Les Fenêtres

Éric Gillon
Thomas Rey

Lycée Marlioz

4 décembre 2012

Introduction

Jusqu'à présent nos programmes manquaient de convivialité.
Nous étions restés à l'informatique des années 80...

source de l'image : `http://histoire.info.free.fr`

Introduction (suite)

Aujourd'hui, nous allons faire un bond en avant et créer des fenêtres.

- Ce diaporama présentera les bases ;
- Ensuite nous verrons comment créer des classes pour simplifier le code de nos programmes (voir document papier distribué).

Sommaire

Boîtes de dialogue

Pour poser une question ou afficher un résultat le mode « console » n'est plus satisfaisant pour les programmeurs que vous êtes. . .Nous allons donc créer et utiliser :

- des boîtes de confirmation ;

Boîtes de dialogue

Pour poser une question ou afficher un résultat le mode « console » n'est plus satisfaisant pour les programmeurs que vous êtes. . .Nous allons donc créer et utiliser :

- des boîtes de confirmation ;
- des boîtes d'entrée de texte ;

Boîtes de dialogue

Pour poser une question ou afficher un résultat le mode « console » n'est plus satisfaisant pour les programmeurs que vous êtes. . .Nous allons donc créer et utiliser :

- des boîtes de confirmation ;
- des boîtes d'entrée de texte ;
- des boîtes de message.

Boîte de confirmation

Une boîte de confirmation demande un « clic » à l'utilisateur (oui/non, OK/Annuler/Ignorer, ...). La réponse est un `int`.
On la déclare ainsi :

Boîte de confirmation

Une boîte de confirmation demande un « clic » à l'utilisateur (oui/non, OK/Annuler/Ignorer, ...). La réponse est un `int`. On la déclare ainsi :

```
int reponse;
reponse = JOptionPane.showConfirmDialog(null,// position
      "Vous avez tout compris ?", // la question
      "Question",// intitulé de la boîte
      JOptionPane.YES_NO_OPTION,// type de boutons
      JOptionPane.INFORMATION_MESSAGE);// icône
```

Boîte de confirmation

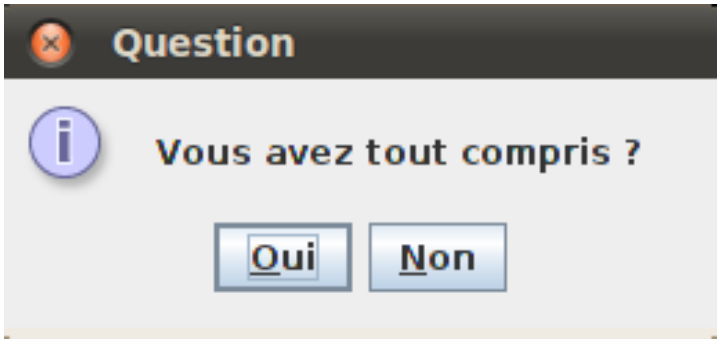
Une boîte de confirmation demande un « clic » à l'utilisateur (oui/non, OK/Annuler/Ignorer, ...). La réponse est un `int`. On la déclare ainsi :

```
int reponse;
reponse = JOptionPane.showConfirmDialog(null,// position
      "Vous avez tout compris ?", // la question
      "Question",// intitulé de la boîte
      JOptionPane.YES_NO_OPTION,// type de boutons
      JOptionPane.INFORMATION_MESSAGE);// icône
```

Penser à indiquer : `import javax.swing.JOptionPane;` dans le préambule (mais Eclipse vous le rappelle...)

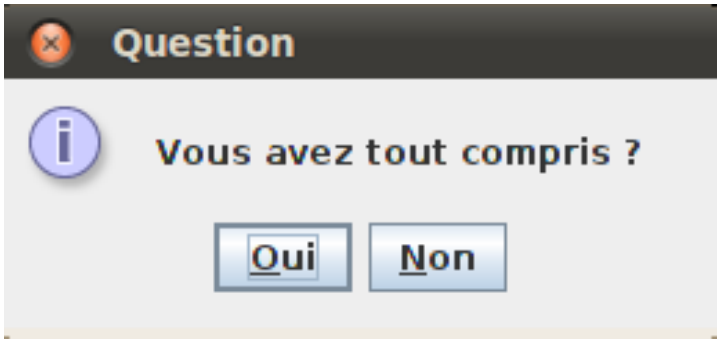
Boîte de confirmation

On obtient :



