Flujos de Entrada y Salida Introducción

Programación

Octubre 2015

Objetivo

Comprender los flujos de entrada y salida en lenguajes de POO

Introducción

- Dispositivos de Almacenamientos
 - Son componentes para la lectura y escritura de datos (temporal o permanentemente)
 - Las computadoras poseen almacenamiento principal (RAM, ROM) y secundario (DD)
- Archivos
 - Son colecciones de datos almacenados permanentemente en dispositivos de almacenamiento
 - Son formados por un conjunto de registros
 - Tienen fin (eof)
 - Tradicionalmente son organizados en registros, los registros en campos, los campos en bytes y los bytes en bits
 - Existen archivos de texto y archivos binarios

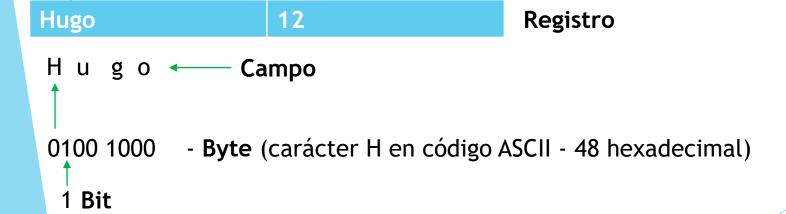
Archivo

Jerarquía de Datos

		_
Nombre	Edad	
Ana	10	
Hugo	12	Aı
Verónica	14	
María	9	
Luis	10	
Juan	13	eo

rchivo

of



Flujos

- Para la manipulación de un archivo almacenado permanentemente es necesario la creación de un flujo
- Los **flujos** son conductos a través del cual se transportan datos hacia o desde un dispositivo
- Cuando los datos van desde la memoria de un programa hacia el dispositivo de almacenamiento es un flujo de salida (output)
- Cuando los datos van desde el dispositivo de almacenamiento hacia la memoria de un programa es un flujo de entrada (input)

Flujos de Datos para Entrada y Salida Estándar

- Son flujos que actúan como canales de comunicación permitiendo la interacción entre un programa y su entorno en el sistema
- El acceso a estos flujos de entrada es desde la clase java.lang.System
- Salida Estándar
 - Está relacionada directamente con la terminal del sistema, de modo que los resultados enviados a este flujo son mostrados en la pantalla (aunque el destino puede ser modificado)
 - Existen dos tipos de salida: i) La regular y ii) la destinada para errores ocurridos en un programa
 - En Java, la salida estándar (StdOut) es representada por un objeto PrintStream llamado out en la clase System
 - La clase *PrintStream* es una implementación de *FilterOutputStream* y por ende también de la clase base *OutputStream*.
- Ejemplo de Salida Estándar:
 - System.out.println("Hola Mundo!"); //La clase System tiene una propiedad estática llamada out del tipo // PrintStream y que define el método println

Salida de Errores (System.err)

- Es otra salida estándar pero con el fin de ser utilizada para errores (StdErr)
- ▶ Al igual que StdOut es representada por un objeto **PrintStream** llamado err
- Los método que pueden ser invocados son los mismo que out
- Si el destino es el mismo para out y err no tendría mucha utilidad si en ambas salidas el color en la consola es el mismo
- System.out.println("Hola!");
- System.err.println("Hola!");

```
C:\Programacion\práctica 5 - Flujos>javac Main.java
C:\Programacion\práctica 5 - Flujos>java Main
Hola!
Hola!
```

Se puede aprovechar de mejor forma si se re-define la salida estándar para out con el archivo "salida.txt" y de err con el archivo "errores.txt"

Redefinición de Salida Estándar

- En Java, para modificar el destino de err y out en dos archivos diferentes, se utilizan los métodos estáticos void setOut(PrintStream out) y void setErr(PrintStream err) de la clase System
- Para ello, es necesario abrir flujos de datos hacia los nuevos destinos de cada salida, para cada uno se crea un objeto FileOutputStream(File file) con un objeto PrintStream(OutputStream out)
 - System.setOut(new PrintStream(new FileOutputStream("salida_normal.txt")));
 - System.setErr(new PrintStream(new FileOutputStream("salida_error.txt")));