

Mobile Computing

Penjadwalan dan Timer

Ramos Somya

Timer dan TimerTask

- Timer dan TimerTasks berfungsi agar Anda bisa melakukan penjadwalan tugas pada suatu waktu. Tugas dapat juga dijadwalkan untuk diulang-ulang sampai interval tertentu.
- Anda dapat membuat tugas dengan menurunkan (extending) TimerTask dan mengimplement method run(). Method run() akan dieksekusi berdasarkan jadwal yang ada pada Timer.



```
class CounterTask extends TimerTask {  
    int counter = 0;  
    public void run() {  
        System.out.println("Counter: " + counter++);  
    }  
}
```



- Untuk menjadwalkan sebuah tugas, buat sebuah Timer dan gunakan method `schedule()` yang ada pada Timer untuk menjadwalkan jalannya tugas.
- Setiap Timer berjalan pada bagian yang terpisah. Method `schedule()` memiliki beberapa bentuk.
- Anda dapat mengatur waktu tugas untuk mulai dengan memberikan delay dalam miliseconds atau dengan memberikan waktu absolut (`java.util.Date`).



```
Timer timer = new Timer();  
TimerTask task = new CounterTask();  
timer.schedule(task, 8000, 1000);
```

- task akan dimulai dalam 8 detik dan diulangi setiap 1 detik



- Anda dapat menghentikan timer dengan menggunakan method `close()`. Method ini dapat menghentikan timer dan mengabaikan tugas yang dijadwalkan.
- Perlu Anda catat, bahwa ketika Timer dihentikan, maka tidak dapat diulangi (direstart) kembali.



| | |
|------|---|
| void | <code>schedule(TimerTask task, Long delay)</code> Melakukan penjadwalan tugas untuk dieksekusi sesudah menentukan delay yang diinginkan (dalam milliseconds) |
| void | <code>schedule(TimerTask task, Long delay, long period)</code> Melakukan penjadwalan tugas untuk dieksekusi berulang-ulang, dimulai sesudah delay yang ditentukan (dalam milliseconds) |
| void | <code>schedule(TimerTask task, Date time)</code> Melakukan penjadwalan tugas agar dapat dieksekusi pada waktu yang ditentukan. |
| void | <code>schedule(TimerTask task, Date time, long period)</code> Melakukan penjadwalan tugas untuk dieksekusi berulang-ulang, dimulai pada waktu yang ditentukan. |
| void | <code>cancel()</code> Menghentikan timer, mengabaikan tugas yang dijadwalkan. |

TimerMidlet.java

```
public class TimerMidlet extends MIDlet implements CommandListener {  
  
    private Command cmdExit, cmdPause;  
    private Form form;  
    private StringItem siTimer;  
    private Display display;  
    private Timer timer = new Timer();  
    private TimerTask tTask = new CounterTask(this);  
    private boolean aktif;  
    private int angka = 0;
```



```
public TimerMidlet() {  
    cmdExit = new Command("Exit", Command.EXIT, 1);  
    cmdPause = new Command("Pause", Command.OK, 1);  
    siTimer = new StringItem("Counter", "");  
    timer.schedule(tTask, 0, 1000);  
    aktif = true;  
    form = new Form("Timer Test");  
    form.addCommand(cmdExit);  
    form.addCommand(cmdPause);  
    form.append(siTimer);  
}
```



```
public void startApp() {  
    display = Display.getDisplay(this);  
    form.setCommandListener(this);  
    display.setCurrent(form);  
}  
  
public void pauseApp() {  
}  
  
public void destroyApp(boolean unconditional) {  
    timer.cancel();  
}
```



```
public void commandAction(Command c, Displayable d) {  
    if (c == cmdExit) {  
        destroyApp(true);  
        notifyDestroyed();  
    } else if (c == cmdPause) {  
        if (aktif) {  
            timer.cancel();  
            aktif = false;  
        } else {  
            timer = new Timer();  
            tTask = new CounterTask(this);  
            timer.schedule(tTask, 0, 1000);  
            aktif = true;  
        }  
    }  
}
```

