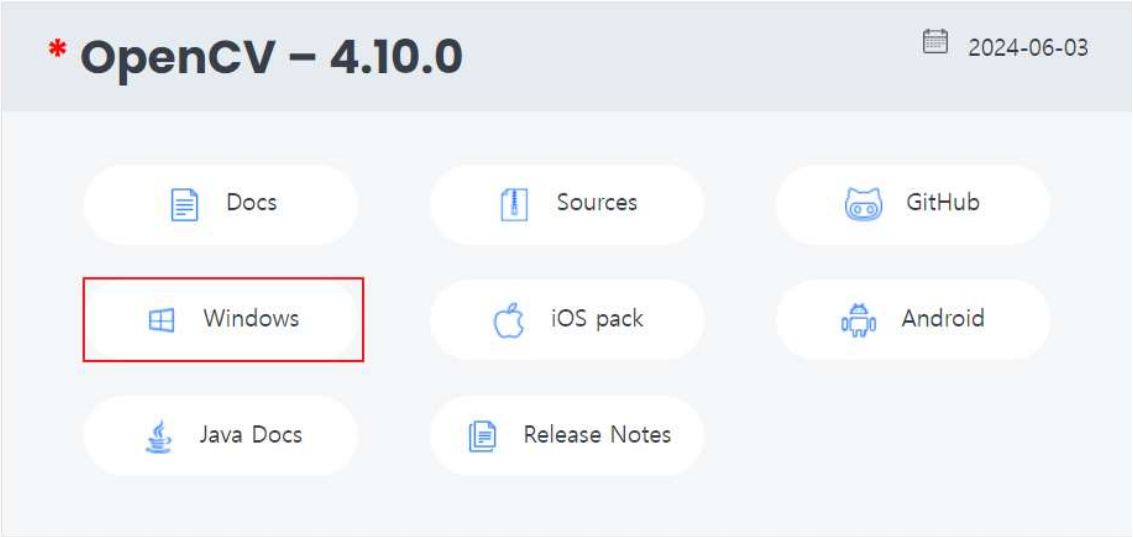
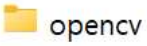
Java에서 Opencv 실행

1. 다음 사이트에서 Windows를 선택한다.
https://opencv.org/releases/

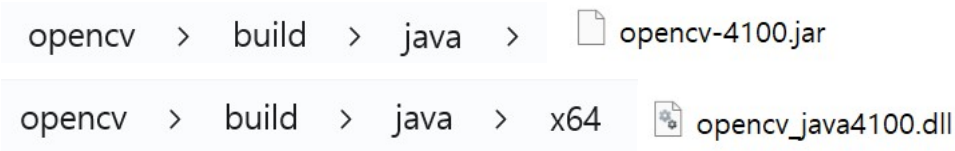


2. 다음 프로그램을 실행한다.

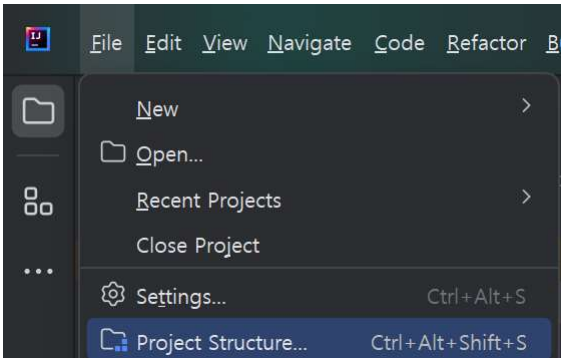
3. 다음과 같이 opencv 폴더가 생성되는 것을 확인한다.



