



UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
FACULTAD DE INGENIERÍA (UNIDAD MEXICALI)

Formato para prácticas de laboratorio

CARRERA	PLAN DE ESTUDIO	CLAVE DE UNIDAD DE APRENDIZAJE	NOMBRE DE LA UNIDAD DE APRENDIZAJE
Ingeniero en Computación	2009-1	12099	Programación Orientada a Objetos

PRÁCTICA No.	LABORATORIO DE	Programación Orientada a Objetos	DURACIÓN (HORAS)
8	NOMBRE DE LA PRÁCTICA	Programas interactivos	2

1. INTRODUCCIÓN

La mayoría de los programas más interesantes son aquellos que interactúan con el usuario; que le hacen una pregunta y toman alguna acción con base en su respuesta. En esta practica conoceremos el funcionamiento de las clase `JOptionPane` y la utilizaremos para crear programas interactivos. Esto significa que los programas solicitarán al usuario que proporcione algunos datos y tome algunas decisiones que nuestros programas utilizarán para manipular objetos. De esta forma, las acciones que se tomen en nuestros programas dependerán de lo que quiera hacer el usuario.

2. OBJETIVO (COMPETENCIA)

Desarrollar programas que interactúen con el usuario utilizando las opciones de entrada de datos de la clase `JOptionPane` con actitud creativa y participativa.

Formuló Cecilia Curlango Rosas Maria Luisa Gonzalez Ramirez	Revisó Gloria E. Chavez Valenzuela	Aprobó	Autorizó M.C. Maximiliano de las Fuentes Lara
Nombre y Firma del Maestro	Nombre y Firma del Responsable de Programa Educativo	Nombre y Firma del Responsable de Gestión de Calidad	Nombre y Firma del Director de la Facultad

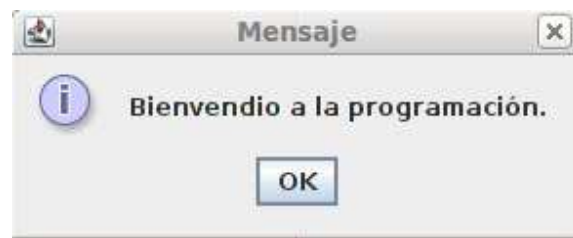
3. FUNDAMENTO

La clase `JOptionPane` facilita crear cuadros de dialogo que piden algún valor al usuario o le informan de algo.





Para mostrar una mensaje al usuario, quizás para informarle que ocurrió algo en el programa, utilizamos el método `showMessageDialog()` como se muestra en la siguiente línea de código.

```
JOptionPane.showMessageDialog(null, "Bienvendio a la programación. ", "Mensaje",  
JOptionPane.INFORMATION_MESSAGE);
```

Al ejecutar la línea anterior, aparece la siguiente ventana.



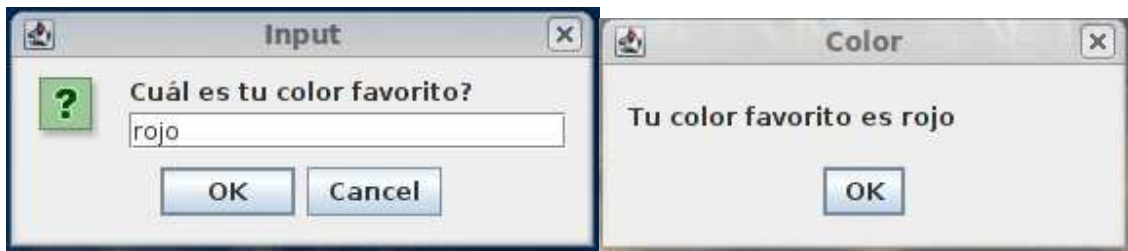
Observe que los `String` que utilizamos como parámetros, corresponden con los mensaje que aparecen en la ventana. El icono que aparece en la parte izquierda de la ventana se generó por que utilizamos la constante `JOptionPane.INFORMATION_MESSAGE`. En la siguiente tabla vemos otros iconos que podemos mostrar.

ICONO JAVA	ICONO WINDOWS	CONSTANTE
		<code>ERROR_MESSAGE</code>
		<code>INFORMATION_MESSAGE</code>
		<code>WARNING_MESSAGE</code>
		<code>QUESTION_MESSAGE</code>
Ningún icono	Ningún icono	<code>PLAIN_MESSAGE</code>

Cuando queremos que el usuario escriba algún dato, entonces utilizamos el método `showInputDialog()` el cual nos regresa un `String` con el texto que escribió el usuario. Las siguientes líneas de código se pueden utilizar para interactuar con el usuario.

```
String tuColor = JOptionPane.showInputDialog("Cuál es tu color favorito?");  
JOptionPane.showMessageDialog(null, "Tu color favorito es " + tuColor,  
"Color", JOptionPane.PLAIN_MESSAGE);
```

