

TD3 : Application graphique

Ce TD, en forme de tutoriel, vous permet d'aborder la programmation d'applications graphiques avec la librairie Swing.

Un exemple complet : Le BMI

Netbeans offre un outil (appelé Matisse) qui facilite la réalisation d'interfaces graphiques mais afin de mieux comprendre le code produit, nous allons d'abord tout faire « à la main » sur un exemple simple : un calcul de BMI.

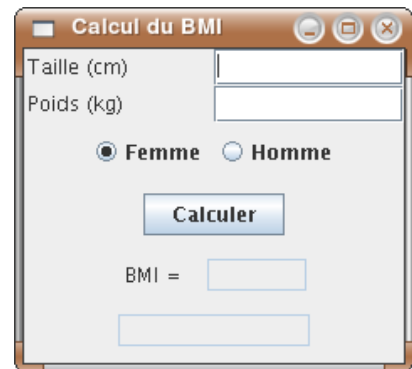
Présentation du problème

Le BMI (Body Mass Index, Indice de masse corporelle) est calculé à partir du poids et de la taille d'un individu. Il renseigne sur sa corpulence.

La formule est : **BMI = poids / (taille)²** où le poids est exprimé en **kg** et la taille en **m**.

Le tableau suivant donne la corpulence en fonction du BMI et du sexe :

	Maigre	Normal	Surcharge	Adiposité	Obésité
Femme	< 19	[19, 24 [[24, 30 [[30, 40 [> 40
Homme	< 20	[20, 25 [[25, 30 [[30, 40 [> 40



Version 1 : Une fenêtre vide

Commençons par le début et affichons une fenêtre vide.

```
package mcd.al2.bmi;
import javax.swing.*;

public class BMIFrame extends JFrame {

    public BMIFrame() {
        super("Calcul du BMI");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        pack();
    }

    public static void main(String[] args) {
        BMIFrame bmi = new BMIFrame();
        bmi.setVisible(true);
    }
}
```

- **super...** : appelle le constructeur de **JFrame** en donnant le titre de la fenêtre.
- **setDefaultCloseOperation...** : indique que l'application doit se terminer si la fenêtre est fermée.
- **pack()** : demande de calculer la taille de la fenêtre ainsi que les tailles et position des ses composants (aucun pour le moment)
- **setVisible()** : rend la fenêtre visible.

Version 2 : Ajout des champs de saisie et du bouton de calcul

On va considérer pour l'instant que l'application ne permet pas de choisir le sexe. De plus, aucune action n'est encore associée au bouton.

```
package mcd.al2.bmi;
import javax.swing.*;
import java.awt.*; // Container est défini dans AWT

public class BMIFrame extends JFrame {

    private JTextField poidsTF;
    private JTextField tailleTF;
    private JButton calculer;
    private JTextField bmiTF;

    public BMIFrame() {
        super("Calcul du BMI");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        poidsTF = new JTextField(3); // Taille en nb de caractères
        tailleTF = new JTextField(3);
        bmiTF = new JTextField(5);
        bmiTF.setEditable(false); // Ce n'est pas un champ éditable
        calculer = new JButton("Calculer");
        setLayout(new FlowLayout());
        // Le layout par défaut (Border) ne nous convient pas
        add(tailleTF);
        add(poidsTF);
        add(calculer);
        add(bmiTF);
        pack();
    }

    public static void main(String[] args) {
        BMIFrame bmi = new BMIFrame();
        bmi.setVisible(true);
    }
}
```

Version 3 : Action associée au bouton

Ajoutons ce qu'il faut pour que la bouton lance effectivement le calcul et affiche le résultat. Ne gérons pas encore les erreur pour l'instant. Il faut importer le package **java.awt.event** qui contient tous les événements. Ceci doit être ajouté au constructeur.

```
calculer.addActionListener( new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        calculer();
    }
});
```

Il faut aussi ajouter une méthode

```

private void calculer() {
    int poids = Integer.parseInt(poidsTF.getText());
    int taille = Integer.parseInt(tailleTF.getText());
    int bmi = poids * 10000 / taille / taille;
    bmiTF.setText(Integer.toString(bmi));
}

```

Version 4 : Gestion des erreurs

Envisageons une boîte de dialogue qui s'affiche si une des entrées est manquantes ou incorrecte. La méthode de calcul devient :

```

private void calculer() {
    int poids=0, taille=0;
    try {
        taille = Integer.parseInt(tailleTF.getText());
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(this,
            "La taille est manquante ou invalide.",
            "Champs invalides",
            JOptionPane.ERROR_MESSAGE);
        return;
    }
    try {
        poids = Integer.parseInt(poidsTF.getText());
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(this,
            "Le poids est manquant ou invalide.",
            "Champs invalides",
            JOptionPane.ERROR_MESSAGE);
        return;
    }

    int bmi = poids * 10000 / taille / taille;
    bmiTF.setText(Integer.toString(bmi));
}

```

Version 5 : Sexe et surcharge

Ajoutons à présent la possibilité de choisir le sexe et affichons le résultat de la surcharge ou non. Pour cela, il faut :

- Déclarer les nouveaux composants

```

private ButtonGroup sexeGroup;
private JRadioButton sexeM;
private JRadioButton sexeF;
private JTextField resultTF; // surcharge ou pas

