Aplicatii JAVA

4

JAVA
Interfata grafica AWT (partea I)

Adrian Runceanu
www.runceanu.ro/adrian

Curs 4

Interfata grafica AWT (Abstract Window Toolkit)

5. Interfata grafica AWT

1. Privire de ansamblu asupra interfetei grafice
2. Componente AWT
3. Gestionari de pozitionare
4. Gruparea componentelor (Clasa Panel)
5. Tratarea evenimentelor
6. Folosirea ferestrelor in AWT

1. Privire de ansamblu asupra interfetei grafice

- Interfata grafica sau, mai bine zis, **interfata grafica cu utilizatorul (GUI)**, este un termen cu înteles larg care se refera la toate tipurile de comunicare vizuala între un program si utilizatorii sai.
- Aceasta este o particularizare a interfetei cu utilizatorul (UI), prin care vom întelege conceptul generic de interactiune între un program si utilizatorii sai.
- Asadar, UI se refera nu numai la ceea ce utilizatorul vede pe ecran ci la toate mecanismele de comunicare între acesta si program.

1. Privire de ansamblu asupra interfetei grafice

- Biblioteca de clase care ofera servicii grafice se numeste **java.awt**, AWT fiind prescurtarea de la **Abstract Window Toolkit** si este pachetul care care a suferit cele mai multe modificari în trecerea de la o versiune JDK la alta.

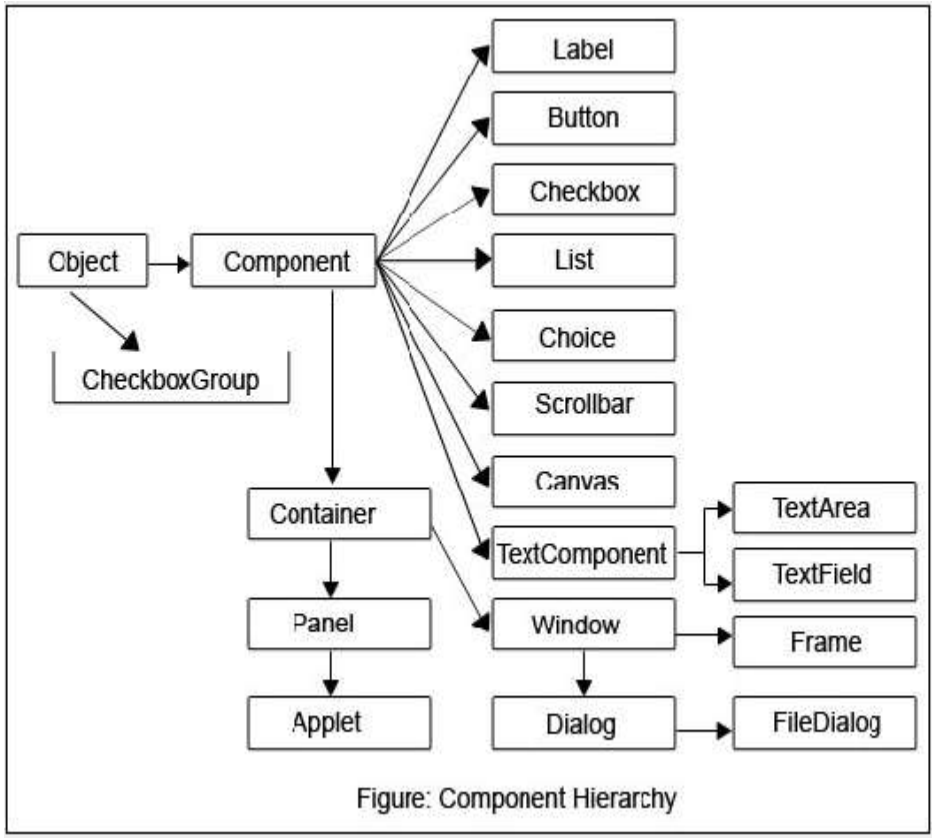


Figure: Component Hierarchy

(*) http://way2java.com/awt-components/java-awt-components/

1. Privire de ansamblu asupra interfetei grafice

In principiu, crearea unei aplicatii grafice presupune urmatoarele activitati:

1. *Crearea unei suprafete de afisare* (cum ar fi o fereastra) pe care vor fi asezate obiectele grafice care servesc la comunicarea cu utilizatorul (butoane, controale de editare, texte, etc)
2. *Crearea si asezarea obiectelor grafice pe suprafata de afisare* în pozitiile corespunzatoare
3. *Definirea unor actiuni* care trebuie sa se execute în momentul când utilizatorul interactioneaza cu obiectele grafice ale aplicatiei
4. *"Ascultarea" evenimentelor generate de obiecte* în momentul interactiunii cu utilizatorul si executarea actiunilor corespunzatoare asa cum au fost ele definite

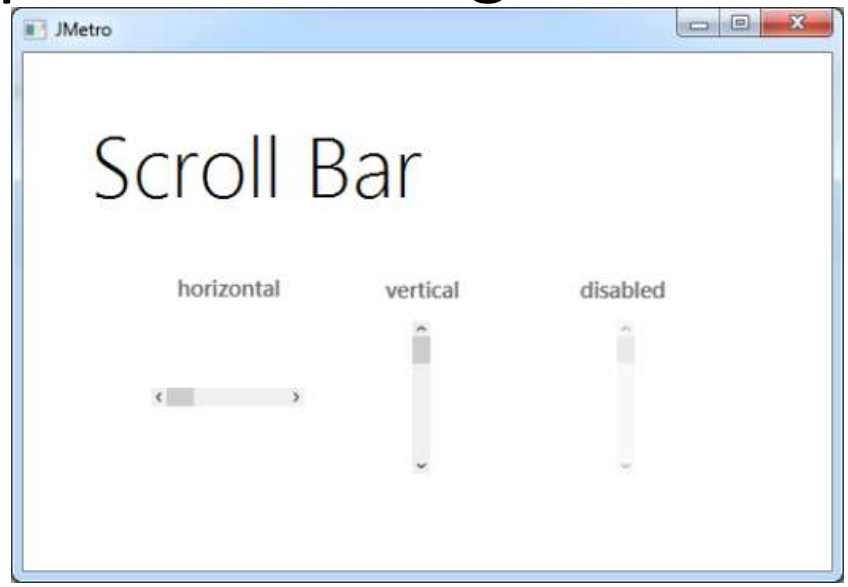
1. Privire de ansamblu asupra interfetei grafice

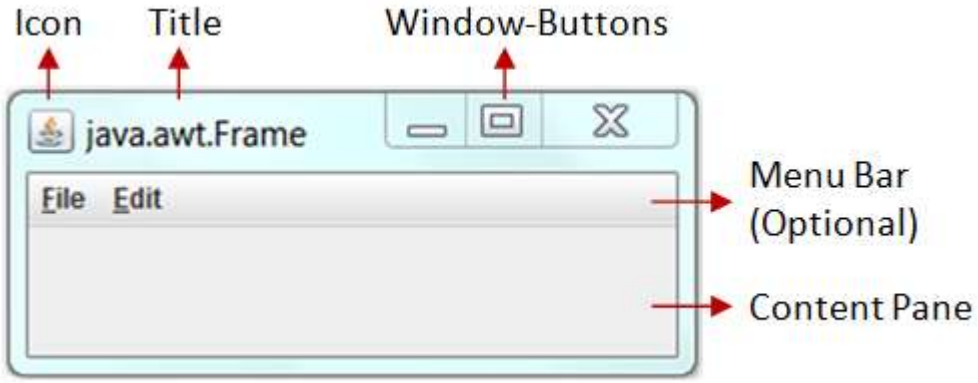
- Majoritatea obiectelor grafice sunt subclase ale clasei **Component**, clasa care defineste generic o componenta grafica care poate interactiona cu utilizatorul
- Singura exceptie o constituie meniurile care provin din clasa *MenuComponent*
- Printr-o componenta sau componenta grafica întelegem *orice obiect care are o reprezentare grafica ce poate fi afisata pe ecran si care poate interactiona cu utilizatorul*

1. Privire de ansamblu asupra interfetei grafice

Exemple de componente sunt:

- ferestre
- butoane
- bare de defilare, etc.





