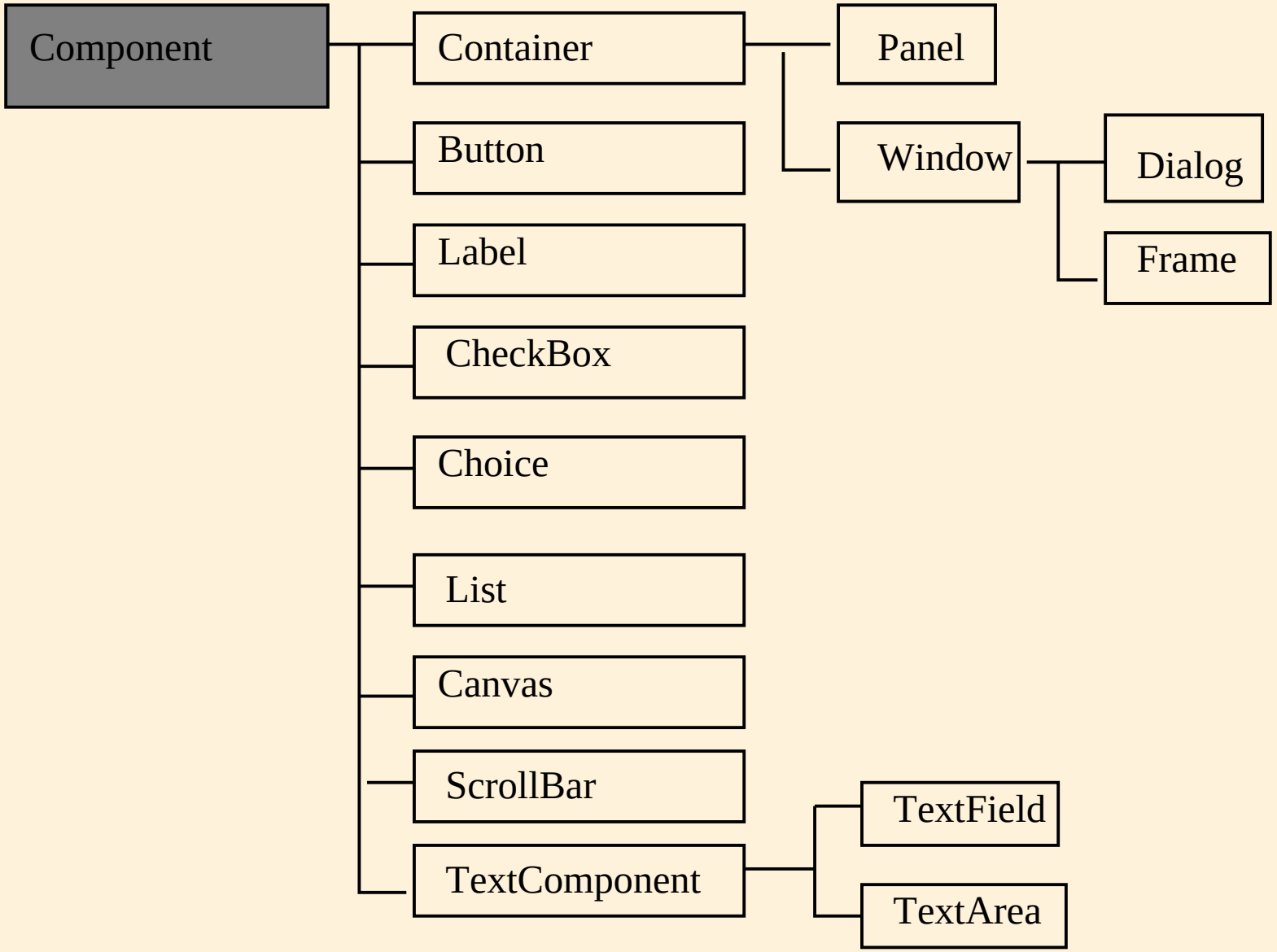
Programmation graphique

Ahmed Zidna
Département Informatique, IUT de Metz
Univerisité de Metz, Ile du Saulcy, F-57045 METZ
Bureau : B37
zidna@univ-metz.fr

Le package java.awt pour l'interface graphique I

- Tous les objets graphiques de l'**AWT** héritent de la classe abstraite **Component**
- Elle gère : l'affichage, les polices de caractères, le focus, le positionnement, les menus..etc.
- Les classes s'appuient sur l'interface graphique native du S.E.
  - Aspect de la plate forme hôte
  - avantage de bonne performance
- Une application AWT est totalement portable.

Le package java.awt pour l'interface graphique I



Les plates formes graphiques I

Pour une applications classique, cet ensemble peut suffire.

- Les fenêtres (frame et boite de dialogues)
- les menues déroulants
- les boutons, check-box, boutons radios
- les champs de saisie texte (mono ou muti-ligne)
- les champs de texte fixes
- les champs graphiques

Le package javax.swing I

- **Swing** permet de réaliser des applications dont l'aspect visuel ne dépend plus du S.E.
- La présentation graphique est assurée par la classe **UIManager**.
- Elle gère directement la présentation des composants graphiques avec la classe **LookAndFeel**
- **Swing** assure une portabilité plus complète des applications graphiques.
  - Inconvénient : applications moins rapides car utilisation de classes compilées.

Le package javax.swing I

- Les classes du package hérite de la classe **Container** de AWT.

