

BUT Informatique 1^{ère} année

R2.02: Développement d'applications avec IHM

Cours 1 : Interfaces graphiques

Sébastien Lefèvre
sebastien.lefevre@univ-ubs.fr

: Interfaces graphiques

Interface graphique

Système de fenêtrage

Interaction plus riche avec l'utilisateur

GUI = Graphical User Interface.

- taille & position des fenêtres
- afficher, saisir données textuelles,
- présenter les informations dans des listes, des menus,
- d'afficher des images, de les modifier,
- interaction avec la souris, le clavier, une surface tactile,
- présentation riche agréable et ergonomique de l'information (texte, image, son, vidéo).

The screenshot shows a window titled "E-mail" with a blue border and standard Windows window controls (minimize, maximize, close). The window is divided into two main sections: "Name" and "E-mail".

Name Section:

- First Name:** A text input field.
- Last Name:** A text input field.
- Title:** A text input field.
- Nickname:** A text input field.
- Format:** A dropdown menu currently showing "Item 1". A list is open below it, showing "Item 1", "Item 2", "Item 3", and "Item 4".

E-mail Section:

- E-mail Address:** A text input field with an "Add" button to its right.
- Address List:** A list box containing "Item 1", "Item 2", "Item 3", "Item 4", and "Item 5". To the right of the list are three buttons: "Edit", "Remove", and "As Default".
- Mail Format:** A section with three radio buttons: "HTML", "Plain Text", and "Custom" (which is selected).

At the bottom right of the window are "OK" and "Cancel" buttons.

Un premier exemple (TP1)



```
import javax.swing.*;

public class HelloWorldSwing extends JFrame {

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            public void run() {

                HelloWorldSwing frame = new HelloWorldSwing();

                frame.pack();

                frame.setVisible(true);

            }

        });

    }

    public HelloWorldSwing() {

        setTitle("HelloWorldSwing");

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JLabel label = new JLabel("Hello World");

        add(label);

    }

}
```

Bonnes et moins bonnes pratiques

Exemple précédent :

- Avantages ?
- Inconvénients ?

