Laborator Nr. 10:
Interfeţe grafice în Java

Întocmit de: Dobrinaş Alexandra

Îndrumător: Asist. Drd. Danciu Gabriel

November 30, 2011

I. NOŢIUNI TEORETICE

Interfaţa grafică, - GUI, este un termen cu înţeles larg care se referă la toate tipurile de comunicare dintre un program şi utilizatorii săi. Aceasta este o particularizare a *interfeţei cu utilizatorul - UI*, prin care se întelege interacţiune dintre un program şi utilizatorii săi. Aşadar, *UI* se referă nu numai la ceea ce utilizatorul vede pe ecran ci şi la toate mecanismele de comunicare între acesta şi program.

Limbajul Java pune la dispoziţie numeroase clase pentru implementarea diverselor funcţionalităţi *UI*, însă în continuare sunt prezentate acelea care permit realizarea unei intefeţe grafice cu utilizatorul (GUI).

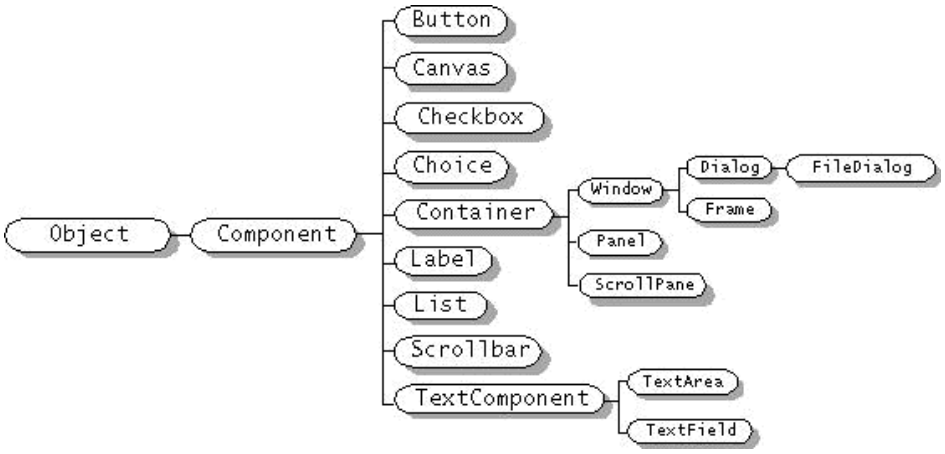
Biblioteca de clase care oferă servicii grafice se numeste `java.awt`, *AWT* fiind prescurtarea de la *Abstract Window Toolkit*. În principiu, crearea unei aplicaţii grafice presupune următoarele:

- Crearea unei suprafeţe de afişare (cum ar fi o fereastră) pe care vor fi aşezate obiectele grafice care servesc la comunicarea cu utilizatorul (butoane, controale de editare, texte, etc);
- Crearea şi aşezarea obiectelor grafice pe suprafaţa de afişare în poziîiile corespunzătoare;
- Definirea unor acţiuni care trebuie să se execute în momentul când utilizatorul interacţionează cu obiectele grafice ale aplicaţiei;
- ,,Ascultarea" evenimentelor generate de obiecte în momentul interacţiunii cu utilizatorul şi executarea acţiunilor corespunzătoare aşa cum au fost ele definite.

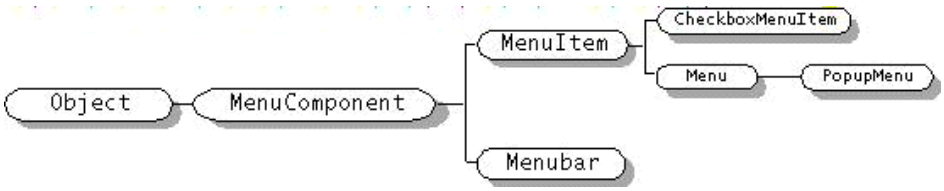
Printr-o *componentă grafică* se înţelege un obiect care are o reprezentare grafică ce poate fi afişată pe ecran şi care poate interacţiona cu utilizatorul. Exemple de componente sunt ferestrele, butoanele, bare de defilare, etc. În general, toate componentele sunt definte de clase proprii ce se gasesc în pachetul `java.awt`. Vezi API aici.

