

## I- La lecture à la console

Les programmes qui affichent des résultats à la console lisent généralement les données à la console.

### Exemple :

```
/*
 * Lecture.java (MODE CONSOLE)
 * Ce programme fait la lecture à la console du nom et de l'âge de l'utilisateur
 * pour ensuite afficher ces informations
 */
import java.io.*;
public class Lecture {

    public static void main (String args[]) throws IOException {

        int age;
        String nom;
        String saisie;

        BufferedReader clavier = new BufferedReader(
            new InputStreamReader(System.in));

        // lecture des données
        System.out.print("\nVotre nom : ");
        nom = clavier.readLine();

        System.out.print("\nVotre âge : ");
        saisie = clavier.readLine();
        age = Integer.parseInt(saisie);

        // affichage des résultats
        System.out.println("\nBonjour " + nom +
            ", vous êtes donc âgé(e) de " + age + " ans.");

        System.out.println("Puissiez-vous vivre encore longtemps!");
    }
}
```

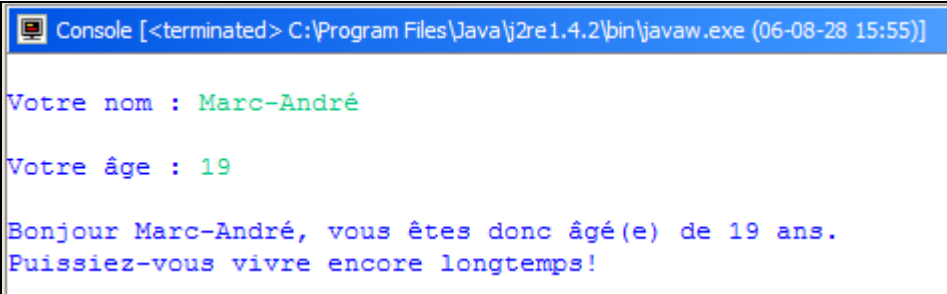
Pour importer la librairie qui contient les classes nécessaires à la lecture

Déclaration d'un tampon de lecture associé à la console

Lire la ligne de type String et la placer dans la variable nom

Extraire l'entier de la ligne lue de type String et le placer dans la variable age

### Résultats obtenus sur la console :



```
Console [<terminated> C:\Program Files\Java\j2re1.4.2\bin\javaw.exe (06-08-28 15:55)]

Votre nom : Marc-André

Votre âge : 19

Bonjour Marc-André, vous êtes donc âgé(e) de 19 ans.
Puissiez-vous vivre encore longtemps!
```

## II- Les boîtes de dialogue

**Exemple 1** (boîtes de dialogue avec le moins d'arguments possible) :

```
/*
 * Lecture1.java (version graphique de Lecture.java)
 * Ce programme fait la lecture du nom et de l'âge de l'utilisateur
 * pour ensuite afficher ces informations
 */

import javax.swing.*;

public class Lecture1 {

    public static void main (String args[]) {

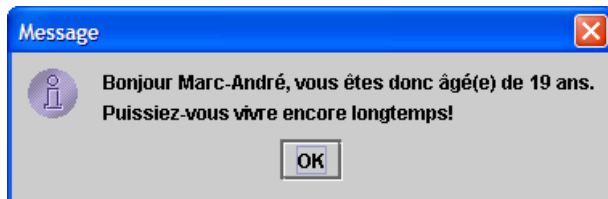
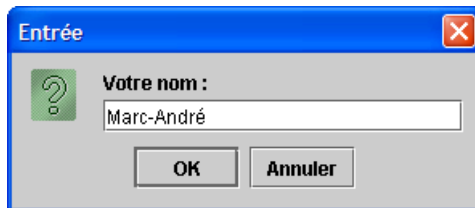
        int age;
        String nom;
        String saisie;
        String phrase;

        // lecture des données
        nom = JOptionPane.showInputDialog("Votre nom : ");

        saisie = JOptionPane.showInputDialog("Votre âge : ");
        age = Integer.parseInt(saisie);

        // affichage des résultats
        phrase = "Bonjour " + nom + ", vous êtes donc âgé(e) de "
                + age + " ans.\nPuisseiez-vous vivre encore longtemps!";
        JOptionPane.showMessageDialog(null, phrase);

        // fin de l'exécution
        System.exit(0);
    }
}
```



