package lte;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.OutputStream;

import java.net.MalformedURLException;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.URISyntaxException;

import java.util.HashMap;

import java.util.Vector;

import java.util.concurrent.TimeUnit;

import java.util.logging.Level;

import java.util.logging.Logger;

import javafx.application.Platform;

import javafx.embed.swing.JFXPanel;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.layout.VBox;

import javafx.scene.media.Media;

import javafx.scene.media.MediaView;

import javafx.util.Duration;

import javax.swing.JFrame;

public class SingleReceiver extends Thread

{

public String node,sysno,portno,onevideo;

int k=0,count=0,starting,ending;

UserController uc;

Observer o;

usermultireceiver umr;

ProgressBind bb;

public HashMap totalvideo = new HashMap();

public Vector totalfile = new Vector();

public Vector storevideo = new Vector();

File f=new File("");

File ff;

private int VID\_WIDTH = 320;

private int VID\_HEIGHT = 180;

private int PLAYER\_WIDTH = 320;

private int PLAYER\_HEIGHT = 265;

// servervideoslist ssl = new servervideoslist();

public SingleReceiver(String nodename,String sysno,String portno,UserController uc,usermultireceiver umr)

{

this.node = nodename;

this.sysno = sysno;

this.portno = portno;

this.uc = uc;

this.bb = uc.getProgressBind();

this.umr = umr;

this.o = uc.getObserver();

ff = new File(f.getAbsolutePath()+"\\Downloads");

ff.mkdir();

start();

}

public void run()

{

try

{

ServerSocket ss = new ServerSocket(Integer.parseInt(portno));

while (true)

{

Socket s = ss.accept();

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

String status = (String) ois.readObject();

if (status.equalsIgnoreCase("userdisplay"))

{

String video =(String)ois.readObject();

String[] vide = video.split("\n");

tablevideo(vide);

Platform.runLater(new Runnable() {

@Override

public void run() {

o.setavailablesourcevideo(video);

}

});

}

else if(status.equalsIgnoreCase("videodesttosource"))

{

String path = (String)ois.readObject();

String videoname = (String)ois.readObject();

byte[] input = (byte[])ois.readObject();

String pathh = path+"\\"+videoname;

uc.videoplay(pathh,videoname);

// store(videoname,input);

}

else if(status.equalsIgnoreCase("Servevideotouser"))

{

// int totalsize = (int)ois.readObject();

String streampath = (String)ois.readObject();

String videoname = (String)ois.readObject();

Vector video =(Vector)ois.readObject();

byte[] bytearr = (byte[])ois.readObject();

uc.videoplay(streampath,videoname);

Platform.runLater(new Runnable() {

@Override

public void run() {

}

});

//

// FileOutputStream out = new FileOutputStream("D:\\Alaguvigneshwar\\"+this.node+"\\"+videoname);

// out.write(bytearr);

}

else if(status.equalsIgnoreCase("DOWNLOADSOURCE"))

{

String currentfile = (String)ois.readObject();

int totalpacksize = (Integer)ois.readObject();

String path = (String)ois.readObject();

String videoname = (String)ois.readObject();

Vector vvv =(Vector)ois.readObject();

byte[] bytearr = (byte[])ois.readObject();

String videofolder = videoname.substring(0, videoname.lastIndexOf("."));

Double b = Double.valueOf(totalpacksize);

Double progressvalue = 1/b;

totalvideo.put(currentfile, bytearr);

totalfile.add(currentfile);

File storepath = new File(ff.getAbsolutePath()+"\\"+this.node);

if(storepath.exists())

{

}

else

{

storepath.mkdir();

}

File storepathh = new File(storepath.getAbsolutePath()+"\\"+videofolder);

storepathh.mkdir();

String strr = storepathh.getAbsolutePath();

FileOutputStream out = new FileOutputStream(strr+"\\"+currentfile);

out.write(bytearr);

if(count == 0)

{

starting = starttime();

}

count++;

Platform.runLater(new Runnable() {

@Override

public void run() {

bb.setdata(progressvalue \*Double.valueOf( totalvideo.size()));

bb.setindata(progressvalue \*Double.valueOf( totalvideo.size()));

}

});

if(count==totalpacksize)