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

- On distingue
  - Des classes qui définissent les composants graphiques qui héritent de **JComponent** qui elle même hérite de **Container**
  - Des classes qui jouent le rôle de containers

Les composants de swing I

- **JButton** $\Rightarrow$ boutons avec des libellés,
- **JCheckBox** $\Rightarrow$ cases à cocher,
- **JcomboBox** $\Rightarrow$ listes déroulantes,
- **JLabel** $\Rightarrow$ composants pour afficher un texte,
- **JList** $\Rightarrow$ composants avec sélection d'une ou plusieurs valeurs.
- **JRadioButton** $\Rightarrow$ boutons à choix exlusif,
- **JScrollBar** $\Rightarrow$ barres de défilement,
- **JScrollPane** $\Rightarrow$ panneaux défilants,
- **JTextComponent**, **JTextField** , **JTextArea** $\Rightarrow$ des textes de saisie.

Les conteneurs de swing I

- **JFrame** qui hérite de **Frame**.
  $\Rightarrow$ construit des fenêtre pourvues de titre et d'une bordure.
- **JPanel** qui hérite de **JComponent** :
  $\Rightarrow$ construit une zone graphique pour placer les compsants.
- **JTabbedPane** qui hérite de **JComponent** :
  $\Rightarrow$ construit un panneau associé à un onglet.

Exemple 0. Création de fenêtre graphique I

- j'importe les librairies

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

- j'instancie un objet de JFrame

```
JFrame m=new JFrame("mon cadre");
m.setLocation(300,200);// coin haut gauche du cadre
m.setSize(50,100);// fixe la taille en pixels
m.setVisible(true);// le cadre est rendu visible
```

Exemple1 : Dérivation de la classe Frame I

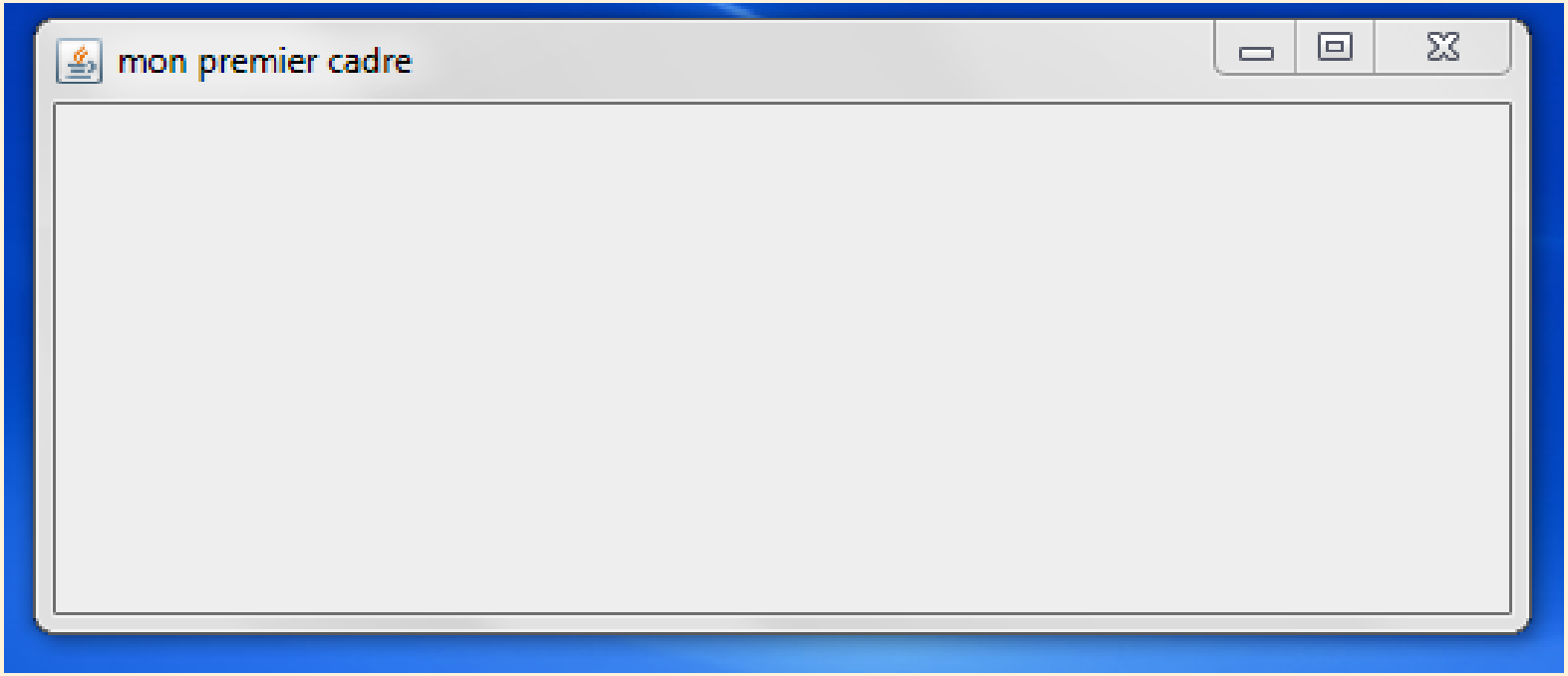
- On peut écrire une classe **Mon1erCadre** qui hérite de **Frame**.

```
import java.awt.*;
class Cadre1 extends JFrame{
// Constructeur
public Cadre1(String titre){
super(titre); setLocation(300,200);
setSize(50, 100);}
}
```

