PERTEMUAN KE – 13
Thread

A. TUJUAN

Menggunakan kelas Threat untuk menangani konkurensi

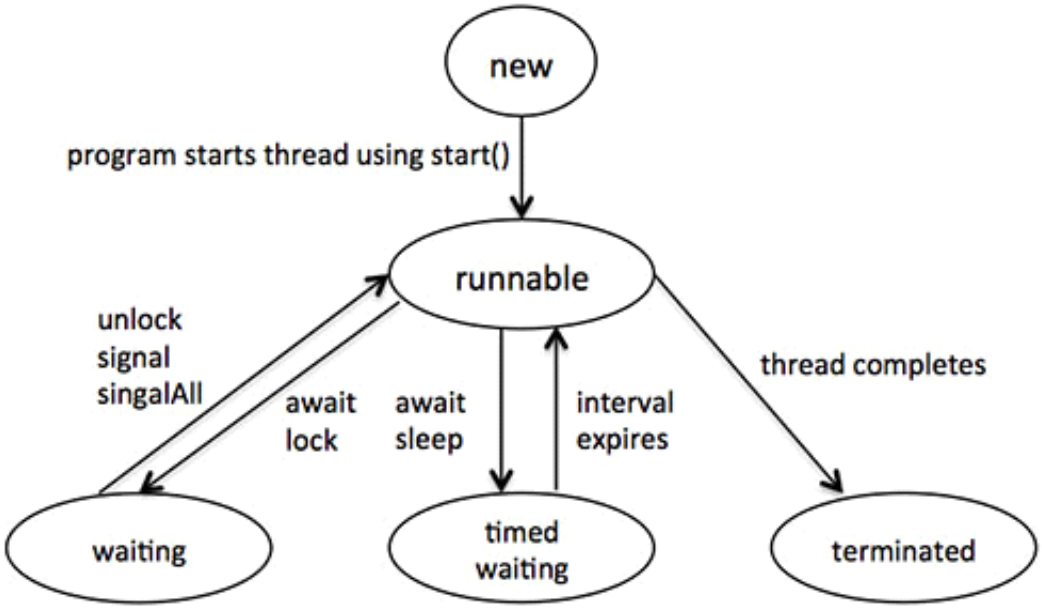
B. TEORI SINGKAT

Java adalah bahasa pemrograman multithreaded yang berarti kita dapat mengembangkan program yang multithreaded menggunakan Java. Sebuah program multithreaded mengandung dua atau lebih bagian yang dapat runnable secara bersamaan dan setiap bagian dapat menangani tugas yang berbeda pada saat yang sama memanfaatkan secara optimal sumber daya yang tersedia khususnya bila komputer Anda memiliki beberapa CPU.

Menurut definisi multitasking adalah ketika beberapa proses berbagi sumber daya umum pengolahan seperti CPU. Multithreading memperluas gagasan multitasking ke aplikasi di mana Anda dapat membagi operasi tertentu dalam satu aplikasi menjadi thread individu. Masing-masing thread dapat runnable secara paralel. OS membagi waktu proses tidak hanya di antara aplikasi yang berbeda, tetapi juga di antara setiap thread dalam aplikasi.

Multithreading memungkinkan untuk menulis dengan cara di mana beberapa kegiatan dapat dilanjutkan bersamaan dalam program yang sama.

Sebuah thread runnable melalui berbagai tahap dalam siklus hidupnya. Sebagai contoh, thread new, start, runnable, dan kemudian terminated. Setelah diagram menunjukkan siklus hidup lengkap thread.



New: Sebuah thread baru start siklus hidupnya di negara baru. Masih dalam keadaan ini sampai program dijalankan thread. Hal ini juga disebut sebagai thread new.

Runnable: Setelah thread baru new start, thread menjadi runnable. Sebuah thread di negara bagian ini dianggap melaksanakan tugasnya.

Waiting: Kadang-kadang, thread transisi ke negara menunggu sementara thread menunggu thread lain untuk melakukan task.A thread transisi kembali ke keadaan runnable hanya ketika thread lain sinyal thread menunggu untuk melanjutkan mengeksekusi.

Timed waiting: Sebuah thread runnable dapat memasuki state menunggu waktunya untuk interval waktu tertentu. Sebuah thread di state bagian ini transisi kembali ke keadaan runnable ketika interval waktu berakhir atau ketika acara itu sedang menunggu terjadi.

Terminated: Sebuah thread runnable memasuki keadaan dihentikan ketika selesai tugasnya atau berakhir.

C. PRAKTIK

Praktik 1: membuat Thread

```
class RunnableDemo implements Runnable {
   private Thread t;
   private String threadName;

   RunnableDemo( String name){
       threadName = name;
       System.out.println("Creating " +  threadName );
   }
   public void run() {
```

```
      System.out.println("Running " +  threadName );
      try {
         for(int i = 4; i > 0; i--) {
            System.out.println("Thread: " + threadName + ", " + i);
            // Let the thread sleep for a while.
            Thread.sleep(50);
         }
     } catch (InterruptedException e) {
         System.out.println("Thread " +  threadName + " interrupted.");
     }
     System.out.println("Thread " +  threadName + " exiting.");
   }

   public void start ()
   {
      System.out.println("Starting " +  threadName );
      if (t == null)
      {
         t = new Thread (this, threadName);
         t.start ();
      }
   }

}
```

Praktik 2. Menggunakan kelas RunnableDemo

```
public class TestThread {
   public static void main(String args[]) {

      RunnableDemo R1 = new RunnableDemo( "Thread-1");
      R1.start();

      RunnableDemo R2 = new RunnableDemo( "Thread-2");
      R2.start();
   }
}
```

Praktik 3. Membuat Thread menggunakan extend Thread

```
class ThreadDemo extends Thread {
   private Thread t;
   private String threadName;

   ThreadDemo( String name){
       threadName = name;
       System.out.println("Creating " +  threadName );
   }
   public void run() {
      System.out.println("Running " +  threadName );
      try {
         for(int i = 4; i > 0; i--) {
            System.out.println("Thread: " + threadName + ", " + i);
            // Let the thread sleep for a while.
            Thread.sleep(50);
```

```
            }
        } catch (InterruptedException e) {
            System.out.println("Thread " +  threadName + " interrupted.");
        }
        System.out.println("Thread " +  threadName + " exiting.");
    }

    public void start ()
    {
        System.out.println("Starting " +  threadName );
        if (t == null)
        {
            t = new Thread (this, threadName);
            t.start ();
        }
    }

}
```

Praktik 4. Menggunakan Thread

```
public class TestThread {
   public static void main(String args[]) {

      ThreadDemo T1 = new ThreadDemo( "Thread-1");
      T1.start();

      ThreadDemo T2 = new ThreadDemo( "Thread-2");
      T2.start();
   }
}
```

D. LATIHAN

- *Latihan diberikan oleh dosen pengampu pada saat praktikum.*
- *Dikerjakan di laboratorium pada jam praktikum.*

E. TUGAS

- *Tugas diberikan oleh dosen pengampu pada akhir praktikum.*
- *Dikerjakan di rumah dan dilampirkan pada laporan.*