Programming Security Surveillance Systems

Mostafa Abd-ElHamid Atwa

MOSTAFA ABD-ELHAMID ATWA

PROGRAMMING SECURITY SURVEILLANCE SYSTEMS

Programming Security Surveillance Systems
1st edition

ISBN 978-87-403-1372-7
Peer reviewed by MSc Mesfer El-Zahrani, King Saud University MSc

CONTENTS

DEDICATION

For Manon Niazi, the Deutsch-Lander. I attended college to see her; did not attend my lessons. I have now moved to the computer academy.

OBJECTIVE

This book is aimed at teaching software engineers how to program surveillance systems and IP camera based solutions. It will deliver coverage of an advanced topic in engineering surveillance systems and advanced byte-level programming concepts in image processing techniques.

SUBJECT

Programming IP camera based solutions and getting your video up and running, getting to know about what an image programmatically is, how to create a video from an image coming from the IP camera, and working on how to broadcast and save any stream to your storage engine, are some of the concepts that will be discussed.

We will also look at the process of creating videos from images and creating image sensors according to the percentage of change from one image to another in a timeline, and will learn how to make surveillance analytics.

Most of the topics of this book will be in Java, PHP and C#.

ABOUT THE AUTHOR AND REVIEWER

Mostafa A. Hamid, Information Systems Engineer

Bachelors Degree Holder [Modern Academy for Computer Science and Management Technology]

Certified from the AUC in Cairo, in Java Programming

Certified From SUNY Potsdam in CISSP, CEH, JS, PHP, ASP.net, Java, C#, Power SHELL REMOTING, and IBM RUP.

Certified IOT Professional x from Massachusetts Institute of technology

Official Reviewer, Mesfer El-Zahrani

Msc in computer science (KSA)
Business Owner (Security SOLUTIONS EST.)

FOREWORD

Programming forms an extensive world of creating programs and delivering services to customers, clients and businesses that can be beneficial in everyday routine tasks or extraordinary unpredictable tasks.

It is the means of giving instructions to the computer and processing these instructions to give the output to the user.

There are three languages that we will focus on in our study in this book, which will be PHP which stands for Zend's Personal Home Page, named after a Danish-Canadian programmer Rasmus Lerdorf, Oracle's Java, and Microsoft's Visual C#.

1 INTRODUCTION TO JAVA AS A PROGRAMMING LANGUAGE

Java as a programming language is the most powerful high level programming language after the C language which is ranked number 2 and comes next to it, the PHP language as the number 1 web programming language in the world, then comes C# as number 3, which we will walk through code in each of them in this book.

Java is an object oriented programming language that uses interfaces, classes and functions.

If you are not familiar with any programming language or do not have any programming experience, I suggest you read any book on programming and do some programming assignments before you continue reading this book.

You might find the following webpage helpful:

https://docs.oracle.com/javase/tutorial/

Oracle's Java tutorials website will help prepare you to handle most of the tasks covered in this book.

We will get familiar with Netbeans and JDK 8 as we continue with our book.

2 INTRODUCTION TO PHP AS A PROGRAMMING LANGUAGE

PHP also is an object oriented programming language that can enable you to have a web interface on apache or nginx, for example.

PHP language also has the capability to let you write byte level code and image processing mechanisms.

We will make a robust PHP application that will enable you to have a sensor which will detect motion in an iterative image repainting mechanism.

With PHP, we will have the capability to transfer the captured image from our device to any cloud service provider and also to transfer it using local integration servers, and finally, make a Facebook post, send an email or send an mms to any phone number, with the captured image.

We will get familiar with Netbeans (developed by Oracle) as our development environment and we will get acquainted with apache server and nginx as the hosting servers for PHP.

We will create a video using PHP and display it on a web page.

3 INTRODUCTION TO C# AS A PROGRAMMING LANGUAGE

C# is a visual language created by Microsoft. It is an object oriented programming language like the other two programming languages. We will undertake in C#, the same tasks that we talked of completing in PHP and Java.

We will create a video in C# from the iterative image capturing process we develop.

We will get to recognise the same functions that we did mentioned in the case of the previous two languages.

Let us understand the Microsoft development environment, Visual Studio 2015 Community Edition.

4 WORKING WITH WEB REQUESTS IN JAVA

As we work with web requests in Java, PHP, and C#, we will start with installing our Java development environment in as follows:

1) Download and install Java JDK 8 from Oracle.
2) Download and install Netbeans from the same website.
3) Alternatively, you need to download OpenJDk from the Java.net website.
4) Then you can work with Notepad++, which is a text editor especially suited for writing many languages which will be an alternative to Netbeans.
5) You will need to have an IP camera which is configured properly. The required configuration as an example, will be:

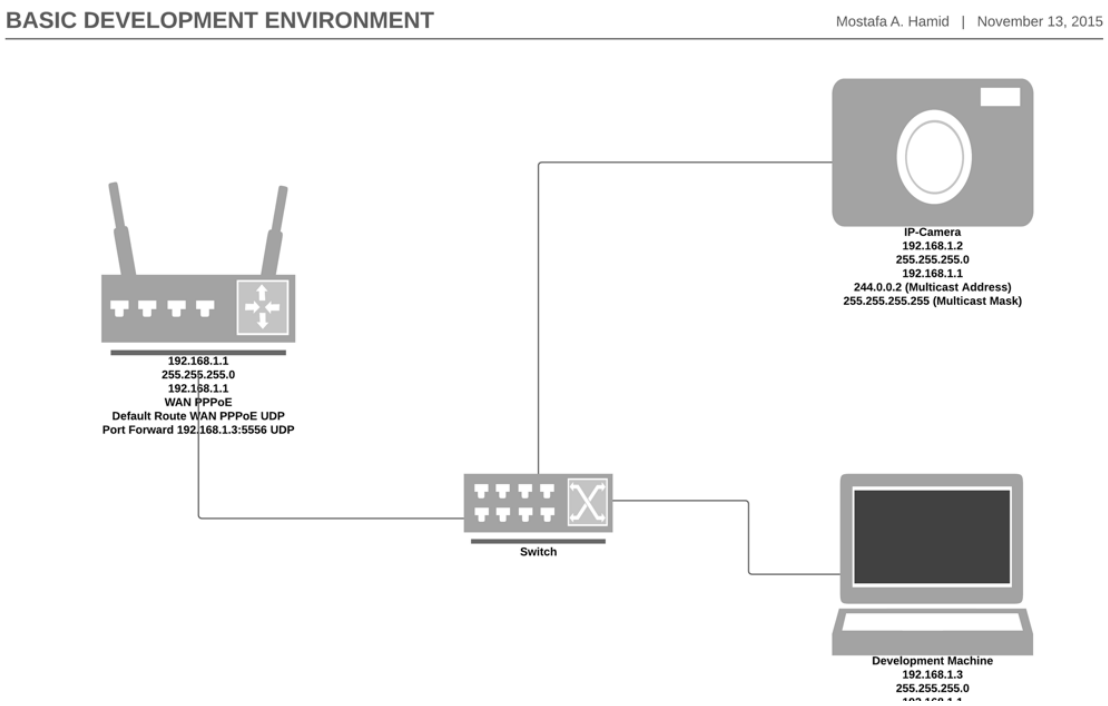
    a. IP address of 192.168.1.2 using the Ethernet network settings page.
    b. Enable unicast on the IP camera and point to the IP address of your development computer and use port 5556.

c. As an alternative, we can use multicasting on an IP address of 244.0.0.1 and subnet mask of 255.255.255.255 and the default gateway of 192.168.1.1.
d. Connect your IP Camera directly to an Ethernet switch using CAT5 (Category 5) or CAT6 (Category 6) UTP (UN-Shielded Twisted Pair) Ethernet Cables.
e. Assign the authentication by going to users or security sections in your IP camera configuration page, then assign the basic authentication mode over HTTP to use the base64 encryption mechanism and put any desired user name and password that will be accepted by your camera's software.

6) You will need an Ethernet switch with at least 4 Ethernet ports.
7) You will need a router or modem exactly like the device that you use with an ADSL (Asymmetric Digital Subscriber Line) ISP (Internet Service Provider) and configured as follows:

a. Assign an IP address to the router or modem as 192.168.1.1 and make the subnet mask 255.255.255.0 and the default gateway of 192.168.1.1 using the Ethernet settings on your router (sometimes called LAN settings).
b. Create a new WAN and prepare it for PPPoE (Point to Point Protocol over Ethernet).
c. Create a default route for this WAN Network and make the default route point to your development environment computer's IP address and make the default route over your newly created WAN if there is no option of pointing it to the IP address of your development machine.
d. Finally enable port forwarding, then set the protocol used in the port forwarding option to: UDP (User Datagram Protocol).

The network should be typical to the following diagram:



At of the time writing this book, an attempt to download and install Netbeans 8.1.2, and JDK 8 from the *downloads* corner of Oracle.com, showed the following screenshots:

Next, we install them and the following screenshots are displayed:

Before moving on, we need to import a library into Netbeans, called javax. This can be downloaded from the following URL:

http://www.java2s.com/Code/Jar/j/Downloadjavaxjar.htm

Now, let us start writing our code, starting with our first web request, as follows:

Create a new file called HTTPClass.java, then if you are using Netbeans, double-click the file to edit it and enter the following code:

```
import java.io.IOException;

        // Importing Exception Class to report exceptions
```

```
import java.net.*;

      // Importing Net Library

import java.awt.Image;

      // Importing Awt Library's Image Class to Create an Image

import java.net.URL;

      // Importing Net's URL class to work with URLs.

import javax.imageio.ImageIO;

      // Importing ImageIO class to input, output and create
         images.
```

```
import javax.swing.*;

      // Importing Swing User Interface Library.

import java.lang.*;

      // Importing Lang Library to have the basic features.

import java.lang.InterruptedException;

      // Importing Interrupted Exception class to report thread
         exception

import java.awt.image.BufferedImage;

      // Importing Buffered Image class to create a buffered
         stream.

import java.awt.image.WritableRaster;

      // Importing Writable Raster class to get image raster
         details

import java.awt.image.DataBufferByte;

      // Importing DataBufferByte to create a data buffer byte
         array

public class HTTPClass{

      public static void main(String[] args){

          try{

               Authenticator.setDefault(new
CustomAuthenticator());
```

```
                // Authenticate using custom authenticator
    at the bottom of the code containing // you IP-Camera
    user name and password

            JFrame frame = new JFrame();

                // This will create a new window in java

            frame.setSize(300,300);

                // This will make the size of the window
that we created

                // 300 x 300 width and height.

            URL url = new URL("http://192.168.1.2/cgi-
bin/viewer/video.jpg");

                // This will create a new connection to
this URL.

                // Starting from /cgi-bin/viewer/video.
jpg changes from one brand to

                // another and one model to another.

        Image img = ImageIO.read(url);

                // Reading the content from the url as
an Image

            JLabel label = new JLabel();

                // Created a new Label

            ImageIcon imgIcon = new ImageIcon(img);

                // Created a new Image Icon.
```

```
                label.setIcon(imgIcon);

                    // Set the icon into the label.

            frame.add(label);

                    // Add the label containing the image
to the frame.

            frame.setVisible(true);

                    // Make the frame visible.

            for(;;){

                    // Loop forever

                Authentication.setDefault(newCustomAuthentica
tion());

                    // Create an authentication instance
```

```
            URL url2 = new URL("http://192.168.1.2/cgi-
bin/viewer/video.jpg");

                // Create a new connection to the URL
of

                // the camera to get the new image every
time

            Image img2 = ImageIO.read(url2);

                // Every iteration create a new image

            WritableRaster imgRaster = img3.getRaster();

                // This will get you how the image is
drawn

            DataBufferByte data =
(DataBufferByte)imgRaster.getDataBuffer();

                // Get the image byte level data

            System.out.println("The raster of the image
is: " + imgRaster);

            System.out.println("The Image Byte Level Data
is: " + data);

            label.setIcon(imgIcon2);

            label.repaint();

            try{

                Thread.sleep(500);

                        // order the loop to sleep for
500 milliseconds
```

```
                } catch (java.lang.InterruptedException ie)
{

                        System.out.println("Interrupted
Exception Occurred: " + ie);

                }

            }

        } catch (MalformedURLException mue) {

            System.out.println("Malformed Exception Occurred:
" + mue);

            }catch(IOException ioe){

                System.out.println("I/O Exception Occurred:
" + ioe);

                }

            }

        }

public static class CustomAuthenticator extends Authenticator{
        protected PasswordAuthentication
getPasswordAuthentication(){

            String username = "admin";

            String password = "manon1982";

            return new PasswordAuthentication(username,passwo
rd.toCharArray());

        }

}
```

4 WORKING WITH WEB REQUESTS IN PHP

Web requests in PHP is done using a function called CURL, this function is executed and can take the authentication parameters within the URL and does not require a specific class as we made in the previous java example as follows:

```
<?php

     namespace  IpCameraCurlCommand\Connect;

     class  Initialize{

          const  URL  =

"http://username@password:192.168.1.2/cgi-bin/viewer/video.jpg";

               // Replace this with your IP camera's image URL.
```

```
        public static function __init(){
            for(;;){
                $ch = curl_init();
                curl_setopt($ch, CURLOPT_URL, URL);
                curl_setopt($ch, CURLOPT_HEADER, 0);
                $img = curl_exec($ch);
                echo '<img src = ' . $img . '>';
                curl_close($ch);
            }
        }
    }
\IpCameraCurlCommand\Connect\Initialize::__init();
```

5 WORKING WITH WEB REQUESTS IN C#

We have made the previous example in Java and PHP. Let us script the same in C#, as the following example shows.

Create a new Windows Forms application in Visual Studio and create a new Windows Form and name it, for example, form1, then insert the following code into the file called form1.cs as in the following code.

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Net;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;


namespace WindowsFormsApplication3

{
```

```
public partial class Form1 : Form
{
    public Bitmap img;
    public Form1()
    {
        InitializeComponent();
        showImage();
    }
    public void showImage()
    {
```

```
this.pictureBox1.Image = null;

// this.label2.Text = "Adjusting…";

for (int i = 0; i < 100; i++) {

Task t = Task.Run(() =>

{

string CameraUrl =
"http://192.168.1.2/cgi-bin/viewer/video.jpg?streamid=0";

if (string.IsNullOrEmpty(CameraUrl)
== false)

{

byte[] buffer = new byte[300000];

int read, total = 0;

// Create a HTTP Request

try {

HttpWebRequest req =
(HttpWebRequest)WebRequest.Create(CameraUrl);

req.Credentials = new
NetworkCredential("username", "password");

// Get Response

try

{
```

```
                    WebResponse resp = req.
GetResponse();

                        // while (true) {

                    Stream stream = resp.
GetResponseStream();

                        //Read data from stream

                    while ((read = stream.
Read(buffer, total, 1000)) != 0)

                    {

                            total += read;

                        }

                            //Get Bitmap

                    MemoryStream memstream
= new MemoryStream(buffer, 0, total);

                        img = (Bitmap)Bitmap.
FromStream(memstream);

                        this.pictureBox1.Image
= img;

                            // Bitmap img =
(Bitmap)Bitmap.FromStream(memstream);

                            // Exception here

                    }

                    catch (IOException ioe)

                    {
```

```
                                    label2.Text = "System
Exception Occurred: " + ioe;

                                }

                            }

                            catch (IOException ex)

                            {

                                label2.Text = "System
Exception Occurred: " + ex;

                            }

                        }

                    });
```

```
                t.Wait(1000);
            } //endfor
        }
        private void button1_Click(object sender, EventArgs
e)
        {
            for (int i = 0; i < 10; i++)
            {
                label2.Text = "Updating…";
                Task t = Task.Run(() =>
                {
                    pictureBox1.Image = null;
                    showImage();
                });
                t.Wait(1000);
                label2.Text = "Done";
            }
        }
    }
}
```

Note that you need to change the camera URL to the address that you assigned to the camera while configuring it.

6 CREATING A USER INTERFACE AS A WEB LAYOUT IN PHP, HTML5, CSS3 AND JS BACKBONE MVC FRAMEWORK

Moving on from our previous sections, let us make an MVC front end layout using backbone as follows:

Create a file called index.html or index.htm and paste the following content into it.

```
<!doctype html>
<html> <!--Starting the document -->
     <head> <!--Creating the head content to be preloaded
before page content -->
          <title>Backbone Js Application</title> <!--will
appear on the browser tab -->
     </head>
     <body>
          <header>Test Application of Backbone MVC with
Handlebars Templating Engine</header>

          <div id="container">Loading…</div>

          <!-- This is a Handlebar Portion -->

          <script type = "text/template" id = "home-template">

               <h1>Camera video</h1>
               {{greetings}}
          </script>

          <div id = "container2"></div>

          <!-- This is the end of the handlebar portion
-->
```

```
        <footer>

            <script src = "js/libs/jquery-2.1.3.min.js"></
script> <!--Can be downloaded over the internet -->

            <script src = "js/libs/underscore-min.js"></
script> <!--Can be downloaded over the internet -->

            <script src = "js/libs/backbone-min.js"></
script> <!--Can be downloaded over the internet-->

            <script src = "js/libs/handlebars-v3.0.1.js"></
script><!--Can be downloaded over the internet -->

            <script src = "js/libs/json2.js"></script><!-
-Can be downloaded over the internet -->
```

```
            <script src = "js/models/MainModel.js"></
script> <!--Our main model that will come later in the
next part-->

            <script src = "js/views/MainView.js"></script>
<!--Our Main View that will be coming later in this chapter
-->

            <script src = "js/scripts.js"></script>

        </footer>

    </body>

</html>
```

In the js folder, let us create the scripts.js file as follows:

```
var assignedDateEnd = null; //assign date and time you
want to loop for

for(var i = Date.now(); i < assignedDateEnd; i++){

var xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {

        if (xhttp.readyState == 4 && xhttp.status == 200)
{

            var img = xhttp.responseText;

        }

xhttp.open("POST", "http://username@password:192.168.1.2/
cgi-bin/viewer/video.jpg", true);

xhttp.setRequestHeader("Content-type", "application/x-www-
form-urlencoded");
```

```
xhttp.send("streamid=0");

};

};

};

var xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {

    xhttp.open("POST", "http://forget@forget:192.168.1.2/
cgi-bin/viewer/video.jpg", true);

    xhttp.setRequestHeader("Content-type", "application/x-
www-form-urlencoded");

    xhttp.send("streamid=0");

};

};

var magazine10 = new Magazine();

magazine10.set("title", "Manon Niazei A. Ghafour");

magazine10.set("pubDate", "12/02/1984");

magazine10.set("image", img);

console.log("The title has been set: " + magazine10.
get("title"));

var magazine1 = new Magazine({

    title:"Manon Niazei",

    pubDate: "12/02",

    image: img,

});
```

```
var MagazineCollection = Backbone.Collection.extend({

    model: Magazine,

});

console.log("start of the foreach");

var magazines = new
MagazineCollection([magazine10,magazine1]);

magazines.forEach(function(model){

    console.log(model.get("image"));

});

console.log("End of the foreach");
```

```
console.log(JSON.stringify(magazine1));

console.log("Magazine 1 title: " + magazine10.get("title"));
```

In our views folder, let us create a file called MainView.js with the following contents:

```
for(var i = Date.now(); i < assignedDateEnd; i++){
var xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {

        if (xhttp.readyState == 4 && xhttp.status == 200)
        {

            var AppView = Backbone.View.extend({

                el: '#container',

                template: Handlebars.compile($("#home-
template").html()),

                initialize: function(){

                this.render();

            },

    render: function(){

    this.$el.html(this.template({greetings: "Welcome to
Manon Niazei Application in Backbone MVC and Handlebars
Template!"}) + " <br /> <b>The Content of the Model: </
b>" + magazineModelInst.get("title") + "<br />Image: <img
src = " + xhttp.responseText + " />");

    },
});
        }
```

```
     xhttp.open("POST", " http://username@password:192.168.1.2/
cgi-bin/viewer/video.jpg",  true);

     xhttp.setRequestHeader("Content-type", "application/x-
www-form-urlencoded");

     xhttp.send("streamid=0");

};

};

};

var  appView  =  new  AppView();

var  AppRouter  =  Backbone.Router.extend({

          routes:  {

               '':  'homeRoute',

               'home':  'homeRoute',

          },

          homeRoute:  function  ()  {

               var  appView  =  new  AppView();

               $("#container2").html(appView.el);

          },

});

var  appRouter  =  new  AppRouter();
```

```
Backbone.history.start();

console.log(appView);
```

Let us now create a model in our models directory and name it MainModel.js, which will have the following contents:

```
var Magazine = Backbone.Model.extend({

    defaults:{

        title:'',

        pubDate: '',

        image: '',

    },
```

```
    initialize: function(){
        console.log("The model has been initialized");
        this.on("change:title", function(){
            console.log("The model's data has been changed");
        });
    }
});
var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (xhttp.readyState == 4 && xhttp.status == 200)
{
            var img = xhttp.responseText;
        }
xhttp.open("POST", "http://username@password:192.168.1.2/
cgi-bin/viewer/video.jpg", true);
xhttp.setRequestHeader("Content-type", "application/x-www-
form-urlencoded");
     xhttp.send("streamid=0");
};
};
```

```
var magazineModelInst = new Magazine({

     title: 'Manon Niazei is a Deutschlander',

     pubDate: '12/02',

     image: img,

});
```

Now, we create our libs folder and download the following:

backbone-min.js

handlebars-v3.0.1.js

handlebars.runtime-v3.0.1.js

jquery-2.1.3.min.js

json2.js

underscore-min.js

After this, you are ready work on the backbone mvc web application with handlebars template engine.

Open the index.html file and you can see the video and when the camera changes view, you will see it yourself on your web application.

We now move on to the next component which will be multi casting and creating our multicast Java application on the SE layout for desktop machines.

7 INTRODUCTION TO BYTES IN PROGRAMMING AND GETTING TO KNOW BYTES

```
/*

 * To change this license header, choose License Headers
in Project Properties.

