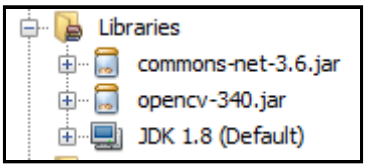
Remote Monitoring with Java

Although the author and publisher have made every effort to ensure that the information in this writing was correct at press time, the author and publisher do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause.

paid link

This is just a simple remote monitoring program. It only requires a PC and a webcam to operate. Not designed to run continously 24/7 for days. The program runs with the help of OpenCV, an open source library for computer vision.

To start with, download the OpenCV library and put the required jar files in your project folder.



This program runs on Windows7 64b machine and requires dll file *opencv_java340* to operate. If you have a similar environment, put the dll file in the root directory of your project.

Here is the complete code:

```
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.time.LocalTime;
import java.util.ArrayList;
import java.util.StringTokenizer;
```

```
import javax.imageio.ImageIO;
import org.opencv.core.Mat;
import org.opencv.videoio.VideoCapture;
import org.apache.commons.net.ftp.FTP;
import org.apache.commons.net.ftp.FTPClient;
import org.opencv.core.Core;

public class RemoteMonitoring{
    private VideoCapture cap;
    private Mat mat;
    private byte[] dat = null;

    private BufferedImage buffImg;
    private int imgW;
    private int imgH;

    private ArrayList<String> arrSched;

    public RemoteMonitoring() throws InterruptedException{
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

        arrSched = new ArrayList();
        arrSched.add('10:52:30');
        arrSched.add('10:53:00');
        arrSched.add('10:54:00');

        PictureCapture pictureCapture = new PictureCapture();
        Thread thr = new Thread(pictureCapture);
        thr.start();
    }

    class PictureCapture implements Runnable{
        @Override
        public void run() {
            imgW = 640;
            imgH = 480;
            buffImg = new BufferedImage(imgW,imgH,BufferedImage.TYPE_3BYTE_BGR);
            LocalTime curTime;

            try{
                while(true){
                    curTime = LocalTime.now();      //Get current time
                    System.out.println(curTime);

                    for(int i=0;i<arrSched.size();i++){
                        StringTokenizer sToken = new StringTokenizer(arrSched.get(i),':');
                        if(curTime.getHour()==Integer.valueOf(sToken.nextToken())%26;%26;
                           curTime.getMinute()==Integer.valueOf(sToken.nextToken())%26;%26;
                           curTime.getSecond()==Integer.valueOf(sToken.nextToken())){
                            cap = new VideoCapture();
                            mat = new Mat();
                            cap.open(0);
                            cap.read(mat);
                            cap.release();

                            if (dat == null || dat.length != imgW * imgH * 3)
                                dat = new byte[imgW * imgH * 3];
                            mat.get(0, 0, dat);

                            buffImg.getRaster().setDataElements(0, 0, mat.cols(), mat.rows(),
dat);

                            String filename = curTime.getHour()+'-'+curTime.getMinute()+'-
'+curTime.getSecond()+'.png';
                            File targetFile = new File('images/'+filename);
                            ImageIO.write(buffImg,'png',targetFile);

                            ftpUpload(targetFile.getPath(),targetFile.getName());
                        }
                    }

                    Thread.sleep(1000);
```

```
                }
            }catch (Exception ex) {
                System.out.println('Error: ' + ex.getMessage());
            }
        }

        private void ftpUpload(String filePath,String fileName){
            String server = 'ftp.domain.com';
            int port = 21;
            String user = 'admin@domain.com';
            String pass = 'abcd1234';

            FTPClient ftpClient = new FTPClient();
            try {
                ftpClient.connect(server, port);
                ftpClient.login(user, pass);
                ftpClient.enterLocalPassiveMode();
                ftpClient.setFileType(FTP.BINARY_FILE_TYPE);

                File localFile = new File(filePath);
                InputStream inputStream = new FileInputStream(localFile);
                ftpClient.storeFile(fileName, inputStream);
                inputStream.close();
            } catch (IOException ex) {
                System.out.println('Error: ' + ex.getMessage());
            } finally {
                try {
                    if (ftpClient.isConnected()) {
                        ftpClient.logout();
                        ftpClient.disconnect();
                    }
                } catch (IOException ex) {
                }
            }
        }
    }

    public static void main(String[] args){
        RemoteMonitoring ap;
        try {
            ap = new RemoteMonitoring();
        } catch (Exception ex) {
        }
    }
}
```

Explanation:

The *VideoCapture* object is needed for image capture. The captured image is stored in the image container *Mat*. The variable *dat* stores the extracted image data.

```
private VideoCapture cap;
private Mat mat;
private byte[] dat = null;
```

The variable *buffImg* manages the captured image. Image size is determined by *imgW* and *imgH*.

```
private BufferedImage buffImg;
private int imgW;
private int imgH;
```

The ArrayList *arrSched* stores the schedules in the format *hour:minute:second*.

```
arrSched = new ArrayList();
arrSched.add('10:52:30');
arrSched.add('10:53:00');
arrSched.add('10:54:00');
```

The class *PictureCapture* implements *Runnable* so that the process will run in the background. The real process is performed in the *while* loop. The variable *curTime* stores the current time. The value will be compared with the item values of the ArrayList *arrSched*. If both contain the same number values, image capturing process begins.