Boîte de confirmation

On obtient :

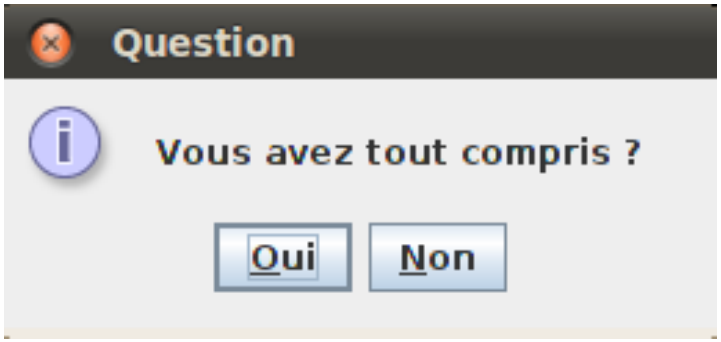


Après le clic la boîte disparaît et la valeur de la variable `reponse` est :

- 0 si on clique sur oui ;

Boîte de confirmation

On obtient :



Après le clic la boîte disparaît et la valeur de la variable `reponse` est :

- 0 si on clique sur oui ;
- 1 si on clique sur non.

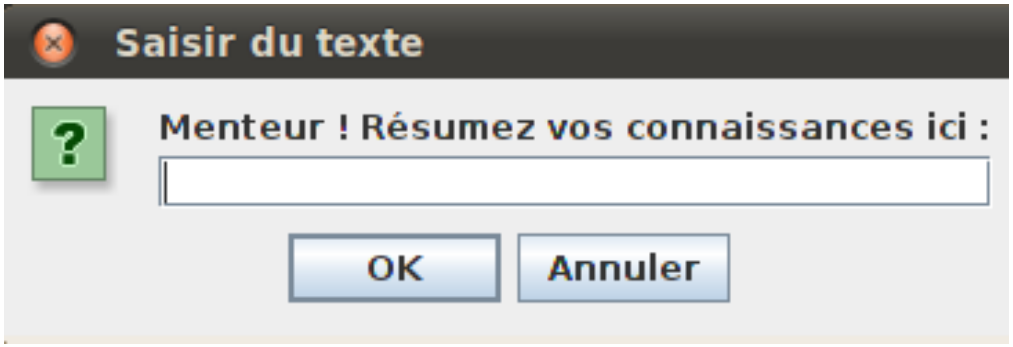
Boîte d'entrée de texte

Sur le même modèle, que fait le code suivant ?

```
String reponse2;
reponse2 = JOptionPane.showInputDialog(null,
    "Menteur ! Résumez vos connaissances ici
       :",
    "Saisir du texte",
    JOptionPane.QUESTION_MESSAGE);
```

Réponse

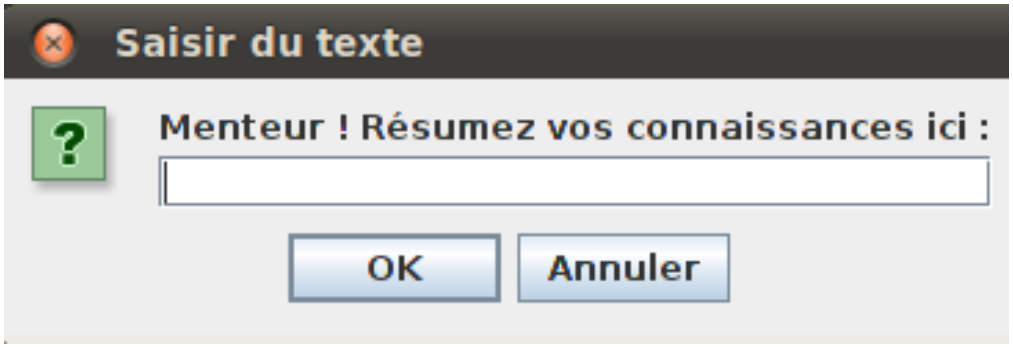
On obtient :



La variable `reponse2` contient :

Réponse

On obtient :



La variable `reponse2` contient :