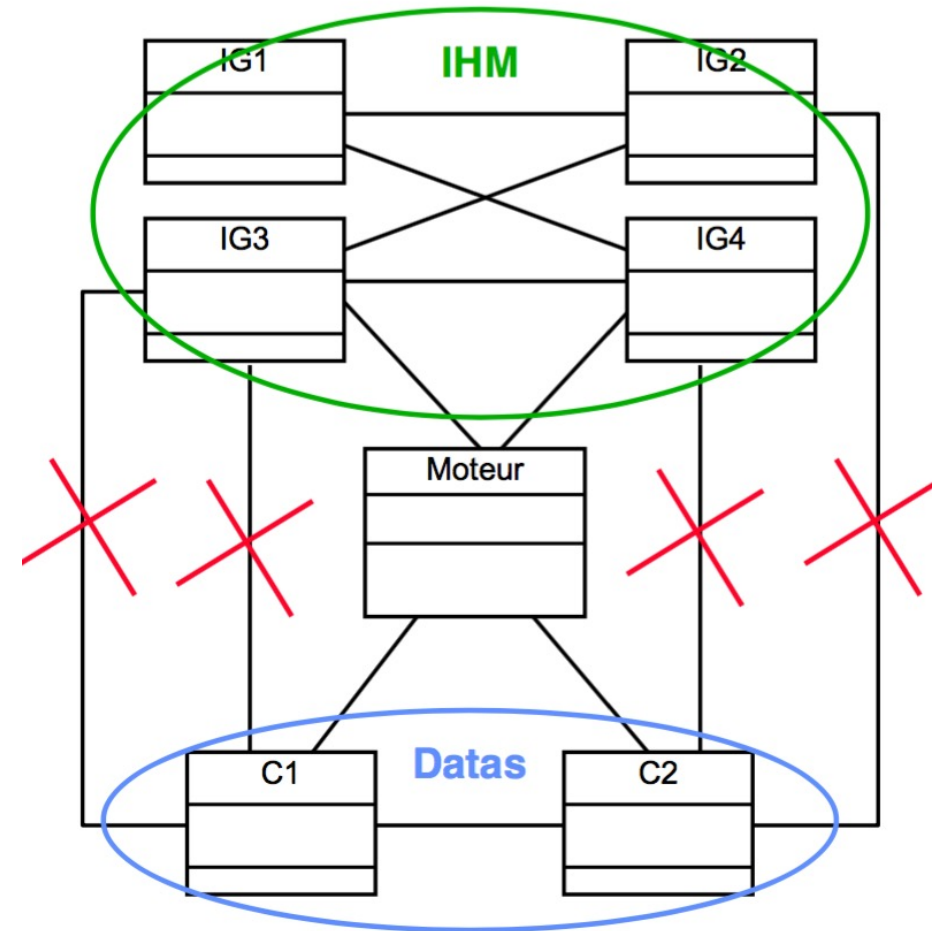
MVC

Patron de conception MVC

= Model View Controller

= Modèle Vue Contrôleur

On sépare les données (M) de leur représentation à l'utilisateur (V), à l'aide de classes dédiées à la gestion des événements utilisateurs (C)



API Java pour les IHM

Java et les IHM = une longue histoire

Au début, Java 1.0 (1995) : AWT (Abstract Windowing Toolkit, `java.awt.*`)
Composants dits lourds (= s'appuient sur du code natif, dessinés par l'OS)

Ensuite, Java 2 = 1.2 (1997) : Swing, `javax.swing`
Composants légers (= 100% Java, dessinés par Java)

Ne pas mixer les 2 ! Préférer Swing !

Un peu plus récemment, JavaFX (2008), inclus dans le JDK...
jusqu'en 2018 !

Il reste une bonne alternative qui sera étudiée plus tard en P4

Les composants graphiques

Widgets (Window Gadgets)

- Boutons
- Cases à cocher
- Liste de sélection
- Menu déroulant
- Barre de défilement
- Zone de saisie de texte
- etc

The image shows a screenshot of a Windows-style dialog box with a blue title bar and standard window controls (minimize, maximize, close). The dialog is divided into two main sections: 'Name' and 'E-mail'.

Name Section:

- First Name:** A text input field.
- Last Name:** A text input field.
- Title:** A text input field.
- Nickname:** A text input field.
- Format:** A dropdown menu currently showing 'Item 1'. A list of items (Item 1, Item 2, Item 3, Item 4) is displayed below the dropdown.

E-mail Section:

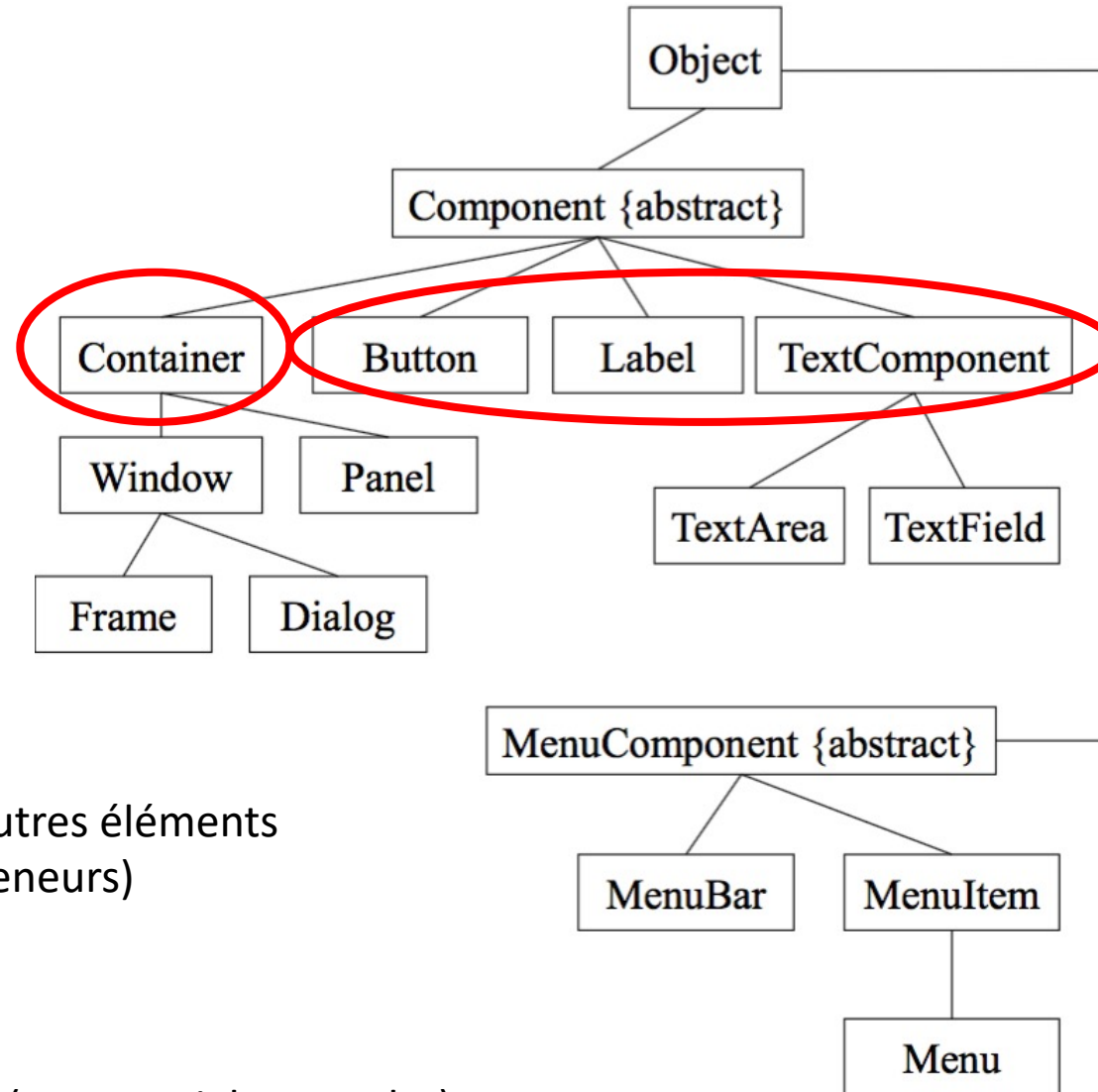
- E-mail Address:** A text input field.
- Add:** A button to add a new email address.
- Edit:** A button to edit a selected email address.
- Remove:** A button to remove a selected email address.
- As Default:** A button to set a selected email address as the default.
- Item List:** A list box containing 'Item 1', 'Item 2', 'Item 3', 'Item 4', and 'Item 5'. It has a vertical scrollbar on the right.

Mail Format:

- Three radio buttons: ☐ HTML, ☐ Plain Text, and ☒ Custom.

Buttons: 'OK' and 'Cancel' buttons are located at the bottom right of the dialog.

AWT



Les conteneurs contiennent d'autres éléments graphiques (y compris des conteneurs)