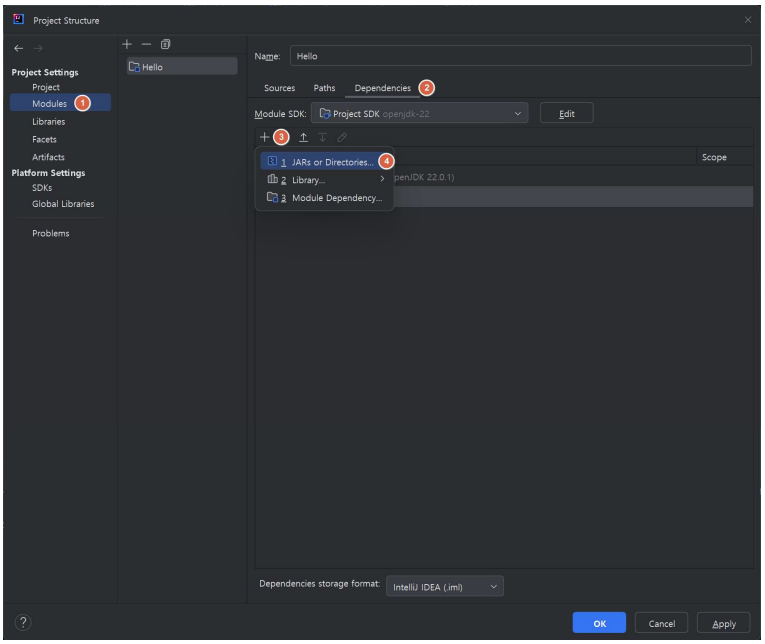
4. 다음 2개의 폴더에서 각각 opencv-4100.jar 파일과 opencv_java4100.dll 파일을 확인한다.



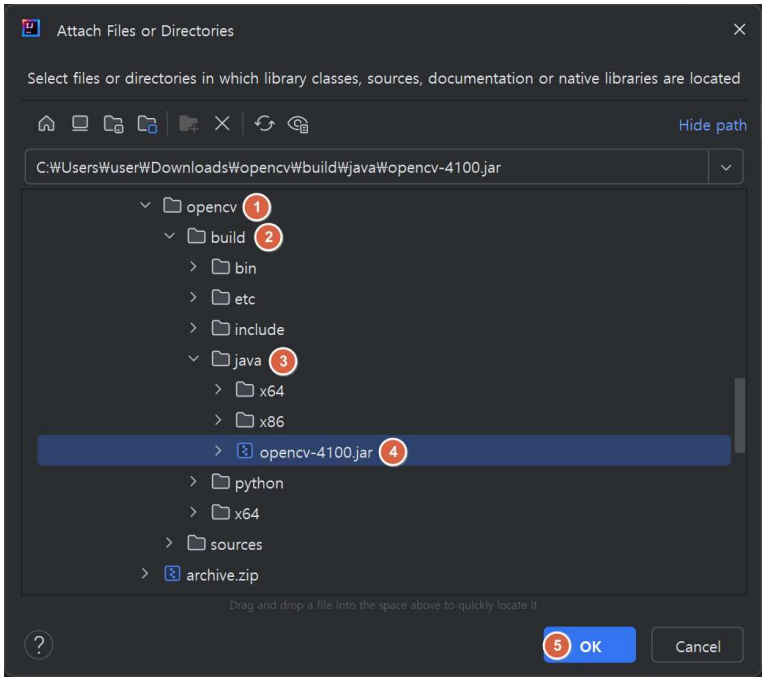
4. IntelliJ에서 다음과 같이 [File]--[Project Structure] 메뉴를 선택한다.



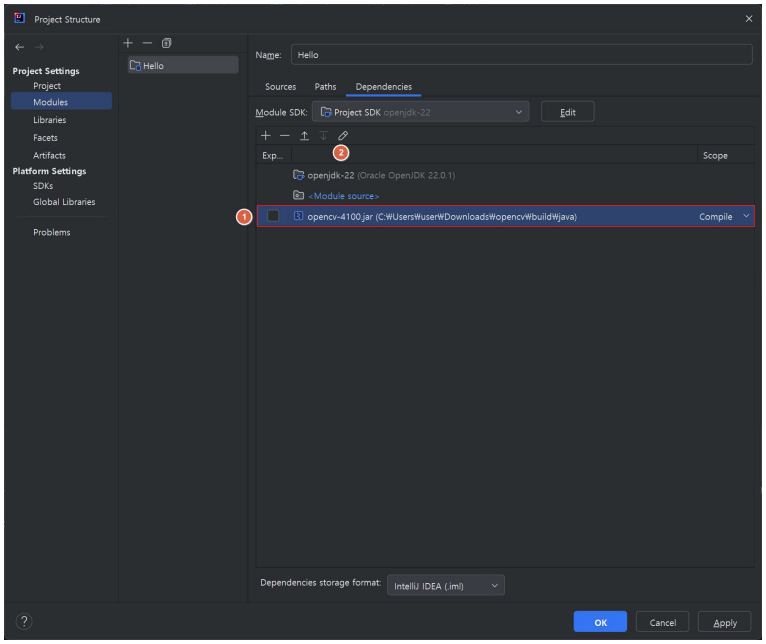
5. 다음과 같이 [Modules]--[Dependencies]--[+]--[1 JAR or Directories...] 순서로 메뉴를 선택한다.