## Exemple 2 (boîtes de dialogue personnalisées) :

```
/*
 * Lecture2.java
 * Ce programme fait la lecture du nom et de l'âge de l'utilisateur
 * pour ensuite afficher ces informations
 */

import javax.swing.*;

public class Lecture2 {

    public static void main (String args[]) {

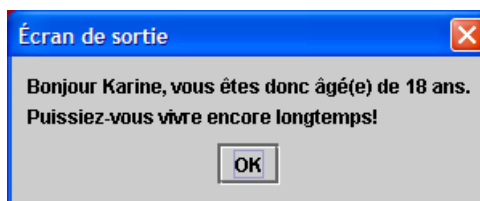
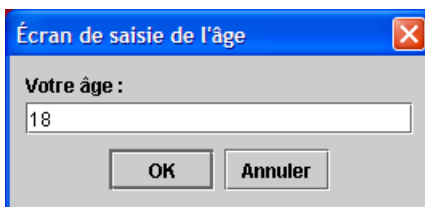
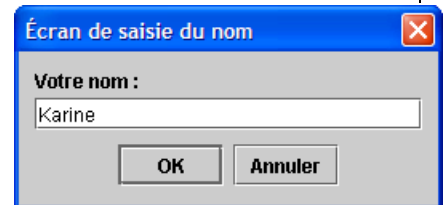
        int age;
        String nom,
            chaine,
            phrase;

        // lecture des données
        nom = JOptionPane.showInputDialog(null, "Votre nom : ",
            "Écran de saisie du nom", JOptionPane.PLAIN_MESSAGE);

        chaine = JOptionPane.showInputDialog(null, "Votre âge : ",
            "Écran de saisie de l'âge", JOptionPane.PLAIN_MESSAGE);
        age = Integer.parseInt(chaine);

        // affichage des résultats
        phrase = "Bonjour " + nom + ", vous êtes donc âgé(e) de "
            + age + " ans.\nPuisseiez-vous vivre encore longtemps!";
        JOptionPane.showMessageDialog(null, phrase, "Écran de sortie",
            JOptionPane.PLAIN_MESSAGE);

        // fin de l'exécution
        System.exit(0);
    }
}
```



### Options possibles pour le type d'icône à afficher :

<code>JOptionPane.PLAIN_MESSAGE</code>	pas d'icône du tout
<code>JOptionPane.ERROR_MESSAGE</code>	icône d'erreur
<code>JOptionPane.INFORMATION_MESSAGE</code>	icône d'information (par défaut avec <code>showMessageDialog</code> )
<code>JOptionPane.WARNING_MESSAGE</code>	icône d'avertissement
<code>JOptionPane.QUESTION_MESSAGE</code>	icône de question (par défaut avec <code>showInputDialog</code> )

### III- Les formats d'affichage

Les données numériques peuvent être affichées avec un format qui contrôle le nombre de décimales, le nombre de chiffres minimum à afficher devant le point décimal, etc.

**Exemple 1** (sans format d'affichage) :

```
/*
 * Facture.java
 * Ce programme lit des informations sur l'achat d'un article,
 * il calcule et affiche le coût total de l'achat avec les taxes
 */

import javax.swing.*;

public class Facture {

    public static void main (String args[]) {

        final double TAUX_TPS = 0.06;
        final double TAUX_TVQ = 0.075;

        int qte;
        double prixUn;
        double total;
        String saisie;

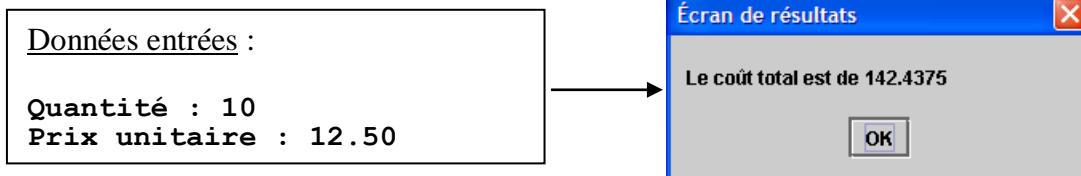
        // lecture des données
        qte = Integer.parseInt(JOptionPane.showInputDialog("Quantité :"));

        saisie = JOptionPane.showInputDialog("Prix unitaire :");
        prixUn = Double.parseDouble(saisie);

        // calcul et affichage des résultats
        total = qte * prixUn * (1 + TAUX_TPS) * (1 + TAUX_TVQ);

        JOptionPane.showMessageDialog(null, "Le coût total est de " + total,
                                     "Écran de résultats", JOptionPane.PLAIN_MESSAGE);

        // fin de l'exécution
        System.exit(0);
    }
}
```