Default camera is opened with the sintax: *cap.open(0)*. To close the opened camera, use: *cap.release()*.

```
cap.open(0);
cap.read(mat);
cap.release();
```

In the lines of code below, image data is extracted.

```
dat = new byte[imgW * imgH * 3];
mat.get(0, 0, dat);
```

The data is then sent to the Java *BufferedImage*. No need to loop through the pixels array. This allows for a faster operation.

```
buffImg.getRaster().setDataElements(0, 0, mat.cols(), mat.rows(), dat);
```

Image filename will be in the format of 'hour-minute-second.png'.

```
String filename = curTime.getHour()+'-'+curTime.getMinute()+'-'+curTime.getSecond()+'.png';
File targetFile = new File('images/'+filename);
ImageIO.write(buffImg,'png',targetFile);
```

If everything works well, image file will be uploaded to the server.

```
ftpUpload(targetFile.getPath(),targetFile.getName());
```

Change the data below with your own FTP account.

```
String server = 'ftp.domain.com';
int port = 21;
String user = 'admin@domain.com';
String pass = 'abcd1234';
```

The uploading process starts with instantiating the *FTPClient* class.

```
FTPClient ftpClient = new FTPClient();
```

The block of code below is self explanatory. It tries to connect to the server with the data provided.

```
ftpClient.connect(server, port);
ftpClient.login(user, pass);
ftpClient.enterLocalPassiveMode();
ftpClient.setFileType(FTP.BINARY_FILE_TYPE);
```

The variable *localFile* holds the image path.

```
File localFile = new File(filePath);
```

The uploading process is performed through the code below.

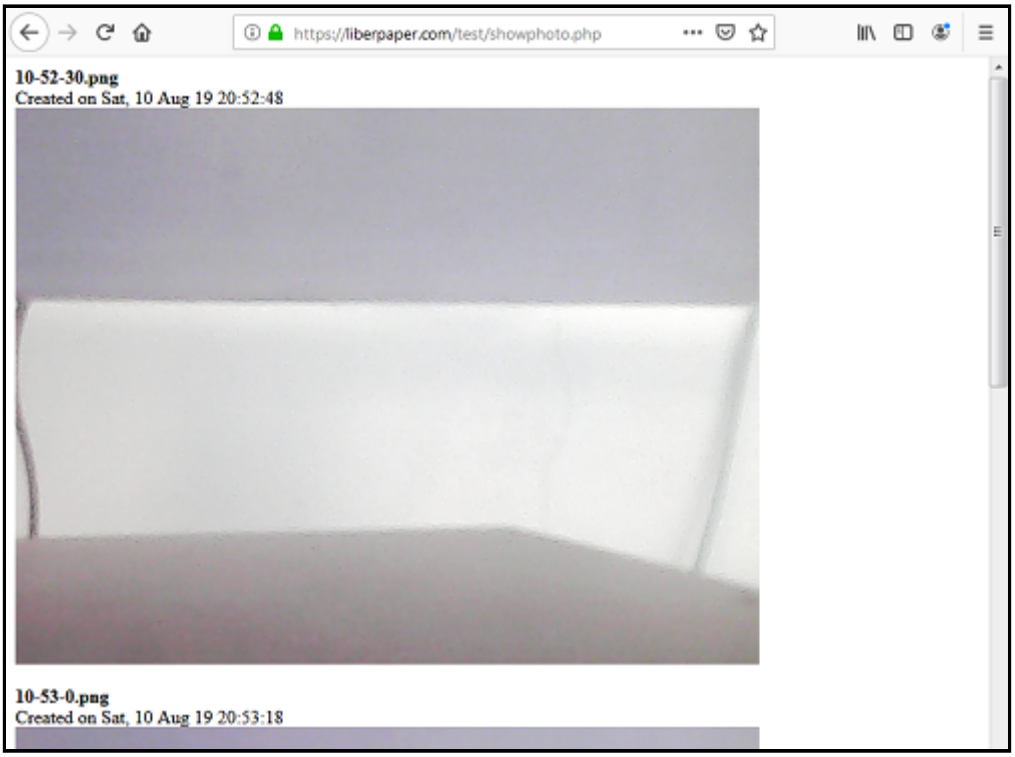
```
ftpClient.storeFile(fileName, inputStream);
```

In order to see the captured images remotely, create a php script to display the latest images captured. Store the file in the same folder as the images. The variable *num_of_files* below determines the number of images to display.

```
$images = glob('*.png', GLOB_BRACE);
$num_of_files = 5;
```

```
foreach($images as $image)
{
    $num_of_files--;
    if($num_of_files > -1)
       echo '<br><img src='.'''.$image.'''.'><br><br>' ;
    else
       break;
}
```

Use your browser to see the captured images remotely:



Feel free to change the images to anything you want.