Crearea obiectelor grafice nu realizează automat şi afişarea lor pe ecran. Mai întâi ele trebuie aăezate pe o suprafaţă de afişare, care poate fi o fereastră sau suprafaţa unui applet, şi vor deveni vizibile în momentul în care suprafaţa pe care sunt afişate va fi vizibilă.

În general, toate componentele sunt definte de clase proprii ce se găsesc în pachetul `java.awt`, clasa `Component` este o *superclasă abstractă* a tuturor acestor clase. Ierarhia acestor clase este sumarizată în diagrama de mai jos.

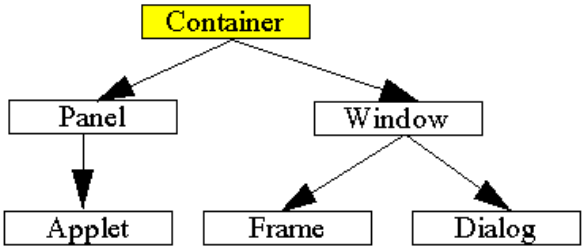


Din cauza modului de imlementare a meniurilor pe diferite sisteme de operare, acestea nu au putut fi integrate ca obiecte de tip `Component`. Superclasa cu ajutorul căreia se pot crea meniuri este `MenuComponent`, iar subclasele sale sunt:



A. Suprafeţe de afişare

Suprafaţa pe care se aşează obiectele grafice se numeşte *suprafaţă de afişare* sau *container* şi reprezintă o instanţă a unei clase obtinută prin extensia superclasei `Container`. O parte din ierarhia a carei rădăcină este `Container` este prezentată în figura de mai jos:



Componentele adăugate sunt memorate într-o listă, iar poziţiile lor din această listă vor defini ordinea de transpunere a acestora în cadrul containerului. Dacă nu este specificat niciun index la adăugarea unei componente, atunci ea va fi adăugată pe ultima pozitie a listei.

1. Adăugarea unei componente

Clasa `Container` pune la dispoziţie metoda `add` pentru adăugarea unei componente pe o suprafaţă de afişare. O componentă nu poate aparţine decât unui singur container, ceea ce înseamnă că pentru a muta un obiect dintr-un container în altul trebuie să-l eliminam mai întâi de pe containerul iniţial. Eliminarea unei componente de pe un container se face cu metoda `remove`.

Exemplu de butoane adăugate într-un container:

```
1  import java.awt.*;
2  public class Test1 {
3     public static void main(String args[]) {
4
5        //creez container-ul - obiect de tip frame
6        Frame f = new Frame("O_fereastra");
7
8        //setez modul de dipunere a ob. pe suprafata ferestrei
9        f.setLayout(new FlowLayout());
10
11       //creez cele doua butoane
12       Button b1 = new Button("OK");
13       Button b2 = new Button("Cancel");
14
15       //adaug primul buton pe suprafata ferestrei
16       f.add(b1);
17       f.pack();
18
19       //adaug al doile buton pe suprafata ferestrei
20       f.add(b2);
21       f.pack();
22
23       //afisez fereastra (o fac vizibila)
24       f.show();
25    }
26 }
```

2. Gestionarea poziţionării componentelor într-un container

Un gestionar de poziţionare *layout manager* este un obiect care controlează dimensiunea şi aranjarea (poziţia) componentelor unui container. Aşadar, modul de aranjare a componentelor pe o suprafaţă de afişare nu este o caracteristică a clasei `Container`. Fiecare obiect de tip `Container`, sau o extensie a lui *(Applet, Frame, Panel)* are asociat un obiect care se ocupă cu dispunerea componentelor pe suprafaţa sa. Toate clasele care instanţiază obiecte pentru gestionarea poziţionării implementează interfaţa `LayoutManager`. La instanţierea unui container se creează implicit un gestionar de poziţionare asociat acestuia. De exemplu, pentru o fereastră (un obiect de tip Window sau o subclasa a sa) gestionarul implict este de tip `BorderLayout`, în timp ce pentru un container de tip Panel este o instanta a clasei `FlowLayout`.