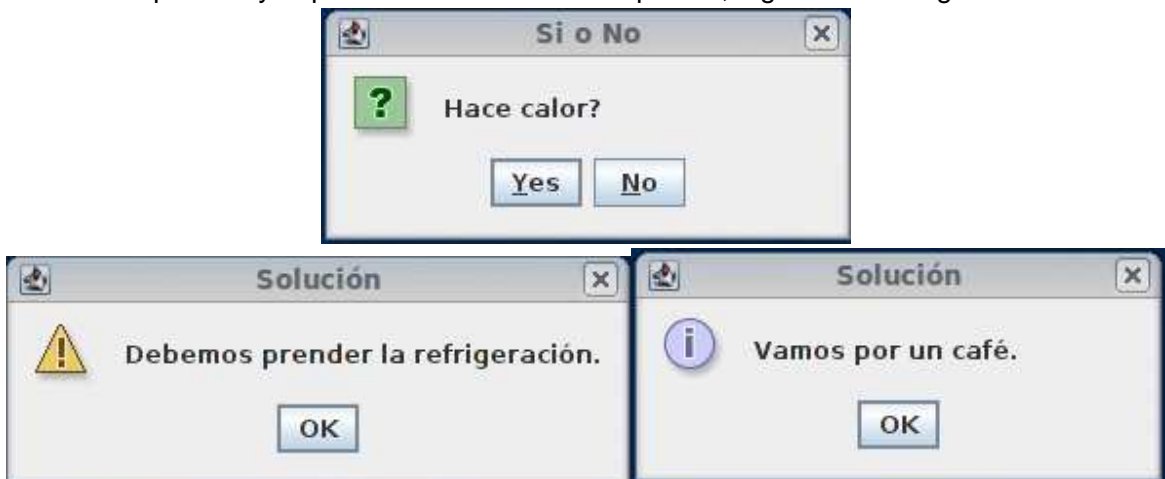
Las ventanas que se muestran son:



Cuando solo queremos que el usuario confirme alguna operación, es decir que nos diga si quiere hacer algo o no, entonces podemos utilizar la ventana que se muestra con el método `showConfirmDialog()`. Este método nos regresa un valor `int` que nos indica que respuesta nos dió el usuario, es decir nos indica que botón presionó. El siguiente código nos demuestra como podemos pedir al usuario una respuesta de si o no.

```
int yesNoResponse = JOptionPane.showConfirmDialog(null, "Hace calor?", "Si o No",
        JOptionPane.YES_NO_OPTION);
if (yesNoResponse == JOptionPane.YES_OPTION) {
    JOptionPane.showMessageDialog(null, "Debemos prender la refrigeración.",
        "Solución", JOptionPane.WARNING_MESSAGE);
} else {
    JOptionPane.showMessageDialog(null, "Vamos por un café.", "Solución",
        JOptionPane.INFORMATION_MESSAGE);
}
```

Observe que comparamos la respuesta del usuario con la constante `JOptionPane.YES_OPTION`. En el API de `JOptionPane` se muestran otras constantes adicionales. El código anterior nos mostrará la siguiente ventana primero y dependiendo de nuestra respuesta, alguna de las siguientes.

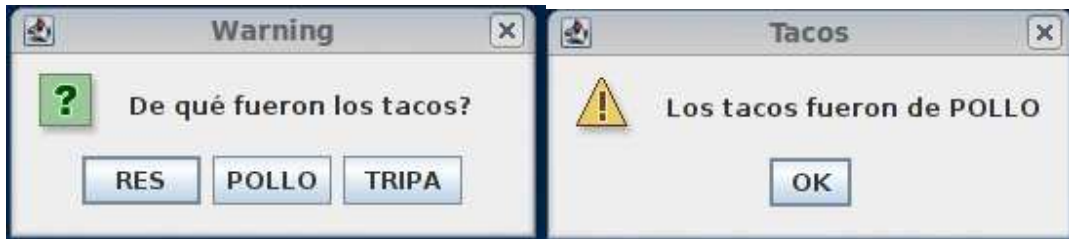


El método `showOptionDialog()` nos permite mostrar ventanas en las que nosotros podemos controlar los textos de los botones y otros aspectos de su presentación lo cual nos permite crear aplicaciones con mayor interactividad.

```
Object[] botones = {"RES", "POLLO", "TRIPA"};
```

```
int tacos = JOptionPane.showOptionDialog(null, "De qué fueron los tacos?",
"Warning",
JOptionPane.DEFAULT_OPTION,
JOptionPane.QUESTION_MESSAGE, null,
botones, botones[0]);
JOptionPane.showMessageDialog(null, "Los tacos fueron de " + botones[tacos],
"Tacos",
JOptionPane.WARNING_MESSAGE);
```

En este caso, el código muestra la siguiente ventana que tiene los tres botones con los mensajes que especificamos. El método nos regresa un número `int` que corresponde al botón que presionó el usuario. Este valor lo utilizamos en este caso como índice para el arreglo de botones para poder así mostrar un mensaje al usuario indicándole la opción que seleccionó.