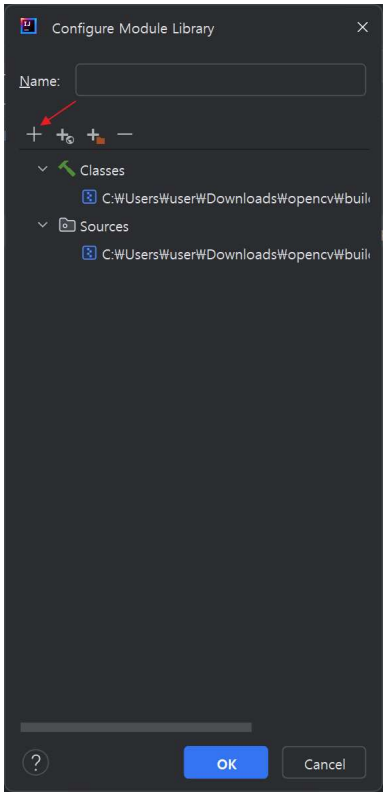
6. 다음과 같이 [opencv]--[build]--[java] 폴더에서 opencv-4100.jar 파일을 선택한다.



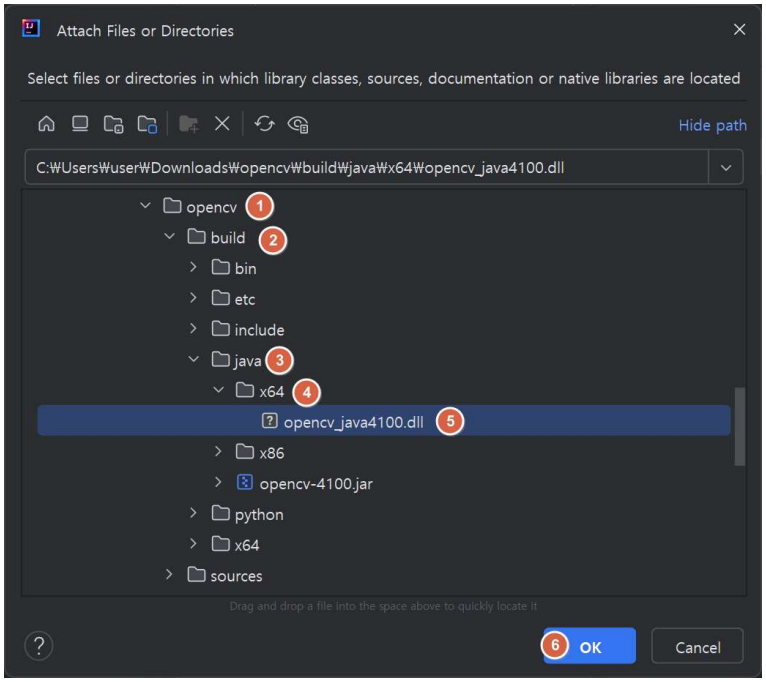
7. 다음과 같이 라이브러리가 추가되는 것을 확인하고, 연필 모양의 아이콘을 누른다.



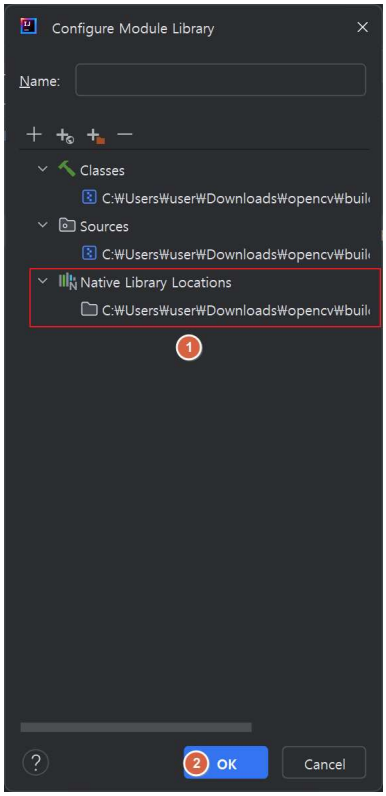
8. 다음 창에서 [+] 아이콘을 누른다.

