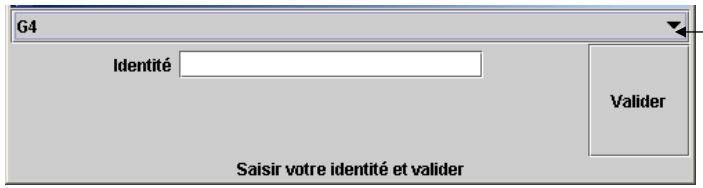
TD JAVA n°7: Interface Graphique

Exercice 1 : Première Application Swing

Nous allons écrire une première application graphique utilisant des composants `Swing`

1. Dans un premier temps, écrire la classe **`PremierPanel`** héritée de **`JPanel`** qui permet de placer les éléments dans un panel de la façon suivante :



Ce composant contient les chaînes : "G4", "G5" et "G6"

2. Ecrire l'application **`TestPremiereFenetre`** qui permet à une fenêtre (largeur 500 - hauteur 150) de recevoir un objet de classe `PremierPanel` et de l'afficher à l'écran.

3. **Gestion des événements :**

↳ 3.1 Lorsque l'utilisateur appuie sur le bouton **Valider**, vous devez récupérer le groupe et l'identité et afficher un message qui tient compte de ces deux paramètres : `Bonjour Toto du groupe G5`

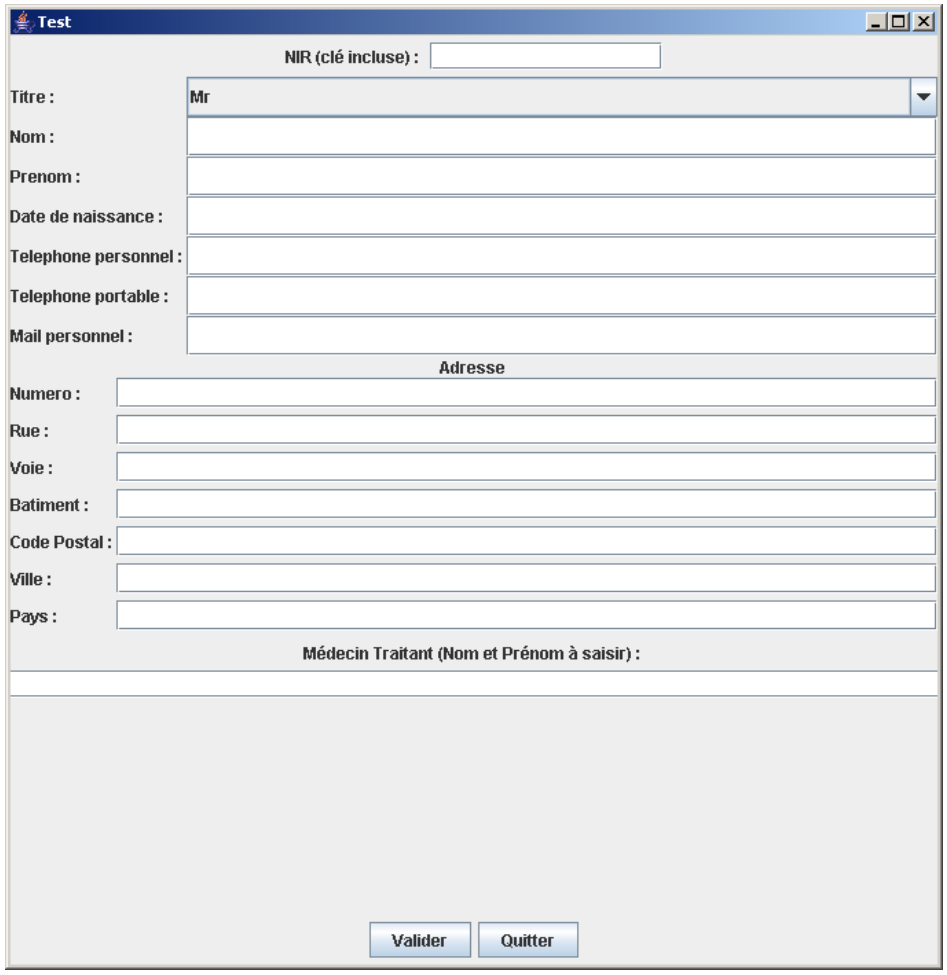
Proposer **3 solutions différentes** pour implémenter ce traitement.
(classe interne anonyme, classe interne membre et objet est son propre écouteur…)

↳ 3.2 Lorsque l'utilisateur **clique sur le panel**, le message : « Saisir votre identité et valider » est affiché comme label et la zone de saisie est vidée (on revient ainsi dans l'état initial)

Pour les plus rapides….