Cei mai utilizaţi gestionari din pachetul `java.awt` sunt:

- FlowLayout
- BorderLayout
- GridLayout
- CardLayout
- GridBagLayout

Detalii despre modul de utilizare a gestionarilor de poziţionare se găsesc aici.

B. Gruparea componentelor (Clasa Panel)

Plasarea componentelor direct pe suprafaţa de afişare poate deveni incomodă în cazul în care avem multe obiecte grafice. Din acest motiv se recomandă gruparea obiectelor grafice înrudite ca funcţii astfel încât să putem fi siguri că, indiferent de gestionarul de poziţionare al suprafeţei de afişare, ele se vor găsi împreună. Gruparea componentelor se face folosind *panel-uri*.

Un *panel* este cel mai simplu model de container. El nu are o reprezentare vizibilă, rolul său fiind de a oferi o suprafaţa de afişare pentru componente grafice, inclusiv pentru alte panel-uri. Clasa care instanţiază aceste obiecte este `Panel`, extensie a superclasei `Container`. Pentru a aranja corespunzator componentele grupate într-un panel, acestuia i se poate specifica un gestionar de poziţionare anume, folosind metoda `setLayout`. Gestionarul implicit pentru containerele de tip `Panel` este `FlowLayout`.

Aşadar, o aranjare eficientă a componentelor unei ferestre înseamnă:

- gruparea componentelor ,,înfrăţite" (care nu trebuie sa fie despartie de gestionarul de pozitionare al ferestrei) în panel-uri;
- aranjarea componentelor unui panel, prin specificarea acestuia a unui gestionar de poziţionare corespunzător;
- aranjarea panel-urilor pe suprafaţa ferestrei, prin specificarea gestionarului de poziţionare al ferestrei.

Exemplu

```
1  import java.awt.*;
2  public class Test2 {
3     public static void main(String args[]) {
4        Frame f = new Frame("Panel");
5
6        Panel panel = new Panel();
7        panel.setLayout(new FlowLayout());
8        panel.add(new Label("Text:"));
9        panel.add(new TextField("", 20));
10       panel.add(new Button("Reset"));
11
12       f.add(panel, BorderLayout.NORTH);
13       f.add(new Button("OK"), BorderLayout.EAST);
14       f.add(new Button("Cancel"), BorderLayout.WEST);
15       f.pack();
16
17       f.show();
18    }
19 }
```

C. Componente grafice

Unele dintre cele mai utilizate componente grafice sunt: *etichetele, butoanele, câmpurile de text, listele derulante, etc.* O listă completă a claselor cu ajutorul cărora se definesc componentele grafice, precum şi proprietăţile lor se găseşte aici.

1. Clasa Label

Un obiect de tip `Label` (etichetă) reprezintă o componentă pentru plasarea unui text pe o suprafaţă de afiâare. O eticheta este formata dintr-o singura linie de text static ce nu poate fi modificat de catre utilizator, dar poate fi modificat din program. Exemplu: Cinci etichete şi adăugate într-un container.

```
import java.awt.*;

public class TestLabel {
   public static void main(String args[]) {
      Frame f = new Frame("TestLabel");
      f.setLayout(new BorderLayout());

      Label nord, sud, est, vest, centru;
      nord = new Label("Nord", Label.CENTER);
      sud = new Label("Sud", Label.CENTER);
      est = new Label("Est", Label.RIGHT);
      vest = new Label("Vest", Label.LEFT);
      centru = new Label("Centru", Label.CENTER);
      centru.setBackground(Color.yellow);
      centru.setFont(new Font("Arial", Font.BOLD, 14));

      f.add(nord, BorderLayout.NORTH);
      f.add(sud, BorderLayout.SOUTH);
      f.add(est, BorderLayout.EAST);
      f.add(vest, BorderLayout.WEST);
      f.add(centru, BorderLayout.CENTER);
      f.pack();

      f.show();
   }
}
```