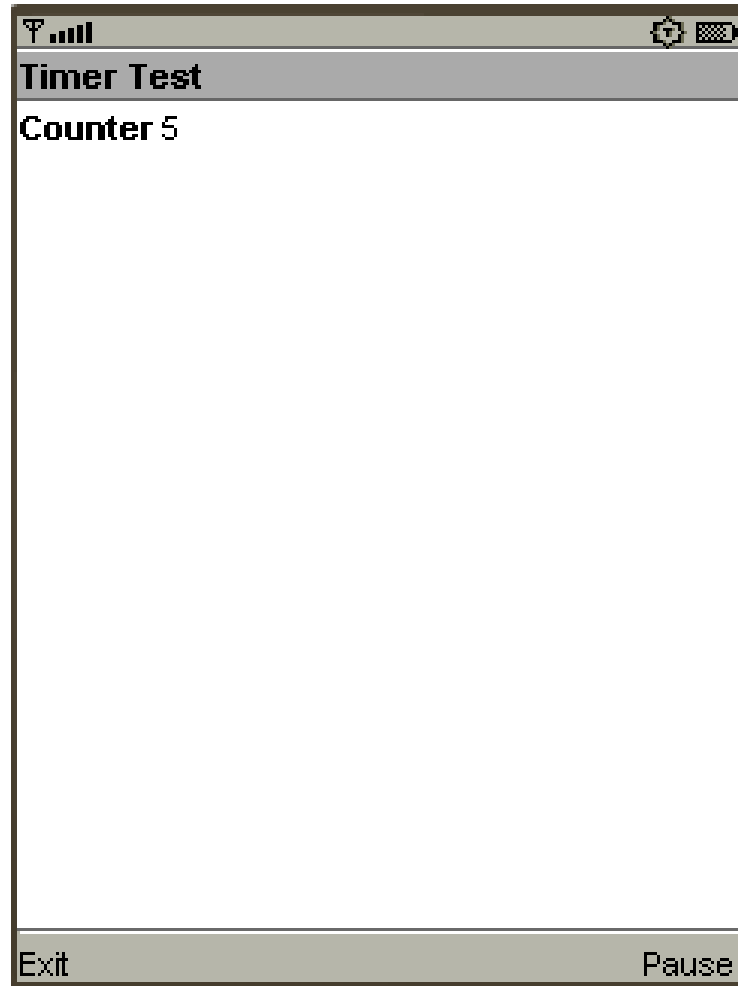


```
public void setText() {  
    siTimer.setText("" + this.angka);  
}  
  
public void tambah() {  
    this.angka++;  
}  
}
```



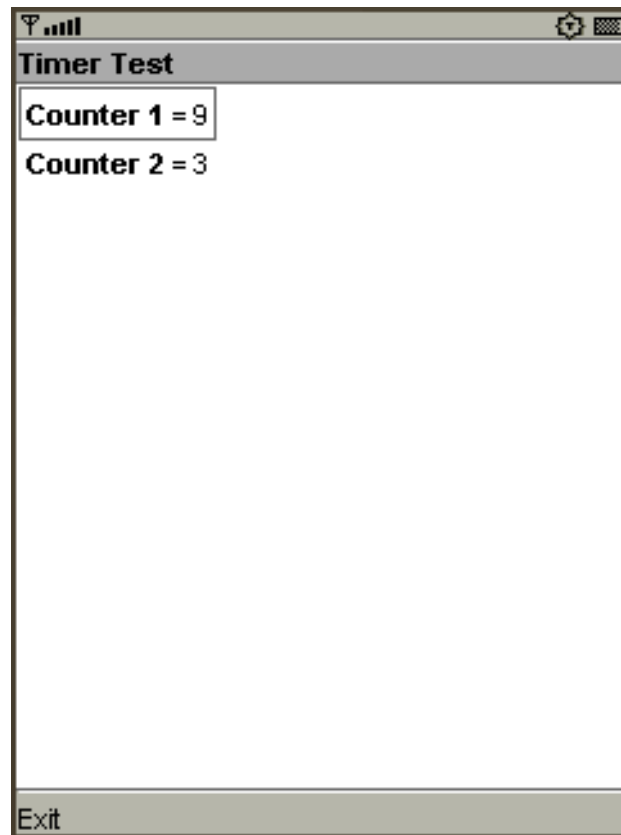
```
class CounterTask extends TimerTask {  
  
    TimerMidlet tmMidlet;  
  
    public CounterTask(TimerMidlet tmMidlet) {  
        this.tmMidlet = tmMidlet;  
    }  
  
    public void run() {  
        tmMidlet.tambah();  
        tmMidlet.setText();  
    }  
}
```

Hasil



Latihan

- Modifikasi contoh sebelumnya dan tambahkan 1 counter lagi dengan interval yang berbeda.



See You Next Week