(*) https://pixelduke.wordpress.com/category/windows-8/page/2/
(*) http://www.herongyang.com/Swing/JMenuBar-Menu-Item-Action-Listener-Test.html
(*) http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

1. Privire de ansamblu asupra interfetei grafice

In general, toate componentele sunt definte de clase proprii ce se gasesc în pachetul **java.awt**, clasa **Component** fiind superclasa abstracta a tuturor acestor clase.

(*) http://way2java.com/awt-components/java-awt-components/

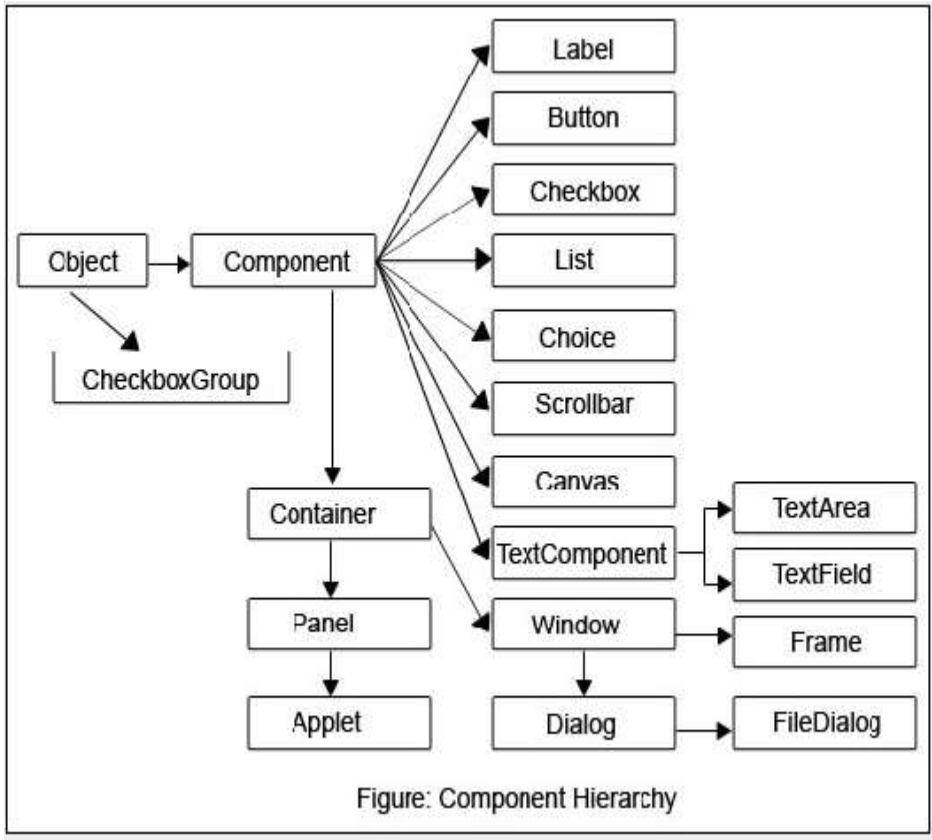


Figure: Component Hierarchy

1. Privire de ansamblu asupra interfetei grafice

- *Crearea obiectelor grafice nu realizeaza automat si afisarea lor pe ecran.*
- Mai întâi ele trebuie asezate pe o suprafata de afisare, care poate fi o fereastra sau suprafata unui applet, si vor deveni vizibile în momentul în care suprafata pe care sunt afisate va fi vizibila.
- O astfel de suprafata pe care se aseaza obiectele grafice reprezinta o instanta a unei clase obtinuta prin extensia clasei Container; din acest motiv suprafetele de afisare se numesc si **containere**.
- Clasa Container este o subclasa aparte a clasei Component, fiind la rândul ei superclasa tuturor suprafetelor de afisare Java (ferestre, applet-uri, etc).

