GENERALITAT VALENCIANA
CONSELLERIA D'EDUCACIÓ, INVESTIGACIÓ, CULTURA I ESPORT
Fondo Social Europeo

Programación

UD 2: Entrada y salida de información

Ciclo Formativo de Grado Superior
Desarrollo de Aplicaciones Web

Francisco Pérez y Ferran Vilanova

Entrada y salida de información

1.– Programación de la consola: entrada y salida de información

2.– Concepto de flujo

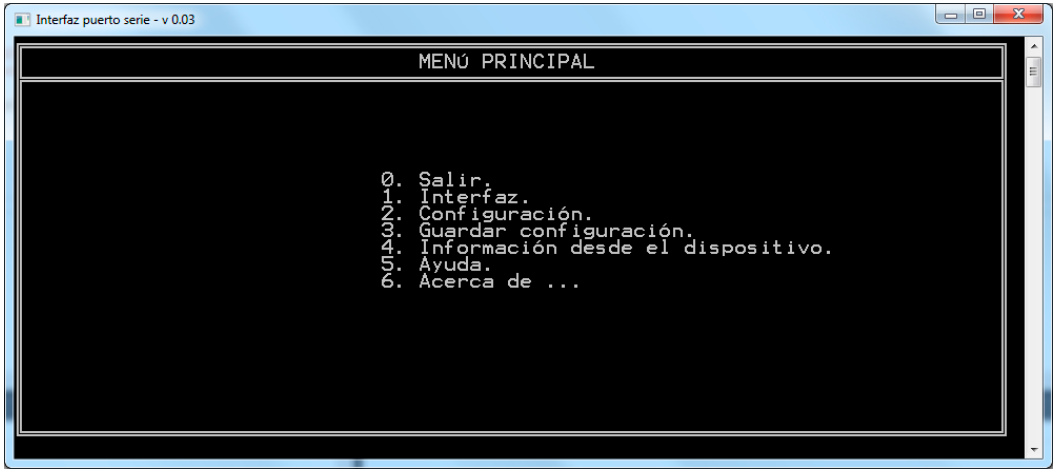
3.– Entrada desde teclado

4.– Salida a pantalla

1.– Programación de la consola

1.–Programación de la consola

Una aplicación de consola es un programa informático diseñado para ser utilizado a través de una interfaz de solo texto, como un terminal de texto, la interfaz de línea de comando de algunos sistemas operativos (Unix, DOS, etc.), o la consola Win32 en Microsoft Windows y la terminal en MacOS.



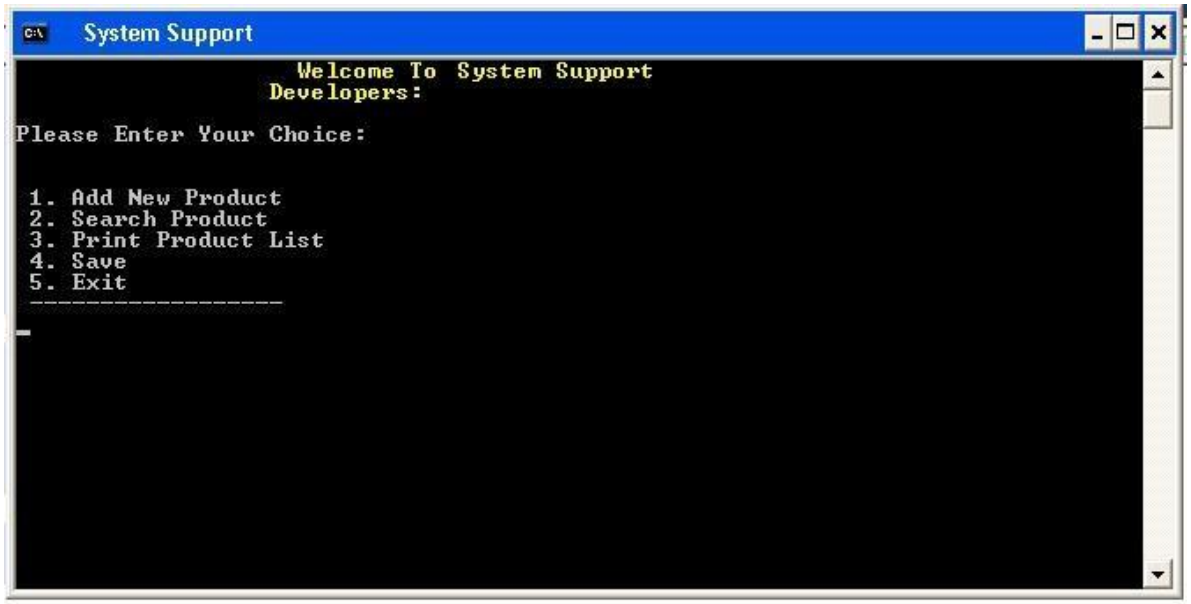
Un usuario generalmente interactúa con una aplicación de consola usando solo un teclado y una pantalla, a diferencia de las aplicaciones de IGU, que normalmente requieren el uso de un mouse u otro dispositivo señalador.

