Akademija tehničko-vaspitačkih strukovnih studija
odsek NIŠ

Savremene računarske tehnologije **SRT**

OBJEKTNO ORIJENTISANO PROGRAMIRANJE - OOP

Prof. dr Zoran Veličković, dipl. inž. el.

2019/2020.

Prof. dr Zoran Veličković, dipl. inž. el.

OBJEKTNO ORIJENTISANO PROGRAMIRANJE - OOP

Apleti – male grafičke aplikacije u Javi

(14)

Sadržaj

Male grafičke aplikacije u Javi

- **APLETI** (engl. *Applets*) **predstavljaju SPECIFIČNU KLASU** Java programa sa **DRUGAČIJOM ARHITEKTUROM** od do sada proučavanih aplikacija.
- **APLETI** su **MALE APLIKACIJE** razvijene u programskom jeziku Java sa sledećim specifičnostima:
  - Smeštene su na SERVERU;
  - Preuzimaju se sa servera PUTEM MREŽE (Interneta);
  - AUTOMATSKI se instaliraju NA KLIJENTU;
  - IZVRŠAVAJU se kao DEO Web stranice.
- Još se može se reći da **APLETI** predstavljaju posebnu **KLASU** u Javi realizovanu u **GRAFIČKOM OKRUŽENJU**.
- **APLETI** koriste Javinu grafičku biblioteku **AWT** (engl. *Abstract Window Toolkit*) i **IZVEDENI** su iz klase **`Applet`**.
- Klasa **`Applet`** se nalazi u paketu: **`java.applet.Applet`**.

Metode Apleta

- U klasi **`Applet`** definisane su sledeće **METODE** za **UPRAVLJANJE IZVRŠAVANJEM APLETA**:
  - **`init()`,**
  - **`start()`,**
  - **`stop()`,**
  - **`destroy()`,**
  - **`getAppleContext()`.**
- Pored pomenutih metoda, klasa **`Applet`** poseduje i metode za **UČITAVANJE** i **PRIKAZIVANJE SLIKA** kao i drugih **MULTIMEDIJALNIH SADRŽAJA,** evo nekoliko njih:
  - **`Image getimage(URL url)`,**
  - **`void resize(int širina, int visina)`,**
  - **`getAudioCllip(URL url, String ime sekvence)`,**
  - **`play(URL url)`** - metode UČITAVANJE i REPRODUKCIJU AUDIO SEKVENCI

Interfejsi Apleta

- Takođe, u paketu **`java.applet`** su definisana i **TRI INTERFEJSA**:
  - **`AppletContext;`**
  - **`AudioClip;`**
  - **`AppletStub.`**
- Apleti imaju **OGRANIČENI PRISTUP RESURSIMA KLIJENATA**, tako da se mogu izvršavati **BEZ BOJAZNI** od virusne infekcije.
- **OKRUŽENJE** u kome se **IZVRŠAVA** aplet naziva se **KONTEJNER APLETA,** i potpuno je odgovoran za njegovo:
  - preuzimanje i
  - životni ciklus apleta.
- Već je napomenuto da su apleti nešto **DRUGAČIJE STRUKTUIRANI** u odnosu na standardne Java aplikacije.
- Primer izgleda **STRUKTURE** Java Apleta je dat na jednostavnom primeru apleta **`SimpleApplet`**.

Primer: SimpleApplet

```
import java.awt.*;
import java.applet.*;

public class SimpleApplet extends Applet {
  public void paint(Graphics g) {
    g.drawString("A Simple Applet", 20, 20);
  }
}
```

Uvoze se **SVE** klase iz grafičkog **AWT** paketa

Uvozi se paket **applet** koji sadrži klasu **Applet**

Dekleracija klase **SimpleApplet**

Kreiranje apleta nasleđivanjem klase **Applet**

Aplet **nema** metodu `Main()` !!

Dekleracija metode **paint()** iz **AWT** paketa, **SimpleApplet** je **REDEFINIŠE**.
Metoda **paint()** se poziva **SVAKI PUT** kada aplet treba da osveži prikaz.

Objekt **g** tipa **Graphics** opisuje **GRAFIČKO OKRUŽENJE** u kome se aplet izvršava.