1. Privire de ansamblu asupra interfetei grafice

- *Interfata grafica serveste interactiunii cu utilizatorul.*
- De cele mai multe ori programul trebuie sa faca o anumita prelucrare în momentul în care utilizatorul a efectuat o actiune si, prin urmare, *obiectele grafice trebuie sa genereze evenimente în functie de actiunea pe care au suferit-o* (actiune transmisa de la tastatura, mouse, etc.).
- Începând cu versiunea 1.1 a limbajului Java evenimentele se implementeaza ca obiecte instanta ale clasei **AWTEvent** sau ale subclaselor ei.

1. Privire de ansamblu asupra interfetei grafice

- Un eveniment este produs de o actiune a utilizatorului asupra unui obiect grafic, deci evenimentele nu trebuie generate de programator.

- In schimb, *într-un program trebuie specificat codul care se executa la aparitia unui eveniment*.

- Interceptarea evenimentelor se realizeaza prin intermediul unor clase de tip *listener* (ascultator, consumator de evenimente), clase care sunt definite în pachetul **java.awt.event**.

5. Interfata grafica AWT

1. Privire de ansamblu asupra interfetei grafice
2. Componente AWT
3. Gestionari de pozitionare
4. Gruparea componentelor (Clasa Panel)
5. Tratarea evenimentelor
6. Folosirea ferestrelor in AWT

2. Componente AWT

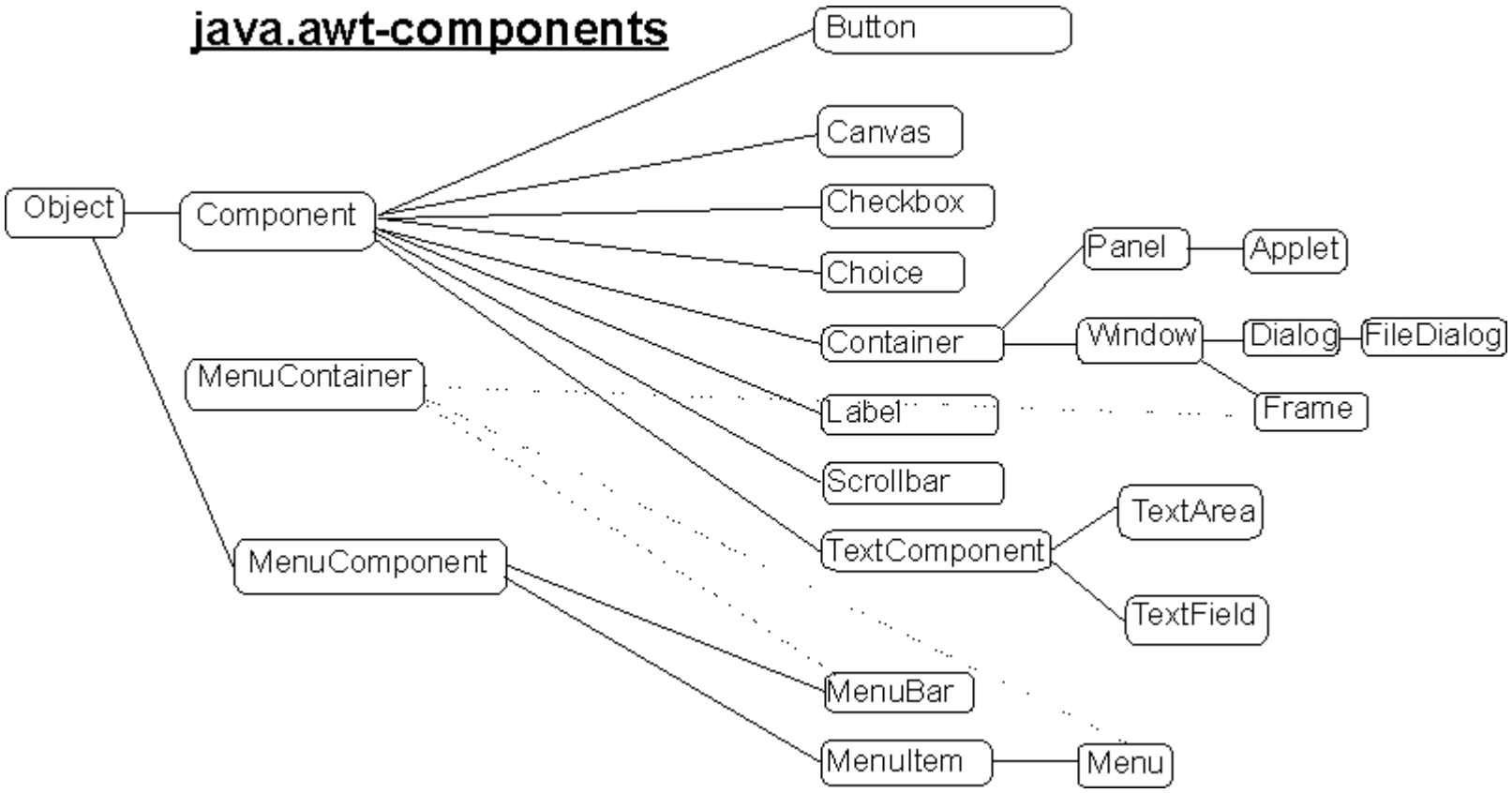
- *Prin componenta vom întelege în continuare orice obiect care are o reprezentare grafica ce poate fi afisata pe ecran si care poate interactiona cu utilizatorul.*