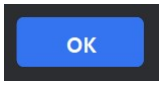


9. 다음과 같이 [opencv]--[build]--[java]--[x64] 폴더에서 opencv_java4100.dll 파일을 선택한다.



10. 다음과 같이 추가된 것을 확인한다.



11. 

버튼을 설정 창을 닫는다.

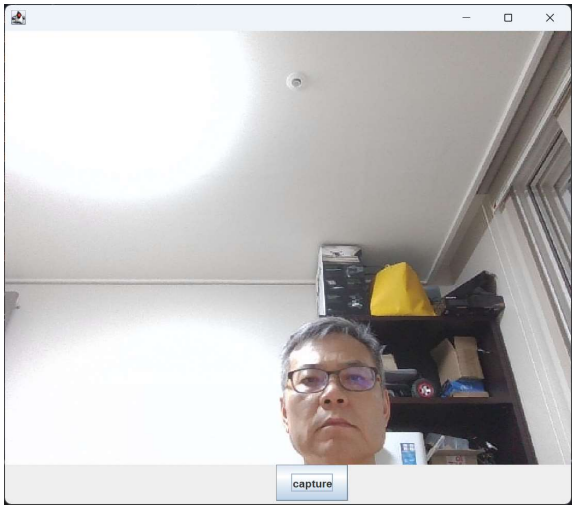
12. 다음과 같이 예제를 작성한다.
https://www.geeksforgeeks.org/taking-a-snapshot-from-system-camera-using-opencv-in-java/

```
// Java Program to take a Snapshot from System Camera
// using OpenCV

// Importing openCV modules
package com.opencvcamera;
// importing swing and awt classes
import java.awt.Dimension;
import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
// Importing date class of sql package
import java.sql.Date;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
```

```java
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.core.MatOfByte;
import org.opencv.imgcodecs.Imgcodecs;
// Importing VideoCapture class
// This class is responsible for taking screenshot
import org.opencv.videoio.VideoCapture;

// Class - Swing Class
public class Camera extends JFrame {

	// Camera screen
	private JLabel cameraScreen;

	// Button for image capture
	private JButton btnCapture;

	// Start camera
	private VideoCapture capture;

	// Store image as 2D matrix
	private Mat image;

	private boolean clicked = false;

	public Camera()
	{

		// Designing UI
		setLayout(null);

		cameraScreen = new JLabel();
		cameraScreen.setBounds(0, 0, 640, 480);
		add(cameraScreen);

		btnCapture = new JButton("capture");
		btnCapture.setBounds(300, 480, 80, 40);
		add(btnCapture);

		btnCapture.addActionListener(new ActionListener() {
			@Override
			public void actionPerformed(ActionEvent e)
			{

				clicked = true;
			}
		});

		setSize(new Dimension(640, 560));
```

```
                setLocationRelativeTo(null);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setVisible(true);
        }

        // Creating a camera
        public void startCamera()
        {
                capture = new VideoCapture(0);
                image = new Mat();
                byte[] imageData;

                ImageIcon icon;
                while (true) {
                        // read image to matrix
                        capture.read(image);

                        // convert matrix to byte
                        final MatOfByte buf = new MatOfByte();
                        Imgcodecs.imencode(".jpg", image, buf);

                        imageData = buf.toArray();

                        // Add to JLabel
                        icon = new ImageIcon(imageData);
                        cameraScreen.setIcon(icon);

                        // Capture and save to file
                        if (clicked) {
                                // prompt for enter image name
                                String name = JOptionPane.showInputDialog(
                                        this, "Enter image name");
                                if (name == null) {
                                        name = new SimpleDateFormat(
                                                        "yyyy-mm-dd-hh-mm-ss")
                                                        .format(new Date(
                                                                HEIGHT,         WIDTH,
getX()));
                                }

                                // Write to file
                                Imgcodecs.imwrite("images/" + name + ".jpg",
                                                                image);

                                clicked = false;
                        }
                }
        }

        // Main driver method
        public static void main(String[] args)
```

```
        {
                System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
                EventQueue.invokeLater(new Runnable() {
                        // Overriding existing run() method
                        @Override public void run()
                        {
                                final Camera camera = new Camera();

                                // Start camera in thread
                                new Thread(new Runnable() {
                                        @Override public void run()
                                        {
                                                camera.startCamera();
                                        }
                                }).start();
                        }
                });
        }
}
```



