MODUL 4
“INSTRUKSI-INSTRUKSI THREAD”

Oleh

SAFIRA MAYA SHOVIE, S.Pd

“Instruksi-Instruksi Thread”

A. TUJUAN PEMBELAJARAN

1. Memahami konsep dasar instruksi *thread* dalam pemrograman aplikasi berorientasi obyek.
2. Merancang, membuat, dan menguji program aplikasi berorientasi obyek dengan penerapan instruksi *thread*.

B. DASAR TEORI

Thread merupakan kemampuan yang disediakan oleh Java untuk membuat aplikasi yang tangguh, karena thread dalam program memiliki fungsi dan tugas tersendiri. Dengan adanya thread, dapat membuat program yang lebih efisien dalam hal kecepatan maupun penggunaan sumber daya, karena kita dapat membagi proses dalam aplikasi kita pada waktu yang sama. Thread umumnya digunakan untuk pemrograman multitasking, networking, yang melibatkan pengaksesan ke sumber daya secara konkuren.

Ada dua cara yang bisa digunakan dalam membuat sebuah thread, yaitu :

- Membuat subclass dari 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. Konstruktor dari cara ini adalah :

```
ClassThread namavar = new ClassThread();
Namavar.start();
```

  Atau dapat juga langsung dengan cara:

```
New ClassThread().start();
```

- Mengimplementasikan 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 mengimplementasi fungsi run(), kita akan mendefinisikan instruksi yang membangun sebuah thread. Konstruktor dari cara ini adalah :

```
ObjekRunnable objek = new ObjekRunnable();
Thread namavar = new Thread(Objek Runnable);
```

  Atau dengan cara singkat seperti :

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

1. **Daemon Dan User Thread**

   Ada dua Macam thread dalam Java, yaitu ***daemon*** dan ***user*** thread. ***Daemon*** thread merupakan thread yang siklus hidupnya tergantung pada thread utama atau induk, sehingga apabila thread induk berakhir, maka otomatis thread-thread daemon juga ikut berakhir. Sedangkan ***user*** thread memiliki sifat berbeda, dimana apabila thread utama sudah selesai, maka user thread akan terus dijalankan.

2. **Sleep**

   Mengatur thread untuk menghentikan prosesnya sejenak dan memberi kesempatan pada thread atau proses lain. Sleep dilakukan dengan cara memanggil method :

   `Sleep(long waktu);`

   Waktu untuk method ini merupakan tipe long dalam milisekon.

3. **Interrupt**

   Apabila menginginkan suatu thread untuk menghentikan proses, maka perlu memanggil method interrupt. Interrupt digunakan untuk memberi signal pada thread untuk menghentikan prosesnya.

C. LATIHAN

Latihan 1

```
public class latihan_thread {

    public static void main(String[] args) {
        int jumlah = 10;
        Thread thread = new Thread(){
            public void run(){
                try{
                    for(int w=1; w<=jumlah; w++){
                        System.out.println("Nomor: "+w);
                        sleep(1000); //Waktu Pending
                    }
                }catch(InterruptedException ex){
                    ex.printStackTrace();
                }
            }
        };
        thread.start();
    }
}
```

Jika kita jalankan program tersebut akan melooping dan mengeluarkan output sebanyak 10 kali, program akan mengeluarkan output 1 kali per detiknya, itulah yang membedakan program yang menggunakan thread dan yang tidak, jika kita tidak menggunakan thread atau hanya for loop saja, maka program akan mengeluarkan semua output tersebut sekaligus, **sleep(1000)** artinya kita mengset waktu pending selama 1000 millisecond/1detik, *try* dan *catch*

digunakan untuk mengkap jenis error ***InterruptedException*** agar program tidak crash saat error itu terjadi.

InterruptedException terjadi karena thread berhenti sementara, jadi kalian bisa melakukan apa saja jika thread berhenti sementara. Selanjutnya untuk contoh kedua, kita akan membuat program multitasking sederhana menggunakan thread, pada program tesebut, kita akan membuat 2 buah method, kedua method tersebut mempunyai output yang berbeda, saat program dijalankan, output tersebut akan keluar secara bersamaan setiap 1 detik sekali.

Latihan 2

```
import static java.lang.Thread.sleep;

public class latihan_thread{

    Thread thread;
    int jumlah = 7;

    public static void main(String[] args) {
        latihan_thread test = new latihan_thread();
        test.proses_satu();
        test.proses_dua();
    }

    void proses_satu(){
        thread = new Thread(){
           public void run(){
               try{
                   for(int w=1; w<=jumlah; w++){
                       System.out.println("Nomor: "+w);
                       sleep(1000); //Waktu Pending
                   }
               }catch(InterruptedException ex){
                   ex.printStackTrace();
               }
           }
        };
        thread.start();
    }
```

```
    void proses_dua(){
            thread = new Thread(){
            public void run(){
                try{
                    for(int w=1; w<=jumlah; w++){
                        System.out.println("Salam Programmer");
                        sleep(1000); //Waktu Pending
                    }
                }catch(InterruptedException ex){
                    ex.printStackTrace();
                }
            }
        };
        thread.start();
    }
}
```