 * To change this template file, choose Tools | Templates

 * and open the template in the editor.

 */
```

```
package multicasting;

/**

 *

 * @author Mostafa

 */

import java.util.*;

public class RTPpacket{

    //size of the RTP header:

    static int HEADER_SIZE =

12;

    //Fields that compose the RTP header
    public int Version;
    public int Padding;
    public int Extension;
    public int CC;
    public int Marker;
    public int PayloadType;
    public int SequenceNumber;
    public int TimeStamp;
    public int Ssrc;

    //Bitstream of the RTP header
    public byte[] header;

    //size of the RTP payload
    public int payload_size;
    //Bitstream of the RTP payload
    public byte[] payload;
    public byte[] byt3;
```

```
    public byte[] createFirst2Bytes(){
        byte[] byt = new byte[1];
        byt[0] = (byte)0xff;
        byt[1] = (byte)0xd8;
        return byt;

    }

    public ArrayList<byte[]> CreateJFIFHeader(float type,
float width, float height, ArrayList<byte[]> tables, float
dri){

        byte[] byt = new byte[17];
        byt[0] = (byte)0xff;
        byt[1] = (byte)0xe0;
        byt[2] = (byte)0x00;
        byt[3] = (byte)0x10;
        byt[4] = (byte)'J';
        byt[5] = (byte)'F';
        byt[6] = (byte)'I';
        byt[7] = (byte)'F';
        byt[8] = (byte)0x00;
        byt[9] = (byte)0x01;
        byt[10] = (byte)0x00;
        byt[11] = (byte)0x00;
        byt[12] = (byte)0x01;
        byt[13] = (byte)0x00;
        byt[14] = (byte)0x01;
        byt[15] = (byte)0x00;
        byt[16] = (byte)0x00;
        tables.add(byt);
        if(dri > 0){

tables.add(CreateDataRestartIntervalMarker(dri));

        }
        return tables;
    }
```

```
public byte[] CreateQuantizationTables(float Q){
    int factor = (int)Q;
    int q;
    if(Q < 1) factor = 1;
    else if (Q > 99) factor = 99;
    if(Q < 50){
        q = 5000 / factor;
    }else{
        q = 200 – factor * 2;
    }
    byte[] dq = this.defaultQuantizers();
    byte[] resultTables = new byte[128];
    for(int i = 0; i < 128; i++){
        int newVal = (dq[i] * q + 50) / 100;
        if(newVal < 1) newVal = 1;
        else if(newVal > 255) newVal = 255;
        resultTables[i] = (byte)newVal;
    }
    return resultTables;
}
```

```
public byte[] defaultQuantizers(){
     byte[] lumaChroma = new byte[128];
     lumaChroma[0] = (byte)16;
     lumaChroma[1] = (byte)11;
     lumaChroma[2] = (byte)12;
     lumaChroma[3] = (byte)14;
     lumaChroma[4] = (byte)12;
     lumaChroma[5] = (byte)10;
     lumaChroma[6] = (byte)16;
     lumaChroma[7] = (byte)14;
     lumaChroma[8] = (byte)13;
     lumaChroma[9] = (byte)14;
     lumaChroma[10] = (byte)18;
     lumaChroma[11] = (byte)17;
     lumaChroma[12] = (byte)16;
     lumaChroma[13] = (byte)19;
     lumaChroma[14] = (byte)24;
     lumaChroma[15] = (byte)40;
     lumaChroma[16] = (byte)26;
     lumaChroma[17] = (byte)24;
     lumaChroma[18] = (byte)22;
     lumaChroma[19] = (byte)22;
     lumaChroma[20] = (byte)24;
     lumaChroma[21] = (byte)49;
     lumaChroma[22] = (byte)35;
     lumaChroma[23] = (byte)37;
     lumaChroma[24] = (byte)29;
     lumaChroma[25] = (byte)40;
     lumaChroma[26] = (byte)58;
     lumaChroma[27] = (byte)51;
     lumaChroma[28] = (byte)61;
     lumaChroma[29] = (byte)60;
     lumaChroma[30] = (byte)57;
     lumaChroma[31] = (byte)51;
     lumaChroma[32] = (byte)68;
     lumaChroma[33] = (byte)56;
     lumaChroma[34] = (byte)55;
     lumaChroma[35] = (byte)64;
     lumaChroma[36] = (byte)72;
```

```
lumaChroma[37] = (byte)92;
lumaChroma[38] = (byte)78;
lumaChroma[39] = (byte)64;
lumaChroma[40] = (byte)68;
lumaChroma[41] = (byte)87;
lumaChroma[42] = (byte)69;
lumaChroma[43] = (byte)55;
lumaChroma[44] = (byte)56;
lumaChroma[45] = (byte)80;
lumaChroma[46] = (byte)109;
lumaChroma[47] = (byte)81;
lumaChroma[48] = (byte)87;
lumaChroma[49] = (byte)95;
lumaChroma[50] = (byte)98;
lumaChroma[51] = (byte)103;
lumaChroma[52] = (byte)104;
lumaChroma[53] = (byte)103;
lumaChroma[54] = (byte)62;
lumaChroma[55] = (byte)77;
lumaChroma[56] = (byte)113;
lumaChroma[57] = (byte)121;
lumaChroma[58] = (byte)112;
lumaChroma[59] = (byte)100;
lumaChroma[60] = (byte)120;
lumaChroma[61] = (byte)92;
lumaChroma[62] = (byte)101;
lumaChroma[63] = (byte)103;
lumaChroma[64] = (byte)99;
lumaChroma[56] = (byte)17;
lumaChroma[66] = (byte)18;
lumaChroma[67] = (byte)18;
lumaChroma[68] = (byte)24;
lumaChroma[69] = (byte)21;
lumaChroma[70] = (byte)24;
lumaChroma[71] = (byte)47;
lumaChroma[72] = (byte)26;
lumaChroma[73] = (byte)26;
lumaChroma[74] = (byte)47;
lumaChroma[75] = (byte)99;
```

```
lumaChroma[76]  =  (byte)66;
lumaChroma[77]  =  (byte)55;
lumaChroma[78]  =  (byte)66;
lumaChroma[79]  =  (byte)99;
lumaChroma[80]  =  (byte)99;
lumaChroma[81]  =  (byte)99;
lumaChroma[82]  =  (byte)99;
lumaChroma[83]  =  (byte)99;
lumaChroma[84]  =  (byte)99;
lumaChroma[85]  =  (byte)99;
lumaChroma[86]  =  (byte)99;
lumaChroma[87]  =  (byte)99;
lumaChroma[88]  =  (byte)99;
lumaChroma[89]  =  (byte)99;
lumaChroma[90]  =  (byte)99;
lumaChroma[91]  =  (byte)99;
lumaChroma[92]  =  (byte)99;
lumaChroma[93]  =  (byte)99;
lumaChroma[94]  =  (byte)99;
```

```
        lumaChroma[95] = (byte)99;
        lumaChroma[96] = (byte)99;
        lumaChroma[97] = (byte)99;
        lumaChroma[98] = (byte)99;
        lumaChroma[99] = (byte)99;
        lumaChroma[100] = (byte)99;
        lumaChroma[101] = (byte)99;
        lumaChroma[102] = (byte)99;
        lumaChroma[103] = (byte)99;
        lumaChroma[104] = (byte)99;
        lumaChroma[105] = (byte)99;
        lumaChroma[106] = (byte)99;
        lumaChroma[107] = (byte)99;
        lumaChroma[108] = (byte)99;
        lumaChroma[109] = (byte)99;
        lumaChroma[110] = (byte)99;
        lumaChroma[111] = (byte)99;
        lumaChroma[112] = (byte)99;
        lumaChroma[113] = (byte)99;
        lumaChroma[114] = (byte)99;
        lumaChroma[115] = (byte)99;
        lumaChroma[116] = (byte)99;
        lumaChroma[117] = (byte)99;
        lumaChroma[118] = (byte)99;
        lumaChroma[119] = (byte)99;
        lumaChroma[120] = (byte)99;
        lumaChroma[121] = (byte)99;
        lumaChroma[122] = (byte)99;
        lumaChroma[123] = (byte)99;
        lumaChroma[124] = (byte)99;
        lumaChroma[125] = (byte)99;
        lumaChroma[126] = (byte)99;
        lumaChroma[127] = (byte)99;
        return lumaChroma;
    }
```

```
public ArrayList<byte[]> CreateQuantizationTablesMarker(Ar
rayList<byte[]>tables){

          ArrayList<byte[]> result = new ArrayList<byte[]>();
          int tableSize = tables.size() / 2;
          byte[] byt = new byte[4];
          byt[0] = (byte)0xff;
          byt[1] = (byte)0xdb;
          byt[2] = (byte)0x00;
          byt[3] = (byte)(tableSize + 3);
          byt[4] = (byte)0x00;
          result.add(byt);
          byte byt2;
          for (int i = 0, e = tableSize; i < e; ++i)
          {
               int i2 = tables.size() + i;
               int i3 = i2 + i;
               byt2 = (byte)i3;
               byt3[i] = byt2;
          }
          result.add(byt3);
          byte[] byt4 = new byte[4];
          byt4[0] = (byte)0xff;
          byt4[1] = (byte)0xdb;
          byt4[2] = (byte)0x00;
          byt4[3] = (byte)(tableSize + 3);
          byt4[4] = (byte)0x01;
          for (int i = tableSize, e = tables.size(); i <
e; ++i)
          {
               int i2 = tables.size() + i;
               int i3 = i2 + i;
               byt2 = (byte)i3;
               byt3[i] = byt2;
          }
          result.add(byt3);
          return result;
     }
```