Metoda **drawString()** je član klase **Graphics** i iscrtava ZNAKOVNI NIZ na x,y lokaciji.

Apleti i HTML

```
// fajl: PokreniApp.html
import java.awt.*;
import java.applet.*;
/*
   <applet code="SimpleApplet" width="200" height="60"> </applet>
*/
public class SimpleApplet extends Applet {
  public void paint(Graphics g) {
    g.drawString("A Simple Applet", 20, 20);
  }
}
```

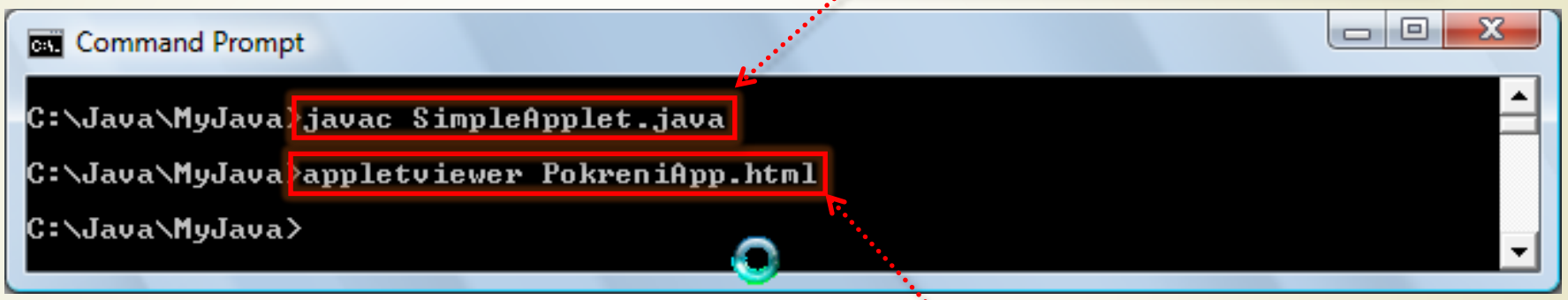
HTML kod koji aplicira aplet **SimpleApplet**

HTML kod je pod komentarom i koristi ga **appletviewer**

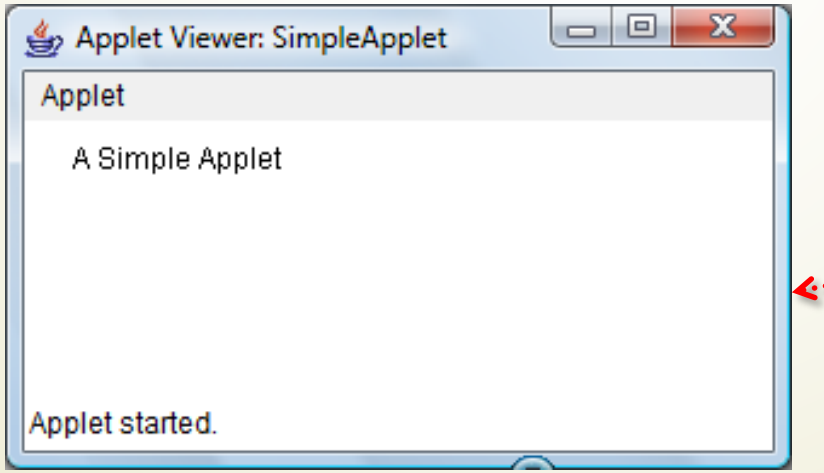
Izvršenje apleta se odvija uz pomoć **ČITAČA Weba** ili uz pomoć programa za prikazivanje apleta: **appletviewer**-a.