- et une classe **TestMon1erCadre** qui permet de la tester

```
class TestCadre1{
public static void main( String [] arg){
Cadre1 m =new Cadre1("mon cadre");
m.setVisible(true);}}
```

Voici mon 1 eme Cadre I



Gestion du positionnements des composants I

Pour positionner les eléments graphiques au sein de l'objet *Frame*, java propose plusieurs classes : les Layouts managers :

- **FlowLayout**
- **BorderLayout** :
- **GridLayout** :
- **GridBagLayout** :
- **CardLayout** :

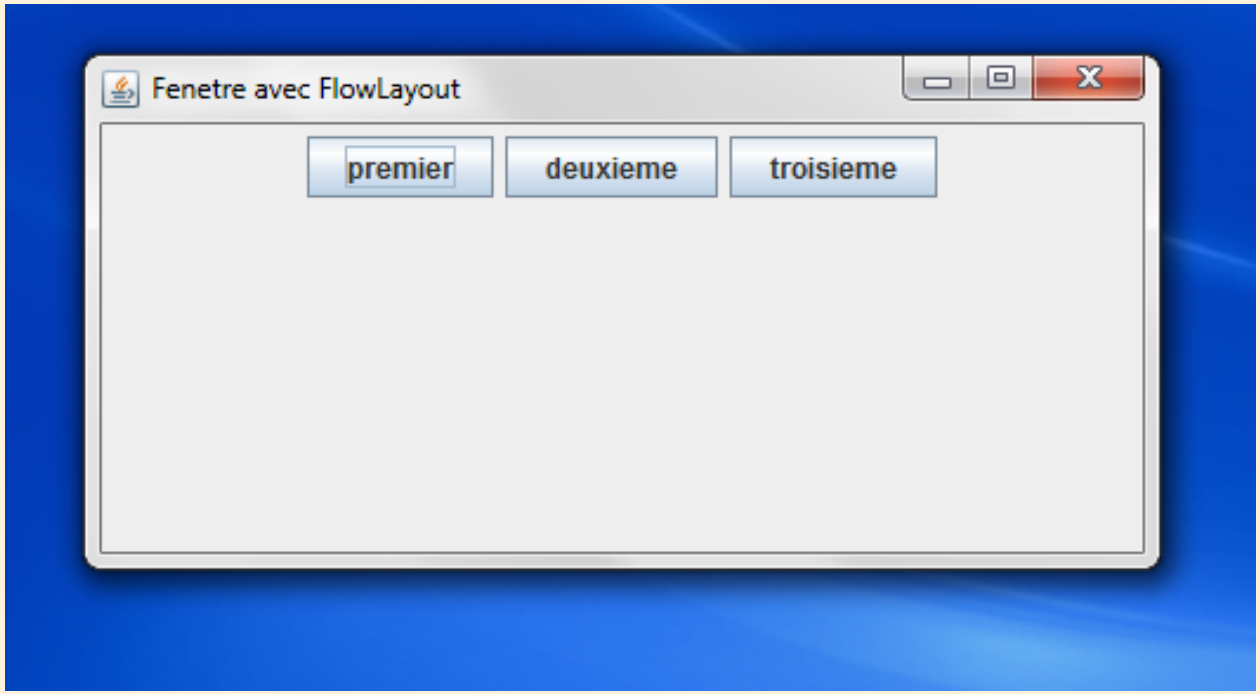
Exemple avec FlowLayout I

⇒ permet de positionner les eléments les uns après les autres

```
import java.awt.*;
public class fentereFlow extends JFrame{
public static void main (String [] args){
fenetreFlow f= new fenetreFlow();
f.setLayout(new FlowLayout());
f.add(new JButton ("premier"));
f.add(new JButton ("deuxieme"));
f.add(new JButton ("troisieme"));
f.setBounds(0,0,600,400);
f.setVisible(true);
}}
```

Exemple de FlowLayout I

- Voici l'affichage



- pour changer l'orientation

```
1   f.setComponentOrientation(ComponentOrientation.
        RIGHT_TO_LEFT)
```