- Exemple de componente sunt ferestrele, butoanele, bare de defilare, etc.

- In general, toate componentele sunt definite de clase proprii ce se gasesc în pachetul **java.awt**, clasa Component fiind superclasa abstracta a tuturor acestor clase.

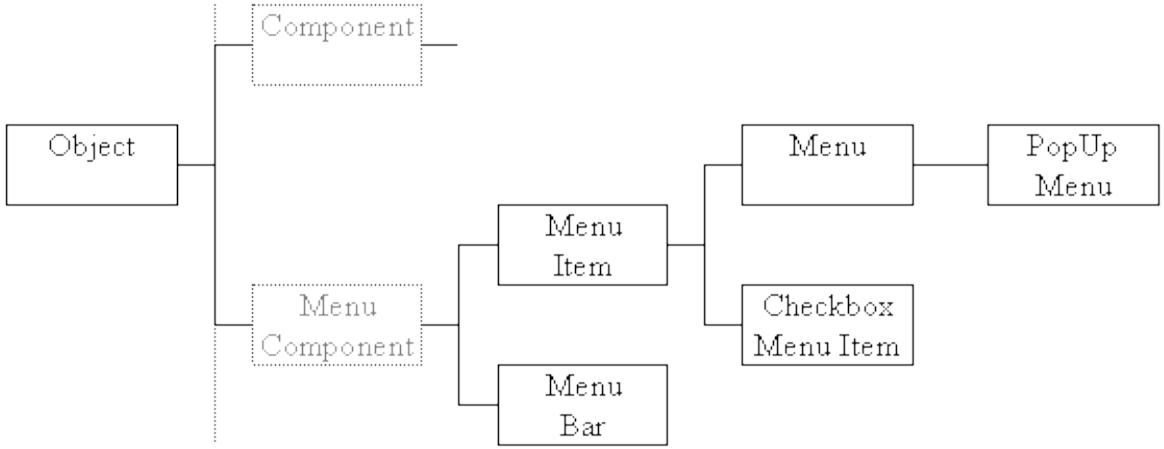
2. Componente AWT

Ierarhia acestor clase:



2. Componente AWT

- Din cauza unor diferente esentiale în implementarea meniurilor pe diferite platforme de operare acestea nu au putut fi integrate ca obiecte de tip Component.
- Superclasa care descrie meniuri este MenuComponent iar ierarhia subclaselor sale este data în diagrama de mai jos:



- Asadar, majoritatea obiectelor grafice sunt subclase ale clasei Component, clasa care defineste generic o component grafica care poate interactiona cu utilizatorul.
- Singura exceptie o constituie meniurile care descind din clasa MenuComponent.