Muchas aplicaciones de consola, como los intérpretes de línea de comandos, son herramientas de línea de comandos, pero también existen numerosos programas de interfaz de usuario basada en texto.

1.–Programación de la consola

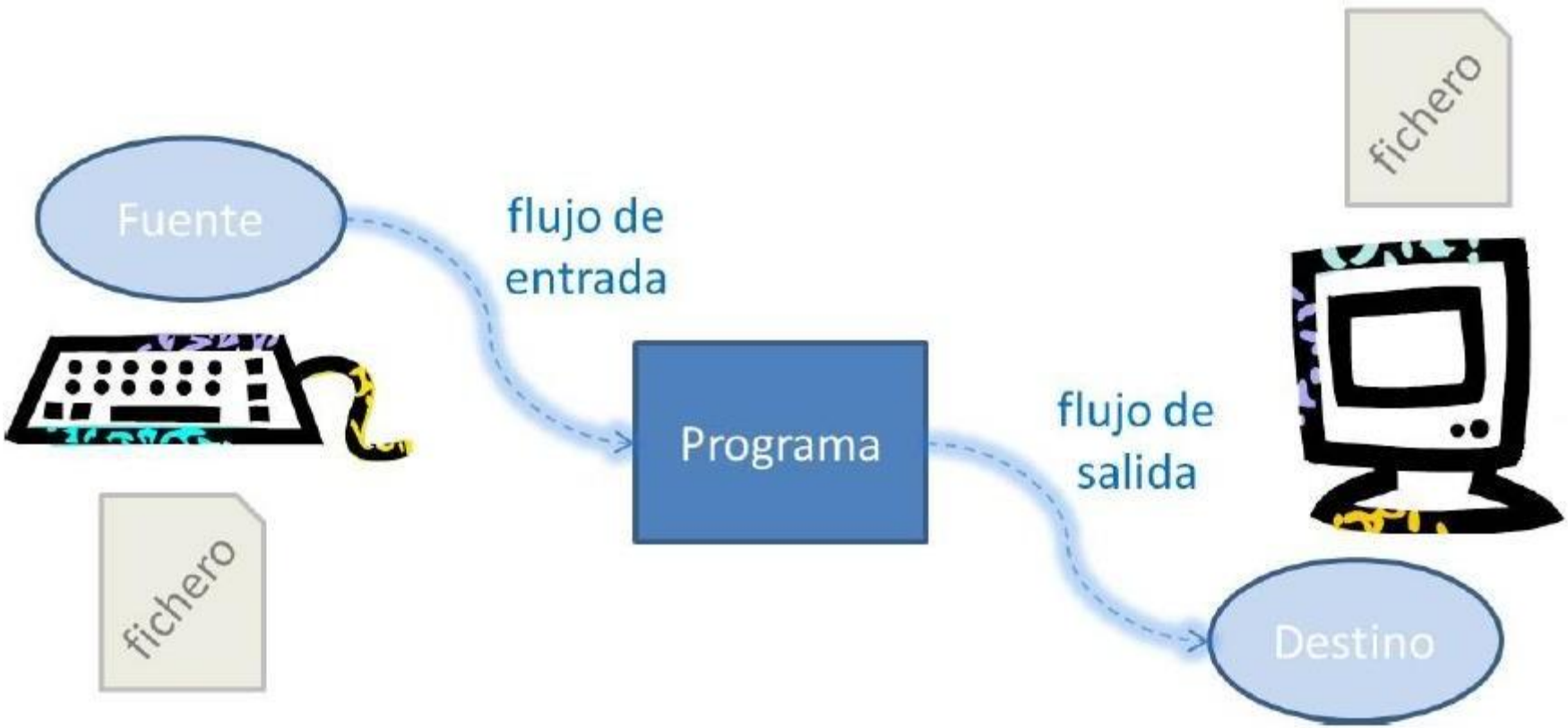
A medida que la velocidad y la facilidad de uso de las aplicaciones de IGU han mejorado con el tiempo, el uso de las aplicaciones de consola ha disminuido en gran medida, pero no ha desaparecido.