Exercice 2 : Application Graphique de `CabinetMedical`

Commencer à réfléchir à l'écriture de la classe **`PanelCreerPatient`** qui permet d'obtenir le panel suivant



Annexe : Extrait de la documentation

javax.swing

Class JComboBox

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JComboBox
```

Object	**getItemAt**(int index) Returns the list item at the specified index.
int	**getItemCount**() Returns the number of items in the list.
JComboBox.KeySelectionManager	**getKeySelectionManager**() Returns the list's key-selection manager.
int	**getMaximumRowCount**() Returns the maximum number of items the combo box can display without a scrollbar
ComboBoxModel	**getModel**() Returns the data model currently used by the JComboBox.
ListCellRenderer	**getRenderer**() Returns the renderer used to display the selected item in the JComboBox field.
int	**getSelectedIndex**() Returns the first item in the list that matches the given item.
Object	**getSelectedItem**() Returns the currently selected item.

Constructor Summary

JLabel() Creates a JLabel instance with no image and with an empty string for the title.
JLabel(Icon image) Creates a JLabel instance with the specified image.
JLabel(Icon image, int horizontalAlignment) Creates a JLabel instance with the specified image and horizontal alignment.
JLabel(String text) Creates a JLabel instance with the specified text.
JLabel(String text, Icon icon, int horizontalAlignment) Creates a JLabel instance with the specified text, image, and horizontal alignment.
JLabel(String text, int horizontalAlignment) Creates a JLabel instance with the specified text and horizontal alignment.

getText

```
public String getText()
```

Returns the text string that the label displays.
Returns:
a String
See Also:
setText(java.lang.String)

setText

```
public void setText(String text)
```

Defines the single line of text this component will display. If the value of text is null or empty string, nothing is displayed.

The default value of this property is null.

This is a JavaBeans bound property.
See Also:
setVerticalTextPosition(int), setHorizontalTextPosition(int), setIcon(javax.swing.Icon)

Constructor Summary

JTextField() Constructs a new TextField.
JTextField(Document doc, String text, int columns) Constructs a new JTextField that uses the given text storage model and the given number of columns.
JTextField(int columns) Constructs a new empty TextField with the specified number of columns.
JTextField(String text) Constructs a new TextField initialized with the specified text.
JTextField(String text, int columns) Constructs a new TextField initialized with the specified text and columns.

Constructor Detail

JTextField

```
public JTextField()
```

Constructs a new TextField. A default model is created, the initial string is null, and the number of columns is set to 0.

JTextField

```
public JTextField(String text)
```

Constructs a new TextField initialized with the specified text. A default model is created and the number of columns is 0.
Parameters:
text - the text to be displayed, or null

JTextField