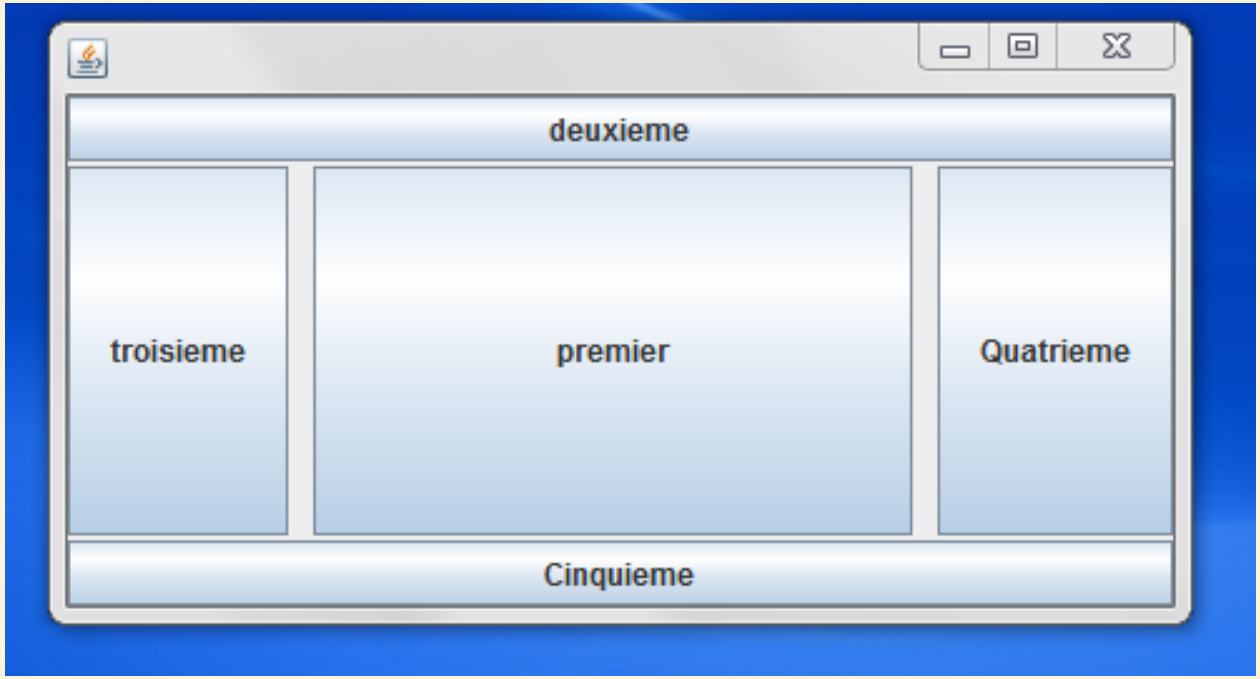
Exemple avec BorderLayout I

$\Rightarrow$ permet de diviser le conteneur en 5 parties correspondants aux points cardinaux et au centre.

```
1  import java.awt.*;
   public class fentereBord extends JFrame{
3  public static void main (String [] args){
   fenetreBord f= new fenetreBord();
5  BorderLayout b=new BorderLayout();
   b.setVgap(2); b.setHgap(10);
7  f.setLayout(b);
   f.add(new JButton ("premier"));
9  f.add("North",new JButton ("deuxieme"));
   f.add(new JButton ("troisieme"), BorderLayout.WEST);
11 f.setBounds(0,0,600,400);
   f.setVisible(true);}}
```

Exemple de BorderLayout I

- Voici l'affichage



- Par défaut, un composant est placé au centre.

Exemple avec GridLayout I

$\Rightarrow$ permet de positionner les composants sur une grille invisible. Il faut spécifier le nombre de lignes et colonnes.

```
import java.awt.*;
public class fentereGrid extends JFrame{
public static void main (String [] args){
fenetreGrid f= new fenetreGrid();
GridLayout g=new GridLayout(2,4);
g.setVgap(5); g.setHgap(5); f.setLayout(g);
f.add(new JButton("premier"));f.add(new JButton("deux"));
f.add(new JButton("trois"));f.add(new JButton("quatre"));
f.add(new JButton("cinq"));f.add(new JButton("six"));
f.add(new JButton("sept"));f.add(new JButton("huit"));
f.setBounds(0,0,600,400);
f.setVisible(true);}}
```

Exemple de GridLayout I

- Voici l'affichage

- Le nombre de lignes est toujours constant. Le nombre de colonnes s'adapte au nombre de composants.

Exemple avec GridBagLayout I

- Il s'appuie sur une grille, mais permet aux composants de s'étaler sur plusieurs lignes ou colonnes en fonction des contraintes (taille, poids , place disponibles..etc).
- Chaque composant est associé à un objet de type **GridBagConstraint**
- Les propriétés de ce type de contraintes sont les suivantes :
  1. **gridx** et **gridy** la ligne et la colonne de placement du composant.
  2. **gridheight** $\Rightarrow$ le nombre de ligne pour la hauteur.
  3. **gridwidth** $\Rightarrow$ le nombre de colonne pour la largeur.
  4. **anchor** $\Rightarrow$ pour la placement du composant si il est petit /espace : **CENTER, NORTH, NORTHEAST,...**
  5. **fill** $\Rightarrow$ pour le réajustement du compsant si il est petit/espace : **NONE, HORIZONTAL,...**

Exemple avec GridBagLayout I

```
import java.awt.*;
public class fenetreGridBag extends JFrame{
public static void main (String [] args){
fenetreGridBag f= new fenetreGridBag();
GridBagLayout gb=new GridBagLayout();
f.setLayout(gb);

JButton b1=new JButton("premier");
JButton b2=new JButton("second");
JButton b3=new JButton("troisieme");
```

Exemple avec GridBagLayout I

- Voici mes contraintes :

```
   GridBagConstraints gbc=new GridBagConstraints();
2  gbc.fill=GridBagConstraints.BOTH;
   gbc.anchor=GridBagConstraints.CENTER;
```

- Voici la position et la taille

```
1  gbc.gridx=1; gbc.gridy=1;
   gbc.gridwidth=4; gbc.gridheight=1;
```

- on associe les contraintes au premier bouton et on l'ajoute

```
   gb.setConstraints(b1,gbc);
2  f.add(b1);
```