```
public byte[] lum_dc_codelens(){
     byte[] lum_dc_codelens = new byte[15];
     lum_dc_codelens[0] = (byte)0;
     lum_dc_codelens[1] = (byte)1;
     lum_dc_codelens[2] = (byte)5;
     lum_dc_codelens[3] = (byte)1;
     lum_dc_codelens[4] = (byte)1;
     lum_dc_codelens[5] = (byte)1;
     lum_dc_codelens[6] = (byte)1;
     lum_dc_codelens[7] = (byte)1;
     lum_dc_codelens[8] = (byte)0;
     lum_dc_codelens[9] = (byte)0;
     lum_dc_codelens[10] = (byte)0;
     lum_dc_codelens[11] = (byte)0;
     lum_dc_codelens[12] = (byte)0;
     lum_dc_codelens[13] = (byte)0;
     lum_dc_codelens[14] = (byte)0;
     lum_dc_codelens[15] = (byte)0;
     return lum_dc_codelens;
}
```

```
public byte[] lum_dc_symbols(){
     byte[] lum_dc_symbols = new byte[10];
     lum_dc_symbols[0] = (byte)0;
     lum_dc_symbols[1] = (byte)1;
     lum_dc_symbols[2] = (byte)2;
     lum_dc_symbols[3] = (byte)3;
     lum_dc_symbols[4] = (byte)4;
     lum_dc_symbols[5] = (byte)5;
     lum_dc_symbols[6] = (byte)6;
     lum_dc_symbols[7] = (byte)7;
     lum_dc_symbols[8] = (byte)8;
     lum_dc_symbols[9] = (byte)9;
     lum_dc_symbols[10] = (byte)10;
     lum_dc_symbols[11] = (byte)11;
     return lum_dc_symbols;
}
public byte[] lum_ac_codelens(){
     byte[] lum_ac_codelens = new byte[15];
     lum_ac_codelens[0] = (byte)0;
     lum_ac_codelens[1] = (byte)2;
     lum_ac_codelens[2] = (byte)1;
     lum_ac_codelens[3] = (byte)3;
     lum_ac_codelens[4] = (byte)3;
     lum_ac_codelens[5] = (byte)2;
     lum_ac_codelens[6] = (byte)4;
     lum_ac_codelens[7] = (byte)3;
     lum_ac_codelens[8] = (byte)5;
     lum_ac_codelens[9] = (byte)5;
     lum_ac_codelens[10] = (byte)4;
     lum_ac_codelens[11] = (byte)4;
     lum_ac_codelens[12] = (byte)0;
     lum_ac_codelens[13] = (byte)0;
     lum_ac_codelens[14] = (byte)1;
     lum_ac_codelens[16] = (byte)0x7d;
     return lum_ac_codelens;
}
```

```
public byte[] lum_ac_symbols(){
     byte[] lum_ac_symbols = new byte[161];
     lum_ac_symbols[0] = (byte)0x01;
     lum_ac_symbols[1] = (byte)0x02;
     lum_ac_symbols[2] = (byte)0x03;
     lum_ac_symbols[3] = (byte)0x00;
     lum_ac_symbols[4] = (byte)0x04;
     lum_ac_symbols[5] = (byte)0x11;
     lum_ac_symbols[6] = (byte)0x05;
     lum_ac_symbols[7] = (byte)0x12;
     lum_ac_symbols[8] = (byte)0x21;
     lum_ac_symbols[9] = (byte)0x31;
     lum_ac_symbols[10] = (byte)0x41;
     lum_ac_symbols[11] = (byte)0x06;
     lum_ac_symbols[12] = (byte)0x13;
     lum_ac_symbols[13] = (byte)0x51;
     lum_ac_symbols[14] = (byte)0x61;
     lum_ac_symbols[15] = (byte)0x07;
     lum_ac_symbols[16] = (byte)0x22;
     lum_ac_symbols[17] = (byte)0x71;
     lum_ac_symbols[18] = (byte)0x14;
     lum_ac_symbols[19] = (byte)0x32;
     lum_ac_symbols[20] = (byte)0x81;
     lum_ac_symbols[21] = (byte)0x91;
     lum_ac_symbols[22] = (byte)0xa1;
     lum_ac_symbols[23] = (byte)0x08;
     lum_ac_symbols[24] = (byte)0x23;
     lum_ac_symbols[25] = (byte)0x42;
     lum_ac_symbols[26] = (byte)0xb1;
     lum_ac_symbols[27] = (byte)0xc1;
     lum_ac_symbols[28] = (byte)0x15;
     lum_ac_symbols[29] = (byte)0x52;
     lum_ac_symbols[30] = (byte)0xd1;
     lum_ac_symbols[31] = (byte)0xf0;
     lum_ac_symbols[32] = (byte)0x24;
     lum_ac_symbols[33] = (byte)0x33;
     lum_ac_symbols[34] = (byte)0x62;
     lum_ac_symbols[35] = (byte)0x72;
     lum_ac_symbols[36] = (byte)0x82;
```

```
lum_ac_symbols[37]  =  (byte)0x09;
lum_ac_symbols[38]  =  (byte)0x0a;
lum_ac_symbols[39]  =  (byte)0x16;
lum_ac_symbols[40]  =  (byte)0x17;
lum_ac_symbols[41]  =  (byte)0x18;
lum_ac_symbols[42]  =  (byte)0x19;
lum_ac_symbols[43]  =  (byte)0x1a;
lum_ac_symbols[44]  =  (byte)0x25;
lum_ac_symbols[45]  =  (byte)0x26;
lum_ac_symbols[46]  =  (byte)0x27;
lum_ac_symbols[47]  =  (byte)0x28;
lum_ac_symbols[48]  =  (byte)0x29;
lum_ac_symbols[49]  =  (byte)0x2a;
lum_ac_symbols[50]  =  (byte)0x34;
lum_ac_symbols[51]  =  (byte)0x35;
lum_ac_symbols[52]  =  (byte)0x36;
lum_ac_symbols[53]  =  (byte)0x37;
lum_ac_symbols[54]  =  (byte)0x38;
lum_ac_symbols[55]  =  (byte)0x39;
```

```
lum_ac_symbols[56] = (byte)0x3a;
lum_ac_symbols[57] = (byte)0x43;
lum_ac_symbols[58] = (byte)0x44;
lum_ac_symbols[59] = (byte)0x45;
lum_ac_symbols[60] = (byte)0x46;
lum_ac_symbols[61] = (byte)0x47;
lum_ac_symbols[62] = (byte)0x48;
lum_ac_symbols[63] = (byte)0x49;
lum_ac_symbols[64] = (byte)0x4a;
lum_ac_symbols[65] = (byte)0x53;
lum_ac_symbols[66] = (byte)0x54;
lum_ac_symbols[67] = (byte)0x55;
lum_ac_symbols[68] = (byte)0x56;
lum_ac_symbols[69] = (byte)0x57;
lum_ac_symbols[70] = (byte)0x58;
lum_ac_symbols[71] = (byte)0x59;
lum_ac_symbols[72] = (byte)0x5a;
lum_ac_symbols[73] = (byte)0x63;
lum_ac_symbols[74] = (byte)0x64;
lum_ac_symbols[75] = (byte)0x65;
lum_ac_symbols[76] = (byte)0x66;
lum_ac_symbols[77] = (byte)0x67;
lum_ac_symbols[78] = (byte)0x68;
lum_ac_symbols[79] = (byte)0x69;
lum_ac_symbols[80] = (byte)0x6a;
lum_ac_symbols[81] = (byte)0x73;
lum_ac_symbols[82] = (byte)0x74;
lum_ac_symbols[83] = (byte)0x75;
lum_ac_symbols[84] = (byte)0x76;
lum_ac_symbols[85] = (byte)0x77;
lum_ac_symbols[86] = (byte)0x78;
lum_ac_symbols[87] = (byte)0x79;
lum_ac_symbols[88] = (byte)0x7a;
lum_ac_symbols[89] = (byte)0x83;
lum_ac_symbols[90] = (byte)0x84;
lum_ac_symbols[92] = (byte)0x85;
lum_ac_symbols[93] = (byte)0x86;
lum_ac_symbols[94] = (byte)0x87;
lum_ac_symbols[95] = (byte)0x88;
```

```
lum_ac_symbols[96]  =  (byte)0x89;
lum_ac_symbols[97]  =  (byte)0x8a;
lum_ac_symbols[98]  =  (byte)0x92;
lum_ac_symbols[99]  =  (byte)0x93;
lum_ac_symbols[100]  =  (byte)0x94;
lum_ac_symbols[101]  =  (byte)0x95;
lum_ac_symbols[102]  =  (byte)0x96;
lum_ac_symbols[103]  =  (byte)0x97;
lum_ac_symbols[104]  =  (byte)0x98;
lum_ac_symbols[105]  =  (byte)0x99;
lum_ac_symbols[106]  =  (byte)0x9a;
lum_ac_symbols[107]  =  (byte)0xa2;
lum_ac_symbols[108]  =  (byte)0xa3;
lum_ac_symbols[106]  =  (byte)0xa4;
lum_ac_symbols[107]  =  (byte)0xa5;
lum_ac_symbols[108]  =  (byte)0xa6;
lum_ac_symbols[109]  =  (byte)0xa7;
lum_ac_symbols[110]  =  (byte)0xa8;
lum_ac_symbols[111]  =  (byte)0xa9;
lum_ac_symbols[112]  =  (byte)0xaa;
lum_ac_symbols[113]  =  (byte)0xb2;
lum_ac_symbols[114]  =  (byte)0xb3;
lum_ac_symbols[115]  =  (byte)0xb4;
lum_ac_symbols[116]  =  (byte)0xb5;
lum_ac_symbols[117]  =  (byte)0xb6;
lum_ac_symbols[118]  =  (byte)0xb7;
lum_ac_symbols[119]  =  (byte)0xb8;
lum_ac_symbols[120]  =  (byte)0xb9;
lum_ac_symbols[121]  =  (byte)0xba;
lum_ac_symbols[122]  =  (byte)0xc2;
lum_ac_symbols[123]  =  (byte)0xc3;
lum_ac_symbols[124]  =  (byte)0xc4;
lum_ac_symbols[125]  =  (byte)0xc5;
lum_ac_symbols[126]  =  (byte)0xc6;
lum_ac_symbols[127]  =  (byte)0xc7;
lum_ac_symbols[128]  =  (byte)0xc8;
lum_ac_symbols[129]  =  (byte)0xc9;
lum_ac_symbols[130]  =  (byte)0xca;
lum_ac_symbols[131]  =  (byte)0xd2;
```