```
public JTextField(int columns)
```

Constructs a new empty TextField with the specified number of columns. A default model is created and the initial string is set to null.
Parameters:
columns - the number of columns to use to calculate the preferred width. If columns is set to zero, the preferred width will be whatever naturally results from the component implementation.

java.awt.event

Interface MouseListener

Method Summary

void	**mouseClicked**(MouseEvent e) Invoked when the mouse has been clicked on a component.
void	**mouseEntered**(MouseEvent e) Invoked when the mouse enters a component.
void	**mouseExited**(MouseEvent e) Invoked when the mouse exits a component.
void	**mousePressed**(MouseEvent e) Invoked when a mouse button has been pressed on a component.
void	**mouseReleased**(MouseEvent e) Invoked when a mouse button has been released on a component.

Correction TD JAVA n°7: Interface Graphique

Exercice 1 :

1. La classe **`PremierPanel`** ⇒ Disposition des composants :

Etape 1 : ***Choisir le gestionnaire***
Etape 2 : ***Positionner et Identifier les composants***

↳ **Au Nord : JComboBox** (Attention pas de JChoice en Swing ⇒ JCombobox => 1 seul choix
JList => plusieurs choix)

↳ **Au Centre :** 2 Composants => **Jlabel et JtextField**

~~jl_monLabel = new JLabel("Identité");~~
~~jt_monText = new JTextField(20);~~
~~add(jl_monLabel,"Center");~~
~~add(jt_monText,BorderLayout.CENTER);~~

⇒ **NON ! ! !** les composants vont se superposer l'un sur l'autre et on n'en verra qu'un (JTextField ; le dernier ajouté)…

⇒ **Solution :** Il faut utiliser un panel avec un nouveau gestionnaire de mise en forme (FlowLayout par défaut pour pouvoir positionner les deux composants **)** ⇒ Création d'un nouveau panel Disponibilité (« panel intermédiare ») qui contiendra ces deux composants ….

↳ **Au Est : Jbutton**
↳ **Au Sud : Jlabel**

Déclaration des composants (**`JButton`**, **`JTextField`**, **`JComboBox`**, **`JLabel`**) *comme attributs de la classe, car on en aura besoin pour la gestion des événements …*


```
import java.awt.*;      // pour le gestionnaire
import javax.swing.*;   // pour les composants swing

public class PremierPanel extends JPanel {

    JComboBox jcb_listeGroupe;
    JPanel jp_panelInfo;
    JLabel jl_monLabel;
    JTextField jt_monText;
    JButton jb_Valider;
    JLabel jl_message;

    public PremierPanel() //Constructeur
    {

        // Choix du gestionnaire
        setLayout (new BorderLayout());

        //JComboxBox
        jcb_listeGroupe= new JComboBox();
        jcb_listeGroupe.addItem("G4");
        jcb_listeGroupe.addItem("G5");
        jcb_listeGroupe.addItem("G6");
        add(jcb_listeGroupe,BorderLayout.NORTH);

        // Deux composants dans la même cellule:il faut les grouper dans un JPanel
        // sinon il s'afficheront l'un sur l'autre et on ne verra que le dernier
        jp_panelInfo = new JPanel();
        jl_monLabel = new JLabel("Identité");
        jp_panelInfo.add(jl_monLabel);
        jt_monText = new JTextField(20);
        // si on ne met pas de texte, on aura un JTextField « applati »
        // là on précise une taille : 20 colonnes
        jp_panelInfo.add(jt_monText);
        add(jp_panelInfo,BorderLayout.CENTER);
```



```
        // Bouton:
        jb_Valider = new JButton("Valider");
        add(jb_Valider,BorderLayout.EAST);

        //Message
    jl_message = new JLabel("Saisir votre identité et valider");
    jl_message.setHorizontalAlignment(JLabel.CENTER); // pour centrer le texte
        add(jl_message,BorderLayout.SOUTH);
    }
}
```

2. L'application **`TestPremiereFenetre`** qui permet à une fenêtre (de largeur 300 et de hauteur 200) de recevoir un objet de classe PremierPanel et de l'afficher à l'écran :

```
public class TestPremiereFenetre {

    public static void main(String args[])
        {
        // Mise en place de la fenetre :  titre et fermeture
        JFrame maFenetre = new JFrame();
        maFenetre.setTitle("Premiere Application graphique");
        maFenetre.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Création le Panel principal
        JPanel mainPanel = new PremierPanel ();
        //Rq : polymorphisme : JPanel ou PremierPanel

        //ajout du panel sur la fenêtre…
        maFenetre.getContentPane().add(mainPanel);

        // Taille de la fenêtre principale et Affichage
        maFenetre.setSize(500,150);   //hauteur:150 pixels-largeur:500
        maFenetre.setVisible(true);   // rend visible la fenêtre

        }// fin main


} // fin classe TestPremiereFenetre
```

Remarque : *Création du Panel principal*

En raison du polymorphisme, on peut choisir **`JPanel`** *ou* **`PremierPanel`** *comme référence :*

```
JPanel mainPanel = new PremierPanel ();     ou
PremierPanel mainPanel = new PremierPanel ();
```

3. Gestion des événements : Modifications à apporter à la classe PremierPanel:

// Ecouteur associé au bouton Valider

➢ 1ère solution : classe interne anonyme

