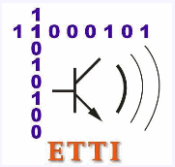
SwRTc ISw
POO – an II

Catedra de Telecomunicatii



08/11/201010

Limbaje de Programare pentru Aplicatii Internet (LPAI)

Laborator 3

Miniaplicatii (applet-uri) Java. Mediul de dezvoltare integrat (IDE) NetBeans. Interfete grafice Swing

3.1. Descrierea laboratorului

In aceasta lucrare de laborator vor fi acoperite urmatoarele probleme:
- Introducere in applet-uri si adaugarea componentelor Swing si a interactivitatii in applet-uri
- Mediul de dezvoltare integrat (IDE) NetBeans
- Crearea unui applet pornind de la zero
- Teme de casa
- Anexe (proceduri de instalare pentru kitul NetBeans)

3.2. Introducere in applet-uri Java

3.2.1. Caracteristicile *applet-urilor* Java

Unul dintre tipurile de programe ce pot fi create cu ajutorul limbajului Java este **applet**-ul. Applet-ul cunoscut si sub numele de mini-aplicatie Java este un program specializat care este executat si vizualizat in cadrul unui browser web (Internet Explorer, Mozilla, etc.). Cand pagina web pe care este plasat applet-ul este încarcata in browser, programul va fi si el incarcat si lansat in executie. Applet-urile nu sunt aplicatii complete, ci componente care ruleaza in mediul browser-ului.

Browser-ul informeaza applet-ul asupra evenimentelor care se petrec pe durata de viata a applet-ului. Serviciile oferite de browser sunt:
- controlul total al ciclului de viata al applet-ului;
- furnizarea informatiilor privind atributele din tag-ul APPLET;
- functia de program/proces principal prin care se executa applet-urile (ofera functia *main()*).

3.2.2. Ciclul de viata al *applet-urilor* Java

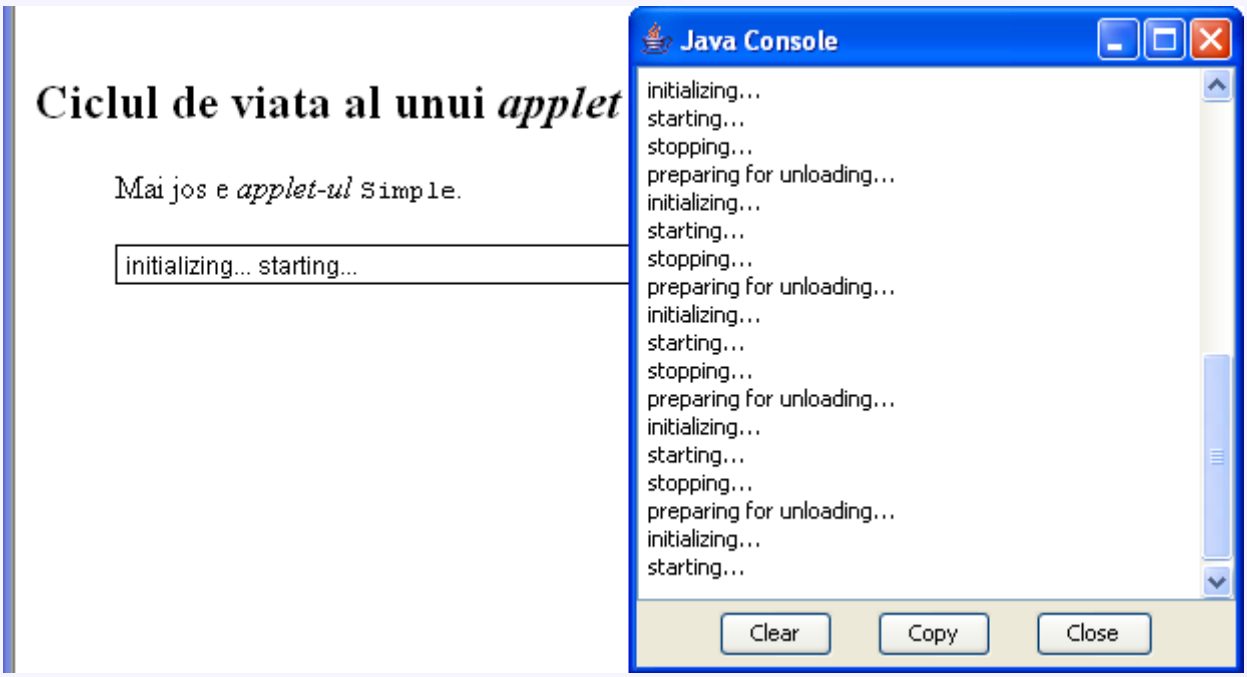
In cazul dezvoltarii unei aplicatii standard in Java, asa cum a fost prezentat in primele laboratoare, este necesara specificarea unei metode *main()*, pe care programul o executa la inceput. In cadrul metodei *main()* este specificata functionalitatea aplicatiei. In cazul unui applet, pe de alta parte, trebuie definite o serie de metode suplimentare care raspund la evenimente invocate de browser pe durata de viata a unui applet. Aceste metode sunt:
- `init()` cand ***incarca applet-ul prima oara***;
- `start()` cand ***un utilizator intra*** sau ***reintra in pagina care contine applet-ul***;
- `stop()` cand ***utilizatorul iese din pagina***;
- `destroy()` ***inaintea terminarii normale.***