Čitač Apleta: Appletviewer

Kompajliranje apleta **SimpleApplet**



Startovanje izvršenja apleta **SimpleApplet** u **Appletviewer-u** koji je implementiran u HTML fajlu **PokreniApp.html**



Rezultat izvršenja apleta: grafički prozor apleta **SimpleApplet**

Događaji Apleta

- Iz pokazanog primera se može uočiti da se aplet izvršava u **GRAFIČKOM PROZORU** (za razliku od kompajlera koji se izvršava sa komandne linije).
- Apleti **NASLEĐUJU** iz klase `Applet` sve potrebne **METODE ZA RAD U GRAFIČKOM OKRUŽENJU** i radu sa **PROZOROM APLETA**.
- Apletima uglavnom **UPRAVLJAJU DOGAĐAJI** koji se mogu uporediti sa prekidnim rutinama kod mikroprocesora.
- **DOGAĐAJI APLETA** se mogu posmatrati na sličan način kao i **DOGAĐAJI** koji potiču od tastature ili miša.
- **APLET ČEKA DOGAĐAJ**, a o njegovom nastanku ga obaveštava **AWT** (grafičko okruženje) pozivom procedure za **OBRADU DOGAĐAJA**.
- Po obavljenom poslu, Aplet vraća kontrola **AWT**-u.
- Sam Aplet samo **KRATKO VREME ZDRŽAVA KONTROLU** nad izvršavanjem.
- Konkretni Aplet treba da **REDEFINIŠE OSNOVNI SKUP METODA** za rad sa apletom.

Apleti i osnovne metode

- U klasi Applet definisane su sledeće četiri metode za upravljanje radom apleta:
  - **`init()`**,
  - **`start()`**,
  - **`stop()`**,
  - **`destroy()`**.
- Takođe, u apletima se koristi i metoda **`paint()`** koja je definisana u klasi **`Component`** iz AWT-a.
- **PRIMER** koji je dat u nastavku samo prikazuje **REDOSLED** kojim AWT poziva metode predefinisanom **STRUKTUROM APLETA**.
- Kada aplet **ZAPOČNE** rad, AWT poziva metode sledećim – **UNAPRED UTVRĐENIM REDOM**:
  - **`init()`**,
  - **`start()`**,
  - **`paint()`**.

Životni ciklus Apleta

- Kada se aplet **ZAVRŠI**, pozivaju se metode sledećim **UTRĐENIM REDOM**:
  - **`stop()`**,
  - **`destroy()`**.

- Za **AŽURIRANJE GRAFIČKIH PODATAKA** u prozoru apleta se koristi metoda **`repaint()`**, koja poseduje **ČETIRI KONSTRUKTORA**:

1. **`void repaint()`**,
2. **`void repaint(int levo, int gore, int širina, int visina)`**,
3. **`void repaint(max_Odlaganje)`**,
4. **`void repaint(max_Odlaganje, int levo, int gore, int širina, int visina)`**.

- Metoda **`Showstatus()`**, prikazuje **PORUKU** u **STATUSNOM PROZORU ČITAČA Weba** ili programa za prikazivanje apleta.
- **STRUKTURA APLETA** je prikazana na primeru apleta **`AppletSkel`**.

Struktura Apleta – (1)

```
import java.awt.*;
import java.applet.*;
/*
   <applet code="AppletSkelet" width="300" height="100"></applet>
*/

public class AppletSkelet extends Applet {
   // Prvi poziv
  public void init() {
   // Ostatak koda za inicijalizaciju
  }
    /* Metoda start() se poziva posle init() metode.
    Takođe poziva se svaki put kada se applet aktivira.*/
  public void start() {
   // Kod za start ili nastavak izvršavanja
  }
```

Uvoze se klase za podršku apleta

Nasleđivanje klase **Applet**

1. poziv **init()** metode

2. poziv **start()** metode

Struktura Apleta – (2)

```
// Poziva se kada je aplet stopiran
   public void stop() {
         // suspenzija izvršavanja
   }
/* Poziva se kada je aplet završen. Ova metoda se poslednja izvršava. */
   public void destroy() {
      // obavljanje završetka aktivnosti
   }
   // Poziva se kada se prozor apleta mora restorirati
   public void paint(Graphics g) {
      // ponovno prikazivanje - osvežavanje sadržaja prozora
     // Java kod zadužen za ponovno iscrtavanje prozora
   }
}
```

3. poziv **stop()** metoda

4. poziv **destroy()** metoda

5. poziv **paint()** metoda

Parametarizovani apleti

- Setite se, oznaka Appleta u HTML-u omogućava **PROSLEĐIVANJE PARAMETARA** apletu, oznaka **<param>**.
- Evo kako smo **PROSLEĐIVALI PARAMETRE** apletu iz **HTML KODA**:

```
<param name = "fontName" value = "Courier">
```