```
import java.awt.*;      // pour le gestionnaire
import javax.swing.*;   // pour les composants swing

// pour la gestion des evenements
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
// ou plus généralement import java.awt.event.*;
```

Dans le constructeur :

```
public PremierPanel() //Constructeur
      {
      ……
      // Bouton:
      jb_Valider = new JButton("Valider");
      add(jb_Valider,BorderLayout.EAST);
      // Ecouteur associé au bouton Valider
      jb_Valider.addActionListener (new ActionListener()   // Ecouteur
            {
            public void actionPerformed(ActionEvent e)
                  {
                  jl_message.setText("Bonjour "+ jt_monText.getText() + " du groupe " +
            jcb_listeGroupe.getSelectedItem());
                  }
                  });……
      }
```

Remarque : `jt_monText` et `jcb_listeGroupe` doivent être déclarés comme attributs de la classe pour être utilisée dans la méthode actionPerformed, puisqu'on aurait très bien pu écrire :

➢ 2ème solution : classe interne membre

```
import java.awt.*;      // pour le gestionnaire
import javax.swing.*;   // pour les composants swing

// pour la gestion des evenements
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
// ou plus généralement import java.awt.event.*;

public class PremierPanel extends JPanel {

      JComboBox jcb_listeGroupe;
      JPanel jp_panelInfo;
      JLabel jl_monLabel;
      JTextField jt_monText;
      JButton jb_Valider;
      JLabel jl_message;

      public PremierPanel() //Constructeur
            {  ……..
            // Bouton:
            jb_Valider = new JButton("Valider");
            add(jb_Valider,BorderLayout.EAST);
            // Ecouteur associé au bouton Valider
            jb_Valider.addActionListener (new MonActionListener());
        ……..
       } // fin Constructeur
```



```
      ////////////////////////////////////////////////////
      /////////// Gestionnaire d'événements implémenté /////
      /////////// en Classe Interne Membre //////////////////
      class MonActionListener implements ActionListener
        {
        public void actionPerformed(ActionEvent e)
          {
            jl_message.setText("Bonjour "+ jt_monText.getText() + " du groupe " +
jcb_listeGroupe.getSelectedItem());
          }
      }/ / fin classe interne MonActionListener

}     // fin Classe principale PremierPanel
```

➢ 3ème solution : objet est son propre écouteur

```
import java.awt.*;      // pour le gestionnaire
import javax.swing.*;   // pour les composants swing

// pour la gestion des evenements
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
// ou plus généralement import java.awt.event.*;

public class PremierPanel extends JPanel implements ActionListener {

      JComboBox jcb_listeGroupe;
      JPanel jp_panelInfo;
      JLabel jl_monLabel;
      JTextField jt_monText;
      JButton jb_Valider;
      JLabel jl_message;

      public PremierPanel() //Constructeur
            {  ……..
            // Bouton:
            jb_Valider = new JButton("Valider");
            add(jb_Valider,BorderLayout.EAST);
            // Ecouteur associé au bouton Valider
            jb_Valider.addActionListener (this);
        ……..
       } // fin Constructeur

      ////////////////////////////////////////////////////////////////////////
      ////Redéfinition de la méthode actionPerformed dans la classe PremierPanel
      //// si objet est son propre écouteur
      public void actionPerformed(ActionEvent arg0) {
            jl_message.setText("Bonjour "+ jt_monText.getText() + " du groupe " +
jcb_listeGroupe.getSelectedItem());
            }// fin actionPerformed

}// fin classe principale PremierPanel
```

➢ **Gestion des clics:** *Pour réagir aux événements de la souris => interface* **MouseListener**

Modifications à apporter dans le constructeur du Panel, rajouter un écouteur aux événments de la souris….