Aceste metode sunt apelate automat de browser si nu trebuie apelate explicit in program. Invocarea ultimelor doua metode conduce la "omorarea" tuturor firelor de executie ale applet-ului si la eliberarea tuturor resurselor alocate acestuia.

Urmatorul applet simplu:

```
 1  import java.applet.Applet;
 2  import java.awt.Graphics;
 3
 4  public class Simple extends Applet {
 5      StringBuffer buffer;
 6
 7      public void init() {
 8          buffer = new StringBuffer();
 9          addItem("initializing... ");
10      }
11      public void start() {
12          addItem("starting... ");
13      }
14      public void stop() {
15          addItem("stopping... ");
16      }
17      public void destroy() {
18          addItem("preparing for unloading...");
19      }
20      void addItem(String newWord) {
21
22          System.out.println(newWord); // afisare in Java Console a browser-ului
23
24          buffer.append(newWord);
25
26          repaint(); // apeleaza paint()
27      }
28      public void paint(Graphics g) {
29          //Draw a Rectangle around the applet's display area.
30          g.drawRect(0, 0, size().width - 1, size().height - 1);
31          //Draw the current string inside the rectangle.
32          g.drawString(buffer.toString(), 5, 15);
33      }
34  }
```

permite, **prin vizualizarea lui**, urmarirea fazelor ciclului de viata ale unui *applet*. Pentru a vedea consola din dreapta, in Internet Explorer se foloseste ***Tools*** si se selecteaza ***Sun Java Console***.



3.3. Utilizarea mediului de dezvoltare integrat (IDE) NetBeans

3.3.1. Utilizarea NetBeans IDE pentru crearea unui applet Java

Pentru crearea unui proiect nou de tip applet Java de la zero trebuie parcursi urmati pasii:

In laborator:

1. Se lanseaza in executie mediul de dezvoltare integrat **NetBeans**;
2. Se selecteaza **File** -> **New Project...**
3. In zona **Categories** se selecteaza **Java**, iar in zona **Projects** se selecteaza **Java Class Library**. Apoi **Next**.
4. Se denumeste proiectul: **`HelloApplet`**
5. Se va specifica locatia unde se va salva proiectul (D:\LPAI\Laborator3\GrupaXYZ). Apoi **Finish**.
6. Din subfereastra **Projects**, cu click dreapta pe nodul de proiect **`HelloApplet`**, se deschide meniul pop-up, **New** -> **Other...**
7. Din zona **Categories** se selecteaza **Java**, iar din zona **File Types** se selecteaza **Japplet**. Apoi **Next**.
8. Se va stabili numele clasei: **`AppletSalut`**, iar in campul **Package**, se introduce: **Test**. Apoi **Finish**.

Mediul integrat de dezvoltare (IDE-ul) NetBeans plasează fişierul sursă applet în pachetul specificat anterior, denumit: **Test**. Codul sursa al applet-ului se va deschide in fereastra principala aflata in partea dreapta (editorul de surse). In mod predefinit aceasta contine urmatoarea secventa de cod:

```
1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5  package Test;
6  import javax.swing.JApplet;
7  /**
8   *
9   * @author LPAI
10  */
11 public class AppletSalut extends JApplet {
12     /**
13      * Initialization method that will be called after the applet is loaded
14      * into the browser.
15      */
16     public void init() {
17         // TODO start asynchronous download of heavy resources
18     }
19     // TODO overwrite start(), stop() and destroy() methods
20 }
```

Codul de mai sus se va inlocui cu o secventa de applet simpla:

```
1  package Test;
2  import javax.swing.*; // clasa importata pentru a se crea un applet
3  import java.awt.*;  // clasa importata pentru a se utiliza scriere grafica
4
5  public class AppletSalut extends JApplet { // se extinde JApplet
6     public void paint(Graphics g) {              // declaratie metoda desenare
7        g.setColor(Color.GREEN);
8        g.drawString("Salut utilizator!", 50, 70); // apel metoda scriere grafica
9     }
10 }
```