- Za **OČITAVANJE PARAMETARA** upućenih Java apletu iz HTML-a koristi se metoda **getParametar()**.
- Zapamtite (ili se setite iz Internet tehnologija), ova metoda vraća vrednost u obliku **ZNAKOVNOG NIZA**!
- Dakle, **LOGIČKE** i **NUMERIČKE** vrednosti se moraju **PRVO PRETVORITI** iz **ZNAKOVNOG NIZA** u njihov interni format.
- Pogledajmo sledeći slajd sa primerom **PROSLEĐIVANJA PARAMETARA** iz HTML koda.

Apleti i param oznaka (1)

```
import java.awt.*;
import java.applet.*;
/*
<applet code="ParamDemo" width="300" height="80" >
    <param name = fontName value = "Courier" >
    <param name = fontSize value = "14" >
    <param name = leading value = "2">
    <param name = accountEnabled value = "true">
</applet>
*/
public class ParamDemo extends Applet {
  String fontName;
  int fontSize;
  float leading;
  boolean active;
```

HTML kôd - dopisano samo radi izvršenja u vreme **razvoja** apleta

Parametri koji se prosleđuju apletu **ParamDemo**

IMENA i **VREDNOSTI** parametara koja se prenose u aplet

Aplet **ParamDemo**

Dekleracija parametara apleta koji će biti preuzeti iz **HTML-a**

Apleti i param oznaka (2)

```
// Inicijalizacija stringa koji treba da se prikaže
  public void start() {
    String param;
   fontName = getParameter("fontName");
    if (fontName == null)
      fontName = "Not Found";
   param = getParameter("fontSize");
    try {
      if (param != null)          // nije pronađen
        fontSize = Integer.parseInt(param);
      else
        fontSize = 0;
    } catch (NumberFormatException e) {
       fontSize = -1;
    }
```

Preuzimanje parametra **fontName**

Provera vrednosti parametra

Preuzimanje parametra **fontSize**

Konverzija tipa **string** u **integer**

Provera vraćene vrednosti

Apleti i param oznaka (3)

```
param = getParameter("leading");
    try {
      if (param != null)              // parametar nađen ?
          leading = Float.valueOf(param).floatValue();
      else
        leading = 0;
    } catch (NumberFormatException e) {
      leading = -1;
    }

param = getParameter("accountEnabled");
    if(param != null)
      active = Boolean.valueOf(param).booleanValue();
  }
```

Apleti i param oznaka (4)

```
// Prikaz parametara, REDEINISANJE METODE paint()
    public void paint(Graphics g) {
        g.drawString("Font name: " + fontName, 0, 10);
        g.drawString("Font size: " + fontSize, 0, 26);
        g.drawString("Leading: " + leading, 0, 42);
        g.drawString("Account Active: " + active, 0, 58);
    }
}
```

Interfejs AppletContext (1)

- Već je pomenuto, u paketu **Java.applet** su definisana **TRI** INTERFEJSA:
  - **AppletContext** (dobavlja podatke o okruženju),
  - **AppletStub** (daje podršku Web čitačima) i
  - **AudioClip** (sa metodama za manipulaciju audio sadržajem: play(), loop() i stop()).
- **AppletContext** je INTERFEJS koji omogućava da se preuzmu podaci iz/o okruženja u kome se aplet izvršava.
- Metode koje ovaj interfejs definiše su:
  - Za applet: **getApplet(String imeApleta)**,
  - Za AudioClip: **getAudioClip(URL url)**,
  - Za sliku: **getImage(URL url)**,
  - Za ulazni strim: **getStream(String ključ)**,
  - Za Učitavanje datoteke: **void showDocument(URL url)**,
  - Za Učitavanje datoteke **void showStatus(String str)**,
  - ...