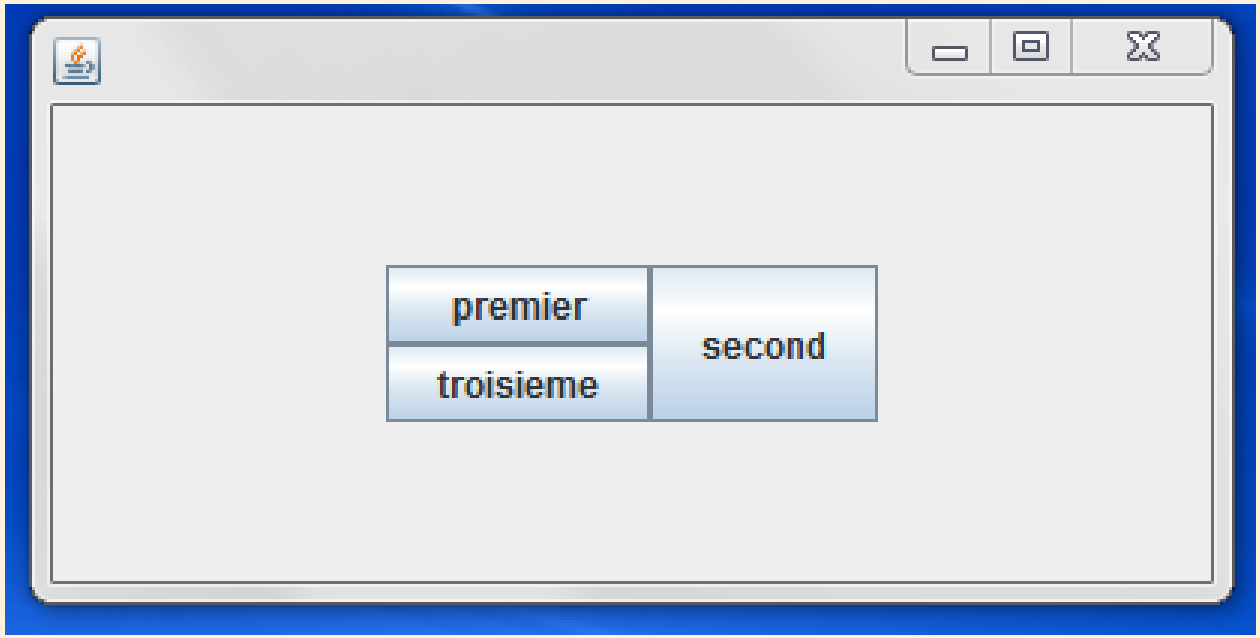
- On fait la même chose avec les autres boutons

Exemple avec GridBagLayout II

```
    gbc.gridx=5; gbc.gridy=1;
2   gbc.gridwidth=2; gbc.gridheight=3;
    gb.setConstraints(b2,gbc);
4   f.add(b2);
    gbc.gridx=1; gbc.gridy=3;
6   gbc.gridwidth=2; gbc.gridheight=1;
    gb.setConstraints(b3,gbc);
8   f.add(b3);
    f.setBounds(0,0,600,400);
10  f.setVisible(true);}}
```

Exemple de GridBagLayout I

- Voici l'affichage



Exemple2 : Décoration de la fenêtre I

Pour décorer la fenêtre, on utilise

- **setBackground()** pour la couleur de fond
- **paint()** appelée automatiquement dès que le contexte l'éxige. Un appel explicite à paint() peut être fait par **rePaint()**.
- Voici un programme qui affiche une fenêtre titré colorée. Elle contient une chaîne de caractères et un oval.

```
public class Cadre2 extends JFrame {
public Cadre2(String titre)
{
  super(titre);
  this.setLocation(300,200);
  this.setSize(200, 200);
  setDefaultCloseOperation(EXIT_ON_CLOSE);
    getContentPane().setBackground(Color.cyan);
}
```

Exemple2 : Décoration de la fenêtre II

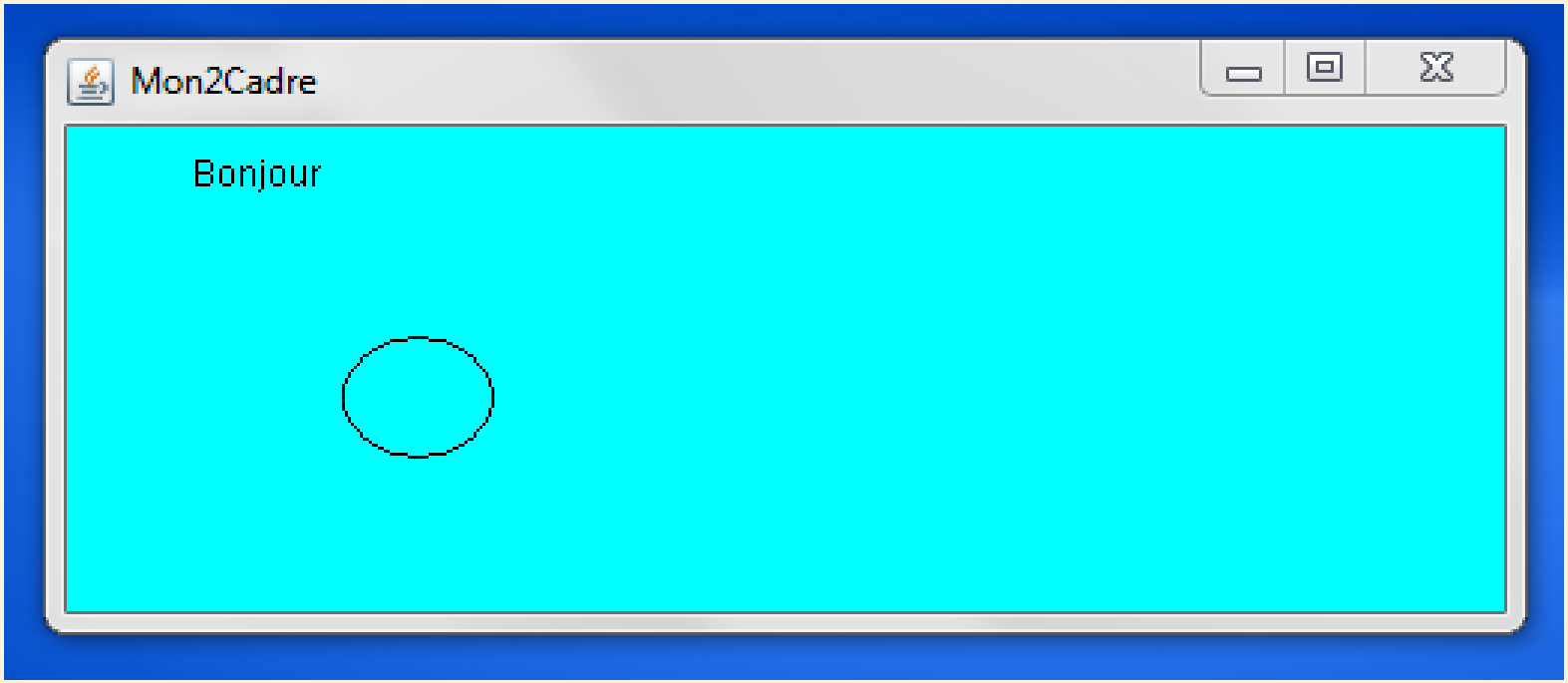
- un **Graphics** pour dessiner, tracer, écrire sur le fond d'un composant

```
1  public void paint(Graphics g)
   {  super.paint(g);
3    g.setColor(Color.black);
     g.drawString("Bonjour", 50, 50);
5    g.drawOval(100, 100, 50, 40);
   }
```