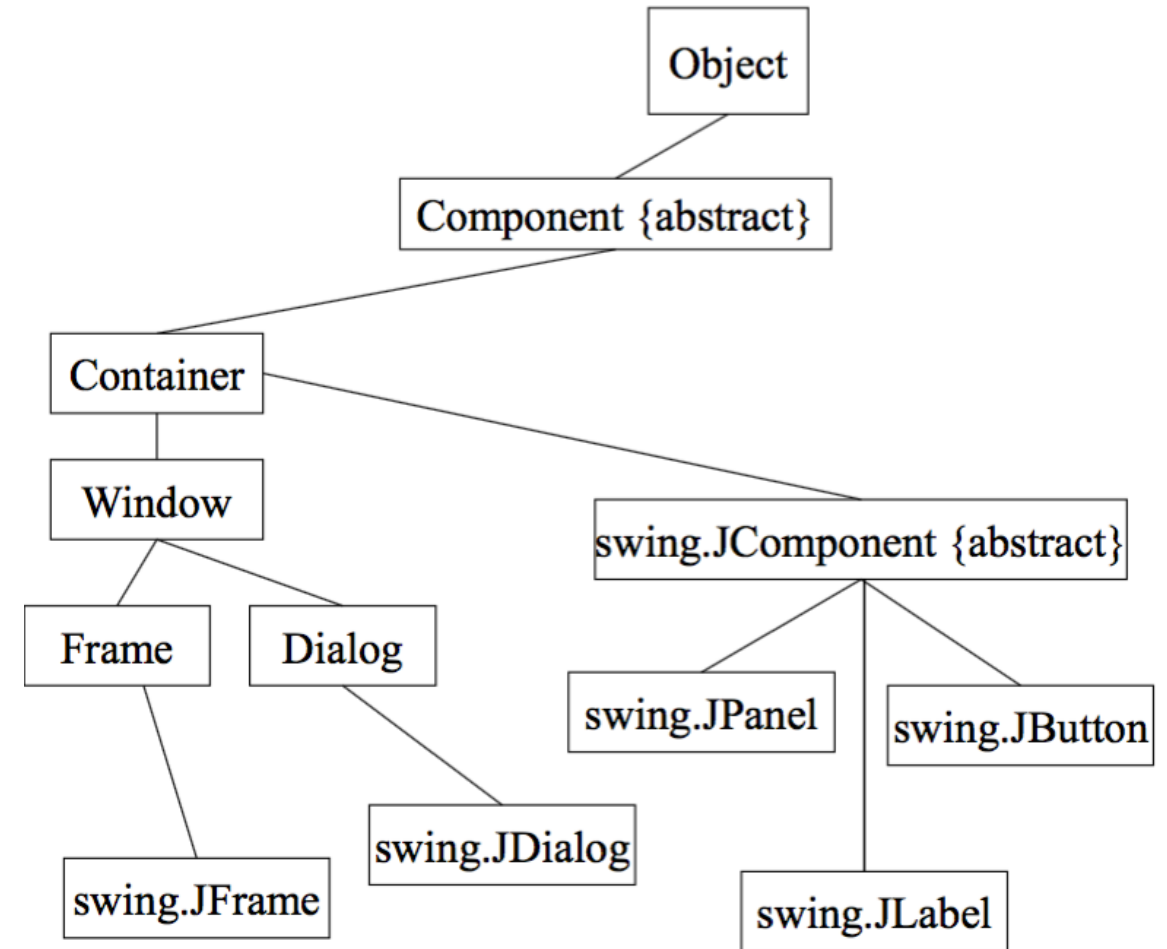
Les composants élémentaires

Les composants de haut-niveau (Frame, Dialog, Applet)

Swing

Les classes Swing héritent des classes AWT

Pour les reconnaître: **J**NomDeLaClasse



Swing

Avantages sur AWT

- Bibliothèque plus fournie
- Widgets plus design
- Meilleure portabilité graphique

Liens avec AWT

- Swing = spécialisation de AWT
(classes Swing héritent des classes AWT)
- Mécanisme de réaction aux événements reste le même

Composants Swing

- JLabel
 - getText()
 - setText()
 - alignement
- JButton
- JComboBox

démo



iutvannes.fr

JFrame

```
import javax.swing.*;

public class Cadre {
    // Attribut
    private JFrame demo;

    // Constructeur
    public Cadre ( String titre ) {
        demo = new JFrame(titre);
        demo.setSize ( 100, 100 );
        demo.setVisible ( true ); }
}
```

OU

```
import javax.swing.*;

public class Cadre extends JFrame {
    // Constructeur
    public Cadre ( String titre ) {
        super(titre);
        this.setSize ( 100, 100 );
        this.setVisible ( true ); }
}
```

Version utilisation du JFrame



Version héritage du JFrame

Ajouts de composants

Pour ajouter un widget à une fenêtre, on utilise la méthode `add(Component comp)` de la classe `Container`

```
import javax.swing.*;

public class Cadre extends JFrame {
    // Constructeur
    public Cadre ( String titre ) {
        super ( titre );
        // Ajout de composants graphiques
        // Un simple JLabel
        JLabel label1 = new JLabel ( "Un JLabel" );
        // Un simple JButton
        JButton myBut = new JButton ( "bouton" );
        // Ajout DANS la fenêtre
        this.add ( label1 );
        this.add ( myBut );
        this.setSize ( 100, 100 );
        this.setVisible ( true );
    }
}
```

démo

Ajouts de composants

Où placer les composants

En précisant les coordonnées et la taille de chaque objet graphique à l'intérieur du Container :

- utilisation de `setBounds (int x, int y, int width, int height)` de la classe `Component`.