## Exemple 2 (avec format d'affichage) :

```
/*
 * Facture1.java
 * Version avec contrôle du format d'affichage pour le résultat
 * Ce programme lit des informations sur l'achat d'un article,
 * il calcule et affiche le coût total de l'achat avec les taxes
 */

import javax.swing.*;
import java.text.*;

public class Facture1 {

    public static void main (String args[]) {

        DecimalFormat monnaie = new DecimalFormat("0.00 $");

        final double TAUX_TPS = 0.06;
        final double TAUX_TVQ = 0.075;

        int qte;
        double prixUn;
        double total;
        String saisie;

        // lecture des données
        qte = Integer.parseInt(JOptionPane.showInputDialog("Quantité :"));

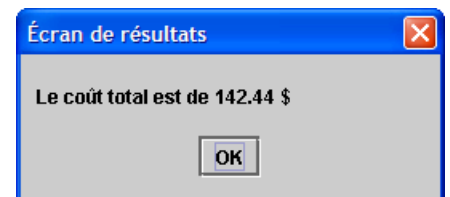
        saisie = JOptionPane.showInputDialog("Prix unitaire :");
        prixUn = Double.parseDouble(saisie);

        // calcul et affichage des résultats
        total = qte * prixUn * (1 + TAUX_TPS) * (1 + TAUX_TVQ);

        JOptionPane.showMessageDialog(null, "Le coût total est de " +
            monnaie.format(total), "Écran de résultats",
            JOptionPane.PLAIN_MESSAGE);

        // fin de l'exécution
        System.exit(0);
    }
}
```

← package nécessaire pour l'utilisation de la classe `DecimalFormat`

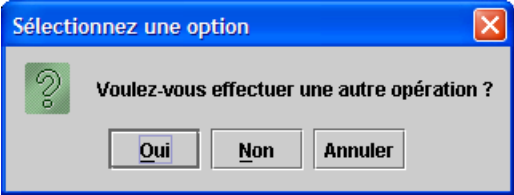
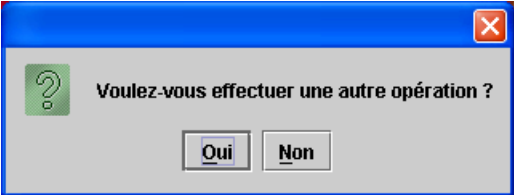
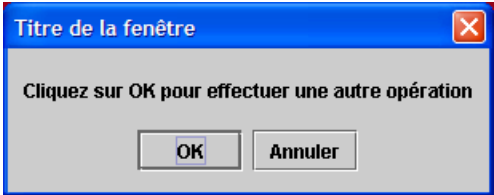


## Exemples de formats :

format	si la valeur formatée vaut			
	0	23148.567	-12.304	0.1
"0.00"	0.00	23148.57	-12.30	0.10
"0.00; (0.00) "	0.00	23148.57	(12.30)	0.10
"#, #00.0#"	00.0	23 148.57	-12.30	00.1
"#, ##0.00 \$"	0.00 \$	23 148.57 \$	-12.30 \$	0.10 \$
"0.00%"	0.00%	2314856.64%	-1230.40%	10.00%