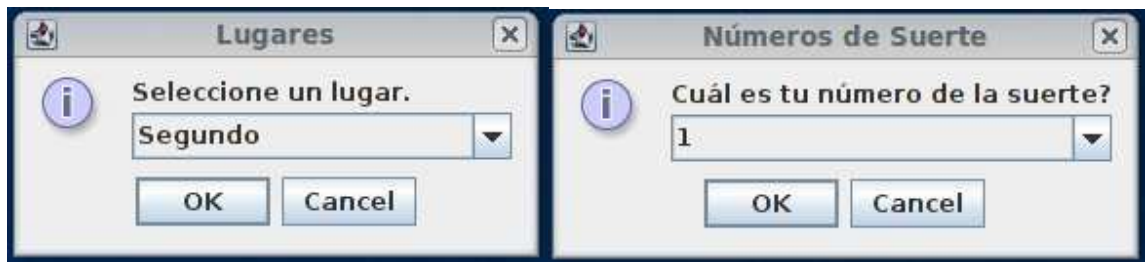
Finalmente, en el siguiente código vemos que más podemos hacer con otra versión del método `showInputDialog()`. Esta versión nos permite enviarle un arreglo de `Object` que inicializamos con tres mensajes. Estos objetos los utiliza `showInputDialog()` para crear un *dropdown list*, una lista desde donde escoge el usuario uno de los valores. También se puede definir cual será el valor que aparecerá seleccionado inicialmente, esto se hace con el último parámetro del método. Una vez que el usuario hace su selección, `showInputDialog()` nos regresa el objeto seleccionado a partir del cual podemos tomar alguna acción dependiendo de su valor.

La segunda parte del listado nos muestra que también podemos utilizar objetos de otros tipos, por ejemplo `Integer`, para que se muestren en el *dropdown list*.

```
Object[] posiblesValores = {"Primero", "Segundo", "Tercero"};
// Indicamos que el valor por default es "Segundo"
Object selectedValue = JOptionPane.showInputDialog(null, "Seleccione un lugar.",
"Lugares", JOptionPane.INFORMATION_MESSAGE, null, posiblesValores,
posiblesValores[1]);
String opcion = selectedValue.toString();
JOptionPane.showMessageDialog(null, "Escogiste: " + opcion, "Resultado",
JOptionPane.INFORMATION_MESSAGE);

// También podemos trabajar con Integer
Integer[] posiblesNumeros = {new Integer(1), new Integer(2), new Integer(3)};
Integer numeroSeleccionado = (Integer) JOptionPane.showInputDialog(null, "Cuál es tu
número de la suerte?", "Números de Suerte",
JOptionPane.INFORMATION_MESSAGE, null, posiblesNumeros,
posiblesNumeros[0]);
JOptionPane.showMessageDialog(null, "Tu número de la suerte es: " +
numeroSeleccionado, "Felicidades!", JOptionPane.INFORMATION_MESSAGE);
```

Las siguientes imágenes muestran como se ven las ventanas que nos genera el código anterior.



Para conocer cuales métodos tiene JOptionPane y su funcionamiento se debe consultar el API de la clase en: <http://download.oracle.com/javase/6/docs/api/javax/swing/JOptionPane.html>

4. PROCEDIMIENTO (DESCRIPCIÓN)

- 1.- Compruebe el funcionamiento de los códigos anteriores.
- 2.- Retome los métodos que se agregaron a la clase Picture para realizar lo siguiente.
 - Utilice JOptionPane para solicitar el color que quiere cambiar de una imagen y a que color lo desea cambiar.
- 3.- Retome los métodos que se agregaron a la clase Turtle para realizar lo siguiente.
 - Utilice JOptionPane y *dropdown list* para darle la opción al usuario de dibujar un:
 - Cuadrado
 - Circulo
 - Triangulo
 - Rectangulo
4. Realice un programa que lea una cadena con JOptionPane y que escriba en un objeto Picture la cadena leída.
- 5.- Realice el proceso del punto 2 pero deberá seguir pidiendo un nuevo color y actualizando la imagen hasta que el usuario seleccione el botón de CANCEL en el objeto JOptionPane.

A) EQUIPO NECESARIO

MATERIAL

Computadoras con capacidad para ejecutar el entorno de desarrollo Netbeans. Paquete misClases. Al menos una imagen para manipularla con los métodos presentados y desarrollados en la practica.

7. REFERENCIAS

Netbeans

<http://netbeans.org/downloads/>

Java 6

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>