

Concurrency: Java Threads

(lectures programs)

Sun OnLine Documentations

Thread Basics

→ **Example 1: SimpleThread:**
✓ *(Run Applet)*

```
public class SimpleThread extends Thread {  
  
    private int countDown = 50;  
    private static int threadCount = 0;  
    private int threadNumber = ++threadCount;  
  
    public SimpleThread () {  
        System.out.println("Making " + threadNumber);  
    }  
  
    public void run() {  
        while (true) {  
            System.out.println("Thread " + threadNumber  
+ "(" + countDown + ")");  
            if (--countDown == 0) return;  
        }  
    }  
  
    public static void main(String[] args) {  
        for(int i = 0; i < 5; i++)  
            new SimpleThread().start();  
        System.out.println("All Threads Started");  
    }  
}
```

- ✓ **Example 2: Counter4**
- ✓ **(Run Applet)**

```
public class Counter4 extends JApplet {

    private JButton startB = new JButton("Start");
    private Ticker[] s;

    class Ticker extends Thread {
        private JButton b = new JButton("Toggle");
        private JTextField t = new JTextField(10);
        private int count = 0;
        private boolean runFlag = true;

        public Ticker() {
            b.addActionListener(new ToggleL());
            JPanel p = new JPanel();
            p.add(t);
            p.add(b);
            getContentPane().add(p);
        }

        class ToggleL implements ActionListener {
            public void actionPerformed(ActionEvent e) {
                runFlag = !runFlag;
            }
        }

        public void run() {
            while (true) {
                if (runFlag) t.setText
(Integer.toString(count++));
                try {
                    sleep(100);
                } catch (InterruptedException e) {
                    System.err.println("Interrupted");
                }
            }
        }
    }

    class StartL implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            startB.setBackground(Color.red);
        }
    }
}
```

```

        startB.setEnabled(false);
        for (int i = 0; i < s.length; i++)
            s[i].start();
    }
}

public void init() {
    Container cp = getContentPane();
    cp.setLayout(new FlowLayout());

    s = new Ticker[4];

    for (int i = 0; i < s.length; i++)
        s[i] = new Ticker();

    startB.addActionListener(new StartL());
    cp.add(startB);
}

public static void main (String[] args) {
    Counter4 applet = new Counter4 ();
    Console.run (applet, 200, 200);
}
}

```

You can run this as an application:

```
% java Counter4
```

Blocking, Notify & Interrupt

→ **Example :** Notifying and Interrupting a blocked thread:

✓ **Run Applet**

```

public class SuspendInterrupt extends JApplet {

    private JTextField txt = new JTextField(10);
    private JButton
        suspendB = new JButton("Suspend"),
        resumeB = new JButton("Resume"),
        interruptB = new JButton("Interrupt");

    private int SuspendCount = 1;
    private SuspendingThread suspendableThread = new
    SuspendingThread();

    class SuspendingThread extends Thread {
        private int count = 0;
        private boolean suspendedFlag = false;

        public SuspendingThread() {
            start();
        }

        public synchronized void funSuspend() {
            suspendedFlag = true;
        }

        public synchronized void funResume() {
            suspendedFlag = false;
            notify();
        }

        public synchronized void funInterrupt() {
            interrupt();
            suspendableThread = null; // to release it
        }

        public void run() {
            while (true) {
                txt.setText("Count: " +
Integer.toString(count++));
                synchronized (this) {
                    if(suspendedFlag) {
                        txt.setText("Suspend # " +
SuspendCount++);
                        try{
                            wait();
                        }
                        catch(InterruptedException e) {
                            System.err.println("wait
Interrupted");
                        }
                    }
                }
            }
        }
    }
}

```



```

        suspendB.setBackground(Color.green);
        resumeB.setEnabled(false);
        resumeB.setBackground(Color.red);
    }
}
);
cp.add(resumeB);
resumeB.setEnabled(false);
resumeB.setBackground(Color.red);

interruptB.addActionListener( new
ActionListener() {
    public
        void actionPerformed(ActionEvent e) {
            suspendB.setEnabled(false);
            suspendB.setBackground(Color.red);
            resumeB.setEnabled(false);
            resumeB.setBackground(Color.red);
            interruptB.setEnabled(false);
            interruptB.setBackground(Color.red);
            txt.setText("Thread Interrupted");
            suspendableThread.funInterrupt();
        }
}
);
cp.add(interruptB);
interruptB.setEnabled(true);
interruptB.setBackground(Color.green);
}
public static void main(String[] args) {
    Console.run(new SuspendInterrupt(), 300, 100);
}
}

```