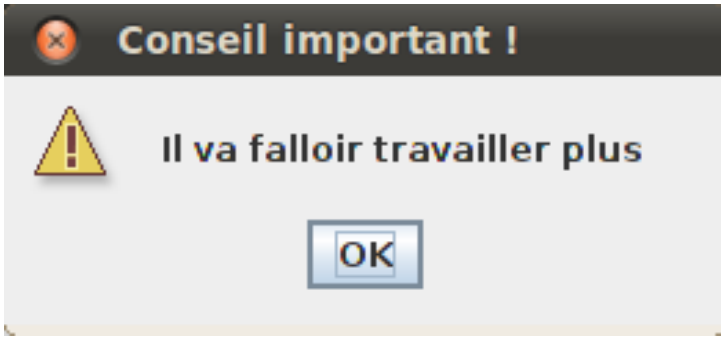
- le texte saisi si on a cliqué sur OK ;
- `null` si on a cliqué sur Annuler.

Boîte de message

Parfois, on souhaite juste afficher une réponse ou une information :

```
JOptionPane.showMessageDialog(null,
    "Il va falloir travailler plus",
    "Conseil important !",
    JOptionPane.WARNING_MESSAGE);
```

On obtient :



Sommaire

Principes

Pour créer une fenêtre plus élaborée en Java, plusieurs méthodes sont possibles (avec `swing`, `awt`, ...)
Nous avons choisi d'utiliser `swing` dont les principes sont les suivants :

Principes

Pour créer une fenêtre plus élaborée en Java, plusieurs méthodes sont possibles (avec `swing`, `awt`, ...)
Nous avons choisi d'utiliser `swing` dont les principes sont les suivants :

- créer une `JFrame` (la fenêtre) ;

Principes

Pour créer une fenêtre plus élaborée en Java, plusieurs méthodes sont possibles (avec `swing`, `awt`, ...)
Nous avons choisi d'utiliser `swing` dont les principes sont les suivants :

- créer une `JFrame` (la fenêtre) ;
- créer un `JPanel` (un panneau) qui contiendra ce que nous placerons dans la fenêtre.

Ajouter une fenêtre

```
import javax.swing.JFrame
public static void main(String[] args) {
   JFrame mafenetre = new JFrame();
   mafenetre.setVisible(true);
   // On définit le titre de la fenêtre :
   mafenetre.setTitle("Devoir Maison d'ISN n 4 - T.Rey");
   //sa taille :
   mafenetre.setSize(400,300);
   // sa position à l'écran :
   mafenetre.setLocation(1400, 200);
   // Enfin on lui indique de quitter lorsqu'on clique
   // sur le bouton de fermeture de la fenêtre
   mafenetre.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
```

Quelques méthodes

On écrit `nomdelafenetre.` suivi de :

Quelques méthodes

On écrit `nomdelafenetre.` suivi de :

- `setResizable(false);` pour empêcher le redimensionnement de la fenêtre ;

Quelques méthodes

On écrit `nomdelafenetre.` suivi de :

- `setResizable(false);` pour empêcher le redimensionnement de la fenêtre ;
- `setAlwaysOnTop(true);` pour garder la fenêtre au premier plan ;

Quelques méthodes

On écrit `nomdelafenetre.` suivi de :

- `setResizable(false);` pour empêcher le redimensionnement de la fenêtre ;
- `setAlwaysOnTop(true);` pour garder la fenêtre au premier plan ;
- `setUndecorate(true);` pour rendre les contours et boutons de contrôles invisibles ;
- ...

Création d'un panneau de contenu

Pour ajouter du contenu à une fenêtre, il faut créer un « panel » ou panneau. Pour cela :

Création d'un panneau de contenu

Pour ajouter du contenu à une fenêtre, il faut créer un « panel » ou panneau. Pour cela :

- importer la classe `javax.swing.JPanel` par la commande `import javax.swing.JPanel` (au tout début du programme) ;

Création d'un panneau de contenu

Pour ajouter du contenu à une fenêtre, il faut créer un « panel » ou panneau. Pour cela :

- importer la classe `javax.swing.JPanel` par la commande `import javax.swing.JPanel` (au tout début du programme) ;
- créer un `JPanel` par : `JPanel panneau = new JPanel();`

Création d'un panneau de contenu

Pour ajouter du contenu à une fenêtre, il faut créer un « panel » ou panneau. Pour cela :

- importer la classe `javax.swing.JPanel` par la commande `import javax.swing.JPanel` (au tout début du programme) ;
- créer un `JPanel` par : `JPanel panneau = new JPanel();`
- indiquer à la fenêtre que c'est ce panneau qui est son « contentPane » :
  `mafenetre.setContentPane(panneau);`