## IV- Les boîtes de dialogue de confirmation

### Exemples :

paramètres à fournir	exemple
1. null 2. String <i>texte à afficher</i>	<pre>reponse = JOptionPane.showConfirmDialog(null,     "Voulez-vous effectuer une autre opération ?");</pre> 
1. null 2. String <i>texte à afficher</i> 3. String <i>titre de la fenêtre</i> 4. int <i>boutons désirés</i>  <u>valeurs possibles pour boutons désirés :</u> DEFAULT_OPTION YES_NO_OPTION YES_NO_CANCEL_OPTION OK_CANCEL_OPTION	<pre>reponse = JOptionPane.showConfirmDialog(null,     "Voulez-vous effectuer une autre opération ?",     "",     JOptionPane.YES_NO_OPTION);</pre> 
1. null 2. String <i>texte à afficher</i> 3. String <i>titre de la fenêtre</i> 4. int <i>boutons désirés</i> 5. int <i>type d'icône</i>  <u>valeurs possibles pour type d'icône :</u> ERROR_MESSAGE INFORMATION_MESSAGE WARNING_MESSAGE QUESTION_MESSAGE PLAIN_MESSAGE	<pre>reponse = JOptionPane.showConfirmDialog(null,     "Cliquez sur OK pour effectuer une autre opération",     "Titre de la fenêtre",     JOptionPane.OK_CANCEL_OPTION,     JOptionPane.PLAIN_MESSAGE);</pre> 

Cette méthode retourne un entier (**int**) dont la valeur dépend du bouton choisi. Les valeurs possibles dépendent des boutons affichés et sont :

```
JOptionPane.YES_OPTION
JOptionPane.NO_OPTION
JOptionPane.CANCEL_OPTION
JOptionPane.OK_OPTION
JOptionPane.CLOSED_OPTION (si on a fermé la fenêtre avec le X)
```

### Exemple 1 (avec showInputDialog) :

```
/*
 * Confirmation1.java
 * Ce programme demande à l'utilisateur s'il désire continuer
 * puis affiche le choix demandé
 */

import javax.swing.*;

public class Confirmation1 {

    public static void main (String args[]) {

        char choix;
        String message = "Vous avez choisi ";

        choix = JOptionPane.showInputDialog(
            "Voulez-vous continuer (O ou N) ?").charAt(0);

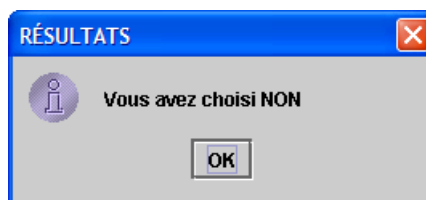
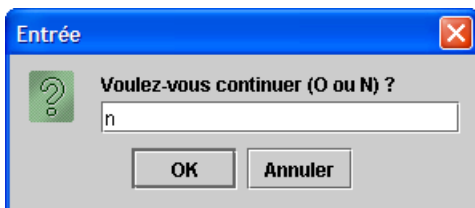
        switch (choix)
        {
            case 'O': case 'o':
                message += "OUI";
                break;

            case 'N': case 'n':
                message += "NON";
                break;

            default:
                message += "une mauvaise valeur";
        }

        JOptionPane.showMessageDialog(null, message, "RÉSULTATS",
            JOptionPane.INFORMATION_MESSAGE);

        // fin de l'exécution
        System.exit(0);
    }
}
```



## Exemple 2 (avec showConfirmDialog) :

```
/*
 * Confirmation2.java
 * Ce programme demande à l'utilisateur s'il désire continuer
 * à l'aide de showConfirmDialog
 * puis affiche le choix demandé
 */

import javax.swing.*;

public class Confirmation2 {

    public static void main (String args[]) {

        int choix;
        String message = "Vous avez choisi ";

        choix = JOptionPane.showConfirmDialog(null,
            "Voulez-vous continuer ?",
            "Fenêtre de confirmation",
            JOptionPane.YES_NO_OPTION);

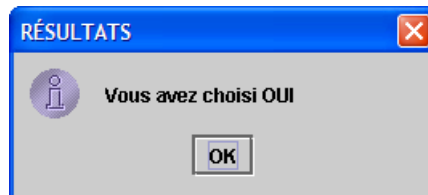
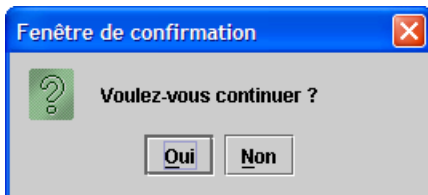
        switch (choix)
        {
            case JOptionPane.YES_OPTION:
                message += "OUI";
                break;

            case JOptionPane.NO_OPTION:
                message += "NON";
                break;

            default:
                message += "de fermer la fenêtre";
        }

        JOptionPane.showMessageDialog(null, message, "RÉSULTATS",
            JOptionPane.INFORMATION_MESSAGE);

        // fin de l'exécution
        System.exit(0);
    }
}
```