Inconvénients ?

Gestionnaire de placement

30 gestionnaires qui implémentent `java.awt.LayoutManager`

Permet le placement automatique dans des zones prédéfinies, et adapté automatiquement en cas de redimensionnement

Utilisation

1. Associer un gestionnaire de placement à l'objet graphique de type Container

```
maFrame.setLayout(new TrucLayout());
```

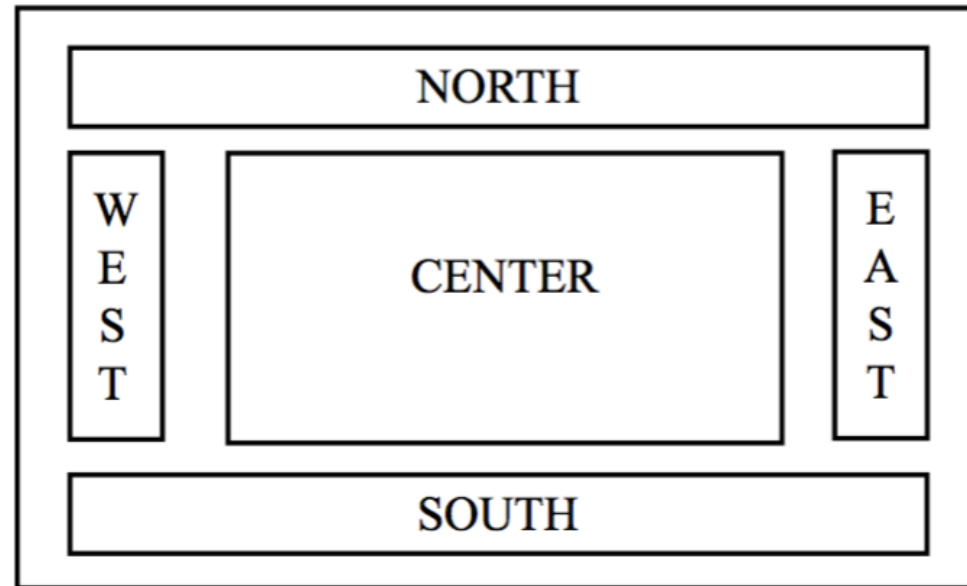
2. Ajouter les composants graphiques dans le Container avec ou sans utilisation d'une contrainte de placement

```
add (Component comp, Object constraints)  
maFrame.add(myButton, "enHautAGauche");
```

OU

```
maFrame.add(myButton);
```

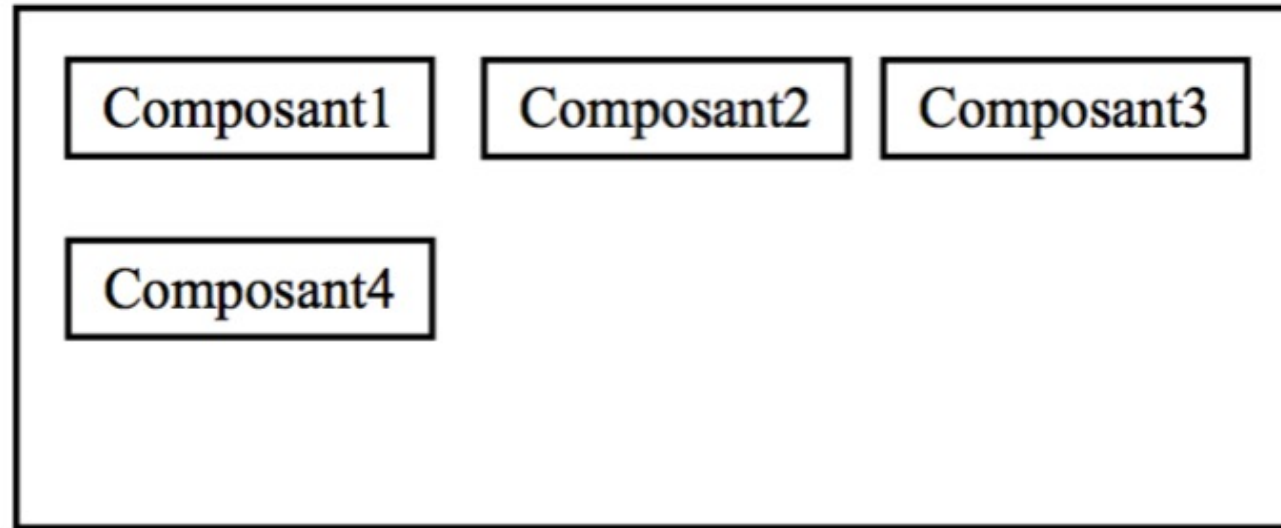
BorderLayout



```
// Définition du gestionnaire de placement BorderLayout
// pour l'objet « monCont » de type Container.
monCont.setLayout ( new BorderLayout() );
monCont.add ( unComposant, BorderLayout.NORTH );
monCont.add ( unComposant, BorderLayout.SOUTH );
monCont.add ( unComposant, BorderLayout.EAST );
monCont.add ( unComposant, BorderLayout.WEST );
monCont.add ( unComposant, BorderLayout.CENTER );
```