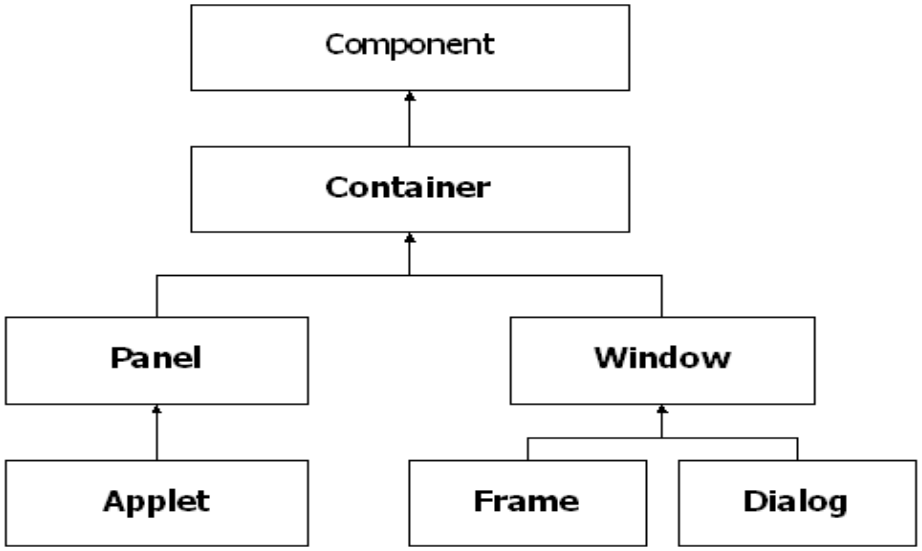
2. Componente AWT

1. Suprafete de afisare (Clasa Container)

- *Crearea obiectelor grafice nu realizeaza automat si afisarea lor pe ecran.*
- Mai întâi ele trebuie asezate pe o suprafata, care poate fi o fereastra sau suprafata unui applet, si vor deveni vizibile în momentul în care suprafata pe care sunt afisate va fi vizibila.
- O astfel de *suprafata pe care se aseaza obiectele grafice se numeste suprafata de afisare sau container* si reprezinta o instanta a unei clase obtinuta prin extensia superclasei Container.

2. Componente AWT

O parte din ierarhia a carei radacina este Container este:



- *Asadar, un container este folosit pentru a adauga componente pe suprafata lui.*
- Componentele adaugate sunt memorate într-o lista iar pozitiile lor din aceasta lista vor defini ordinea de traversare "front-to-back" a acestora în cadrul containerului.
- Daca nu este specificat nici un index la adaugarea unei componente atunci ea va fi adaugata pe ultima pozitie a listei.

2. Componente AWT

Exemplu: crearea unei ferestre ce contine doua butoane

```
import java.awt.*;
public class TestAWT1 {
public static void main(String args[]) {
// se creeaza fereastra - un obiect de tip Frame
Frame f = new Frame("O fereastra");
// se seteaza modul de dispunere a obiectelor pe suprafata ferestrei
f.setLayout(new FlowLayout());
```

2. Componente AWT

```
// se creeaza cele doua butoane
Button b1 = new Button("OK");
Button b2 = new Button("Cancel");
// se adauga primul buton pe suprafata ferestrei
f.add(b1); f.pack();
// se adauga al doilea buton pe suprafata ferestrei
f.add(b2); f.pack();
// se afiseaza fereastra (devine vizibila)
f.show();
}
}
```