- Voici la classe **TestMon2emeCadre**

```
   public class TestCadre2 {
2    public static void main(String[] args) {
       Cadre2 c=new Cadre2("Mon2Cadre");
4      c.setVisible(true);
     }
6  }
```

Voici mon 2 eme Cadre I



Exemple 3 : Positionner et dimensionner une fenêtre I

Exemple : Supposons que l'interface occupe le quart de l'écran et soit placée au tiers de la diagonale et la chaîne "Salut !" soit placée au centre de la fenêtre.

- Pour paramétrer le cadre, on utilse la classe **Toolkit** qui fournit les dimensions de l'écran (l'unité est le pixel).

```
1  import java.awt.*;
   class Cadre3 extends JFrame{
3  public Cadre3(String titre){
   super(titre);
5  Toolkit tk = Toolkit.getDefaultToolkit();
```

- On récupère les dimensions de l'ecran

```
1  Dimension d = tk.getScreenSize();
   int Ox, Oy;//coin haut gauche de la fenetre
3  //recupere la hauteur et la largeur de l'ecran
   int he=(int)(d.getHeight());
5  int le=(int)(d.getWidth());
```

Exemple 3 : Positionner et dimensionner une fenêtre II

- la fenêtre doit prendre 1/4 de l'ecran et placée au 1/3 de la diagonale.

```
1   int hf = he/2; int lf = le/2;
    Ox = le/3; Oy = he/3;
```

- place le cadre et lui donne une couleur de fond

```
    setLocation(Ox,Oy);
2   setSize(lf, hf);
    setBackground(Color.pink);
4   }
```

- Voici ma fonction **paint**

Exemple 3 : Positionner et dimensionner une fenêtre III

```
   public void paint(Graphics g){
2  int l = getWidth(); //recupere largeur de la fenetre
   int h = getHeight(); //recupere hauteur de la fenetre
4  super.paint(g);
   g.setColor(Color.blue);
6  //on ecrit a partir du milieu
   g.drawString("Salut!",l/2, h/2);
8  }}
```

- Et pour tester tout cela

```
   class TestCadre3{
2  public static void main( String [] arg){
   Cadre3 m=new Cadre3("mon 3eme cadre");
4  m.setVisible(true);
   }}
```

Voici mon 3 eme Cadre I

Exemple 4. Ajouter des composants et FlowLayout I

Le programme affiche une fenêtre qui contient 3 champs de texte et 3 étiquettes et un bouton gérée par FlowLayout

```
1  import java.awt.*;
   class Cadre4 extends JFrame{
3  JTextField operandeGauche, operandeDroit,resultat;
   JLabel etiquetteGauche, etiquetteDroite,etiqResultat;
5  JButton bouton;
   //----- Constructeur -------------
7  public Cadre4(String titre){
   super(titre);
9  ..........
   setLayout(new FlowLayout());
```