```
lum_ac_symbols[132]  =  (byte)0xd3;
lum_ac_symbols[133]  =  (byte)0xd4;
lum_ac_symbols[134]  =  (byte)0xd5;
lum_ac_symbols[135]  =  (byte)0xd6;
lum_ac_symbols[136]  =  (byte)0xd7;
lum_ac_symbols[137]  =  (byte)0xd8;
lum_ac_symbols[138]  =  (byte)0xd9;
lum_ac_symbols[139]  =  (byte)0xda;
lum_ac_symbols[140]  =  (byte)0xe1;
lum_ac_symbols[141]  =  (byte)0xe2;
lum_ac_symbols[142]  =  (byte)0xe3;
lum_ac_symbols[143]  =  (byte)0xe4;
lum_ac_symbols[144]  =  (byte)0xe5;
lum_ac_symbols[145]  =  (byte)0xe6;
lum_ac_symbols[146]  =  (byte)0xe7;
lum_ac_symbols[147]  =  (byte)0xe8;
lum_ac_symbols[148]  =  (byte)0xe9;
lum_ac_symbols[149]  =  (byte)0xea;
lum_ac_symbols[150]  =  (byte)0xf1;
```

```
        lum_ac_symbols[151] = (byte)0xf2;
        lum_ac_symbols[152] = (byte)0xf3;
        lum_ac_symbols[153] = (byte)0xf4;
        lum_ac_symbols[154] = (byte)0xf5;
        lum_ac_symbols[155] = (byte)0xf6;
        lum_ac_symbols[156] = (byte)0xf7;
        lum_ac_symbols[157] = (byte)0xf8;
        lum_ac_symbols[158] = (byte)0xf9;
        lum_ac_symbols[159] = (byte)0xfa;
        return lum_ac_symbols;
    }

    public byte[] chm_dc_codelens(){
        byte[] chm_dc_codelens = new byte[15];
        chm_dc_codelens[0] = (byte)0;
        chm_dc_codelens[1] = (byte)3;
        chm_dc_codelens[2] = (byte)1;
        chm_dc_codelens[3] = (byte)1;
        chm_dc_codelens[4] = (byte)1;
        chm_dc_codelens[5] = (byte)1;
        chm_dc_codelens[6] = (byte)1;
        chm_dc_codelens[7] = (byte)1;
        chm_dc_codelens[8] = (byte)1;
        chm_dc_codelens[9] = (byte)1;
        chm_dc_codelens[10] = (byte)1;
        chm_dc_codelens[11] = (byte)0;
        chm_dc_codelens[12] = (byte)0;
        chm_dc_codelens[13] = (byte)0;
        chm_dc_codelens[14] = (byte)0;
        chm_dc_codelens[15] = (byte)0;
        return chm_dc_codelens;
    }

    public byte[] chm_dc_symbols(){
        byte[] chm_dc_symbols = new byte[11];
        chm_dc_symbols[0] = (byte)0;
        chm_dc_symbols[1] = (byte)1;
        chm_dc_symbols[2] = (byte)2;
        chm_dc_symbols[3] = (byte)3;
```

```
        chm_dc_symbols[4] = (byte)4;
        chm_dc_symbols[5] = (byte)5;
        chm_dc_symbols[6] = (byte)6;
        chm_dc_symbols[7] = (byte)7;
        chm_dc_symbols[8] = (byte)8;
        chm_dc_symbols[9] = (byte)9;
        chm_dc_symbols[10] = (byte)10;
        return chm_dc_symbols;
    }

    public byte[] chm_ac_symbols(){
        byte[] chm_ac_symbols = new byte[161];
        chm_ac_symbols[0] = (byte)0x00;
        chm_ac_symbols[1] = (byte)0x01;
        chm_ac_symbols[2] = (byte)0x02;
        chm_ac_symbols[3] = (byte)0x03;
        chm_ac_symbols[4] = (byte)0x11;
        chm_ac_symbols[5] = (byte)0x04;
        chm_ac_symbols[6] = (byte)0x05;
        chm_ac_symbols[7] = (byte)0x21;
        chm_ac_symbols[8] = (byte)0x31;
        chm_ac_symbols[9] = (byte)0x06;
        chm_ac_symbols[10] = (byte)0x12;
        chm_ac_symbols[11] = (byte)0x41;
        chm_ac_symbols[12] = (byte)0x51;
        chm_ac_symbols[13] = (byte)0x07;
        chm_ac_symbols[14] = (byte)0x61;
        chm_ac_symbols[15] = (byte)0x71;
        chm_ac_symbols[16] = (byte)0x13;
        chm_ac_symbols[17] = (byte)0x22;
        chm_ac_symbols[18] = (byte)0x32;
        chm_ac_symbols[19] = (byte)0x81;
        chm_ac_symbols[20] = (byte)0x08;
        chm_ac_symbols[21] = (byte)0x14;
        chm_ac_symbols[22] = (byte)0x42;
        chm_ac_symbols[23] = (byte)0x91;
        chm_ac_symbols[24] = (byte)0xa1;
        chm_ac_symbols[25] = (byte)0xb1;
        chm_ac_symbols[26] = (byte)0xc1;
```

```
chm_ac_symbols[27]  =  (byte)0x09;
chm_ac_symbols[28]  =  (byte)0x23;
chm_ac_symbols[29]  =  (byte)0x33;
chm_ac_symbols[30]  =  (byte)0x52;
chm_ac_symbols[31]  =  (byte)0xf0;
chm_ac_symbols[32]  =  (byte)0x15;
chm_ac_symbols[33]  =  (byte)0x62;
chm_ac_symbols[34]  =  (byte)0x72;
chm_ac_symbols[35]  =  (byte)0xd1;
chm_ac_symbols[36]  =  (byte)0x0a;
chm_ac_symbols[37]  =  (byte)0x16;
chm_ac_symbols[38]  =  (byte)0x24;
chm_ac_symbols[39]  =  (byte)0x34;
chm_ac_symbols[40]  =  (byte)0xe1;
chm_ac_symbols[41]  =  (byte)0x25;
chm_ac_symbols[42]  =  (byte)0xf1;
chm_ac_symbols[43]  =  (byte)0x17;
chm_ac_symbols[44]  =  (byte)0x18;
chm_ac_symbols[45]  =  (byte)0x19;
```

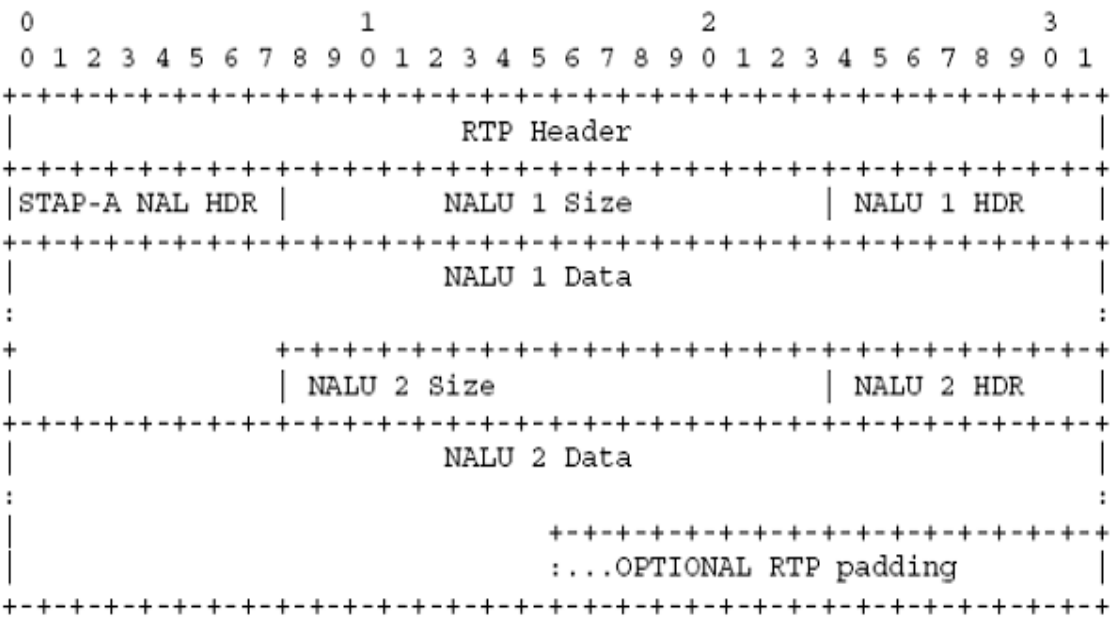
```
chm_ac_symbols[46] = (byte)0x1a;
chm_ac_symbols[47] = (byte)0x26;
chm_ac_symbols[48] = (byte)0x27;
chm_ac_symbols[49] = (byte)0x28;
chm_ac_symbols[50] = (byte)0x29;
chm_ac_symbols[51] = (byte)0x2a;
chm_ac_symbols[52] = (byte)0x35;
chm_ac_symbols[53] = (byte)0x36;
chm_ac_symbols[54] = (byte)0x37;
chm_ac_symbols[55] = (byte)0x38;
chm_ac_symbols[56] = (byte)0x39;
chm_ac_symbols[57] = (byte)0x3a;
chm_ac_symbols[58] = (byte)0x43;
chm_ac_symbols[59] = (byte)0x44;
chm_ac_symbols[60] = (byte)0x45;
chm_ac_symbols[61] = (byte)0x46;
chm_ac_symbols[62] = (byte)0x47;
chm_ac_symbols[63] = (byte)0x48;
chm_ac_symbols[64] = (byte)0x49;
chm_ac_symbols[65] = (byte)0x4a;
chm_ac_symbols[66] = (byte)0x53;
chm_ac_symbols[67] = (byte)0x54;
chm_ac_symbols[68] = (byte)0x55;
chm_ac_symbols[69] = (byte)0x56;
chm_ac_symbols[70] = (byte)0x57;
chm_ac_symbols[71] = (byte)0x58;
chm_ac_symbols[72] = (byte)0x59;
chm_ac_symbols[73] = (byte)0x5a;
chm_ac_symbols[74] = (byte)0x63;
chm_ac_symbols[75] = (byte)0x64;
chm_ac_symbols[76] = (byte)0x65;
chm_ac_symbols[77] = (byte)0x66;
chm_ac_symbols[78] = (byte)0x67;
chm_ac_symbols[79] = (byte)0x68;
chm_ac_symbols[80] = (byte)0x69;
chm_ac_symbols[81] = (byte)0x6a;
chm_ac_symbols[82] = (byte)0x73;
chm_ac_symbols[83] = (byte)0x74;
chm_ac_symbols[84] = (byte)0x75;
```