2. Componente AWT

Fereastra afisata de acest program va arata astfel:

2. Componente AWT

2. Adaugarea unei componente

- Clasa Container pune la dispozitie metoda add pentru adaugarea unei componente pe o suprafata de afisare.
- *O componenta nu poate apartine decât unui singur container*, ceea ce înseamna ca pentru a muta un obiect dintr-un container în altul trebuie sa-l eliminam mai întâi de pe containerul initial.
- Eliminarea unei componente de pe un container se face cu metoda remove.

```
Frame f = new Frame("O fereastra");
Button b = new Button("OK");
f.add(b); // se adauga butonul pe suprafata ferestrei
f.show();
```

2. Componente AWT

3. Gestionarea pozitionarii

Exemplu de program **Java** care afiseaza 5 butoane pe o fereastra:

```
import java.awt.*;
public class TestLayout {
public static void main(String args[]) {
Frame f = new Frame("Grid Layout");
f.setLayout(new GridLayout(3, 2)); //*
Button b1 = new Button("Button 1");
Button b2 = new Button("2");
Button b3 = new Button("Button 3");
Button b4 = new Button("Long-Named
Button 4");
Button b5 = new Button("Button 5");
f.add(b1); f.add(b2); f.add(b3);
f.add(b4); f.add(b5);
f.pack();
f.show();
}
}
```

2. Componente AWT

- Se poate modifica linia marcata cu '*' ca mai jos, lasând neschimbat restul programului:

```
Frame f = new Frame("Flow Layout");
f.setLayout(new FlowLayout());
```

- Fereastra afisata dupa aceasta modificare va avea o cu totul altfel de dispunere a componentelor sale:

5. Interfata grafica AWT

1. Privire de ansamblu asupra interfetei grafice
2. Componente AWT
3. Gestionari de pozitionare
4. Gruparea componentelor (Clasa Panel)
5. Tratarea evenimentelor
6. Folosirea ferestrelor in AWT

3. Gestionari de pozitionare

- Un *gestionar de pozitionare (layout manager)* este un obiect care controleaza dimensiunea si aranjarea (pozitia) componentelor unui container.
- Asadar, modul de aranjare a componentelor pe o suprafata de afisare nu este o caracteristica a clasei **Container**.
- Fiecare obiect de tip **Container**, sau o extensie a lui (Applet, Frame, Panel) are asociat un obiect care se ocupa cu dispunerea componentelor pe suprafata sa: *gestionarul de pozitionare (layout manager)*.

3. Gestionari de pozitionare

- Toate clasele care instantiaza obiecte pentru gestionarea pozitionarii implementeaza interfata LayoutManager.

- La instantierea unui container se creeaza implicit un gestionar de pozitionare asociat acestui container.

- De exemplu pentru o fereastra (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.

3. Gestionari de pozitionare

Folosirea gestionarilor de pozitionare

- Asa cum am vazut, orice container are un gestionar implicit de pozitionare - un obiect care implemeneaza interfata LayoutManager, acesta fiindu-i atasat automat la crearea sa.
- In cazul în care acesta nu corespunde necesitatilor, el poate fi schimbat cu usurinta.

Cei mai utilizati gestionari din pachetul **java.awt** sunt:

1. FlowLayout
2. BorderLayout
3. GridLayout
4. CardLayout
5. GridBagLayout

3. Gestionari de pozitionare

- Atasarea explicita a unui gestionar de pozitionare la un container se face cu metoda setLayout a clasei Container.
- Metoda poate primi ca parametru orice instanta a unei clase care implementeaza interfata LayoutManager.
- Secventa de atasare a unui gestionar pentru un container este:

```
FlowLayout gestionar = new FlowLayout();
container.setLayout(gestionar);
```

sau, mai uzual:

```
container.setLayout(new FlowLayout());
```

3. Gestionari de pozitionare

- Una din facilitatile cele mai utile oferite de gestionarii de pozitionare este *rearanjarea componentele unui container atunci când acesta este redimesionat*.