Interfejs AppletContext (2)

- Zbog **BEZBEDNOSNIH** razloga, Java dozvoljava da aplet učita podatke **SAMO IZ FOLDERA** u kome se nalazi HTML datoteka ili klasa apleta.
- Imena ovih foldera mogu se dobiti sledećim metodama:
  - `getDocumentBase()` i
  - `getCodeBase()`.
- Ove metode pripadaju velikoj klasi **URL**.
- Za **UČITAVANJE DATOTEKE** koristi se metoda `showDocument(URL url)` iz interfejsa `AppletContext`.
- Kada se dobije **KONTEKST** u kome se aplet izvršava, može se prikazati **BILO KOJI DOKUMENT** metodom `showDocument(URL url)`.

Interfejs AppletContext (3)

```
/* Korišćenje konteksta apleta: getCodeBase(), showDocument() za prikaz HTML fajla.
*/
import java.awt.*;
import java.applet.*;
import java.net.*;
/*  <applet code="ACDemo" width="300" height="50"> </applet> */
public class ACDemo extends Applet{
  public void start() {
    AppletContext ac = getAppletContext();
    URL url = getCodeBase();    // čitanje url-a ovog apleta
```

Interfejs AppletContext (4)

```
try {
                ac.showDocument(new URL(url+"Test.html"));
            }
        catch (MalformedURLException e) {
          showStatus("URL not found");
        }
    }
}
```

Eventualno bačeni izuzetak

PRIKAZIVANJE dokumenta Test.html iz prethodno dobavljenog foldera

- Datoteka **Test.html** koja se prikazuje **MORA BITI** u ISTOM FOLDERU gde se nalazi i sam aplet!

Interfejs AudioClip

- **AudioClip** je **INTERFEJS** koji definiše sledeće metode:
  - **play()** – reprodukuje sekvencu od početka
  - **stop()** – prekida reprodukciju
  - **loop()** – reprodukcija u kontinualnoj formi
- **UČITAVANJE AUDIO SEKVENCE** se obavlja metodom **getAudioClip()**, dok se reprodukcija dobavljene audio sekvence obavlja metodom **play()**.
- INTERFEJS **AppletStub** obezbeđuje saradnju APLETA i ČITAČA.
- Prikazivanje na konzoli se često koristi u procesu debagovanja.
- Ako se u apletu pozove metoda **System.out.println()**, podaci se **NE ŠALJU** u prozor apleta.
- Za potrebe debagovanja treba koristiti metodu **drow-String()**.

Java Play Sound korišćenje Appleta (1)

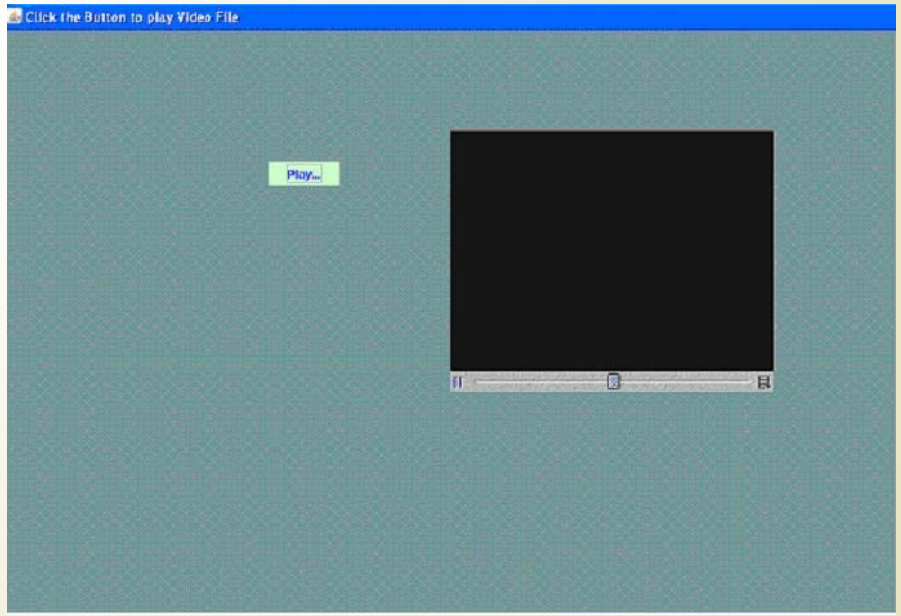
```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class PlaySoundApplet extends Applet implements ActionListener
{
    Button play,stop;
    AudioClip audioClip;
    public void init()
     {
        play = new Button("  Play_Loop  ");
        add(play);
        play.addActionListener(this);
        stop = new Button("  Stop_Play  ");
        add(stop);
        stop.addActionListener(this);
        audioClip = getAudioClip(getCodeBase(), "Play_Sound.wav");
     }
```