{

ending = stoptime();

int totaltime = ending-starting;

Double t = Double.valueOf(TimeUnit.MILLISECONDS.toSeconds(totaltime));

Platform.runLater(new Runnable()

{

@Override

public void run()

{

o.setdownloadtime(String.valueOf(t)+"s");

}

});

File[] join = storepathh.getAbsoluteFile().listFiles();

OutputStream outputStream = new FileOutputStream(storepath.getAbsolutePath()+"\\"+videoname);

if (join.length != 0)

{

for (File file : join)

{

InputStream inputStream = new FileInputStream(file);

int readByte = 0;

while ((readByte = inputStream.read()) != -1) {

outputStream.write(readByte);

}

inputStream.close();

}

outputStream.close();

if(join!=null)

{

for(File fl :join)

{

fl.delete();

storepathh.delete();

}

}

else

{

storepathh.delete();

}

}

}

}

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

public void send(String choose,String video,String nodename,String location) throws IOException

{

if (!umr.supersourcename.contains(video))

{

new Thread(new Runnable()

{

@Override

public void run()

{

String sysnum = (String) umr.gatewaysysno.get(location).toString();

String portnum = (String) umr.gatewayportno.get(location).toString();

try

{

Socket s = new Socket(sysnum, Integer.parseInt(portnum));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("VideoRequest");

oos.writeObject(choose);

oos.writeObject(nodename);

oos.writeObject(location);

oos.writeObject(video);

oos.close();

s.close();

}

catch (IOException ex)

{

Logger.getLogger(SingleReceiver.class.getName()).log(Level.SEVERE, null, ex);

}

}

}).start();

}

else if (umr.supersourcename.contains(video))

{

String sysnum = (String) umr.supersourcesysno.get(video).toString();

String portnum = (String) umr.supersourceportno.get(video).toString();

new Thread(new Runnable()

{

@Override

public void run()

{

try

{

Socket s = new Socket(sysnum, Integer.parseInt(portnum));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("VideoRequesttosupersource");

oos.writeObject(choose);

oos.writeObject(nodename);

oos.writeObject(location);

oos.writeObject(video);

oos.close();

s.close();

}

catch (IOException ex)

{

Logger.getLogger(SingleReceiver.class.getName()).log(Level.SEVERE, null, ex);

}

}

}).start();

}

}

public void availvideo(String node,String localg) throws IOException

{

String sysnum = (String) umr.gatewaysysno.get(localg).toString();

String portnum = (String) umr.gatewayportno.get(localg).toString();

Socket s = new Socket(sysnum, Integer.parseInt(portnum));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("availvideo");

oos.writeObject(node);

oos.close();

s.close();

}

public void videoplay(String path,String videoname) throws MalformedURLException, URISyntaxException

{

String VIDURL=path.replace("\\", "/");

File ff = new File(VIDURL);

System.out.println("=============>"+VIDURL);

final String vurl = "file:///"+VIDURL;

System.out.println("=============>"+vurl);

final JFXPanel panel = new JFXPanel();

Platform.runLater(new Runnable()

{

@Override

public void run()

{

initFX(panel, vurl);

}

});

}

private void initFX(JFXPanel panel, String url)

{

final Media clip = new Media(url);

final javafx.scene.media.MediaPlayer player = new javafx.scene.media.MediaPlayer(clip);

final MediaView view = new MediaView(player);

MediaView mediaView = new MediaView(player);

mediaView.setFitWidth(VID\_WIDTH);

mediaView.setFitHeight(VID\_HEIGHT);

mediaView.getMediaPlayer().play();

uc.videoplay.getChildren().addAll(mediaView);

}

public int starttime()

{

int startTime = (int) System.currentTimeMillis();

return startTime;

}

int stoptime()

{

int stopTime = (int) System.currentTimeMillis();

return stopTime;

}

void tablevideo(String[] video)

{

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

{

onevideo=video[i];

if(!storevideo.contains(onevideo))

{

storevideo.add(onevideo);

if (uc.servervideos.get(0).getColumnvideo().equals(""))

{

uc.servervideos.clear();

}

uc.servervideos.add(new servervideoslist(onevideo));

}

}

}

}