Applet-ul implementat mosteneste clasa **`Japplet`** care este accesibila prin importarea pachetului de clase **javax.swing.*** . Pe langa acestea, applet-ul contine elemente de interfata grafica pentru dialogul cu utilizatorul (prin importul pachetului **java.awt.**). Appletul va afisa mesajul “Salut utilizator!” in cadrul ferestrei appletului (datorita apelului metodei **drawString()**), in culoarea verde

(prin apelul metodei **setColor()**). De fiecare data cand trebuie sa se afiseze sau sa se actualizeze fereastra appletului, Java apeleaza metoda **paint()**. Acest applet poate fi vazut fie cu ajutorul unui browser web, fie cu ajutorul programului NetBeans care contine un utilitar de vizualizare de appleturi.

In laborator:

1. Din fereastra **Projects**, se deschide meniul pop-up cu click dreapta pe nodul de proiect **`HelloApplet`** si se selecteaza **Build** (Este creat astfel in directorul **dist**, fisierul arhiva **AppletSalut.jar**).

2. Din fereastra **Projects**, se deschide meniul pop-up cu click dreapta pe nodul de proiect **`HelloApplet`** si se selecteaza **Run File** (Este creat astfel in directorul **build**, fisierul html care include appletul ***`AppletSalut.html`***). Appletul va fi lansat automat cu ajutorul **AppletViewer**-ului.

Continutul fisierului ***`AppletSalut.html`*** generat automat in urma executiei este urmatorul:

```
<HTML>
   <HEAD>
      <TITLE>Applet HTML Page</TITLE>
   </HEAD>
   <BODY>
   <H3><HR WIDTH="100%">Applet HTML Page<HR WIDTH="100%"></H3>
   <P>
   <APPLET codebase="classes" code="Test/NewJApplet.class" width=350 height=200></APPLET>
   </P>

   <HR WIDTH="100%"><FONT SIZE=-1><I>Generated by NetBeans IDE</I></FONT>
   </BODY>
</HTML>
```

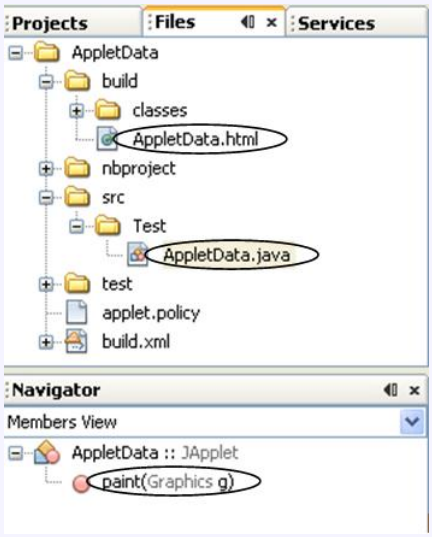
Acest cod poate fi vizualizat si eventual modificat prin selectarea tabului „**Files**”, apoi se selecteaza Build -> classes -> ***`AppletSalut.html`***. Codul sursa „html” se va afisa in fereastra principala aflata in partea dreapta.

In laborator:

1. Un exemplu mai complex de implementare a appleturilor este dat de urmatorul cod, care va fi testat in laborator.

```
 1  package Test;
 2  import javax.swing.JApplet;
 3  import java.awt.Graphics;
 4  import java.util.Calendar;
 5
 6  public class AppletData extends JApplet {
 7
 8     public void paint(Graphics g) {
 9        Calendar acum = Calendar.getInstance();
10        int minutul = acum.get(Calendar.MINUTE);
11        int ora = acum.get(Calendar.HOUR_OF_DAY);
12        int aziCaZiALunii = acum.get(Calendar.DAY_OF_MONTH);
13        int luna = acum.get(Calendar.MONTH);
14        int anul = acum.get(Calendar.YEAR);
15        int aziCaZiASaptamanii = acum.get(Calendar.DAY_OF_WEEK);
16        g.drawString("Salut utilizator!", 50, 25);
17        g.drawString("Azi suntem in " + aziCaZiALunii + "."
18                + luna + "." + anul + ".", 50, 65);
19        g.drawString("Conform ceasului calculatorului este ora "
20                + ora + ":" + minutul + ".", 50, 85);
21        switch (aziCaZiASaptamanii) {
22           case 1:
23              g.drawString("Azi este duminica.", 50, 45);
24              break;
25           case 2:
26              g.drawString("Azi este luni.", 50, 45);
27              break;
28           case 3:
29              g.drawString("Azi este marti.", 50, 45);
30              break;
31           case 4:
32              g.drawString("Azi este miercuri.", 50, 45);
```

```
33              break;
34           case 5:
35              g.drawString("Azi este joi.", 50, 45);
36              break;
37           case 6:
38              g.drawString("Azi este vineri.", 50, 45);
39              break;
40           case 7:
41              g.drawString("Azi este sambata.", 50, 45);
42              break;
43           default:
44              g.drawString("Acest text nu ar trebui sa poata fi afisat.", 50, 45);
45        }
46     }
47  }
```