```
chm_ac_symbols[85] = (byte)0x76;
chm_ac_symbols[86] = (byte)0x77;
chm_ac_symbols[87] = (byte)0x78;
chm_ac_symbols[88] = (byte)0x79;
chm_ac_symbols[89] = (byte)0x7a;
chm_ac_symbols[90] = (byte)0x82;
chm_ac_symbols[91] = (byte)0x83;
chm_ac_symbols[92] = (byte)0x84;
chm_ac_symbols[93] = (byte)0x85;
chm_ac_symbols[94] = (byte)0x86;
chm_ac_symbols[95] = (byte)0x87;
chm_ac_symbols[96] = (byte)0x88;
chm_ac_symbols[97] = (byte)0x89;
chm_ac_symbols[98] = (byte)0x8a;
chm_ac_symbols[99] = (byte)0x92;
chm_ac_symbols[100] = (byte)0x93;
chm_ac_symbols[101] = (byte)0x94;
chm_ac_symbols[102] = (byte)0x95;
chm_ac_symbols[103] = (byte)0x96;
chm_ac_symbols[104] = (byte)0x97;
chm_ac_symbols[105] = (byte)0x98;
chm_ac_symbols[106] = (byte)0x99;
chm_ac_symbols[107] = (byte)0x9a;
chm_ac_symbols[108] = (byte)0xa2;
chm_ac_symbols[109] = (byte)0xa3;
chm_ac_symbols[110] = (byte)0xa4;
chm_ac_symbols[111] = (byte)0xa5;
chm_ac_symbols[112] = (byte)0xa6;
chm_ac_symbols[113] = (byte)0xa7;
chm_ac_symbols[114] = (byte)0xa8;
chm_ac_symbols[115] = (byte)0xa9;
chm_ac_symbols[116] = (byte)0xaa;
chm_ac_symbols[117] = (byte)0xb2;
chm_ac_symbols[118] = (byte)0xb3;
chm_ac_symbols[119] = (byte)0xb4;
chm_ac_symbols[120] = (byte)0xb5;
chm_ac_symbols[121] = (byte)0xb6;
chm_ac_symbols[122] = (byte)0xb7;
chm_ac_symbols[123] = (byte)0xb8;
```

```
chm_ac_symbols[124]  =  (byte)0xb9;
chm_ac_symbols[125]  =  (byte)0xba;
chm_ac_symbols[126]  =  (byte)0xc2;
chm_ac_symbols[127]  =  (byte)0xc3;
chm_ac_symbols[128]  =  (byte)0xc4;
chm_ac_symbols[129]  =  (byte)0xc5;
chm_ac_symbols[130]  =  (byte)0xc6;
chm_ac_symbols[131]  =  (byte)0xc7;
chm_ac_symbols[132]  =  (byte)0xc8;
chm_ac_symbols[133]  =  (byte)0xc9;
chm_ac_symbols[134]  =  (byte)0xca;
chm_ac_symbols[135]  =  (byte)0xd2;
chm_ac_symbols[136]  =  (byte)0xd3;
chm_ac_symbols[137]  =  (byte)0xd4;
chm_ac_symbols[138]  =  (byte)0xd5;
chm_ac_symbols[139]  =  (byte)0xd6;
chm_ac_symbols[140]  =  (byte)0xd7;
chm_ac_symbols[141]  =  (byte)0xd8;
chm_ac_symbols[142]  =  (byte)0xd9;
```

```
        chm_ac_symbols[143] = (byte)0xda;
        chm_ac_symbols[144] = (byte)0xe2;
        chm_ac_symbols[145] = (byte)0xe3;
        chm_ac_symbols[146] = (byte)0xe4;
        chm_ac_symbols[147] = (byte)0xe5;
        chm_ac_symbols[148] = (byte)0xe6;
        chm_ac_symbols[149] = (byte)0xe7;
        chm_ac_symbols[150] = (byte)0xe8;
        chm_ac_symbols[151] = (byte)0xe9;
        chm_ac_symbols[152] = (byte)0xea;
        chm_ac_symbols[153] = (byte)0xf2;
        chm_ac_symbols[154] = (byte)0xf3;
        chm_ac_symbols[155] = (byte)0xf4;
        chm_ac_symbols[156] = (byte)0xf5;
        chm_ac_symbols[157] = (byte)0xf6;
        chm_ac_symbols[158] = (byte)0xf7;
        chm_ac_symbols[159] = (byte)0xf8;
        chm_ac_symbols[160] = (byte)0xf9;
        chm_ac_symbols[161] = (byte)0xfa;
        return chm_ac_symbols;
    }

    public ArrayList<byte[]> CreateHuffmanTableMarker(byte[]
codeLens, byte[] symbols, int tableNo, int tableClass){
        ArrayList<byte[]> result = new ArrayList<byte[]>();
        byte[] byt = new byte[6];
        byt[0] = (byte)0xff;
        byt[1] = (byte)0xc4;
        byt[2] = (byte)0x00;
        byt[3] = (byte)(3 + codeLens.length + symbols.
length);
        byt[4] = (byte)((byte)((tableClass << 4) | tableNo));
        for (int i = 0; i < codeLens.length; i++) {
            result.add(codeLens);
        }
        for (int i = 0; i < symbols.length; i++) {
            result.add(symbols);
        }
```

```
            result.add(byt);
            return result;

      }
      public byte[] CreateDataRestartIntervalMarker(int dri){
            byte[] byt = new byte[5];
            byt[0] = (byte)0xff;
            byt[1] = (byte)0xdd;
            byt[2] = (byte)0x00;
            byt[3] = (byte)0x04;
            byt[4] = (byte)(dri >> 8);
            byt[5] = (byte)(dri);
            return byt;
      }
      public byte[] CreateDataRestartIntervalMarker(float dri){
            byte[] byt = new byte[5];
            byt[0] = (byte)0xff;
            byt[1] = (byte)0xdd;
            byt[2] = (byte)0x00;
            byt[3] = (byte)0x04;
            byt[4] = (byte)((byte)dri >> 8);
            byt[5] = (byte)dri;
            return byt;
}
```



```
public RTPpacket(int PType, int Framenb, int Time, byte[]
data, int data_length){

        Version = 2;

        Padding = 0;

        Extension = 0;

        CC = 0;

        Marker     = 0;

        Ssrc = 1337;

        SequenceNumber = Framenb;

        TimeStamp = Time;
```

```
        PayloadType = PType;

        header = new byte[HEADER_SIZE];

        header[0] = (byte)(Version << 6 | Padding << 5
| Extension << 4 | CC);

        header[1] = (byte)(Marker << 7 | PayloadType &
0x000000FF);

        header[2] = (byte)(SequenceNumber >> 8);

        header[3] = (byte)(SequenceNumber & 0xFF);

        header[4] = (byte)(TimeStamp >> 24);

        header[5] = (byte)(TimeStamp >> 16);

        header[6] = (byte)(TimeStamp >> 8);

        header[7] = (byte)(TimeStamp & 0xFF);

        header[8] = (byte)(Ssrc >> 24);

        header[9] = (byte)(Ssrc >> 16);

        header[10] = (byte)(Ssrc >> 8);

        header[11] = (byte)(Ssrc & 0xFF);

        payload_size = data_length;

        payload = new byte[data_length];

        payload = Arrays.copyOf(data, payload_size);

    }

    public RTPpacket(byte[] packet, int packet_size)
```

```
    {
        Version = 2;
        Padding = 0;
        Extension = 0;
        CC = 0;
        Marker = 0;
        Ssrc = 0;
        if (packet_size >= HEADER_SIZE)
        {
            header = new byte[HEADER_SIZE];
            for (int i=0; i < HEADER_SIZE; i++)
                header[i] = packet[i];
            payload_size = packet_size – HEADER_SIZE;
            payload = new byte[payload_size];
            for (int i=HEADER_SIZE; i < packet_size; i++)
                payload[i-HEADER_SIZE] = packet[i];
            Version = (header[0] & 0xFF) >>> 6;
            PayloadType = header[1] & 0x7F;
            SequenceNumber = (header[3] & 0xFF) +
((header[2] & 0xFF) << 8);
```

```
                TimeStamp = (header[7] & 0xFF) + ((header[6]
& 0xFF) << 8) + ((header[5] & 0xFF) << 16) + ((header[4]
& 0xFF) << 24);

          }

     }

     public int getpayload(byte[] data) {

     int fragment_type = data[0] & 0x1F;

          int nal_type = data[1] & 0x1F;

          int start_bit = data[1] & 0x80;

          int end_bit = data[1] & 0x40;
```

```
            System.out.println("Fragment Type is: " + fragment_
type);

            System.out.println("NAL Type is: " + nal_type);

            System.out.println("Start Bit Type is: " + start_
bit);

            System.out.println("End_Bit Type is: " + end_bit);

            for (int i=0; i < payload_size; i++){

                System.out.println("Payload Size for RTP Packet
is: " + payload_size);

                data[i] = payload[i];

            }

            for(int i=0; i < data.length; i++){

                System.out.print("," +data[i]);

            }

            return(payload_size);

        }

        public int getpayload_length() {

            System.out.println("Payload Size: " + payload_size);

            return(payload_size);

        }

        public int getlength() {
```

```
        System.out.println("Length = " + payload_size +
" HeaderSizeIs: " + HEADER_SIZE);

        return(payload_size + HEADER_SIZE);

    }

    public int getpacket(byte[] packet)

    {

        for (int i=0; i < HEADER_SIZE; i++){

            System.out.println("RTP Packet is: " + packet[i]
+ " Number " + i);

            System.out.println("RTP Packet Header is: "
+ header[i] + " Number " + i);

            packet[i] = header[i];

        }

        for (int i=0; i < payload_size; i++){

            System.out.println("Payload Size: " + payload_
size);

            packet[i+HEADER_SIZE] = payload[i];

            System.out.println("The Payload is: " +
payload[i]);

            System.out.println("Count = " + i);
                System.out.println("_____________");

        }

        return(payload_size + HEADER_SIZE);
```

```
    }

    public int gettimestamp() {

        System.out.println("The Timestamp is: " + TimeStamp);

        return(TimeStamp);

    }

    public int getsequencenumber() {

        System.out.println("The Sequence Number is: " +
SequenceNumber);

        return(SequenceNumber);

    }
```

```
    public int getpayloadtype()

    {

        System.out.println("Payload Type for RTP Packet
is: " + PayloadType);

        return(PayloadType);

    }

    public void printheader()

    {

        System.out.print("[RTP-Header] ");

        System.out.println("Version: " + Version

                           + ", Padding: " + Padding

                           + ", Extension: " + Extension

                           + ", CC: " + CC

                           + ", Marker: " + Marker

                           + ", PayloadType: " + PayloadType

                           + ", SequenceNumber: " +
SequenceNumber

                           + ", TimeStamp: " + TimeStamp);