## V- Les zones de texte JTextArea

**Exemple 1** (utiliser un JTextArea dans une boîte de dialogue) :

```
/*
 * Exemple1JTextArea.java
 * Exemple d'utilisation d'un JTextArea pour afficher
 * des résultats en colonnes
 * méthode setText pour assigner un String au contenu du JTextArea
 */

import javax.swing.*;

public class Exemple1JTextArea
{
    public static void main(String[] args)
    {
        JTextArea sortie = new JTextArea(10, 30);
        String texte;

        int    valeur1;
        double valeur2;

        texte = "nombre\tdouble\tquart\n";

        for (int nombre = 1; nombre < 7; nombre++)
        {
            valeur1 = nombre * 2;
            valeur2 = nombre / 4.0;
            texte += "\n" + nombre + "\t" + valeur1 + "\t" + valeur2;
        }

        sortie.setText(texte);
        JOptionPane.showMessageDialog(null, sortie,
            "liste des nombres", JOptionPane.PLAIN_MESSAGE);

        System.exit(0);
    }
}
```

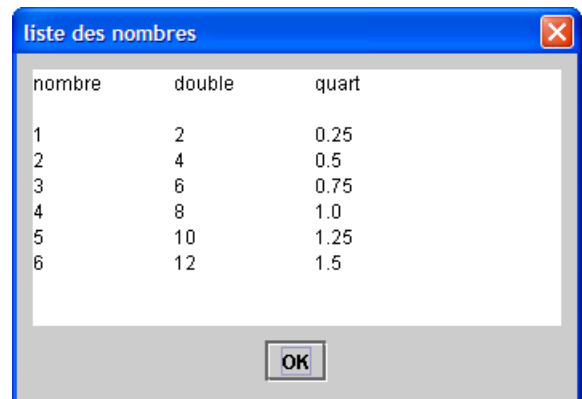
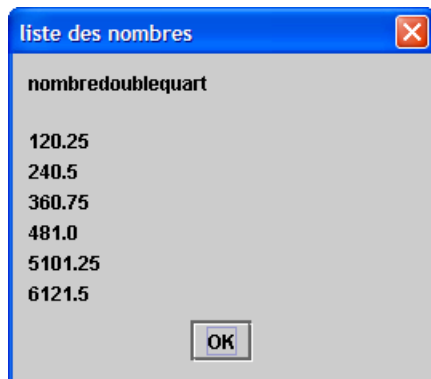
package nécessaire à toutes les classes commençant par **J** (**J** indique **swing**)

demander 10 lignes de 30 colonnes (ou plus au besoin)

**+=** pour concaténer au texte (ajouter à la fin du texte)

mettre le texte dans le JTextArea

Si on avait mis **texte** directement au lieu de **sortie** dans le **showMessageDialog** :



**Exemple 2** (éviter l'utilisation d'une variable String pour le contenu d'un JTextArea) :

```
/*
 * Exemple2JTextArea.java
 * Exemple d'utilisation d'un JTextArea pour afficher
 * des résultats en colonnes
 * méthode append pour ajouter à la fin du JTextArea
 */

import javax.swing.*;

public class Exemple2JTextArea
{
    public static void main(String[] args)
    {
        JTextArea sortie = new JTextArea(10, 30);
        int    valeur1;
        double valeur2;

        sortie.append("nombre\tdouble\tquart\n");

        for (int nombre = 1; nombre < 7; nombre++)
        {
            valeur1 = nombre * 2;
            valeur2 = nombre / 4.0;
            sortie.append("\n" + nombre + "\t" + valeur1 + "\t" + valeur2);
        }

        JOptionPane.showMessageDialog(null, sortie,
            "liste des nombres", JOptionPane.PLAIN_MESSAGE);

        System.exit(0);
    }
}
```