- On ajoute les composants : 2 **JLabel**, 2**JTextField** et un **JButton**

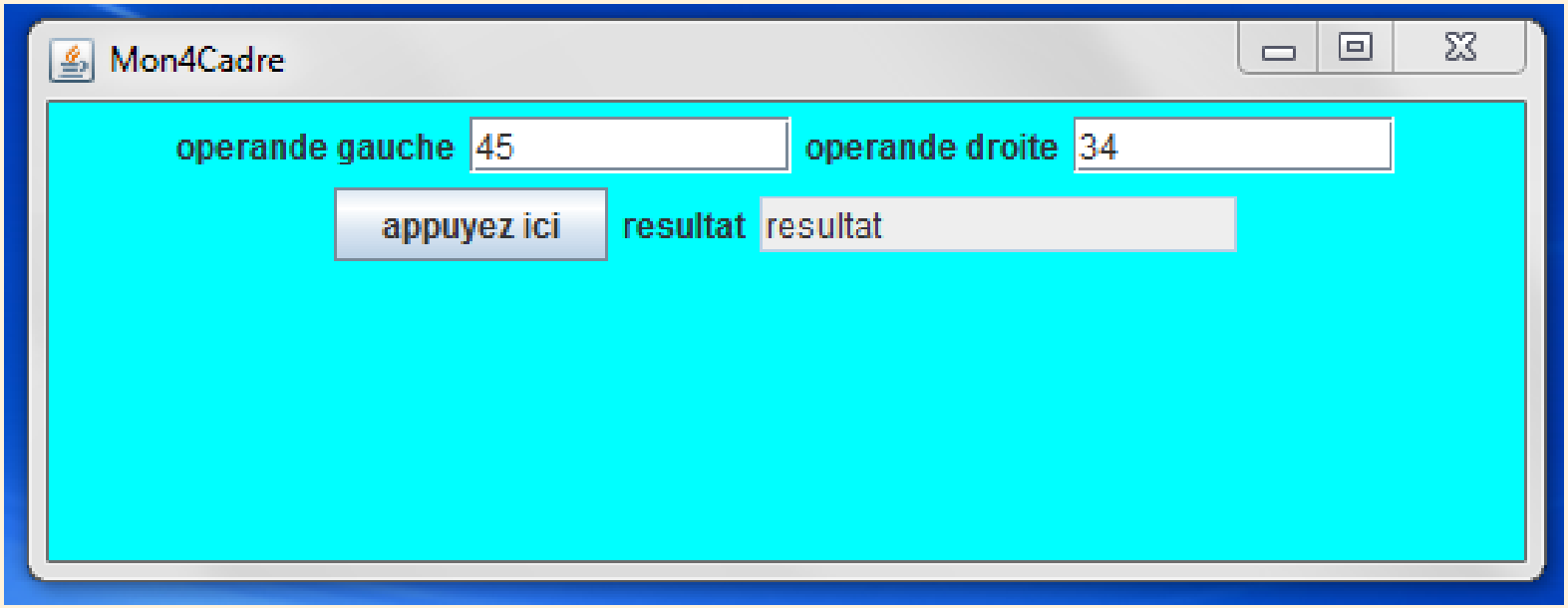
Exemple 4. Ajouter des composants et FlowLayout II

```
    etiquetteGauche = new JLabel("operande␣gauche");
2   add(etiquetteGauche); // accroche le composant
    operandeGauche = new JTextField("entree␣texte",10);
4   add(operandeGauche);// accroche le composant
    etiquetteDroite = new JLabel("operande␣droite");
6   add(etiquetteDroite); // accroche le composant
    operandeDroit = new JTextField("entree␣texte",10);
8   add(operandeDroit);// accroche le composant

10  bouton = new JButton("appuyez␣ici");
    add(bouton);
12
    etiqResultat = new JLabel("resultat");
14  add(etiqResultat);
    resultat = new JTextField("resultat␣", 15);
16  resultat.setEditable(false);
    add(resultat);
18  }
```

- Le programme devient illisible, il devient difficile d'en assurer la maintenance.

Voici mon 4 eme Cadre I



Exemple 5. Structurer le cadre :JPanel I

- Le cadre est organisé en 3 panneaux. Le layout est un **Borderlayout**
- Les composants sont accrochés aux panneaux au lieu d'être accrochés au cadre.
- Le cadre et les panneaux peuvent utiliser des gestionnaires de mise en place différents.
- La méthode **place()** permet de gèrer tout ce qui concerne le positionnement et les dimensions du cadre.

```
import java.awt.*;
class Mon5emeCadre extends JFrame{
JPanel  panneauNord,  // panneau nord
JPanel  panneauCentre, // panneau centre
JPanel  panneauSud; // panneau sud
JTextField  operandeGauche, operandeDroit,resultat;
JLabel  etiquetteGauche, etiquetteDroite,etiquetteResultat;
JButton bouton;
```

Exemple 5. Structurer le cadre :JPanel II

- fonction place() gère le positionnement et dimensions du cadre

```
    private void place(){
2   Toolkit tk = Toolkit.getDefaultToolkit();
    Dimension d = tk.getScreenSize();
4   int he,hf,le,lf;
    int Ox, Oy;
6   he=(int)(d.getHeight());
    le=(int)(d.getWidth());
8   hf = he/2; lf = le/2;
    Ox = le/3; Oy = he/3;
10  setLocation(Ox,Oy);
    setSize(lf, hf);
12  }
```

- Constructeur Mon5emeCadre

Exemple 5. Structurer le cadre :JPanel III

```
public Mon5emeCadre(String titre){
super(titre);
place()ă;
setBackground(Color.pink);
// pour disposer les composants sur la fenetre
setLayout(new BorderLayout());
```

- Création des composants

```
etiquetteGauche = new JLabel("operande␣gauche");
operandeGauche = new JTextField("entree␣texte",10);
etiquetteDroite = new JLabel("operande␣droite");
operandeDroit = new JTextField("entree␣texte",10);
bouton = new JButton("+");
etiquetteResultat = new JLabel("resultat");
resultat = new JTextField("resultat␣de␣l'operat", 15);
resultat.setEditable(false);
```