- Pozitiile si dimensiunile componentelor nu sunt fixe, ele fiind ajustate automat de catre gestionar la fiecare redimensionare astfel încât sa ocupe cât mai "estetic" suprafata de afisare.

3. Gestionari de pozitionare

- Sunt însa situatii când dorim sa plasam componentele la anumite pozitii fixe iar acestea sa ramâna acolo chiar daca redimensionam containerul.
- Folosind un gestionar de pozitionare aceasta *pozitionare absoluta a componentelor* nu este posibila si deci trebuie cumva sa renuntam la gestionarea automata a containerul.
- Acest lucru se realizeaza prin trimiterea argumentului null metodei setLayout:

```
// pozitionare absoluta a componentelor in container
container.setLayout(null);
```

3. Gestionari de pozitionare

- Folosind pozitionarea absoluta, nu va mai fi suficient sa adaugam cu metoda add componentele în container ci va trebui sa specificam pozitia si dimensiunea lor - acest lucru era facut automat de gestionarul de pozitionare.

Exemplu:

```
container.setLayout( null );
Button b = new Button("Buton");
b.setSize(10, 10); // se stabileste dimensiunea butonului
b.setLocation (0, 0); // se stabileste locatia in container-ul repstectiv
b.add();
```

5. Interfata grafica AWT

1. Privire de ansamblu asupra interfetei grafice
2. Componente AWT
3. Gestionari de pozitionare
4. Gruparea componentelor (Clasa Panel)
5. Tratarea evenimentelor
6. Folosirea ferestrelor in AWT

4. Gruparea componentelor (Clasa Panel)

- Plasarea componentelor direct pe suprafata de afisare poate deveni incomoda în cazul în care avem multe obiecte grafice.

- Din acest motiv se recomanda *gruparea obiectelor grafice înrudite ca functii* astfel încât sa putem fi siguri ca, indiferent de gestionarul de pozitionare al suprafetei de afisare, ele se vor gasi împreuna.

- Gruparea componentelor se face în panel-uri.

4. Gruparea componentelor (Clasa Panel)

- *Un panel este cel mai simplu model de container.*
- El nu are o reprezentare vizibila, rolul sau fiind de a oferi o suprafata de afisare pentru componente grafice, inclusiv pentru alte panel-uri.
- Clasa care instantiaza 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 pozitionare anume, folosind metoda setLayout.
- Gestionarul implicit pentru containerele de tip Panel este FlowLayout.

4. Gruparea componentelor (Clasa Panel)

Asadar, o aranjare eficienta a componentelor unei ferestre înseamna:

1. gruparea componentelor "înfratite" (care nu trebuie sa fie despartite de gestionarul de pozitionare al ferestrei) în panel-uri
2. aranjarea componentelor unui panel, prin specificarea acestuia a unui gestionar de pozitionare corespunzator
3. aranjarea panel-urilor pe suprafata ferestrei, prin specificarea gestionarului de pozitionare al ferestrei

4. Gruparea componentelor (Clasa Panel)

Exemplu

```
import java.awt.*;
public class TestPanel {
public static void main(String args[]) {
        Frame f = new Frame("Panel");
        Panel panel = new Panel();
        panel.setLayout(new FlowLayout());
        panel.add(new Label("Text:"));
        panel.add(new TextField("", 20));
        panel.add(new Button("Reset"));
        f.add(panel, BorderLayout.NORTH);
        f.add(new Button("OK"), BorderLayout.EAST);
        f.add(new Button("Cancel"), BorderLayout.WEST);
        f.pack();
        f.show();
        }
}
```

Referinte

- Curs practic de Java, Cristian Frasinaru – capitolul Interfata grafica cu utilizatorul
- http://docs.oracle.com/javase/6/docs/technotes/guides/awt/
- http://www.tutorialspoint.com//awk/index.htm
- http://archive.oreilly.com/oreillyschool/courses/java3/archive/java3.2/java306.html

Întrebări?