```
import java.awt.*;      // pour le gestionnaire
import javax.swing.*;   // pour les composants swing

// pour la gestion des evenements
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
// ou plus généralement import java.awt.event.*;

public class PremierPanel extends JPanel {

      JComboBox jcb_listeGroupe;
      JPanel jp_panelInfo;
      JLabel jl_monLabel;
      JTextField jt_monText;
      JButton jb_Valider;
      JLabel jl_message;

      public PremierPanel() //Constructeur
            {
            ……
            ……

            // Pour réagir aux événements de la souris
            // => interface MouseListener
            this.addMouseListener ( new MouseAdapter()
            {
            public void mouseClicked (MouseEvent e)
                  {
                  jl_message.setText("Saisir votre identité et valider");
                  jt_monText.setText("");
                  }
            });
       } // fin Constructeur

      ……
      ……

}     // fin Classe principale PremierPanel
```


Il vaut mieux déclarer les* `JButton` *et* `JTextField` *comme attribut (variable globale), car on en aura besoin pour la gestion des événements…

***Exercice 2 : Application Graphique de* `CabinetMedical`**

…. Ne pas corriger en TD : Les laisser chercher ! ! !

```
public class PanelCreerPatient extends JPanel {

      //tous les composants nécessaires
      // Attributs du patient
      private static final String [] TITRES={"Mr","Mme","Melle"};
      private JComboBox jcb_saisieTitre = new JComboBox (TITRES);
      private JTextField jtx_saisieNom = new JTextField (30);
      private JTextField jtx_saisiePrenom = new JTextField (30);
      private JTextField jtx_saisieTelPerso = new JTextField (10);
      private JTextField jtx_saisieTelPortable = new JTextField (10);
      private JTextField jtx_saisieMailPerso = new JTextField (120);

      private JTextField jtx_saisieNIR = new JTextField (15);
      private JTextField jtx_saisieMedecin = new JTextField (120);
      private JTextField jtx_saisieDate = new JTextField(10);

      // Attributs de l'adresse

      private JTextField jtx_saisieAdresseNumero= new JTextField (6);
      private JTextField jtx_saisieAdresseRue = new JTextField (120);
      private JTextField jtx_saisieAdresseVoie = new JTextField (120);
      private JTextField jtx_saisieAdresseBatiment = new JTextField (120);
      private JTextField jtx_saisieAdresseCodePostal = new JTextField (60);
      private JTextField jtx_saisieAdresseVille = new JTextField (120);
      private JTextField jtx_saisieAdressePays = new JTextField (120);


      // Panels intermédiaires nécessaires à la mise en page

      // Boutons de validation
      JButton jb_Valider = new JButton("Valider");
      JButton jb_Quitter = new JButton("Quitter");


      ////////////////////////////////
      // Constructeur
      ////////////////////////////////
      public PanelCreerPatient(){

            //////////////////////////
            // Panel NIR
            //////////////////////////
            JPanel jp_panelNIR = new JPanel(); //flowlayout par défaut ...
            jp_panelNIR.add(new JLabel ("NIR (clé incluse) : "));
            jp_panelNIR.add(this.jtx_saisieNIR);
      jtx_saisieNIR.setToolTipText("Saisir ici les 15 carcatères du NIR sans espace");

            ////////////////////////////////////////////////////////////////
            // Panel Caractéristique Personne (hors adresse et descandant)
            ////////////////////////////////////////////////////////////////
            JPanel jp_panelPersonne= new JPanel();
            // sera composé de 2 panneaux :
            // => panneau des labels et panneaux des saisies...

            // Creation du panneau des labels : 6 lignes, 1 colonne
            // avec un écart de 5 pixels entre les composants
            // car les infos sont alignées !!!
            JPanel jp_panelLabels = new JPanel(new GridLayout(7,1));
            jp_panelLabels.add(new JLabel ("Titre : "));
            jp_panelLabels.add(new JLabel ("Nom : "));
            jp_panelLabels.add(new JLabel ("Prenom : "));
            jp_panelLabels.add(new JLabel ("Date de naissance : "));
            jp_panelLabels.add(new JLabel ("Telephone personnel : "));
```

```
            jp_panelLabels.add(new JLabel ("Telephone portable : "));
            jp_panelLabels.add(new JLabel ("Mail personnel : "));
            //jp_panelLabels.add(new JLabel ("Adresse : "));