Java Play Sound korišćenje Appleta (2)

```
public void actionPerformed(ActionEvent ae)
    {
        Button source = (Button)ae.getSource();
         if (source.getLabel() == "  Play Loop  ")
          {
            audioClip.play();
          }
         else if(source.getLabel() == "  Stop Play  ")
          {
            audioClip.stop();
          }
    }
}
```

Java Play video korišćenjem Apleta (1)

```
import java.util.*;
import java.net.*;
import java.io.*;
import javax.media.*;
import javax.swing.*;
import java.awt.*;
public class Client2 extends javax.swing.JFrame implements ControllerListener, Runnable {
    String filepath;
    Player player;
    Component visualComponent;
    Component controlComponent;
    Component progressBar;
    boolean firstTime;
    long CachingSize;
    int controlPanelHeight;
    int videoWidth;
    int videoHeight;
```



Java Play video korišćenjem Apleta(2)

```
/** Creates new form Client2 */
    public Client2() {
        System.out.println("Inside the Client2 Constructor");

        initComponents();
        setBounds(50,50,880,600);
    }

    public void run(){
        System.out.println(" Inside the Run mehtod");
        while(true){

        }
    }
```

Java Play video korišćenjem Apleta (3)

```
/* Ovaj metod se poziva iz konstrukzora da bi inicijalizovao formu. */
        private void initComponents() {        //GEN-BEGIN:initComponents
        jPanel1 = new javax.swing.JPanel();
        getFile = new javax.swing.JButton();
        panel = new java.awt.Panel();
        getContentPane().setLayout(null);
        setTitle("Click the Button to play Video File");
        addWindowListener(new java.awt.event.WindowAdapter() {
          public void windowClosing(java.awt.event.WindowEvent evt) {
                exitForm(evt);
            }
        });
       jPanel1.setLayout(null);
       jPanel1.setBackground(new java.awt.Color(123, 153, 153));
       getFile.setBackground(new java.awt.Color(204, 255, 204));
       getFile.setForeground(new java.awt.Color(51, 51, 255));
       getFile.setText("PlayFile");
       getFile.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
                getFileActionPerformed(evt);
            }
        });
```

Java Play video korišćenjem Apleta (4)

```
            jPanel1.add(getFile);
            getFile.setBounds(260, 130, 72, 26);
            panel.setLayout(null);
            panel.setName("panel");
            jPanel1.add(panel);
            panel.setBounds(440, 100, 340, 280);
            getContentPane().add(jPanel1);
            jPanel1.setBounds(0, 0, 880, 580);
            pack();
        }    //GEN-END: initComponents
        private void FilesActionPerformed(java.awt.event.ActionEvent evt) {    //GEN-FIRST:
event_FilesActionPerformed
        }    //GEN-LAST:event_FilesActionPerformed
        private void peersActionPerformed(java.awt.event.ActionEvent evt) {    //GEN-FIRST:
event_peersActionPerformed
        }    //GEN-LAST:event_peersActionPerformed
        private void ExitButtonActionPerformed(java.awt.event.ActionEvent evt) {    //GEN-FIRST:
event_ExitButtonActionPerformed
            System.exit(0);
        }    //GEN-LAST:event_ExitButtonActionPerformed
        private void getFileActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:
event_getFileActionPerformed
```

Java Play video korišćenjem Apleta (5)

```
VedioAudioPlayer("C:\\P2PProxyfinal\\Client2\\TCache\\Temp.mpg");
    }      //GEN-LAST:event_getFileActionPerformed
  /** Exit the Application */
    private void exitForm(java.awt.event.WindowEvent evt) {  //GEN-FIRST:event_exitForm
    }      //GEN-LAST:event_exitForm
    /* @param args the command line arguments */
    public static void main(String args[]) {
        System.out.println("Client2 is Ready to Start");
        new Client2().show();
    }
    public void  VedioAudioPlayer(String file) {
        System.out.println("The Asking File is ready to Play");
        player = null;
        visualComponent = null;
        controlComponent = null;
        progressBar = null;
        firstTime = true;
        CachingSize = 0L;
        controlPanelHeight = 0;
        videoWidth = 0;
        videoHeight = 0;
        filepath=file;
        init();
    }
```