démo

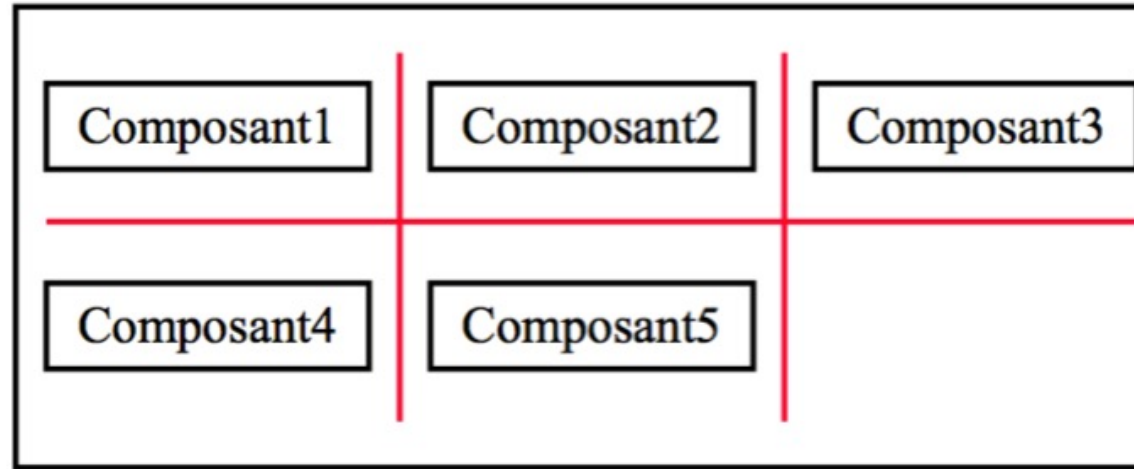
FlowLayout



```
// Définition du gestionnaire de placement FlowLayout
// pour l'objet « monCont » de type Container.
// Disposition des composants de gauche à droite.
monCont.setLayout ( new FlowLayout() );
monCont.add ( composant1 );
monCont.add ( composant2 );
monCont.add ( composant3 );
monCont.add ( composant4 );
```