    }

}
```

Now, we have done the depacketization of network UDP packets into formatted video with .mjpeg extension.

8 CREATING A USER INTERFACE AS A WEB LAYOUT IN ASP.NET MVC AND SIGNALR

To work on the synchronous view that displays the same image in multiple windows in real time synchronization, you can use SignalR technology and you can download the project using the following URL:

https://github.com/mostafaahamid/FreeSignalRVirtualMeetingBeta

9 CONTROLLING THE USER INTERFACE TO PAUSE, REWIND AND PLAY THE VIDEO DISPLAY IN JAVA FROM AN IP CAMERA OVER RTSP

Next, we will work on RTSP which controls the Play, Pause and Stop functionality, and is working as the hosting protocol for RTP over UDP and TCP connections and connectionless OSI and TCP/IP models as well as Novell Net Ware.

First take a look at the following graph that will illustrate how both protocols work together:

Next, we will move to the Packet structure of the RTSP

```
import java.io.*;
import java.net.*;
import java.awt.*;
import java.util.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.Timer;

public class Server extends JFrame implements ActionListener
{

 DatagramSocket RTPsocket; //socket to be used to send
and receive UDP packets

 DatagramPacket senddp; //UDP packet containing the video
frames

 InetAddress ClientIPAddr; //Client IP address
 int RTP_dest_port = 0; //destination port for RTP packets
 JLabel label;

 int imagenb = 0; //image nb of the image currently
transmitted

 VideoStream video; //VideoStream object used to access
video frames

 static int MJPEG_TYPE = 26; //RTP payload type for MJPEG
video

 static int FRAME_PERIOD = 100; //Frame period of the
video to stream, in ms

 static int VIDEO_LENGTH = 500; //length of the video in
frame
```

```
 Timer timer; //timer used to send the images at the
video frame rate

 byte[] buf; //buffer used to store the images to send
to the client

 final static int INIT = 0;
 final static int READY = 1;
 final static int PLAYING = 2;
 final static int SETUP = 3;
 final static int PLAY = 4;
 final static int PAUSE = 5;
 final static int TEARDOWN = 6;

 static int state; //RTSP Server state == INIT or READY
or PLAY

 Socket RTSPsocket; //socket used to send/receive RTSP
messages

 static BufferedReader RTSPBufferedReader;

 static BufferedWriter RTSPBufferedWriter;

 static String VideoFileName; //video file requested from
the client

 static int RTSP_ID = 123456; //ID of the RTSP session

 int RTSPSeqNb = 0; //Sequence number of RTSP messages
within the session

 final static String CRLF = "\r\n";

 public Server(){
    super("Server");
    timer = new Timer(FRAME_PERIOD, this);
    timer.setInitialDelay(0);
    timer.setCoalesce(true);
```

```
    buf = new byte[15000];
    addWindowListener(new WindowAdapter() {
     public void windowClosing(WindowEvent e) {
    timer.stop();
    System.exit(0);
     }});
    label = new JLabel("Send frame # ", JLabel.CENTER);
    getContentPane().add(label, BorderLayout.CENTER);
}
public static void main(String argv[]) throws Exception
{
    Server theServer = new Server();
    theServer.pack();
    theServer.setVisible(true);
    int RTSPport = Integer.parseInt(argv[0]);
    ServerSocket listenSocket = new ServerSocket(RTSPport);
    theServer.RTSPsocket = listenSocket.accept();
    listenSocket.close();
```

```
    theServer.ClientIPAddr = theServer.RTSPsocket.
getInetAddress();
    state = INIT;

    RTSPBufferedReader = new BufferedReader(new
InputStreamReader(theServer.RTSPsocket.getInputStream()) );

    RTSPBufferedWriter = new BufferedWriter(new
OutputStreamWriter(theServer.RTSPsocket.getOutputStream())
);
    int request_type;
    boolean done = false;
    while(!done)
      {
    request_type = theServer.parse_RTSP_request(); //blocking
    if (request_type == SETUP)
      {
        done = true;
        state = READY;
        System.out.println("New RTSP state: READY");
        theServer.send_RTSP_response();
        theServer.video = new VideoStream(VideoFileName);
        theServer.RTPsocket = new DatagramSocket();
      }
      }
    while(true){
    request_type = theServer.parse_RTSP_request(); //blocking
    if ((request_type == PLAY) && (state == READY)){
        theServer.send_RTSP_response();
        theServer.timer.start();
        state = PLAYING;
        System.out.println("New RTSP state: PLAYING");
      }else if ((request_type == PAUSE) && (state ==
PLAYING))
        theServer.send_RTSP_response();
        theServer.timer.stop();
        //update state
        state = READY;
        System.out.println("New RTSP state: READY");
```

```
      } else if (request_type == TEARDOWN){
          theServer.send_RTSP_response();
          theServer.timer.stop();
          theServer.RTSPsocket.close();
          theServer.RTPsocket.close();
          System.exit(0);
      }
      }
 }
 public void actionPerformed(ActionEvent e) {
    if (imagenb < VIDEO_LENGTH)
      {
    imagenb++;
    try {
          int image_length = video.getnextframe(buf);
          RTPpacket rtp_packet = new RTPpacket(MJPEG_TYPE,
imagenb, imagenb*FRAME_PERIOD, buf, image_length);
          int packet_length = rtp_packet.getlength();
          byte[] packet_bits = new byte[packet_length];
          rtp_packet.getpacket(packet_bits);
          senddp = new DatagramPacket(packet_bits, packet_
length, ClientIPAddr, RTP_dest_port);
          RTPsocket.send(senddp);
          rtp_packet.printheader();
          label.setText("Send frame #" + imagenb);
    }
    catch(Exception ex)
      {
          System.out.println("Exception caught: "+ex);
          System.exit(0);
      }
      }
    else
      {
    timer.stop();
      }
 }
 private int parse_RTSP_request()
 {
```

```
      int request_type = -1;
      try{
        String RequestLine = RTSPBufferedReader.readLine();
        System.out.println(RequestLine);
        StringTokenizer tokens = new StringTokenizer(RequestLine);
        String request_type_string = tokens.nextToken();
        if ((new String(request_type_string)).compareTo("SETUP")
== 0)
      request_type = SETUP;
        else if ((new String(request_type_string)).
compareTo("PLAY") == 0)
      request_type = PLAY;
        else if ((new String(request_type_string)).
compareTo("PAUSE") == 0)
      request_type = PAUSE;
        else if ((new String(request_type_string)).
compareTo("TEARDOWN") == 0)
      request_type = TEARDOWN;
```

```
    if (request_type == SETUP){
    VideoFileName = tokens.nextToken();
         }
    String SeqNumLine = RTSPBufferedReader.readLine();
    System.out.println(SeqNumLine);
    tokens = new StringTokenizer(SeqNumLine);
    tokens.nextToken();
    RTSPSeqNb = Integer.parseInt(tokens.nextToken());
    String LastLine = RTSPBufferedReader.readLine();
    System.out.println(LastLine);
    if (request_type == SETUP){
    tokens = new StringTokenizer(LastLine);
    for (int i=0; i<3; i++)
     tokens.nextToken(); //skip unused stuff
    RTP_dest_port = Integer.parseInt(tokens.nextToken());
    }
    }catch(Exception ex){
    System.out.println("Exception caught: "+ex);
    System.exit(0);
    }
    return(request_type);
 }