Latihan 3

Runnable digunakan agar thread bisa menjalankan tugasnya, cara paling sederhana untuk membuat thread adalah dengan mengimplementasikan interface Runnable pada class java kita, jadi kalian cukup mengimplementasiakan fungsi dari method *run*.

```
import static java.lang.Thread.sleep;

public class latihan_thread implements Runnable{

    int jumlah = 10;

    public static void main(String[] args) {
        latihan_thread test = new latihan_thread();
        test.run();
    }

    @Override
    public void run() {
       try{
          for(int w=1; w<=jumlah; w++){
            System.out.println("ID: "+w);
            sleep(1000); //Waktu Pending
          }
       }catch(InterruptedException ex){
              ex.printStackTrace();
       }
    }
}
```

Jalankan program tersebut, hasilnya akan sama dengan program sebelumnya, kedua cara tersebut sama saja fungsinya, tergantung kebutuhan.

Latihan 4

```
package coba.thread;
public class CobaThread {
    public static void main (String [] args) {
    Baju baju1 = new Baju("Baju-1");
    Baju baju2 = new Baju("Baju-2");
    baju1.start();
    baju2.start();
    }
}
    class Baju extends Thread {
        // konstruktor
        public Baju (String id) {
            super (id);
        }
        // Mendefinisikan sendiri run()
    @Override
    public void run() {
        String nama = getName();
        for (int i=0; i<5; i++) {
            try {
                sleep(1000); // Tunggu 1 detik
}
            catch(InterruptedException ie) {
                System.out.println("Terinterupsi");
            }
            System.out.println("Thread " + nama + ":Posisi" + i );
        }
}}
```

Latihan 5

```
package coba.thread;
public class CobaThreadRunnable {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Thread celana1 = new Thread(new Celana("Celana-1"));
        Thread celana2 = new Thread(new Celana("Celana-2"));
        celana1.start();
        celana2.start();
    }
}
```

```
class Celana implements Runnable {

    String nama;
// konstruktor
    public Celana(String id) {
        nama = id;
    }

    public void run() {
        for (int i = 0; i < 5; i++) {
            try {
                Thread.currentThread().sleep(1000);
            } catch (InterruptedException ie) {
                System.out.println("Terinterupsi");
            }
            System.out.println("Thread " + nama + ":Posisi" + i);
        }
    }}
```

Latihan 6

```
package latihan_coba;
public class Latihan_COBA {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        int i;
        int a;

        Thread kopasus = new Thread();
        kopasus.start();
        while(true){

            for(i=1; i<=20; i++){
                System.out.println("Pasukkan Kopasus Berhasil Menyelamatkan Ibu Hamil ke : " + i);

                try{
                    kopasus.sleep(1000);
                }catch(Exception e){
                    e.getStackTrace();
                }
              if(i==20){
                  System.out.println("Pasukkan Marinir Datang Ke TKP ");

              }
    }
```

```
        Thread marinir = new Thread();
        marinir.start();

            for(a=1; a<=50; a++){
                System.out.println("Pasukkan Marinir Berhasil Membunuh Teroris Ke : " + a);

                if(a<=20){
                    System.out.println("Pasukkan Kopasus Berhasil Menyelamatkan Warga Manula Ke : " +a);
                }

                try{
                    marinir.sleep(1000);
                }catch(Exception e){
                    e.getStackTrace();

                }
                if(a==50){
                        System.out.println("Pasukkan Infanteri Datang ke TKP ");
                    }
            }
            break;
    }

    }
}
```

Latihan 7

Silahkan membuat program thread dengan menggunakan perintah *Extend* dan *Runnable* serta berikan penjelasan!

Catatan :

1. Silahkan mengerjakan semua latihan yang terdapat diatas, bisa didiskusikan dengan teman kalian. Tetapi untuk pengerjaan tugasnya tetap individu.
2. Materi diatas hanya sedikit rangkuman dari berbagai fungsi kode program yang sudah saya buat. Silahkan untuk pemantaban pemahaman lebih dalam, kalian cari referensi contoh program yang lain dengan menggunakan beberapa fungsi dari kode program diatas.
3. Deadline pengumpulan tugas, 3 hari setelah materi dibagikan. Jika telat dalam pengumpulan, tugas tidak diterima, terkecuali bila ada kendala, misal : laptop rusak dsb, maka setelah laptop kembali normal tugas bisa menyusul.
4. Tugas dikumpulkan melalui link berikut : https://drive.google.com/drive/folders/1bew2ImhiLp-25UdyhYdMPLb0XRDr-gO3?usp=sharing dengan catatan yang dikumpulkan adalah project pada netbeans yang sudah dikerjakan, dan laporan yang berisi screenshoot syntax program dan output program serta penjelasan program.
5. Jangan lupa format folder tugas dan laporan yaitu **No.Absen_Nama Lengkap_Kelas.**

SELAMAT MENGERJAKAN ^_^