2. Clasa Button

Un obiect de tip `Button` se foloseşte pentru plasarea unui buton etichetat pe o suprafata de afisare. Exemplu: Doua butoane adăugate pe o fereastră;

```
import java.awt.*;

public class TestButton {
   public static void main(String args[]) {
      Frame f = new Frame("Button");
      f.setLayout(new FlowLayout ( ));
      f.setSize(200, 200);

      Button b1 = new Button("OK");
      b1.setBounds(30, 30, 50, 70);
      b1.setFont(new Font("Arial", Font.BOLD, 14));
      b1.setBackground(java.awt.Color.orange);
      f.add(b1);

      Button b2 = new Button("Cancel");
      b2.setBounds(100, 30, 70, 50);
      b2.setForeground(java.awt.Color.blue);
      f.add(b2);
      f.pack();
      f.show();

   }
}
```

3. Clasa Checkbox

Un obiect de tip `Checkbox` reprezintă o componentă care are două stări : *selectată* sau *neselectată*. Este folosit pentru a prelua anumite opţiuni de la utilizator.

```
import java.awt.*;

public class TestCheckBox {
   public static void main(String args[]) {
      Frame f = new Frame("CheckBox");
      f.setLayout(new GridLayout(5, 1));
      f.setSize(200, 200);

      Label label1 = new Label("Ingrediente Pizza:", Label.CENTER);
      label1.setBackground(Color.orange);
      Label label2 = new Label("");
      label2.setBackground(Color.lightGray);

      Checkbox cbx1 = new Checkbox("cascaval");
      Checkbox cbx2 = new Checkbox("sunca");
      Checkbox cbx3 = new Checkbox("ardei");

      f.add(label1);
      f.add(label2);
      f.add(cbx1);
      f.add(cbx2);
      f.add(cbx3);

      f.pack();
      f.show();

   }
}
```

4. Clasa Choice

Un obiect de tip `Choice` defineşte o listă de opţiuni din care utilizatorul poate selecta una singură. La un moment dat, din întreaga listă doar o singura opţiune este vizibilă, cea selectată în momentul curent.

```
import java.awt.*;

public class Choice {
   public static void main(String args[]) {
      Frame f = new Frame("Choice");
      f.setLayout(new GridLayout(4, 1));
      f.setSize(200, 200);

      Label label = new Label("Alegeti culoarea");
      label.setBackground(Color.red);

      Choice culori = new Choice();
      culori.add("Rosu");
      culori.add("Verde");
      culori.add("Albastru");
      culori.select("Rosu");

      f.add(label);
      f.add(culori);

      f.pack();
      f.show();

   }
}
```

5. Clasa TextField

Un obiect de tip `TextField` defineşte un control de editare a textului pe o singură linie. Este util pentru interogarea utilizatorului asupra unor valori.

```
import java.awt.*;

public class TestText {
   public static void main(String args[]) {
      Frame f = new Frame("Text");
      f.setLayout(new GridLayout(3, 1));
      f.setSize(200, 200);
      f.setBackground(Color.lightGray);

      TextField nume = new TextField("", 30);
      TextField parola = new TextField("", 10);
      parola.setEchoChar('*');

      Panel p1 = new Panel();
      p1.setLayout(new FlowLayout(FlowLayout.LEFT));
      p1.add(new Label("Nume:"));
      p1.add(nume);

      Panel p2 = new Panel();
      p2.setLayout(new FlowLayout(FlowLayout.LEFT));
```

```
21          p2.add(new Label("Parola:"));
22          p2.add(parola);
23
24          Label acces = new Label("Introduceti_numele_si_parola!",                    Label.CENTER);
25          f.add(p1);
26          f.add(p2);
27          f.add(acces);
28
29          f.pack();
30          f.show();
31
32      }
33  }
```

II. TEMA

1. Compilaţi şi rulaţi aplicaţile prezentate. Modificaţi programele astfel încât să se pună în evidenţă toate tipurile de ,,aranjare" a componentele pe container.

2. Realizaţi un program, pe o anumită temă, în care să utilizaţi cât mai multe diintre componentele ce pot fi adăugate pe un container. Folosiţi cel puţin o componentă care nu a fost prezentată în acest laborator(vedeţi API-ul.)

3. Realizaţi o interfaţă pentru un calculator. Vezi calculatorul din Windows. Container-ul va trebui să conţină butoanele pentru cifre, butoanele pentru operaţii aritmetice, ,,fereastra" pentru afişare, precum şi alte butoane necesare efectuării calculelor aritmetice..