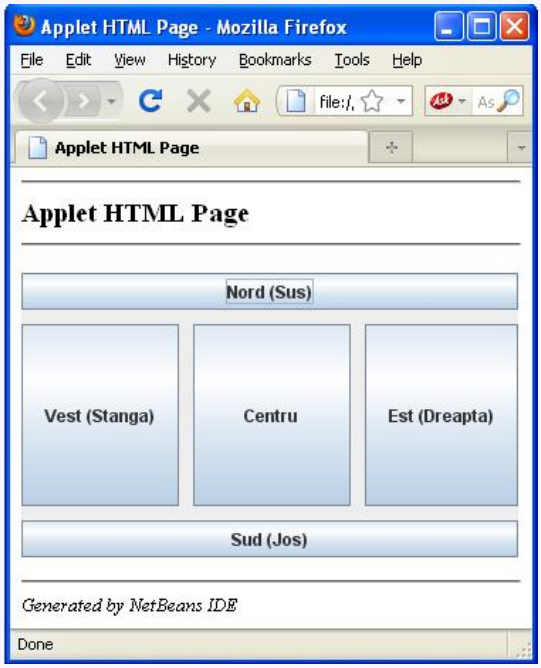
3.4. Adaugarea componentelor Swing si a interactivitatii in applet-uri

3.4.1. Crearea interactivitatii in interfetele grafice

Programul **`ExtensieJApplet`** ilustreaza crearea unei miniaplicatii (*applet*) prin **extinderea clasei `JApplet`**, si asezarea componentelor Swing relativ la margini - **`BorderLayout`**.

```
1  import java.awt.*;
2  import javax.swing.*;
3  /**
4   * Demonstreaza extinderea JApplet pentru a o miniaplicatie Java.
5   */
6  public class ExtensieJApplet extends JApplet {
7      /**
8       * Metoda de initializare a appletului. Apelata de browser la prima
9       * utilizare a appletului, stabileste layout-ul (modul de dispunere a
10      * componentelor in panoul de continut) si adauga componentele in panou.
11      */
12     public void init() {
13        // Obtinerea panoului de continut intern cadrului (container de componente)
14        Container container = getContentPane();
15
16        // Asezarea componentelor in panou (la 10 pixeli de marginea panoului)
17        container.setLayout(new BorderLayout(10, 10));
18
19        // Adaugarea a 5 butoane la panoul appletului
20        container.add(new JButton("Est (Dreapta)"), BorderLayout.EAST);
21        container.add(new JButton("Sud (Jos)"), BorderLayout.SOUTH);
22        container.add(new JButton("Vest (Stanga)"), BorderLayout.WEST);
23        container.add(new JButton("Nord (Sus)"), BorderLayout.NORTH);
24        container.add(new JButton("Centru"), BorderLayout.CENTER);
25     }
26 }
```

Applet-ul rezultat poate fi vizualizat intr-un browser care va arata astfel:



Se va urmari in continuare, imbunatatirea programului **`ExtensieJApplet`** prin **introducerea interactivitatii**, (detalii privind interfetele grafice *swing* in Java) ceea ce presupune tratarea evenimentelor din interfata grafica. Spunem ca un program este controlat de evenimente daca acesta asteapta sa i se ceara sa faca o anumita actiune.

In Java exista mai multe moduri de tratare a evenimentelor.

(1) implementand o metoda numita `action();`
(2) implementand o metoda numita `handleEvent();`
(3) implementand metode specifice interfetelor grafice Swing;

Metodele (1) si (2) sunt valabile incepand chiar din prima versiune Java – JDK 1.0, in timp ce metoda (3) a fost implementata incepand cu versiunea JDK 1.1.

Acest mod de tratare a evenimentelor necesita 3 activitati din partea programatorului:

- Varianta A – Implementare interfata:

I. Declararea unei clase care sa implementeze o interfata «Action Listener» (ascultator de evenimente), (contine metode ce trebuie implementate de utilizator pentru tratarea evenimentului respectiv);

II. Implementarea tuturor metodelor definite in interfata « Action Listener »;

III. Inregistrarea unui obiect din clasa declarata in pasul I, la fiecare dintre componentele grafice (tinta / sursa) pentru care se vrea tratarea evenimentului respectiv.