démo

GridLayout



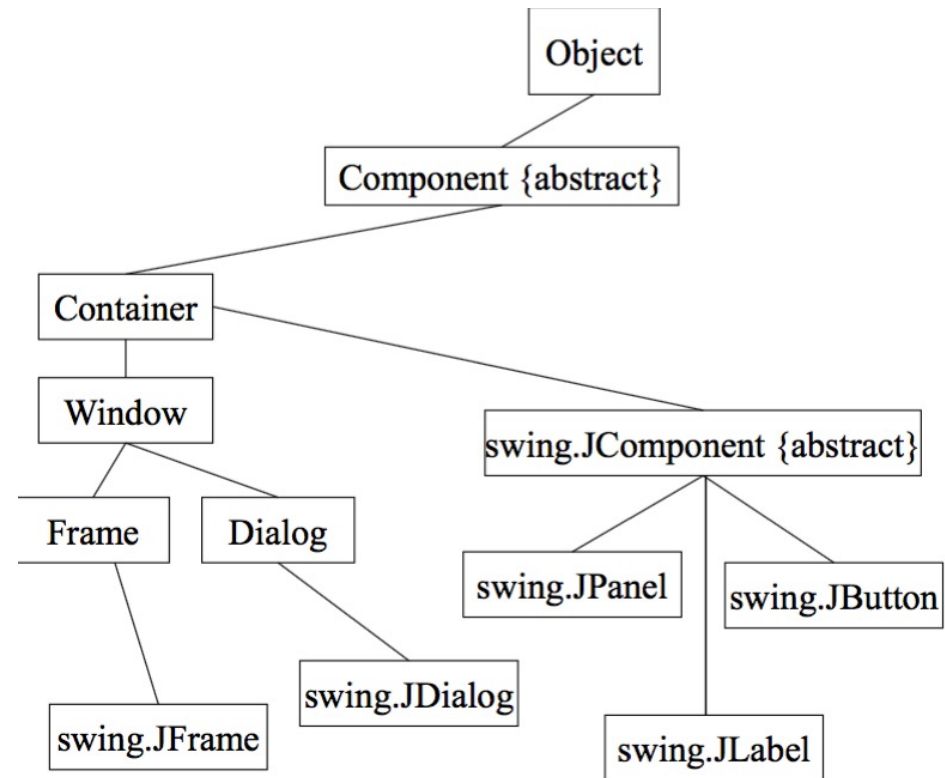
```
// Définition du gestionnaire de placement GridLayout
// pour l'objet « monCont » de type Container.
// Disposition des composants de gauche à droite et de haut en bas.
monCont.setLayout ( new GridLayout (2, 3) );
monCont.add ( composant1 );
monCont.add ( composant2 );
monCont.add ( composant3 );
monCont.add ( composant4 );
monCont.add ( composant5 );
```

démo

Les classes Container

Les seuls objets qui peuvent contenir d'autres objets graphiques.

Construction de l'interface par emboîtement de poupées russes.



Exemple

1. Créer une JFrame maJ
2. Associer à maJ le gestionnaire GridLayout (1,2)
3. A gauche mettre un JPanel myP
4. Associer à myP le gestionnaire GridLayout(2,1)
5. Dans myP, ajouter 2 JButton (Clic et Clac)
6. Ajouter une JList à droite

démo:
code à suivre

Code

```
import javax.swing.*;
import java.awt.*;

public class Cadre extends JFrame {
    // Déclarer ICI en private/protected TOUS les widgets ...
    private JButton clicButton;
    private JButton clacButton;
    private List listeDroite;
    // Constructeur
    public Cadre ( String titre ) {
        super ( titre );
        // Mise en place de TOUT le décor
        miseEnPlaceToutDecor( );
        // TOUTES DERNIERES INSTRUCTIONS
        // Fermeture de la fenêtre (clic sur la croix)
        this.setDefaultCloseOperation ( EXIT_ON_CLOSE );
        this.setSize ( 500, 500 );
        this.setVisible ( true );
    }
    // Lanceur
    public static void main ( String[] args ) {
        Cadre monC = new Cadre ( "Ma super démo" ); }
}
```

```
// Méthode qui place TOUS les widgets dans la fenêtre
private void miseEnPlaceToutDecor( ) {
    clicButton = new JButton ( "Clic" );
    clacButton = new JButton ( "Clac" );
    listeDroite = new List();
    JPanel panGche = new JPanel();
    // Panneau de gauche
    panGche.setLayout ( new GridLayout ( 2, 1 ) );
    panGche.add ( clicButton );
    panGche.add ( clacButton );
    // Liste à droite
    listeDroite.add ( "texte1" );
    listeDroite.add ( "texte2" );
    // Placement dans la fenêtre elle-même
    this.setLayout ( new GridLayout ( 1, 2 ) );
    this.add ( panGche );
    this.add ( listeDroite );
}
...
}
```