le contenu de départ est vide

ajouter à la fin du contenu du JTextArea

### Exemple 3 (modifier la police de caractères d'un JTextArea) :

```
/*
 * Exemple3JTextArea.java
 * Exemple d'utilisation d'un JTextArea pour afficher
 * des résultats en colonnes
 * méthode setText pour assigner un String au contenu du JTextArea
 * méthode setFont pour changer la police de caractères du JTextArea
 */

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

public class Exemple3JTextArea
{
    public static void main(String[] args)
    {
        JTextArea sortie = new JTextArea(10, 30);
        String texte;

        texte = "nombre\tdouble\tquart\n";

        for (int nombre = 1; nombre < 7; nombre++)
        {
            texte += "\n" + nombre + "\t" + (nombre * 2)
                    + "\t" + (nombre / 4.0);
        }

        sortie.setFont(new Font("Courier", Font.PLAIN, 12));
        sortie.setText(texte);
        JOptionPane.showMessageDialog(null, sortie,
            "liste des nombres en Courier 12", JOptionPane.PLAIN_MESSAGE);

        sortie.setFont(new Font("Comic Sans MS", Font.BOLD + Font.ITALIC, 14));
        JOptionPane.showMessageDialog(null, sortie,
            "liste des nombres en Comic Sans MS 14", JOptionPane.PLAIN_MESSAGE);

        System.exit(0);
    }
}
```

package nécessaire à la classe **Font**

changer la police du contenu du JTextArea pour **Courier 12pt**

#### Valeurs possibles pour le nom de la police :

toute police acceptée par le système +

**Serif**

**Monospaced**

**SansSerif**

voir page suivante pour comparer les deux écrans obtenus

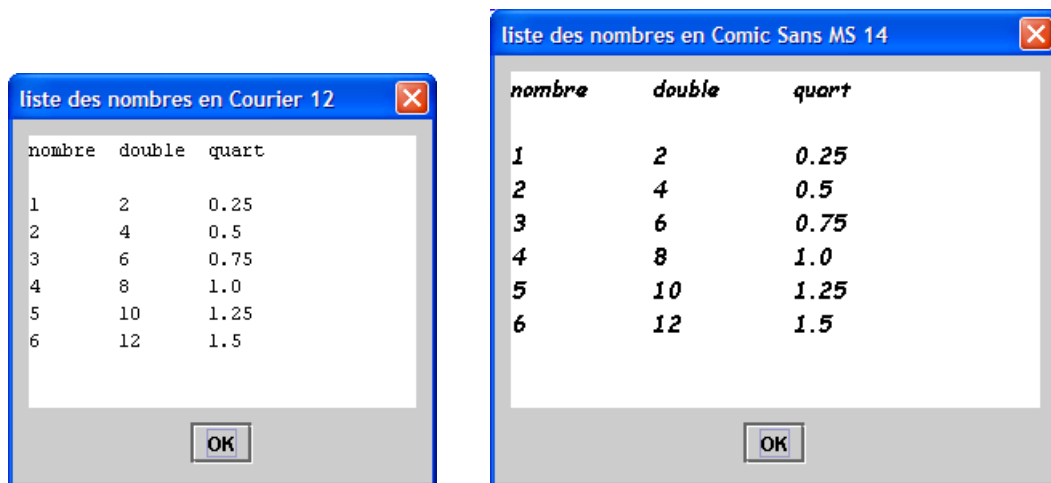
#### Valeurs possibles pour les attributs de la police :

**Font.PLAIN**

**Font.BOLD**

**Font.ITALIC**

**Font.BOLD + Font.ITALIC**



### Autres méthodes pour personnaliser un JTextArea :

exemples
// pour empêcher l'utilisateur de modifier le texte d'un JTextArea <code>zoneTexte.setEditable(false);</code>
// pour rendre un JTextArea transparent <code>zoneTexte.setOpaque(false);</code>
// pour modifier la couleur de fond d'un JTextArea <code>zoneTexte.setBackground(Color.blue);</code>
// pour modifier la couleur du texte d'un JTextArea <code>zoneTexte.setForeground(Color.white);</code>
// pour modifier la police de caractères d'un JTextArea <code>zoneTexte.setFont(new Font("Courier", Font.BOLD, 14));</code>
// pour modifier la valeur d'un \t dans un JTextArea <code>zoneTexte.setTabSize(15);</code> (la valeur par défaut est 8)