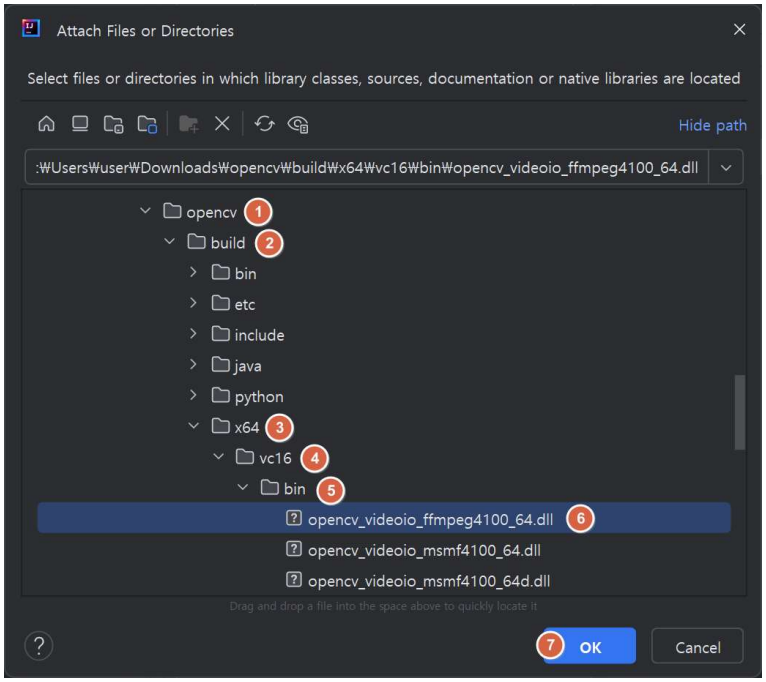
HTTP IP 스트리밍 서버

다음 사이트를 참조한다.
https://github.com/mesutpiskin/opencv-live-video-stream-over-http/tree/master/src/main/java

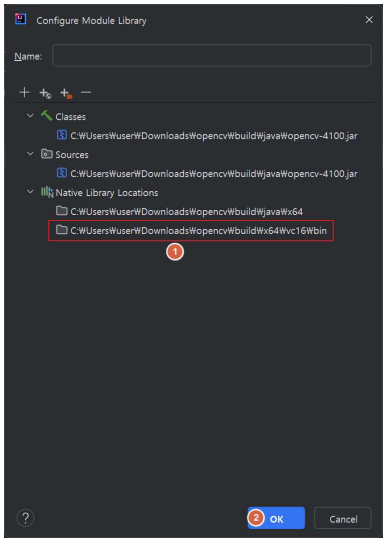
HTTP IP 스트리밍 클라이언트

다음 사이트를 참조한다.
https://stackoverflow.com/questions/26535645/ip-camera-with-opencv-in-java

1. 다음과 같이 opencv_video_ffmpeg4100_64.dll 파일을 선택한다.



2. 다음과 같이 추가된 것을 확인하고 [OK] 버튼을 누른다.



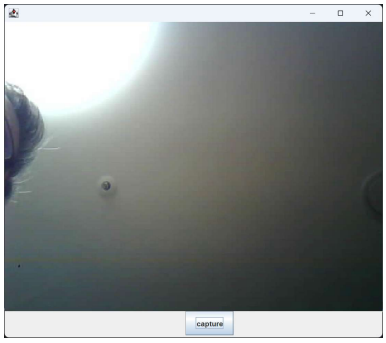
3. 앞에서 수행했던 webcam 예제의 Camera 클래스에 다음과 같이 초기화 블록을 추가한다. 생성자 바로 앞에 추가한다.

```
44      {
45          System.loadLibrary( libname: "opencv_videoio_ffmpeg4100_64");
46      }
47
48      public Camera()  1 usage
49      {
```

4. 다음과 같이 IP 주소를 설정한다.

```
78      public void startCamera()  1 usage
79      {
80          capture = new VideoCapture( filename: "http://192.168.4.1:81/stream");
```

5. 결과를 확인한다.



