**ENTRADAS Y SALIDAS A LA CONSOLA DEL SISTEMA**

**1.- ESCRITURA DE DATOS EN LA CONSOLA DEL SISTEMA**

El objeto System.out de la clase PrintStream representa la consola de salida del sistema,

y cuenta, entre otros, con los métodos **print** y **println** que permiten enviar cadenas a la consola del sistema. El método **printf** permite como su homónimo en C dar formato a los datos de salida.

Ejemplo:

**public class Escritura //Sin lectura**

**{**

**public static void main(String[] args)**

**{**

**int res;**

**int num1=20;**

**int num2=40;**

**res=num1+num2;**

**System.out.print("Vamos a calcular la suma de los numeros: " +num1+" y ");**

**//System.out.print(num1);**

**System.out.println(num2);**

**System.out.print ("El resultado de la suma es: ");**

**System.out.println( res);**

**}**

**}**

**Salida:**

**Vamos a calcular la suma de los numeros: 20 y 40**

**El resultado de la suma es: 60**

**2.- LECTURA DE DATOS EN LA CONSOLA DEL SISTEMA**

**2.1.- LECTURA DE DATOS A TRAVÉS DE LA CLASE Scanner**

La clase Scanner está disponible a partir de Java 5 y facilita la lectura de datos en los programas Java.

Para utilizar Scanner en el programa tendremos que hacer lo siguiente:

**1.** **Escribir el import**

La clase Scanner se encuentra en el paquete java.util por lo tanto se debe incluir al inicio del programa la instrucción:

**import java.util.Scanner;**

**2.** **Crear un objeto Scanner**

Tenemos que crear un objeto de la clase Scanner asociado al dispositivo de entrada.

Si el dispositivo de entrada es el teclado escribiremos:

**Scanner teclado = new Scanner(System.in);**

Se ha creado el objeto **teclado** asociado al teclado representado por *System.in*

Una vez hecho esto podemos leer datos por teclado.

Para leer podemos usar el método **nextXxx()** donde Xxx indica un tipo, por ejemplo nextInt() para leer un entero, nextDouble() para leer un double, etc.

**Ejemplo** de lectura por teclado de un **número entero**:

int n;

System.out.print("Introduzca un número entero: ");

n = teclado.nextInt();

**Ejemplo** de lectura de un número de tipo **double**:

double x;

System.out.print("Introduzca número de tipo double: ");

n = teclado.nextDouble();

**Ejemplo** de lectura de una **cadena de caracteres**:

String s;

System.out.print("Introduzca texto: ");

s = teclado.nextLine();

**Ejemplo** de lectura de un **caracter**:

char caracter;

String s;

System.out.print("Introduzca un caracter: ");

s = teclado.next();

caracter=s.charAt(0);

**Ejemplo de programa Java con lectura de datos con Scanner**:

El programa pide que se introduzca el nombre de la persona y lo muestra por pantalla. A continuación lee por teclado el radio de una circunferencia de tipo double y muestra su longitud. Además lee un entero y muestra su cuadrado.

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); //crear un objeto Scanner

String nombre;

double radio;

int n;

System.out.print("Introduzca su nombre: ");

nombre = sc.nextLine(); //leer un String

System.out.println("Hola " + nombre + "!!!");

System.out.print("Introduzca el radio de la circunferencia: ");

radio = sc.nextDouble(); //leer un double

System.out.println("Longitud de la circunferencia: " + 2\*Math.PI\*radio);

System.out.print("Introduzca un número entero: ");

n = sc.nextInt(); //leer un entero

System.out.println("El cuadrado es: " + (n\*n));

}

}

**Cómo limpiar el buffer de entrada en Java**

Cuando en un programa se leen por teclado datos numéricos y datos de tipo carácter o String debemos tener en cuenta que al introducir los datos y pulsar intro estamos también introduciendo en el buffer de entrada el intro.

Es decir, cuando en un programa introducimos un datos y pulsamos el intro como final de entrada, el carácter intro también pasa al buffer de entrada.

Buffer de entrada si se introduce un 5: 5\n

En esta situación, la instrucción:

n = sc.nextInt();

Asigna a n el valor 5 pero el intro permanece en el buffer

Buffer de entrada después de leer el entero: \n

Si ahora se pide que se introduzca por teclado una cadena de caracteres:

System.out.print("Introduzca su nombre: ");

nombre = sc.nextLine(); //leer un String

El método nextLine() extrae del buffer de entrada todos los caracteres hasta llegar a un intro y elimina el intro del buffer.