- Varianta B – Extindere clasa abstracta (adaptor):

I. Declararea unei clase care extinde o clasa predefinita (abstracta, adaptor pentru interfata) care implementeaza o interfata « Action Listener » (ascultator de evenimente);
II. Reimplementarea metodelor dorite din clasa care implementeaza interfata;
III. Inregistrarea unui obiect din clasa declarata in pasul I, la fiecare dintre componentele grafice (tinta / sursa) pentru care se vrea tratarea evenimentului respectiv

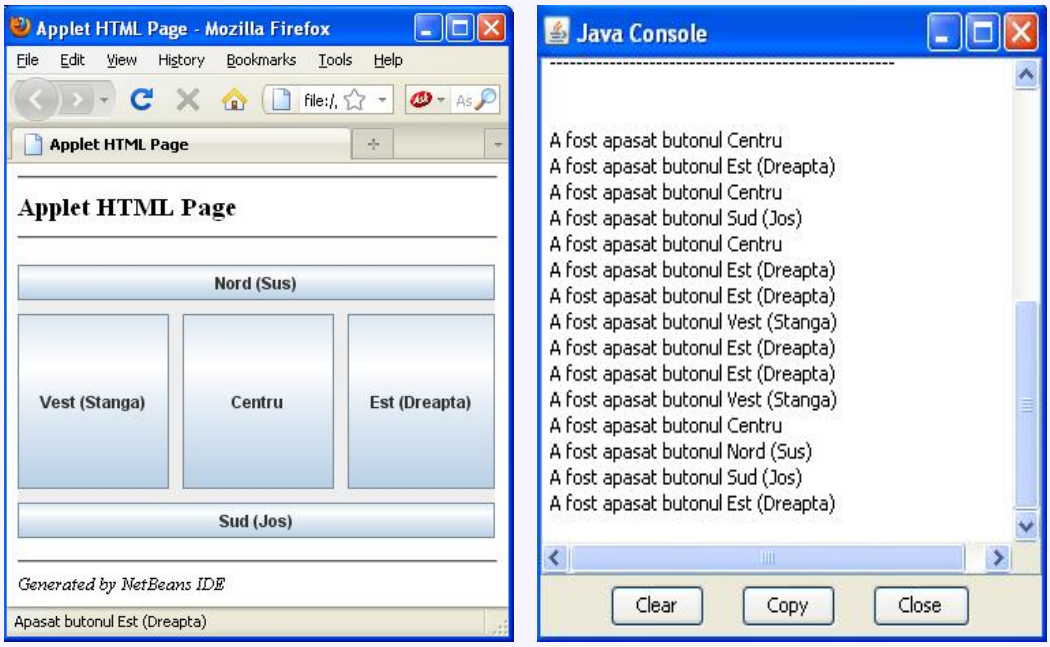
Programul **`ExtensieInteractivaJApplet`** ilustreaza:

- crearea unei miniaplicatii (*applet*) prin **extinderea clasei `Japplet`**;

- tratarea evenimentului « actionare » pentru componentele de tip buton prin implementarea variantei A;