Java Play video korišćenjem Apleta (6)

```
public void init() {
        String s = null;
        MediaLocator medialocator = null;
        Object obj = null;
        URL url;
        if(filepath == null)
            Fatal("Invalid media file parameter");

        try {             //url = new URL(filepath);
            if((medialocator = new MediaLocator("file:"+filepath)) == null)
                Fatal("Can't build URL for " + filepath);
            try {
                player = Manager.createPlayer(medialocator);
                player.start();
                player.addControllerListener(this);
                player.realize();
            }
            catch(NoPlayerException noplayerexception) {
                System.out.println(noplayerexception);
                Fatal("Could not create player for " + medialocator);
            }
        }
        catch(MalformedURLException _ex) {
            Fatal("Invalid media file URL!");
        }
        catch(IOException _ex) {
            Fatal("IO exception creating player for " + medialocator);
        }
    }
```

Java Play video korišćenjem Apleta (7)

```
public synchronized void controllerUpdate(ControllerEvent controllerevent) {
   if(player == null)
       return;
   if(controllerevent instanceof RealizeCompleteEvent) {
       if(progressBar != null) {
           panel.remove(progressBar);
           progressBar = null;
       }
       int i = 320; int j = 0;
      if(controlComponent == null && (controlComponent = player.getControlPanelComponent()) != null) {
           controlPanelHeight = controlComponent.getPreferredSize().height;
           panel.add(controlComponent);
           j += controlPanelHeight;
       }
       if(visualComponent == null && (visualComponent = player.getVisualComponent()) != null) {
           panel.add(visualComponent);
           Dimension dimension = visualComponent.getPreferredSize();
           videoWidth = dimension.width;
           videoHeight = dimension.height;
           i = videoWidth;
           j += videoHeight;
           visualComponent.setBounds(0, 0, videoWidth, videoHeight);
       }
       //panel.setBounds(0, 0, i, j);
       if(controlComponent != null) {
           controlComponent.setBounds(0, videoHeight, i, controlPanelHeight);
           controlComponent.invalidate();
       }
```

Java Play video korišćenjem Apleta (8)

```
} else
    if(controllerevent instanceof CachingControlEvent) {
         if(player.getState() > 200)
         return;
      CachingControlEvent cachingcontrolevent = (CachingControlEvent)controllerevent;
  CachingControl cachingcontrol = cachingcontrolevent.getCachingControl(); if(progressBar == null && (progressBar =
cachingcontrol.getControlComponent()) != null){
                    panel.add(progressBar);
                    panel.setSize(progressBar.getPreferredSize());
                    validate();
                }
            } else
                if(controllerevent instanceof EndOfMediaEvent) {
                    player.setMediaTime(new Time(0L));
                    player.stop();
                } else
                    if(controllerevent instanceof ControllerErrorEvent) {
                        player = null;
                        Fatal(((ControllerErrorEvent)controllerevent).getMessage());
                    } else
                        if(controllerevent instanceof ControllerClosedEvent)
                            panel.removeAll();
    }
```

Java Play video korišćenjem Apleta (9)

```
void Fatal(String s) {
        System.err.println("FATAL ERROR: " + s);
        throw new Error(s);
    }
    public void destroy() {
        if(player != null)
            player.close();
    }
    public void start() {
        if(player != null)
            player.start();
    }

    public void stop() {
        if(player != null) {
            player.stop();
            player.deallocate();
        }
    }
  // Variables declaration - do not modify //GEN-BEGIN: variables
     private javax.swing.JButton getFile;
     private javax.swing.JLabel jLabel2;
     private javax.swing.JPanel jPanel1;
     public java.awt.Panel panel;
  // End of variables declaration//GEN-END:variables
}
```