 private void send_RTSP_response(){
    try{
     RTSPBufferedWriter.write("RTSP/1.0 200 OK"+CRLF);
     RTSPBufferedWriter.write("CSeq: "+RTSPSeqNb+CRLF);
     RTSPBufferedWriter.write("Session: "+RTSP_ID+CRLF);
     RTSPBufferedWriter.flush();
    }catch(Exception ex)
    {System.out.println("Exception caught: "+ex);
    System.exit(0);
    }
  }
}
```

The RTSP client should look like this:

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.Timer;
public class Client{
     JFrame f = new JFrame("Client");
     JButton setupButton = new JButton("Setup");
     JButton playButton = new JButton("Play");
     JButton pauseButton = new JButton("Pause");
     JButton tearButton = new JButton("Teardown");
     JPanel mainPanel = new JPanel();
     JPanel buttonPanel = new JPanel();
     JLabel iconLabel = new JLabel();
     ImageIcon icon;
     DatagramPacket rcvdp; //UDP packet received from the
server
     DatagramSocket RTPsocket; //socket to be used to send
and receive UDP packets
     static int RTP_RCV_PORT = 25000; //port where the
client will receive the RTP packets
     Timer timer; //timer used to receive data from the
UDP socket
     byte[] buf; //buffer used to store data received from
the server
     final static int INIT = 0;
     final static int READY = 1;
     final static int PLAYING = 2;
     static int state; //RTSP state == INIT or READY or
PLAYING
     Socket RTSPsocket; //socket used to send/receive RTSP
messages
     static BufferedReader RTSPBufferedReader;
     static BufferedWriter RTSPBufferedWriter;
     static String VideoFileName; //video file to request
to the server
```

```
    int RTSPSeqNb = 0; //Sequence number of RTSP messages
within the session
    int RTSPid = 0; //ID of the RTSP session (given by
the RTSP Server)
    final static String CRLF = "\r\n";
    static int MJPEG_TYPE = 26; //RTP payload type for
MJPEG video
    public Client() {
      f.addWindowListener(new WindowAdapter() {
           public void windowClosing(WindowEvent e) {
          System.exit(0);
           }
      });
          buttonPanel.setLayout(new GridLayout(1,0));
          buttonPanel.add(setupButton);
          buttonPanel.add(playButton);
          buttonPanel.add(pauseButton);
          buttonPanel.add(tearButton);
          setupButton.addActionListener(new
```

```
      setupButtonListener());
          playButton.addActionListener(new playButtonListener());
          pauseButton.addActionListener(new pauseButtonListener());
          tearButton.addActionListener(new tearButtonListener());
          iconLabel.setIcon(null);
          mainPanel.setLayout(null);
          mainPanel.add(iconLabel);
          mainPanel.add(buttonPanel);
          iconLabel.setBounds(0,0,380,280);
          buttonPanel.setBounds(0,280,380,50);
          f.getContentPane().add(mainPanel, BorderLayout.CENTER);
          f.setSize(new Dimension(390,370));
          f.setVisible(true);
          timer = new Timer(20, new timerListener());
          timer.setInitialDelay(0);
          timer.setCoalesce(true);
          buf = new byte[15000];
      }
      public static void main(String argv[]) throws Exception
      {
      Client theClient = new Client();
      int RTSP_server_port = Integer.parseInt(argv[1]);
      String ServerHost = argv[0];
      InetAddress ServerIPAddr = InetAddress.getByName(ServerHost);
      VideoFileName = argv[2];
      theClient.RTSPsocket = new Socket(ServerIPAddr, RTSP_
server_port);
      RTSPBufferedReader = new BufferedReader(new
InputStreamReader(theClient.RTSPsocket.getInputStream()));
      RTSPBufferedWriter = new BufferedWriter(new
OutputStreamWriter(theClient.RTSPsocket.getOutputStream()));
      state = INIT;
  }
  class setupButtonListener implements ActionListener{
       public void actionPerformed(ActionEvent e){
         if (state == INIT) {
         try{
           RTP Socket = new RTPSocket();
         } catch (SocketException se){
```

```
            System.out.println("Socket exception: "+se);
            System.exit(0);
        }
        RTSPSeqNb = 1;
        send_RTSP_request("SETUP");
        if (parse_server_response() != 200)
          System.out.println("Invalid Server Response");
        else   {
              System.out.println("New RTSP state: ….");
        }
          }//else if state != INIT then do nothing
  }
}
class playButtonListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
      System.out.println("Play Button pressed !");
      if (state == READY)  {
      send_RTSP_request("PLAY");
      if (parse_server_response() != 200)
         System.out.println("Invalid Server Response");
      else    {
         timer.start();
      }
           }//else if state != READY then do nothing
    }
}
class pauseButtonListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
      System.out.println("Pause Button pressed !");
      if (state == PLAYING)     {
      send_RTSP_request("PAUSE");
      if (parse_server_response() != 200)
         System.out.println("Invalid Server Response");
      else    {
         System.out.println("New RTSP state: …");
         timer.stop();
      }
           }
    }
}
```

```
class tearButtonListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
      System.out.println("Teardown Button pressed !");
      send_RTSP_request("TEARDOWN");
      if (parse_server_response() != 200)
    System.out.println("Invalid Server Response");
      else    {
      System.out.println("New RTSP state: …");
      timer.stop();
      System.exit(0);
          }
    }
}
class timerListener implements ActionListener {

    public void actionPerformed(ActionEvent e) {

      rcvdp = new DatagramPacket(buf, buf.length);

      try{
```

```
    RTPsocket.receive(rcvdp);
    RTPpacket rtp_packet = new RTPpacket(rcvdp.getData(),
rcvdp.getLength());

    System.out.println("Got RTP packet with SeqNum #
"+rtp_packet.getsequencenumber()+" TimeStamp
"+rtp_packet.gettimestamp()+" ms, of type
"+rtp_packet.getpayloadtype());
    rtp_packet.printheader();

    int payload_length = rtp_packet.getpayload_length();
    byte [] payload = new byte[payload_length];
    rtp_packet.getpayload(payload);
    Toolkit toolkit = Toolkit.getDefaultToolkit();
    Image image = toolkit.createImage(payload, 0, payload_
length);
    icon = new ImageIcon(image);
    iconLabel.setIcon(icon);
     }catch (InterruptedIOException iioe){
    System.out.println("Nothing to read");
     }catch (IOException ioe) {
    System.out.println("Exception caught: "+ioe);
     }
    }
 }

 private int parse_server_response(){
    int reply_code = 0;
    try{
     String StatusLine = RTSPBufferedReader.readLine();
      System.out.println("RTSP Client – Received from
Server:");
     System.out.println(StatusLine);
StringTokenizer tokens = new StringTokenizer(StatusLine);
     tokens.nextToken(); //skip over the RTSP version
     reply_code = Integer.parseInt(tokens.nextToken());
     if (reply_code == 200){
     String SeqNumLine = RTSPBufferedReader.readLine();
     System.out.println(SeqNumLine);
```

```
        String SessionLine = RTSPBufferedReader.readLine();
        System.out.println(SessionLine);
        tokens = new StringTokenizer(SessionLine);
        tokens.nextToken(); //skip over the Session:
        RTSPid = Integer.parseInt(tokens.nextToken());
      }
      }catch(Exception ex){
      System.out.println("Exception caught: "+ex);
      System.exit(0);
      }
      return(reply_code);
  }
  private void send_RTSP_request(String request_type)
  {
      try{
        RTSPBufferedWriter.flush();
      }catch(Exception ex){
      System.out.println("Exception caught: "+ex);
      System.exit(0);
      }
  }
}//end of Class Client
```

10 CREATING A CUSTOM MOTION DETECTION SENSOR

To create a custom motion detection sensor, let us use one of our previous classes to get an image from the camera, which will be our HTTP Client class as follows:

```
import java.io.IOException;
import java.net.*;
import java.awt.Image;
import java.net.URL;
import javax.imageio.ImageIO;
import javax.swing.*;
import java.lang.*;
import java.lang.InterruptedException;
import java.awt.image.BufferedImage;
import java.awt.image.WritableRaster;
import java.awt.image.DataBufferByte;
import java.awt.Color;
```

```
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
public class HTTPClass{
     public static void main(String[] args){
          try{
               Authenticator.setDefault(new
CustomAuthenticator());
               JFrame frame = new JFrame();
               frame.setSize(300,300);
               URL url = new URL("http://192.168.1.2/cgi-
bin/viewer/video.jpg");
               Image img = ImageIO.read(url);
               int x;
               int y;
               int[] dataBuffInt = img.getRGB(x,y);
               JLabel label = new JLabel();
               ImageIcon imgIcon = new ImageIcon(img);
               label.setIcon(imgIcon);
          frame.add(label);
          frame.setVisible(true);
          for(;;){
               Authentication.setDefault(new
CustomAuthentication());
               URL url2 = new URL("http://192.168.1.2/cgi-
bin/viewer/video.jpg");
               Image img2 = ImageIO.read(url2);
               WritableRaster imgRaster = img3.getRaster();
               DataBufferByte data = (DataBufferByte)imgRaster.
getDataBuffer();
               System.out.println("The raster of the image
is: " + imgRaster);
               System.out.println("The Image Byte Level Data
is: " + data);
               label.setIcon(imgIcon2);
               label.repaint();
               try{
                    Thread.sleep(500);
```

```
                } catch (java.lang.InterruptedException
ie) {
                    System.out.println("Interrupted Exception
Occurred: " + ie);
                }
            }
        } catch (MalformedURLException mue) {
            System.out.println("Malformed Exception Occurred:
" + mue);
        }catch(IOException ioe){
            System.out.println("I/O Exception Occurred: " +
ioe);
        }
    }
}

public static class CustomAuthenticator extends Authenticator{
protected PasswordAuthentication getPasswordAuthentication(){
            String username = "admin";
            String password = "manon1982";
            return new PasswordAuthentication(username,passwo
rd.toCharArray());

        }

}
```

Now, we need to work with FFMPEG and the following class illustrates how to work with the FFMPEG library in PHP:

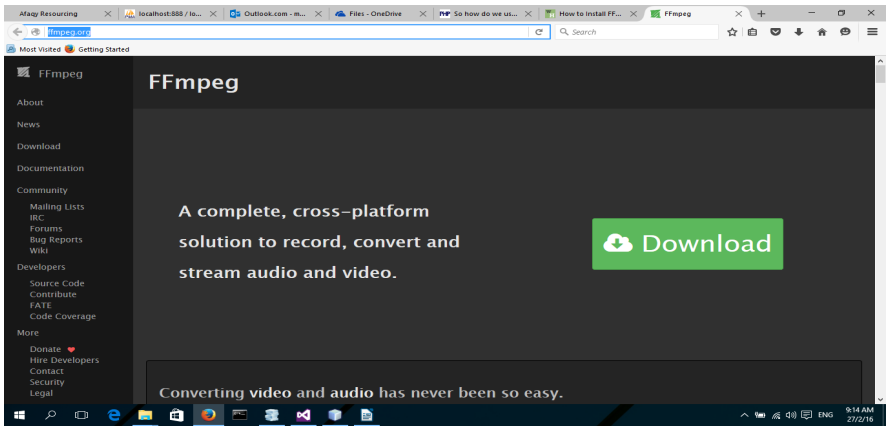
```
<?php

namespace SaveFile\From;

class WebCamera {

    public static function __init($filePath, $directory =
'E:\\', $ffmpegPath = "C:\\Bitnami\\wampstack-7.0.0RC7-0\\
apache2\\htdocs\\libs\\FFMPEG\\Bin\\ffmpeg", $cameraName =
"Lenovo EasyCamera"){
```

```
            shell_exec('taskkill /im ffmpeg.exe /t /f');
//Windows
     shell_exec('sudo taskkill /im ffmpeg /t /f'); //Linux
            $commandExecute = $ffmpegPath . ' -f dshow -i
video="'. $cameraName . '" -t 5 ' . $filePath;
            $filePath = $directory . md5(date('Y--d H-i-s'))
. '.mpeg';
            shell_exec($commandExecute);
            echo '<object id="MediaPlayer1"
CLASSID="CLSID:22d6f312-b0f6-11d0-94ab-0080c74c7e95"
codebase="http://activex.microsoft.com/activex/controls/mpla
yer/en/nsmp2inf.cab#Version=5,1,52,701"
                standby="Loading Microsoft Windows® Media
Player components…" type="application/x-oleobject"
 width="280" height="256">
<param name="fileName" value="' .
$filePath . '">
                <param name="animationatStart" value="true">
                <param name="transparentatStart"
```

```
value="true">
                <param name="autoStart" value="true">
                <param name="showControls" value="true">
                <param name="Volume" value="-450">
                <embed type="application/x-mplayer2"
pluginspage="http://www.microsoft.com/Windows/MediaPlayer/"
src="'  .  $filePath  .  '"  name="MediaPlayer1"  width=280
height=256 autostart=1 showcontrols=1 volume=-450>
                </object>';
     }
}
```

Download FFMPEG from this URL for the suitable operating system

http://ffmpeg.org/



First of all, please download the ffmpeg library that is suitable for your system (Windows/ Linux)

Second, get to know the path of the executable file in Windows ffmpeg.exe in Windows or in ffmpeg.sh in LINUX.

Third, you need to use the function and class as follows:

In your file that will output the video:

```
<?php

use\SaveFile\From;

require'WebCamera.php';

\SaveFile\From\WebCamera::__init("FileName.mpg", "/Directory/
Of/Putput/Path/To/Be/Combined/To/File/Name",  "/Path/To/
FFMPEG/File",  "/Web/Camera/Driver/Name/From/ControlPanel/
Or/Devices/List/In/Linux");
```

The previous class shows how to work with FFMPEG and an example of the usage in PHP with the installation instructions on FFMPEG on Linux and Windows Operating Systems.