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4
5  public class ExtensieInteractivaJApplet extends JApplet {
6
7      public void init() {
8        // Obtinerea panoului de continut (content pane) creat de browser pentru
9        // executia appletului (container in care vor fi plasate componentele)
10       Container container = getContentPane();
11       // Stabilirea layout-ului panoului, BorderLayout cu spatiu 10 pixeli
12       container.setLayout(new BorderLayout(10, 10));
13
14       // Adaugarea a 5 butoane la panoul appletului
15       // Referintele create vor fi necesare si inregistrarii ascultatorilor
16       JButton b1 = new JButton("Est (Dreapta)");
17       JButton b2 = new JButton("Sud (Jos)");
18       JButton b3 = new JButton("Vest (Stanga)");
19       JButton b4 = new JButton("Nord (Sus)");
20       JButton b5 = new JButton("Centru");
21       container.add(b1, BorderLayout.EAST);
22       container.add(b2, BorderLayout.SOUTH);
23       container.add(b3, BorderLayout.WEST);
24       container.add(b4, BorderLayout.NORTH);
25       container.add(b5, BorderLayout.CENTER);
26
27       // Crearea unui obiect "ascultator" de "evenimente actionare"
28       // (pe care le trateaza)
29       ActionListener obiectAscultatorActionare = new ActionListener() {
30         // Tratarea actionarii unui buton
31         public void actionPerformed(ActionEvent ev) {
32           // Mesaj informare in consola Java
33           System.out.println("A fost apasat butonul " + ev.getActionCommand());
34
35           // Mesaj informare in bara de stare
36           showStatus("Apasat butonul " + ev.getActionCommand());
37         }
38       };
39       // Inregistrarea "ascultatorului" de "evenimente actionare" la "sursele"
40       // de evenimente
41       b1.addActionListener(obiectAscultatorActionare);
42       b2.addActionListener(obiectAscultatorActionare);
43       b3.addActionListener(obiectAscultatorActionare);
44       b4.addActionListener(obiectAscultatorActionare);
45       b5.addActionListener(obiectAscultatorActionare);
46     }
47 }
```

Applet-ul rezultat poate fi vizualizat intr-un browser care va arata ca in figura de mai jos.



3.4.2. Applet-uri standalone

In Java se pot creea de asemenea si appleturi ca aplicatii de sine statatoare (desktop) care pot fi executate cu ajutorul functiei *main()*.

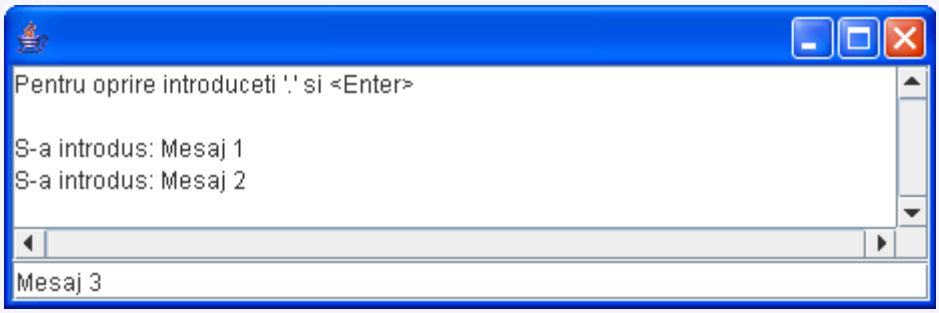
Programul **AppletAplicatie** ce va fi prezentat in continuare exemplifica:

- crearea unei miniaplicatii (applet) prin **extinderea clasei `JApplet`**:
  - care utilizeaza **componente grafice** de tip:
    - **linie / intrare de text** – **`JTextField`**;
    - **zone de text** – **`JtextArea;`**
- **tratarea evenimentelor « action »** pentru componentele intrare de text;
- **transformarea miniaplicatiei in aplicatie de sine statatoare** prin:
  - **adaugarea metodei main();**
  - utilizarea unui **obiect de tip `JFrame`** cu functionalitate echivalenta obiectului de tip `JApplet`.

```
 1  import java.awt.*;
 2  import java.awt.event.*;
 3  import javax.swing.*;
 4
 5  /**
 6   * Ecou text grafic (applet si aplicatie de sine statatoare in acelasi timp).
 7   */
 8  public class AppletAplicatie extends JApplet {
 9
10      public void init() {
11        // Obtinerea panoului de continut (content pane) creat de browser pentru
12        // executia appletului (container in care vor fi plasate componentele)
13        Container containerCurent = this.getContentPane();
14
15        // Stabilirea layout-ului panoului
16        containerCurent.setLayout(new BorderLayout());
17
18        final JTextArea outTextGrafic = new JTextArea(5, 40); // Zona non-editabila
19        JScrollPane scrollPane = new JScrollPane(outTextGrafic,
20                      JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
21                      JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
22        final JScrollBar vertical = scrollPane.getVerticalScrollBar();
23        containerCurent.add("Center", scrollPane);
24        outTextGrafic.setEditable(false);
25        outTextGrafic.append("Pentru oprire introduceti '.' si <Enter>\n\n");
26
27        final JTextField inTextGrafic = new JTextField(40); // Camp editabil intrare
28        containerCurent.add("South", inTextGrafic);
29
30        // Crearea unui obiect "ascultator" de "evenimente actionare"
31        // (pe care le trateaza)
32        ActionListener obiectAscultatorActionare = new ActionListener() {
33
34           // Tratarea actionarii intrarii de text
35           public void actionPerformed(ActionEvent ev) {
36
37              final String sirCitit = inTextGrafic.getText(); // Citire din intrare
38
39              inTextGrafic.setText("");                       // Golire intrare text
40
41              outTextGrafic.append("S-a introdus: "+sirCitit+"\n"); //Scriere in zona
42
43              vertical.setValue(vertical.getMaximum() - vertical.getVisibleAmount());
```