```

- Les créer dans le constructeur et associer les boutons au même groupe

```

sexeGroup = new ButtonGroup();
sexeF = new JRadioButton("Femme");
sexeF.setSelected(true);
sexeGroup.add(sexeF);
sexeM = new JRadioButton("Homme");
sexeGroup.add(sexeM);
resultTF = new JTextField(10);
resultTF.setEditable(false);

```

- Les ajouter à la fenêtre

```

add(sexeF);
add(sexeM);
add(resultTF);

```

- Nous devons aussi modifier la méthode de calcul. Afin de ne pas tout mélanger, nous en extrayons les parties purement de calcul

```

private int calculerBMI(int taille, int poids) {
    return poids * 10000 / taille / taille;
}

private String calculerSurcharge(int bmi, boolean homme) {
    if( homme )
        if (bmi<20) return "Maigre";
        else if (bmi<25) return "Normal";
        else if (bmi<30) return "Surcharge";
        else if (bmi<40) return "Adiposite";
        else return "Obesite";

    else
        if (bmi<19) return "Maigre";
        else if (bmi<24) return "Normal";
        else if (bmi<30) return "Surcharge";
        else if (bmi<40) return "Adiposite";
        else return "Obesite";
}

```

- La fin de la méthode calculer() devient

```

int bmi = calculerBMI(taille,poids);
bmiTF.setText(Integer.toString(bmi));
boolean homme = sexeM.isSelected();
String surcharge = calculerSurcharge(bmi,homme);
resultTF.setText(surcharge);

```

Version 6 : Séparation de la vue et du modèle

Pour l'instant, nous mélangeons dans une même classe le code lié au calcul (le modèle) et celui lié à l'interface graphique (la vue). Nous verrons plus tard, l'architecture MVC (Modèle-Vue-Contrôleur) qui permet une bonne séparation entre ces 2 parties. Ici, pour faire simple, sortons les méthodes de calcul et plaçons les dans une classe dédiée (par exemple : BMI).

- Il faut remplacer le mot `private` par `static`. Vous vous rappelez de ce que cela signifie ?
- Il faut ajouter quelque chose lors de l'appel. Vous voyez quoi ?

Version 7 : Format des entrées

Il est possible de contrôler précisément ce qui peut être accepté dans les champs de saisie via des `JFormattedTextField`.

Exemple pour le poids.

- Changez la déclaration de poidsTF
- Remplacez sa création par (il faudra le bon import pour NumberFormat) :

```
poidsTF = new JFormattedTextField(NumberFormat.getIntegerInstance());
poidsTF.setColumns(3);
```

Version 8 : Mise en page

Attaquons-nous à présent à la mise en page correcte des éléments graphiques.

```
JPanel panel0 = new JPanel();
panel0.setLayout(
    new BorderLayout(panel0, BorderLayout.PAGE_AXIS));
JPanel panel1 = new JPanel(new GridLayout(2,2));
panel1.add(new JLabel("Taille (cm)"));
panel1.add(tailleTF);
panel1.add(new JLabel("Poids (kg)"));
panel1.add(poidsTF);
panel0.add(panel1);
JPanel panel2 = new JPanel();
panel2.add(sexeF);
panel2.add(sexeM);
panel0.add(panel2);
JPanel panel3 = new JPanel();
panel3.add(calculer);
panel0.add(panel3);
JPanel panel4 = new JPanel();
panel4.add(new Label("BMI = "));
panel4.add(bmiTF);
panel0.add(panel4);
JPanel panel5 = new JPanel();
panel5.add(resultTF);
panel0.add(panel5);
setLayout(new FlowLayout());
add(panel0);
```

Ceci n'est qu'une proposition obtenue avec des gestionnaires de mise en page simples. Un meilleur résultat peut-être obtenu avec un gestionnaire plus élaboré comme le `GridBagLayout`. Matisse propose même un gestionnaire propre encore plus élaboré.

Version 9 : Un menu

Ajoutons un petit menu.

```
JMenuBar menuBar = new JMenuBar();
JMenu menuFile = new JMenu("BMI");
JMenuItem menuExit = new JMenuItem("Quitter");
menuExit.addActionListener( new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
});
JMenuItem menuAbout = new JMenuItem("A propos...");
menuAbout.addActionListener( new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        about();
    }
});
menuFile.add(menuAbout);
menuFile.add(menuExit);
menuBar.add(menuFile);
setJMenuBar(menuBar);
```

```
private void about() {
    JOptionPane.showMessageDialog(this,
        "Atelier Logiciel 2G - Programme de test !");
}
```

L'outil Netbeans : Matisse

Matisse vous permet de développer plus rapidement vos interfaces graphiques. Voyez le tutoriel fourni avec Netbeans : <http://www.netbeans.org/kb/55/quickstart-gui.html>

Application : BMI et Matisse

Refaites l'application BMI en utilisant Matisse.

Vous pouvez aussi y apporter les améliorations suivantes :

- Le bouton n'est pas actif si un des champs de saisie est vide.
- Enlever le bouton. Le résultat est adapté dès que les valeurs des champs de saisie sont modifiées