UF3.2
Interacción con el usuario en Java

Centro Profesional
Universidad Europea Madrid
LAUREATE INTERNATIONAL UNIVERSITIES

CONTENIDOS

1. Paquetes (API)
2. Envoltorios
3. Entrada / Salida

PAQUETES

Definición

Un paquete es el equivalente a una libreria de C. En lugar de utilizar la orden de C:

```
# include
```

en Java utilizaremos la orden:

```
import paquete;
```

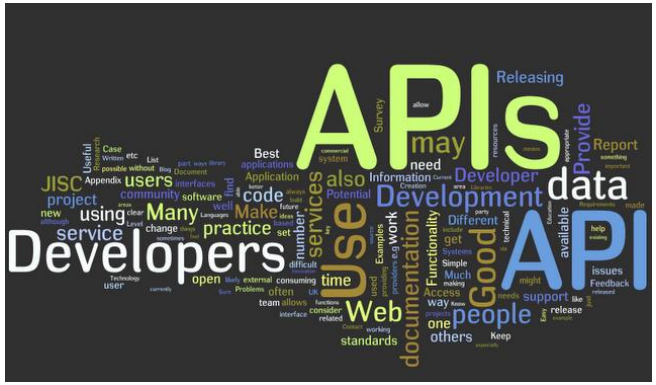
PAQUETES

API

Los paquetes facilitan extraordinariamente la labor del programador, ya que existen miles de líneas de código que otros programadores ya han hecho por nosotros.

El conjunto de todos los paquetes oficiales en Java es lo que se denomina la Interfaz de Programación de Aplicaciones (API, del inglés Application Programming Interface) de Java.

En cada versión de la API se incluye un mayor número de paquetes y también existe documentación exhaustiva para que cualquier programador pueda consultar cómo hacer uso de ella.



ENVOLTORIOS

Definición

Además de los tipos de datos básicos que ya conocemos, Java sigue con su vocación de utilizar objetos y nos ofrece un envoltorio de los tipos de datos básicos para convertirlos en objetos.

Existen nueve envoltorios: Integer, Long, Float, Double, Short, Byte, Character, Boolean y Void.

Fijaos que todos ellos tienen nombres prácticamente idénticos a los tipos básicos, pero con la diferencia de que la primera letra es mayúscula (recordad que Java es sensible a mayúsculas).

Envoltorio	Tipo
Integer	int
Long	long
Float	float
Double	double
Short	short
Byte	byte
Character	char
Boolean	boolean
Void	void

Utilización

Los envoltorios son muy útiles para realizar conversiones entre tipos de manera sencilla, ya que la API nos ofrece métodos para hacer casi todo lo que deseemos.
Por ejemplo, la conversión de un número entero introducido por teclado por un usuario, que por tanto es una cadena de caracteres, es una tarea muy sencilla en Java:

```
int numeroConvertido = Integer.parseInt(numeroString);
```

Fijaos que hemos utilizado el método parseInt() de un objeto Integer. Este método recibe como parámetro un String y devuelve el número entero correspondiente de hacer la conversión de ese String.

Nota: En este ejemplo podéis apreciar que, como no teníamos ningún objeto Integer, hemos podido utilizar el mismo nombre de la clase para hacer uso de su método. Son facilidades que nos da Java.

¿Complicado?

Ya hemos visto anteriormente que la salida en Java es muy fácil de utilizar.

En cambio, no ocurre lo mismo con la entrada, donde Java muestra una de sus incomodidades más palpable. Veamos cómo se trabaja.

- Salida estándar. Se utiliza System.out que se encuentra en el paquete java.lang.
- Entrada estándar. Se utiliza System.in que se encuentra en el paquete java.lang.
- Salida de errores. Se utiliza System.err que se encuentra en el paquete java.lang.

Ejemplo

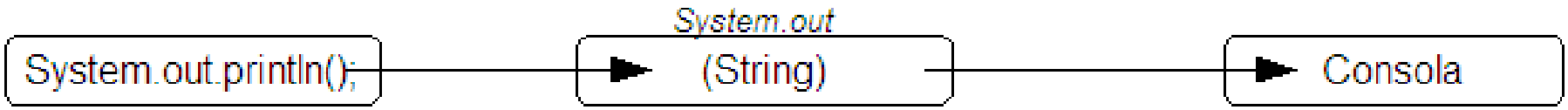
Imprimir por pantalla

Utilizaremos los métodos print() o println() (este último incluye el retorno de carro al final de la salida).
Uso:

System.out.print("Se imprime este mensaje sin el retorno de carro");

System.out.println("Se imprime este mensaje con un retorno de carro");

Gráficamente:



Teclado

Leer de teclado

A bajo nivel se utiliza el método read(), que lee un único carácter.

Uso:

```
char c = (char) System.in.read();
```

Esta manera de leer del teclado es muy poco práctica y sería tedioso programar la lectura de una cadena de caracteres. En su lugar, es habitual hacer uso del paquete **java.util** que nos ofrece ayuda para acceder de manera más cómoda a la lectura del teclado.

Veámoslo

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;
```

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

Si el dispositivo de entrada es el teclado escribiremos:

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

String s;

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

s = sc.nextLine();
```