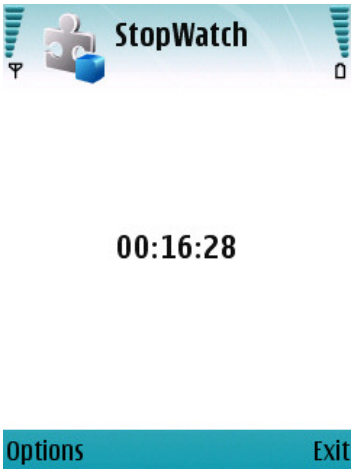
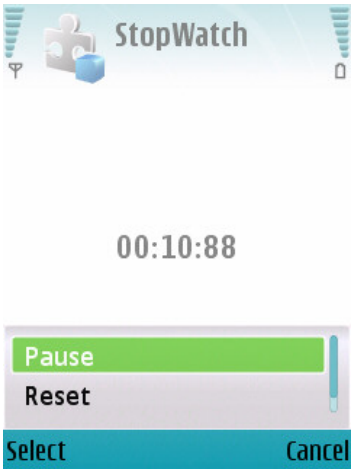
A stopwatch in Java ME

The article contains an example stopwatch application, which has the ability to pause/resume and reset.





StopWatch.java

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class StopWatch extends MIDlet {

    private Display display;
    private StopWatchCanvas canvas;

    public StopWatch() {
        display=Display.getDisplay(this);
        canvas=new StopWatchCanvas(this);
    }

    public void startApp() {
 display.setCurrent(canvas);
    }

    public void pauseApp() {

    }

    public void destroyApp(boolean unconditional) {

    }

}
```

StopWatchCanvas.java

```
import javax.microedition.lcdui.*;

public class StopWatchCanvas extends Canvas implements CommandListener {

    private Command exit;
    private Command start;
    private Command pause;
    private Command reset;
    private StopWatch midlet;
    private StopWatchThread thread;
    private final String startValue="00:00:00";
    private Image image=null;

    public StopWatchCanvas(StopWatch midlet){
 this.midlet=midlet;

        //Start the thread, stopwatch will remain paused
 thread=new StopWatchThread(this);
 thread.start();

 //Create the commands
 exit=new Command("Exit", Command.EXIT, 1);
 start=new Command("Start", Command.ITEM, 1);
 pause=new Command("Pause", Command.ITEM, 1);
 reset=new Command("Reset", Command.SCREEN, 2);

 addCommand(exit);
 addCommand(start);
 setCommandListener(this);
    }

    public void setImage(Image img) {
 this.image=img;
    }
```


(C) Copyright Nokia 2013. All rights reserved.

```
    protected void paint(Graphics g) {
        if(image!=null)
            //Draw text updated by the thread
            g.drawImage(image, getWidth()/2, getHeight()/2, Graphics.VCENTER|Graphics.HCENTER);
        else {
            //Display a white rectangle and black text
            g.setColor(255,255,255);
            g.fillRect(0, 0, this.getWidth(), this.getHeight());
            g.setColor(0, 0, 0);
            g.setFont(Font.getFont(Font.FACE_SYSTEM, Font.STYLE_BOLD, Font.SIZE_LARGE));
            g.drawString(startValue, this.getWidth()/2, this.getHeight()/2, Graphics.BASELINE|Graphics.HCENTER);
 }
    }

    public void commandAction(Command c, Displayable d) {
        if(c==exit) {
            //Stop the thread
            thread.killThread();
            //Exit the midlet
            midlet.notifyDestroyed();
        }
        else if(c==start) {
            //Start or resume the stopwatch
            thread.startWatch();
            removeCommand(start);
            addCommand(pause);
            addCommand(reset);
        }
        else if(c==pause) {
            //Pause the stopwatch
            thread.pauseWatch();
            removeCommand(pause);
            addCommand(start);
            addCommand(reset);
        }
        else {
            //Reset the counter
            if(!thread.isPaused()) {
                thread.pauseWatch();
                thread.resetWatch();
                removeCommand(pause);
                removeCommand(reset);
                addCommand(start);
            }
            else {
                thread.resetWatch();
                removeCommand(reset);
            }
            //Clear the image
            setImage(null);
        }
            repaint();
    }

}
```

StopWatchThread.java

```
import java.util.*;
import javax.microedition.lcdui.*;

public class StopWatchThread extends Thread {

    private StopWatchCanvas canvas;
    boolean paused=true;
    boolean stopped=false;
    private Date startTime, pauseTime;
    private long totaltime=0;
    private Image img;

    public StopWatchThread(StopWatchCanvas canvas) {
 this.canvas=canvas;
    }

    //Stops the thread
    public synchronized void killThread() {
 stopped=true;
    }

    //Determines whether or not the thread is running
    public synchronized boolean threadIsRunning() {
 return !stopped;
    }

    //Starts the stopwatch (or resumes after being paused)
    public synchronized void startWatch() {
        if(startTime==null)
            startTime=new Date();
 if ( img == null) img = Image.createImage(240, 160);
 paused=false;
        //Wake up the thread
 notifyAll();
    }

    //Pauses the stopwatch
    public synchronized void pauseWatch() {
        paused=true;
 pauseTime=new Date();
 totaltime=pauseTime.getTime()-startTime.getTime()+totaltime;
 //Make startTime null so that on startWatch() we get a new startTime
 startTime=null;
    }

    //Resets the stopwatch
    public synchronized void resetWatch() {
 //Make startTime null
        startTime=null;
 //Reset totaltime
 totaltime=0;
    }

    public synchronized boolean isPaused() {
```


(C) Copyright Nokia 2013. All rights reserved.

```
        return paused;
    }

    private String getTime() {
Calendar c=Calendar.getInstance();
Date rightNow=new Date();
c.setTime(new Date(rightNow.getTime()-startTime.getTime()+totaltime));

        //We work with minutes, seconds and centiseconds
int min=c.get(c.MINUTE);
int sec=c.get(c.SECOND);
        int csec=c.get(c.MILLISECOND)/10;
String smin;
String ssec;
        String scsec;

//Format the values so that they will look appropriate when displayed
if(min<10)
            smin="0"+min;
        else
            smin=String.valueOf(min);
if(sec<10)
            ssec="0"+sec;
        else
            ssec=String.valueOf(sec);
        if(csec<10)
            scsec="0"+csec;
        else
            scsec=String.valueOf(csec);

return smin+":"+ssec+":"+scsec;
    }

    public void run() {
        while(threadIsRunning()) {
            try {
                if(!isPaused()) {
                    try {
    Graphics g=img.getGraphics();
   g.setColor(255,255,255);
   g.fillRect(0, 0, img.getWidth(), img.getHeight());
   g.setColor(0,0,0);
   g.setFont(Font.getFont(Font.FACE_SYSTEM, Font.STYLE_BOLD, Font.SIZE_LARGE));
   g.drawString(getTime(), img.getWidth()/2, img.getHeight()/2, Graphics.BASELINE|Graphics.HCENTER);
   canvas.setImage(img);
   canvas.repaint();
   }
                    catch(Exception e) {

                    }
                    Thread.sleep(10);
  }
                else {
                    synchronized(this) {
                    wait();
                    }
  }
            }
            catch(InterruptedException e) {

            }
 }
    }

}
```

Download

- Project containing source code, JAR and JAD

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.