En este caso asigna una cadena vacía a la variable nombre y limpia el intro. Esto provoca que el programa no funcione correctamente, ya que no se detiene para que se introduzca el nombre.

Solución:

Se debe **limpiar el buffer** de entrada si se van a leer datos de tipo carácter a continuación de la lectura de datos numéricos.

La forma más sencilla de limpiar el buffer de entrada en Java es ejecutar la instrucción:

sc.nextLine();

Lo podemos comprobar si cambiamos el orden de lectura del ejemplo y leemos el nombre al final:

import java.util.Scanner;

public class JavaApplication335 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String nombre;

double radio;

int n;

System.out.print("Introduzca el radio de la circunferencia: ");

radio = sc.nextDouble();

System.out.println("Longitud de la circunferencia: " + 2\*Math.PI\*radio);

System.out.print("Introduzca un número entero: ");

n = sc.nextInt();

System.out.println("El cuadrado es: " + Math.pow(n,2));

System.out.print("Introduzca su nombre: ");

nombre = sc.nextLine(); //leemos el String después del double

System.out.println("Hola " + nombre + "!!!");

}

}

Si lo ejecutamos obtendremos:

Introduzca el radio de la circunferencia: 34

Longitud de la circunferencia: 213.62830044410595

Introduzca un número entero: 3

El cuadrado es: 9.0

Introduzca su nombre: Hola !!!

Comprobamos que no se detiene para pedir el nombre.

La solución es escribir la instrucción

**sc.nextLine();**

después de la lectura del int y antes de leer el String:

import java.util.Scanner;

public class JavaApplication335 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String nombre;

double radio;

int n;

System.out.print("Introduzca el radio de la circunferencia: ");

radio = sc.nextDouble();

System.out.println("Longitud de la circunferencia: " + 2\*Math.PI\*radio);

System.out.print("Introduzca un número entero: ");

n = sc.nextInt();

System.out.println("El cuadrado es: " + Math.pow(n,2));

sc.nextLine();

System.out.print("Introduzca su nombre: ");

nombre = sc.nextLine();

System.out.println("Hola " + nombre + "!!!");

}

}

Ahora la ejecución es correcta:

Introduzca el radio de la circunferencia: 23

Longitud de la circunferencia: 144.51326206513048

Introduzca un número entero: 5

El cuadrado es: 25.0

Introduzca su nombre: Lucas

Hola Lucas!!!

**2.2.- LECTURA DE DATOS A TRAVÉS DE LA CLASE JOptionPane**

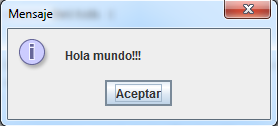
Los cuadros de dialogo (**JOptionPane**) son “herramientas” muy útiles al momento de ingresar datos y mostrar información; digo util ya que con estas no necesitamos crear objetos de tipo **BufferedReader** y “tirar” mas código para poder usar correctamente los datos que se ingresan como lo expliqué brevemente en el [anterior post](http://jagonzalez.org/clase-leer/). Entre los JOptionPane entontraremos: **showConfirmDialog,  showInputDialog,showMessageDialog y showOptionDialog**. Los más usados son los tres primeros.

**showMessageDialog**

Este cuadro de dialogo es ideal para mostrar información de cualquier tipo, este reemplaza el System.out.print(); La sintaxis es:

|  |  |
| --- | --- |
|  | JOptionPane.showMessageDialog(**null**, "Hola mundo!!!"); |

Como observamos en la anterior linea, lo que va dentro de las “…” es el mensaje que deseemos mostrar, para estos mensajes también se puede usar la concatenación para mostrar mensajes y variables a la misma vez.  
- Lo que se ve al ejecutar la aplicacion es lo siguiente:

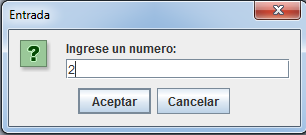


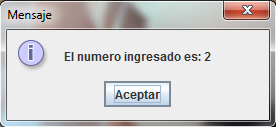
**showInputDialog**

Este cuadro de dialogo es ideal para ingresar datos, hay que tener en cuenta que todo lo que se reciba se tomara como un **String**, pero esto no es problema para trabajar con otros tipos de datos, en tal caso es necesario convertir la cadena que se recibe en el tipo de dato que necesitemos (int, Float, double, etc). La sintaxis es las siguiente:

|  |  |
| --- | --- |
| 1  2 | String ax = JOptionPane.showInputDialog("Ingrese un numero: ");  JOptionPane.showMessageDialog(**null**, "El numero ingresado es: "+ax); |

- En la linea 1 cree una variable de tipo **String** la cual llame “ax”, en esta variable almaceno el dato que se ingresa con el **showInputDialog**, al igual que el **showMessageDialog** lo que va dentro del “…” es el mensaje que mostramos al usuario.  
- En la linea 2 muestro el dato que recibí atravez de la variable ax.  
- Al ejecutarse la aplicación veremos lo siguiente:





**showConfirmDialog**

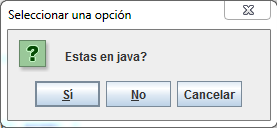
Este cuadro de dialogo es de tipo entero, por lo tanto hay que crear una variable de tipo **int** para manejar las opciones que este cuadro de dialogo ofrece, las anteriores son si, no, y cancelar; la sintaxis es la siguiente:

|  |  |
| --- | --- |
| 1  2  3  4  5 | **int** ax = JOptionPane.showConfirmDialog(**null**, "Estas en java?");  **if**(ax == JOptionPane.YES\_OPTION)  JOptionPane.showMessageDialog(**null**, "Has seleccionado SI.");  **else** **if**(ax == JOptionPane.NO\_OPTION)  JOptionPane.showMessageDialog(**null**, "Has seleccionado NO."); |

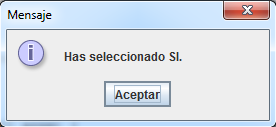
En la linea 1 se puede observar que cree una variable entera, el **null** que se antepone al mensaje es necesario, ya que este se toma cuando el usuario pulsa la tecla **Esc** o presiona **Cancelar**, el mensaje a mostrar se usa como en los anteriores cuadros de dialogo.

En las lineas 2 y 4 se usan condicionales para tomar las acciones respectivas de acuerdo a la opción que haya dado el usuario.

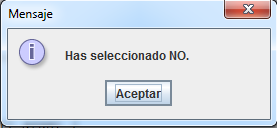
En las lineas 3 y 5 se muestran mensaje correspondientes a la respectiva opción que haya dado el usuario.



Si seleccionamos **SÍ** veremos el siguiente cuadro de dialogo:



Si por el contrario seleccionamos **NO** veremos el siguiente cuadro de dialogo:



Para finalizar mostrare el codigo necesario para trabajar con números **int**, **Float**, **Double**, **Long**:

|  |  |
| --- | --- |
| 1  2  3  4  5 | String ax = JOptionPane.showInputDialog(**null**,"");  **int** a = Integer.parseInt(ax);*//se convierte el dato a entero*  Float b = Float.parseFloat(ax);*//se convierte el dato a flotante*  Double c = Double.parseDouble(ax);*// se convierte el dato a double*  Long d = Long.parseLong(ax);*//se convierte el dato a long* |

- Hay que tener en cuenta que lo anterior se hace con **JOptionPane.showInputDialog** y que en las **“…”** va el mensaje que deseamos mostrar al usuario para guiarlo en la operación.

Para poder usar estos cuadros de dialogo hay que importar la **librería** necesaria para que esta se ejecute, la cual es:

**import javax.swing.JOptionPane;**

**import javax.swing.JOptionPane;**

**public class LeerDatosOptionPane**

**{**

**public static void main(String args[])**

**{**

**JOptionPane.showMessageDialog(null, "Hola mundo!!!"); //pruebo**

**String texto=JOptionPane.showInputDialog("Escriba un nombre");**

**String texto1=JOptionPane.showInputDialog("Escriba su edad");**

**int edad=Integer.parseInt(texto1);**

**String texto2=JOptionPane.showInputDialog("Escriba su estatura");**

**Double estatura=Double.parseDouble(texto2);**

**System.out.println("Nombre "+texto+"\nEdad:"+edad+"\nAltura: "+estatura);**

**}**

**}**

|  |  |
| --- | --- |
|  |  |
|  |  |