Algunos usuarios simplemente prefieren las aplicaciones basadas en consola, mientras que algunas organizaciones aún dependen de las aplicaciones de consola existentes para manejar las tareas clave de procesamiento de datos.



2.– Concepto de flujo

2.–Concepto de flujo



3.– Entrada desde teclado

3. Entrada desde teclado

La clase Scanner de Java provee métodos para leer valores de entrada de varios tipos y está localizada en el paquete java.util.

Los valores de entrada pueden venir de varias fuentes, incluyendo valores que se entren por el teclado o datos almacenados en un archivo.

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.

3. Entrada desde teclado

Principales constructores y métodos de la clase Scanner	
public **Scanner** (InputStream source)	Crea un nuevo Scanner a partir de un flujo de entrada de datos como es el caso de System.in (para poder leer desde teclado).
public String **next** () public String **next** (String pattern)	Obtiene el siguiente elemento leído del teclado como un String (si coincide con el patrón especificado). Lanza NoSuchElementException si no quedan más elementos por leer.
public String **nextLine** ()	Se lee el resto de línea completa, descartando el salto de línea. Devuelve el resultado como un String. Lanza NoSuchElementException si no quedan más elementos por leer.
public int **nextInt** () public long **nextLong** () public short **nextShort** () public byte **nextByte** () public float **nextFloat** () public double **nextDouble** () public boolean **nextBoolean** ()	Devuelve el siguiente elemento como un int siempre que se trate de un int. Ídem para long, short, byte, float, double y boolean. Lanza InputMismatchException en caso de no poder obtener un valor del tipo apropiado. Lanza NoSuchElementException si no quedan más elementos por leer.
public boolean **hasNext** ()	Devuelve true si queda algún elemento por leer.
public boolean **hasNextLine** ()	Devuelve true si queda alguna línea por leer.
public boolean **hasNextInt** () public boolean **hasNextLong** () public boolean **hasNextShort** () public boolean **hasNextByte** () public boolean **hasNextFloat** () public boolean **hasNextDouble** () public boolean **hasNextBoolean** ()	Devuelve true si el siguiente elemento a obtener se puede interpretar como un int. Ídem para long, short, byte, float, double y boolean.
public Scanner **useLocale** (Locale l)	Establece la configuración local del Scanner a la configuración especificada por el Locale l.

3. Entrada desde teclado

Ejemplo:

```
Scanner sc = new Scanner(System.in);

int numClase;
String nombre;
double nota;

System.out.println("Introduce el número de clase:");
numClase = sc.nextInt();

//NOTA: esta línea es para capturar el retorno de carro
sc.nextLine();

System.out.println("Introduce el nombre del alumno:");
nombre = sc.nextLine();

System.out.println("Introduce la nota del exámen:");
nota = sc.nextDouble();
```

4.– Salida a pantalla

4.–Salida a pantalla

Salida por pantalla

*System.**out**.*println

*System.**out**.*print

Salida por pantalla formateada

*System.**out**.*printf

4.–Salida a pantalla

Imprimir números enteros con *System.**out**.*printf

```
…

//Declaración de variables
int a = 8;
int b = 3;
int resultado = 0;

//%d se sustituye por la variable entera, resultado
//%n indica un salto de línea
resultado = (a + b);
System.out.printf(“La suma es: %d %n”, resultado);


resultado = (a - b);
System.out.printf("La resta es: %d %n", resultado);

…
```

4.–Salida a pantalla

Imprimir texto con *System.**out**.*printf

```
…

//%s se sustituye por la variable de texto, imprime en minúsculas
//%S se sustituye por la variable de texto, imprime en mayúsculas
//El salto de línea se puede indicar con \n o %n

String texto = "Mayor";

//Imprime: El resultado es Mayor
System.out.printf("El resultado es: %s \n", texto);

//Imprime: El resultado es MAYOR
System.out.printf("El resultado es: %S %n", texto);

…
```