```
44              validate();
45              repaint();
46
47              if (sirCitit.equals(new String(".")))
48                 System.exit(0); // Conditie oprire
49           }
50        };
51
52        //Inregistrare obiect "ascultator" de "evenimente actionare" la "obiect sursa"
53        inTextGrafic.addActionListener(obiectAscultatorActionare);
54     }
55
56     public static void main(String[] args) {
57        AppletAplicatie applet = new AppletAplicatie(); // Applet-ul are rol de panou
58
59        JFrame frame = new JFrame();
60        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
61        Container containerCurent = frame.getContentPane();
62        containerCurent.setLayout(new BorderLayout());
63        containerCurent.add("Center", applet); // Applet-ul e inserat in fereastra
64
65        applet.init();          // Initializarile grafice ale applet-ului refolosite
66
67        frame.pack();
68        frame.setVisible(true); // Fereastra devine vizibila
69
70        applet.start();
71     }
72  }
```

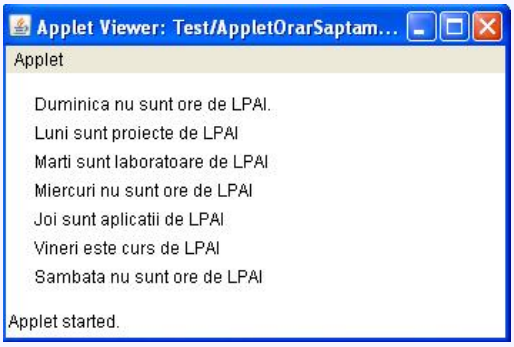
Programul **AppletAplicatie** rulat ca aplicatie de sine statatoare:



3.4.3. Crearea unui orar non-interactiv al disciplinei LPAI

Programul **AppletOrarSaptamanal** afiseaza orarul saptamanal al disciplinei LPAI (in anul universitar 2010-2011), fara detalii privind orele si grupele si fara a oferi interactivitate.

```
1  import java.applet.Applet;
2  import java.awt.Graphics;
3  import java.util.Calendar;
4
5  public class AppletOrarSaptamanal extends javax.swing.JApplet {
6
7     public void paint(Graphics g) {
8        g.drawString("Duminica nu sunt ore de LPAI.", 20, 25);
9        g.drawString("Luni sunt proiecte de LPAI", 20, 45);
10       g.drawString("Marti sunt laboratoare de LPAI", 20, 65);
11       g.drawString("Miercuri nu sunt ore de LPAI", 20, 85);
12       g.drawString("Joi sunt aplicatii de LPAI", 20, 105);
13       g.drawString("Vineri este curs de LPAI", 20, 125);
14       g.drawString("Sambata nu sunt ore de LPAI", 20, 145);
15    }
16 }
```

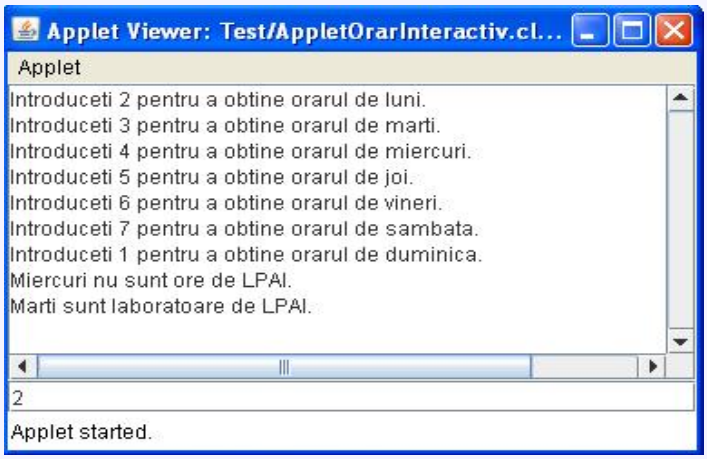


3.4.4. Crearea unui orar interactiv al disciplinei LPAI

Programul **`AppletOrarInteractiv`** este o varianta a applet-ului **`AppletOrarSaptamanal`** care **afiseaza interactiv orarul saptamanal al disciplinei LPAI**, fara detalii privind orele si grupele, dar oferind **interactivitate prin intermediul unei linii/intrari de text** careia ii este atasat si **tratat evenimentul de tip actionare** (in acest caz un **<enter>** dat cand cursorul este in interiorul liniei/intrarii de text).