- Création et remplissage du panneau nord ——————

Exemple 5. Structurer le cadre :JPanel IV

```
   panneauNord = new Panel();
2  panneauNord.setLayout(new GridLayout(2,2));
   panneauNord.setBackground(Color.yellow);
4  panneauNord.add(etiquetteGauche);
   panneauNord.add(etiquetteDroite);
6  panneauNord.add(operandeGauche);
   panneauNord.add(operandeDroit);
```

- Création et remplissage du panneau Centre

```
1  panneauCentre = new JPanel();
   panneauCentre.setLayout(new FlowLayout());
3  panneauCentre.setBackground(Color.pink);
   panneauCentre.add(bouton);
```

- Création et remplissage du panneau Sud

Exemple 5. Structurer le cadre :JPanel V

```
    panneauSud = new JPanel();
2   panneauSud.setLayout(new FlowLayout());
    panneauSud.setBackground(Color.blue);
4   panneauSud.add(etiquetteResultat);
    panneauSud.add(resultat);
```

- accroche lees panneaux au cadre

```
1   add(panneauNord, BorderLayout.NORTH);
    add(panneauCentre, BorderLayout.CENTER);
3   add(panneauSud, BorderLayout.SOUTH);
    }
5   } // Mon5emeCadre
```

- Une classe pour tester

Exemple 5. Structurer le cadre :JPanel VI

```
1  class TestMon5emeCadre{
   public static void main( String [] arg)
3  {
   Mon5emeCadre m;
5  m = new Mon5emeCadre("mon␣5eme␣cadre");
   m.setVisible(true);
7  } // main
```

Structurer la fenêtre :une classe par composant I

L'introdution de nouvelles classes va permettre de regouper au sein des constructeurs les opérations de création propres à chaque composant. Pour traiter le problème précédent, on utilise les classes suivantes :

- **PanneauNord , PanneauCentre, PanneauSud** pour les trois panneaux
- **Etiquette** pour toutes les étiquettes
- **Entree** pour les champs de texte de saisie
- **Sortie** pour le champ de texte résultat de calcul
- **Bouton** pour la classe bouton $+$

- Voici ma classe Etiquette

Structurer la fenêtre :une classe par composant II

```
1  class Etiquette extends JLabel{
   public Etiquette(String titre, Container c){
3    super(titre);
     c.add(this);
5    }
   }
```

- Voici ma classe Entree

```
   class Entree extends JTextField // champ de texte de saisie
       {
2  public Entree(String message, Container c){
     super(message, 10);
4    c.add(this);
     }
6  } // Entree
```

- Voici ma classe JBouton

Structurer la fenêtre :une classe par composant III

```
class Bouton extends Button{
public Bouton(String libelle, Container c){
  super(libelle);
  c.add(this);
  }
} // Bouton
```

- Voici ma classe Sortie

```
class Sortie extends JTextField{
public Sortie(String message, Container c){
  super(message, 15);
  setEditable(false);
  c.add(this);}
} // Sortie
```

Mon PanneauNord I

- Mon PanneauNord avec 2 entrées et deux etiquettes

```
class PanneauNord extends JPanel{
Entree operandeGauche, operandeDroit;
Etiquette etiquetteGauche, etiquetteDroite;

public PanneauNord(){
 setBackground(Color.yellow);
 setLayout(new GridLayout(2,2));
 etiquetteGauche=new Etiquette("oper gauche", this);
 etiquetteDroite=new Etiquette("oper droit", this);
 operandeGauche=new Entree("entree de nombre", this);
 operandeDroit=new Entree("entree de nombre", this);
}
} // PanneauNord
```

Mon PanneauCentre I

- Voici mon panneau PanneauCentre une etiquette et un bouton.

```
1  class PanneauCentre extends JPanel{
   Bouton bouton;
3  public PanneauCentre(){
   setLayout(new FlowLayout());
5  bouton = new Bouton("+", this);
   }
7  } // PanneauCentre
```

Mon PanneauSud I

- Voici mon panneau PanneauSud une etiquette et une sortie

```
1  class PanneauSud extends JPanel{
   Sortie  resultat;
3  Etiquette etiquetteResultat;
   public PanneauSud(){
5  setLayout(new FlowLayout());
   etiquetteResultat = new Etiquette("resultat", this);
7  resultat = new Sortie("resultat de l'oper", this);
   }
9  }// PanneauSud
```

Mon Cadre I

Voici mon Mon5emeCadre qui comporte 3 panneaux

```
1  class Mon5emeCadre extends Frame{
   PanneauNord  panneauNord;
3  PanneauCentre panneauCentre;
   PanneauSud  panneauSud;
5  void place(){.......} // place
   //---Constructeur Mon10emeCadre--------
7  public Mon5emeCadre(String titre){
   super(titre); place();
9  setLayout(new BorderLayout());
   //accroche les panneaux Nord, Centre et Sud au cadre
11 panneauNord = new PanneauNord();
   add(panneauNord, BorderLayout.NORTH);
13 panneauCentre = new PanneauCentre();
   add(panneauCentre, BorderLayout.CENTER);
15 panneauSud = new PanneauSud();
   add(panneauSud, BorderLayout.SOUTH);}
17 }
```

Mon Cadre II

```
1  class TestMon5emeCadre{
   public static void main( String [] arg){
3  Mon8emeCadre m=new Mon5emeCadre("mon␣5eme␣cadre");
   m.setVisible(true);
5  }}
```

Voici mon 4 eme Cadre I