객체 인식

다음 사이트를 참조한다.
https://github.com/mesutpiskin/opencv-object-detection

opencv-object-detection / src / DeepNeuralNetwork /

- Application.java
- DnnObject.java
- DnnProcessor.java

안드로이드 JPEG IP 스트리밍

https://answers.opencv.org/question/15812/ip-camera-frames-manipulation/
https://github.com/fury999io/public-ip-cams?tab=readme-ov-file
https://yottu.tistory.com/20

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                EdgeToEdge.enable(this);
                setContentView(R.layout.activity_main);
                ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
                        Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
                        v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
                        return insets;
                });

                LinearLayout main=findViewById(R.id.main);
                MyVideoView myVideoView = new MyVideoView(this);
                main.addView(myVideoView);
        }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:id="@+id/main"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        tools:context=".MainActivity" />
```

MyVideoView.java

```
public class MyVideoView extends SurfaceView implements SurfaceHolder.Callback, Runnable {

        Thread thread;
        boolean threadRunning = true;

        public MyVideoView(Context context) {
                super(context);
                getHolder().addCallback(this);

                thread = new Thread(this);
        }

        @Override
        public void surfaceCreated(@NonNull SurfaceHolder holder) {
                thread.start();
        }

        @Override
```

```
public void surfaceChanged(@NonNull SurfaceHolder holder, int format, int width, int height) {

}

@Override
public void surfaceDestroyed(@NonNull SurfaceHolder holder) {
        threadRunning = false;
        try {
                thread.join();
        } catch (InterruptedException e) {}
}

byte[] arr=new byte[1000000];

@Override
public void run() {

        SurfaceHolder holder=getHolder();

        try {
                URL url=new URL("http://131.95.3.163/mjpg/video.mjpg")://"http://192.168.4.1:81/stream");
                //URL url=new URL("http://220.233.144.165:8888/mjpg/video.mjpg");
                HttpURLConnection connection=(HttpURLConnection) url.openConnection();
                InputStream inputStream=connection.getInputStream();

                while (threadRunning) {
                        int i=0;
                        // check for jpeg soi sequence: ff d8
                        for(;i<1000;i++) {
                                int b=inputStream.read();
                                if(b==0xff) {
                                        int b2=inputStream.read();
                                        if(b2==0xd8) break;
                                }
                        }
                        if(i>999) {
                                Log.e("MJPG", "bad head!");
                                continue;
                        }
                        arr[0]=(byte)0xff;
                        arr[1]=(byte)0xd8;
                        i=2;
                        // check for jpeg eoi sequence: ff d9
                        for(;i<100000;i++) {
                                int b=inputStream.read();
                                arr[i]=(byte)b;
                                if(b==0xff) {
                                        i++;
                                        int b2=inputStream.read();
                                        arr[i]=(byte)b2;
                                        if(b2==0xd9) break;
                                }
                        }
                        i++;

                        int nBytes=i;
                        Log.e("MJPG", "got an image, "+nBytes+" bytes");

                        Bitmap bmp= BitmapFactory.decodeByteArray(arr,0,nBytes);

                        Canvas canvas=holder.lockCanvas();

                        canvas.drawBitmap(bmp,0,0,new Paint());

                        holder.unlockCanvasAndPost(canvas);
                }
        } catch (Exception e) {
```

```
                                Log.e("MJPG", e.toString());
                        }

                }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools">

        <application
                android:allowBackup="true"
                android:dataExtractionRules="@xml/data_extraction_rules"
                android:fullBackupContent="@xml/backup_rules"
                android:icon="@mipmap/ic_launcher"
                android:label="@string/app_name"
                android:roundIcon="@mipmap/ic_launcher_round"
                android:supportsRtl="true"
                android:theme="@style/Theme.MyApplication"
                android:usesCleartextTraffic="true"
                tools:targetApi="31">
                <activity
                        android:name=".MainActivity"
                        android:exported="true">
                        <intent-filter>
                                <action android:name="android.intent.action.MAIN" />

                                <category android:name="android.intent.category.LAUNCHER" />
                        </intent-filter>
                </activity>
        </application>
        <uses-permission android:name="android.permission.INTERNET"/>

</manifest>
```