            // Creation du panneau de saisie : 6 lignes, 1 colonne
            // avec un écart de 5 pixels entre les composants
            // Saisie des caractéristiques d'une personne
            JPanel jp_panelSaisiePers = new JPanel(new GridLayout(7,1));
            jp_panelSaisiePers.add(this.jcb_saisieTitre);
            jp_panelSaisiePers.add(this.jtx_saisieNom);
            jp_panelSaisiePers.add(this.jtx_saisiePrenom);
            jp_panelSaisiePers.add(this.jtx_saisieDate);
jtx_saisieDate.setToolTipText("Vous devez saisir votre date au format jj/mm/aaaa");

            jp_panelSaisiePers.add(this.jtx_saisieTelPerso);
            jp_panelSaisiePers.add(this.jtx_saisieTelPortable);
            jp_panelSaisiePers.add(this.jtx_saisieMailPerso);

            ////////////////////////////////////////////////////////////
            // Panel Caractéristiques des Adresses
            ////////////////////////////////////////////////////////////
            JPanel jp_panelAdresse= new JPanel();

            // labels des adresse
            JPanel jp_panelAdresse_Labels = new JPanel(new GridLayout(7,1,5,5));
            jp_panelAdresse_Labels.add(new JLabel ("Numero : "));
            jp_panelAdresse_Labels.add(new JLabel ("Rue : "));
            jp_panelAdresse_Labels.add(new JLabel ("Voie : "));
            jp_panelAdresse_Labels.add(new JLabel ("Batiment : "));
            jp_panelAdresse_Labels.add(new JLabel ("Code Postal : "));
            jp_panelAdresse_Labels.add(new JLabel ("Ville : "));
            jp_panelAdresse_Labels.add(new JLabel ("Pays : "));
            // champs de saisie des adresses
            JPanel jp_panelAdresse_SaisiePers = new JPanel(new GridLayout(7,1,5,5));
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseNumero);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseRue);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseVoie);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseBatiment);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseCodePostal);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdresseVille);
            jp_panelAdresse_SaisiePers.add(this.jtx_saisieAdressePays);

            jp_panelAdresse.setLayout(new BorderLayout());
      jp_panelAdresse.add(new JLabel ("Adresse ",JLabel.CENTER),BorderLayout.NORTH);
            jp_panelAdresse.add(jp_panelAdresse_Labels,BorderLayout.WEST);
            jp_panelAdresse.add(jp_panelAdresse_SaisiePers,BorderLayout.CENTER);
            ////////////////////////////////////////////////////////////
            //jp_panelSaisiePers.add(jp_panelAdresse);


            // Ajout des sous-panneaux ...
            // utilisation d'un gestionnaire Borderlayout
            // avec écart de 5 pixels entre les composants
            jp_panelPersonne.setLayout(new BorderLayout());
            jp_panelPersonne.add(jp_panelLabels,BorderLayout.WEST);
            jp_panelPersonne.add(jp_panelSaisiePers,BorderLayout.CENTER);

            ////////////////////////
            // Panel Medecin
            ////////////////////////
            JPanel jp_panelMedecin= new JPanel();//flowlayout par défaut ...
      jp_panelMedecin.add(new JLabel ("Médecin Traitant (Nom et Prénom à saisir) : "));
            jp_panelMedecin.add(this.jtx_saisieMedecin);

            ////////////////////////////////////////////////////////////
            // Panel Patient => regroupant les panels Personne et Medecin HORS NIR
```



```
            ////////////////////////
            JPanel jp_panelPatient= new JPanel();
            jp_panelPatient.setLayout(new GridLayout(3,1));
            jp_panelPatient.add(jp_panelPersonne);
            jp_panelPatient.add(jp_panelAdresse);
            jp_panelPatient.add(jp_panelMedecin);

            // Panel Boutons
            ////////////////////////
            JPanel jp_panelBoutons= new JPanel();//flowlayout par défaut ...
            jp_panelBoutons.add(jb_Valider);
            jp_panelBoutons.add(jb_Quitter);


            ////////////////////////////////////
            // AU FINAL
            this.setLayout(new BorderLayout());
            this.add(jp_panelNIR,BorderLayout.NORTH);
            this.add(jp_panelPatient,BorderLayout.CENTER);
            this.add(jp_panelBoutons,BorderLayout.SOUTH);

    } // fin constructeur

    }// fin Classe PanelCreerPatient
```