```
 1  import java.applet.Applet;
 2  import java.awt.*;
 3  import java.awt.event.*;
 4  import javax.swing.*;
 5  import java.util.Calendar;
 6
 7  public class AppletOrarInteractiv extends javax.swing.JApplet {
 8      public void init() {
 9        // Obtinerea panoului de continut (content pane) creat de browser pentru
10        // executia appletului (container in care vor fi plasate componentele)
11        Container containerCurent = this.getContentPane();
12
13        // Stabilirea layout-ului panoului
14        containerCurent.setLayout(new BorderLayout());
15
16        final JTextArea outGrafic = new JTextArea(8, 40); // Zona non-editabila
17        JScrollPane scrollPane = new JScrollPane(outGrafic,
18                       JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
19                       JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
20        final JScrollBar vertical = scrollPane.getVerticalScrollBar();
21        containerCurent.add("Center", scrollPane);
22        outGrafic.setEditable(false);
23        outGrafic.append("Introduceti 2 pentru a obtine orarul de luni.\n" +
24                    "Introduceti 3 pentru a obtine orarul de marti.\n" +
25                    "Introduceti 4 pentru a obtine orarul de miercuri.\n" +
26                    "Introduceti 5 pentru a obtine orarul de joi.\n" +
27                    "Introduceti 6 pentru a obtine orarul de vineri.\n" +
28                    "Introduceti 7 pentru a obtine orarul de sambata.\n" +
29                    "Introduceti 1 pentru a obtine orarul de duminica.\n");
30        final JTextField inGrafic = new JTextField(40); // Camp editabil intrare
31        containerCurent.add("South", inGrafic);
32
33        // Crearea unui obiect "ascultator" de "evenimente actionare"
34        // (pe care le trateaza)
35        ActionListener obiectAscultatorActionare = new ActionListener() {
36
37           // Tratarea actionarii intrarii de text
38           public void actionPerformed(ActionEvent ev) {
39              String rez;
40              final String sirCitit = inGrafic.getText(); // Citire din intrare
41              try {
42                 int numarZi = Integer.parseInt(sirCitit);
43                 switch (numarZi) {
44                    case 1: rez = "Duminica nu sunt ore de LPAI."; break;
45                    case 2: rez = "Luni sunt proiecte de LPAI"; break;
46                    case 3: rez = "Marti sunt laboratoare de LPAI."; break;
47                    case 4: rez = "Miercuri nu sunt ore de LPAI."; break;
```

```
48                      case 5: rez = "Joi sunt aplicatii de LPAI."; break;
49                      case 6: rez = "Vineri este curs de LPAI."; break;
50                      case 7: rez = "Sambata nu sunt ore de LPAI."; break;
51                      default: rez="Numarul e prea mare sau negativ. Mai incercati.";
52                   }
53                } catch (NumberFormatException ne) {
54                   rez = "Nu ati introdus un numar. Mai incercati.";
55                }
56
57                inGrafic.setText("");          // Golire intrare text
58                outGrafic.append(rez + "\n"); // Scriere in zona
59                vertical.setValue(vertical.getMaximum()-vertical.getVisibleAmount());
60                validate(); repaint();
61             }
62          };
63
64          //Inregistrare "ascultator" de "evenimente actionare" la "sursa" acestora
65          inGrafic.addActionListener(obiectAscultatorActionare);
66       }
67    }
```



3.5. Teme pentru acasa

3.5.1. Tema de casa 1: personalizarea orarului cu detalii privind orele, grupele si subgrupele

Se va dezvolta (concepe, edita, compila si executa in NetBeans IDE) un program Java sub forma de **applet** numit **`AppletOrarInteractivDetaliat`** care sa permita afisarea interactiva a orarului la una dintre disciplinele din acest semestru, cu urmatoarele specificatii:

- organizarea programului va fi similara programului `AppletOrarInteractiv` in sensul ca **interactivitatea** va fi realizata **prin intermediul unei linii/intrari de text** careia **ii este atasat si tratat evenimentul de tip actionare**,

- fiecare grup de 2 studenti ai unei subgrupe (cei 2 studenti care lucreaza la acelasi calculator in cadul laboratorului) **isi va alege o alta disciplina pentru care sa realizeze orarul** (care va trebui **sa corespunda celui real din acest semestru),**

- spre deosebire de programul **`AppletOrarInteractiv`** noul program va permite aflarea orarului detaliat (incluzand orele de desfasurare) la respectiva disciplina, si personalizat pentru grupa si subgrupa din care fac parte studentii care il realizeaza,

- adaptarea applet-ului pentru a putea rula si **ca aplicatie de sine statatoare** (vezi exemplul applet-ului AppletAplicatie).

Temele vor fi **predate la lucrarea urmatoare**, cate **un exemplar pentru fiecare grup de 2 studenti, pe hartie** (avand numele celor doi studenti scrise pe prima pagina sus), fie scrise **de mana**, fie sub forma de **listing**.

3.5.1. Tema de casa 2: personalizarea orarului cu elemente grafice

Adaugarea in applet-ul realizat la tema1 a altor elemente grafice:

1. **buton cu rol echivalent** unui **<enter>** dat **cand cursorul este in interiorul liniei/intrarii de text;**

2. **utilizarea unor componente de tip radioButton ca alternativa la linia/intrarea de text**. Vezi si **detalii privind interfetele grafice *swing* in Java**.

Anexa

1. Instalarea kitului NetBeans (DUPA instalarea kitului Java)

1.1. Se lanseaza netbeans-6.1-windows.exe,

1.2. Se confirma TOATE optiunile implicite!!

1.3. Cand se cere specificarea kitului Java se selecteaza:

C:\Program Files\Java\jdk1.5.0_12