthread [Java Applicati
Thread 11 Sedang Berjalan
Thread 15 Sedang Berjalan
Thread 14 Sedang Berjalan
Thread 13 Sedang Berjalan
Thread 12 Sedang Berjalan
Thread 10 Sedang Berjalan
Thread 16 Sedang Berjalan
Thread 19 Sedang Berjalan
Thread 18 Sedang Berjalan
Thread 17 Sedang Berjalan
```

2. Melalui Implements Interface Runnable

Cara ini merupakan cara yang paling sederhana dalam membuat thread. Runnable merupakan unit abstrak, kelas yang mengimplementasikan interface ini hanya cukup mengimplementasikan fungsi run(). Dalam mengimplementasi fungsi run(), kita akan mendefinisikan instruksi yang membangun sebuah thread. Contoh:

```
1  package thread;
2
3  class Count implements Runnable{
4      public void run() {
5          try{
6              System.out.println("Thread " + Thread.currentThread().getId() +" Sedang Berjalan");
7          } catch(Exception e) {
8              System.out.println("Proses Terganggu");
9          }
10     }
11 }
12
13 public class Exthread {
14
15     public static void main(String[] args) {
16         for(int i = 1; i <= 10; i++) {
17 //          --Extends Thread--
18 //          Count satu = new Count();
19 //          satu.start();
20 //          --Runnable Interface--
21 //          Runnable dua = new Count();
22             Thread jalan = new Thread(new Count());
23             jalan.start();
24         }
25
26     }
27
28 }
29
```

Output :

```
Console
<terminated> Exthread [Java Application]
Thread 10 Sedang Berjalan
Thread 12 Sedang Berjalan
Thread 11 Sedang Berjalan
Thread 13 Sedang Berjalan
Thread 15 Sedang Berjalan
Thread 14 Sedang Berjalan
Thread 16 Sedang Berjalan
Thread 17 Sedang Berjalan
Thread 18 Sedang Berjalan
Thread 19 Sedang Berjalan
```

Perhatikan perbedaan cara pembentukan objek dari class Count dalam kedua acara tersebut.

Melalui Extends Class Thread yaitu Count satu = new Count();

Melalui Implements Interface Runnable yaitu Thread jalan = new Thread (new Count());

D. PRIORITAS THREAD

Dalam Java kita dapat membuat prioritas suatu thread relatif terhadap thread yang lain. Sehingga thread yang mempunyai prioritas lebih tinggi mempunyai kesempatan lebih besar untuk mengakses suatu sources. JVM memilih thread yang runnable dengan prioritas tertinggi. Semua thread java mempunyai prioritas dari 1 sampai 10. Prioritas tertinggi 10 dan berakhir dengan 1 sebagai prioritas terendah. Sedangkan prioritas normal adalah 5.

Thread.MIN_PRIORITY = thread dengan prioritas terendah.

Thread.MAX_PRIORITY = thread dengan prioritas tertinggi.

Thread.NORM_PRIORITY = thread dengan prioritas normal.

Saat thread baru dibuat ia mempunyai prioritas yang sama dengan thread yang menciptakannya. Prioritas thread dapat diubah dengan menggunakan setpriority() method.

E. MULTITHREADING

Multithreading va adalah proses mengeksekusi dua atau lebih thread secara bersamaan untuk pemanfaat CPU secara maksimal.

Contoh :

```
1  package thread;
2
3  class Hello extends Thread{
4      public void run(){
5          for (int i = 1; i <= 10; i++) {
6              System.out.println("Hello");
7              try { Thread.sleep(500);} catch(Exception e){}
8          }
9      }
10 }
11 class World extends Thread{
12     public void run(){
13         for (int i = 1; i <= 10; i++) {
14             System.out.println("World");
15             try { Thread.sleep(500);} catch(Exception e){}
16         }
17     }
18 }
```

```
20 public class multithread {
21
22     public static void main(String[] args) {
23          Hello satu = new Hello();
24             World dua = new World();
25
26             satu.start();
27             try { Thread.sleep(10);} catch(Exception e){}
28             dua.start();
29
30         }
31
32     }
33
```

Output:

```
Console
<terminated> multithread [Java Application]
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
Hello
World
```