Création d'un panneau de contenu

Pour ajouter du contenu à une fenêtre, il faut créer un « panel » ou panneau. Pour cela :

- importer la classe `javax.swing.JPanel` par la commande `import javax.swing.JPanel` (au tout début du programme) ;
- créer un `JPanel` par : `JPanel panneau = new JPanel();`
- indiquer à la fenêtre que c'est ce panneau qui est son « contentPane » :
  `mafenetre.setContentPane(panneau);`

On peut changer la couleur de fond :

```
panneau.setBackground(Color.cyan);
```

Résultat

On obtient :

Sommaire

Les « conteneurs »

Nous avons désormais une belle fenêtre sur fond bleu cyan, mais elle est vide ! Pour la remplir, on peut :

Les « conteneurs »

Nous avons désormais une belle fenêtre sur fond bleu cyan, mais elle est vide ! Pour la remplir, on peut :

- créer des conteneurs (ou components en anglais) : textes, boutons, zones de textes, cases à cocher, ...

Les « conteneurs »

Nous avons désormais une belle fenêtre sur fond bleu cyan, mais elle est vide ! Pour la remplir, on peut :

- créer des conteneurs (ou components en anglais) : textes, boutons, zones de textes, cases à cocher, ...
- ajouter ceux-ci au panneau ;

Les « conteneurs »

Nous avons désormais une belle fenêtre sur fond bleu cyan, mais elle est vide ! Pour la remplir, on peut :

- créer des conteneurs (ou components en anglais) : textes, boutons, zones de textes, cases à cocher, ...
- ajouter ceux-ci au panneau ;
- nous verrons plus tard comment les organiser.

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

- un texte à afficher :

```
JLabel monTexte = new JLabel("Coucou !");
```

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

- un texte à afficher :

```
JLabel monTexte = new JLabel("Coucou !");
```

- un bouton :

```
JButton bouton1 = new JButton("J'ai compris");
```

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

- un texte à afficher :

```
JLabel monTexte = new JLabel("Coucou !");
```

- un bouton :

```
JButton bouton1 = new JButton("J'ai compris");
```

- un champ de texte (à saisir) :

```
JTextField champTexte = new JTextField(20);
```

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

- un texte à afficher :
```
JLabel monTexte = new JLabel("Coucou !");
```
- un bouton :
```
JButton bouton1 = new JButton("J'ai compris");
```
- un champ de texte (à saisir) :
```
JTextField champTexte = new JTextField(20);
```
- une zone de texte (à saisir) :
```
JTextArea zoneTexte = new JTextArea(3,20);
```

Exemples de conteneurs

Voici une liste (non exhaustive) de conteneurs :

- un texte à afficher :
```
JLabel monTexte = new JLabel("Coucou !");
```
- un bouton :
```
JButton bouton1 = new JButton("J'ai compris");
```
- un champ de texte (à saisir) :
```
JTextField champTexte = new JTextField(20);
```
- une zone de texte (à saisir) :
```
JTextArea zoneTexte = new JTextArea(3,20);
```

Attention : penser aux « import »

Pour chacun des conteneurs précédents, penser à importer la bibliothèque correspondante (mais Eclipse vous le rappelle) avant le début de la classe :

```
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

Attention : penser aux « import »

Pour chacun des conteneurs précédents, penser à importer la bibliothèque correspondante (mais Eclipse vous le rappelle) avant le début de la classe :

```
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

ou, pour être sûr de n'en oublier aucun :

```
import.javax.swing.*;
```

Enfin, on les affiche !

Une fois ces conteneurs créés, il suffit de les ajouter à notre panneau :

```
monPanneau.add(monTexte);
monPanneau.add(bouton1);
monPanneau.add(champTexte);
monPanneau.add(zoneTexte);
```

Résultat

On obtient la jolie fenêtre ci-dessous :

Sommaire

Obtenir des informations

Dans l'exemple précédent, nous avons créé des conteneurs que l'utilisateur va (ou non) utiliser :

- boutons ;
- zones de textes ;
- cases cochées, ...

Il faut maintenant récupérer ces informations :

- quel bouton a été pressé ?
- quel texte a été saisi ?
- quelles cases sont cochées ?
- ...

Les « listeners »