I/O: il package java.io

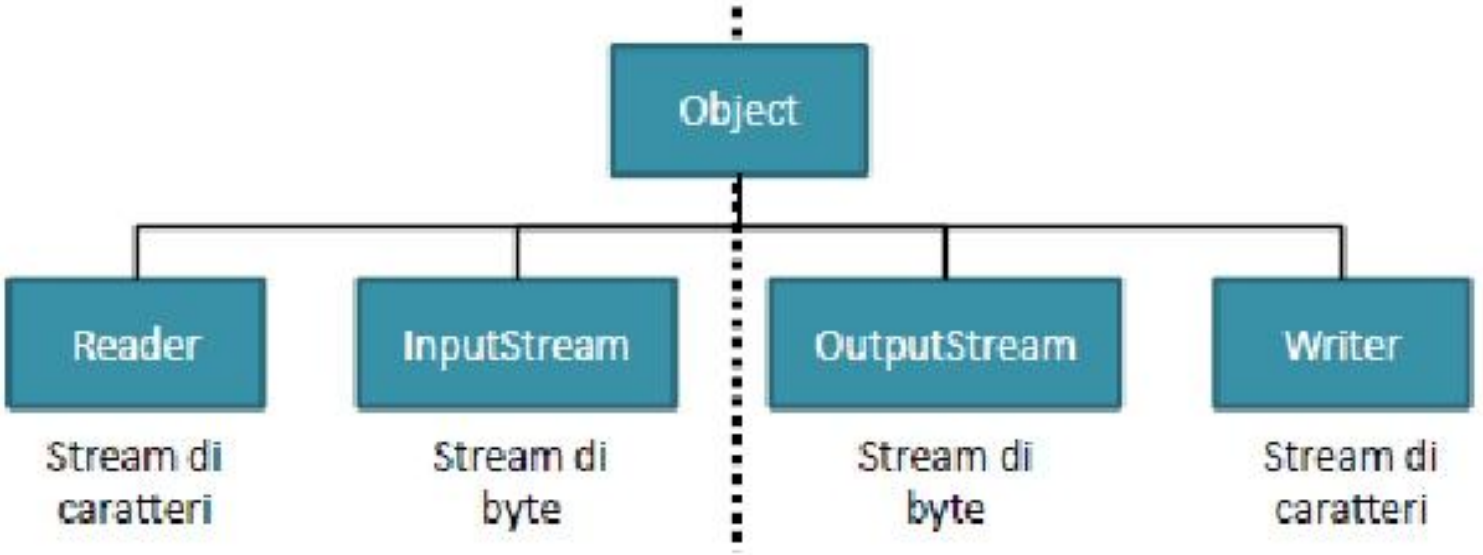
In Java si usano gli stream per poter leggere/scrivere dai/sui file

Uno **stream** è un canale di comunicazione :

1. monodirezionale (o è input o è output)
2. ad uso generale
3. che trasferisce byte (o anche caratteri)

Il **package java.io** distingue tra stream di byte (file binari) e stream di caratteri (file testuali)

Questi stream si traducono poi in una serie di classi di base; queste sono alla base della tassonomia delle classi di stream realizzate da Java



Input e Output

Java offre **numerosi classi per la gestione dell'input e dell'output**

- La tastiera è normalmente collegata al campo statico in presente nella classe di sistema System
- System offre tre campi statici che, se non opportunamente ridefiniti dal programmatore, rappresentano:
  1. **in**: stream di input (Coincide con la tastiera)
  2. **out**: stream di output (Coincide con la console in cui eseguite il programma)
  3. **err**: stream d'errore (Coincide con la console in cui eseguite il programma)

System.in è un oggetto di tipo InputStream (stream orientato ai byte): non è facilmente utilizzabile, perciò si usano altre classi di supporto per facilitare la vita al programmatore …

Input da tastiera

Possiamo utilizzare la **classe Scanner**

- È una classe offerta dalla libreria standard Java ed è presente nel **package java.util**
- Mette a disposizione un insieme di operazioni che consentono la lettura dell'input fornito dall'utente

Ad ***esempio***, per leggere una stringa immessa dall'utente:

```
import java.util.*;
public class TestLettura {
   public static void main(String args[]){
        Scanner input = new Scanner(System.in);
        String s = input.nextLine();
        System.out.println("Ho letto: " + s);
   }
}
```

Input da tastiera: metodi della classe Scanner

- **oggetto.nextLine()** restituisce sotto forma di oggetto di tipo String la successiva riga d'ingresso fornita in ingresso fino alla pressione del tasto Enter
- **oggetto.next()** restituisce sotto forma di oggetto di tipo String la successiva parola fornita in ingresso (per parola s'intende una sequenza di caratteri che finisce con una spaziatura)

- **oggetto.nextDouble()** restituisce sotto forma di dato double il numero reale fornito in ingresso

- **oggetto.nextInt()** restituisce sotto forma di dato int il numero intero fornito in ingresso

Esempio

```
import java.util.*;
public class TestLettura {
    public static void main(String args[]){
        System.out.println(“Immetti un intero “);
        Scanner input = new Scanner(System.in);
        int x = input.nextInt();
        System.out.println("Ho letto: " + x);
    }
}
```

Input da file

• Per leggere dati da un file presente sul disco, la classe Scanner si affida ad un'altra classe, **File** , che descrive file e cartelle presenti in un file system.

• Per prima cosa si costruisce un oggetto di tipo File, fornendo il nome del file da leggere:

File inFile = new File("input.txt");

• Successivamente si utilizza tale oggetto per costruire un oggetto di tipo Scanner:

Scanner in = new Scanner(inFile);

Input da file

• Quest’ oggetto Scanner legge il testo contenuto nel file input.text .

•Per leggere dati potete usare i soliti metodi della classe (next, nextLine, nextInt, nextDouble)

•Ci sono anche altri metodi tra cui ad es. il metodo oggetto.hasNextLine() restituisce false se non c’è nessun’altra riga nell’input dello scanner altrimenti true.

Output da file

Per scrivere dati in un file si costruisce un **oggetto di tipo PrintWriter** , fornendo il nome del file:

PrintWriter out = new PrintWriter(“output.txt”)

•Se il file in cui si scrive esiste già, viene svuotato prima di scrivervi nuovi dati. Se il file non esiste viene creato un file nuovo

•Con un oggetto di tipo PrintWriter potete usare gli usuali metodi print, println

•Quando avete terminato di scrivere in un file accertatevi di chiudere l’oggetto PrintWriter:

out.close()

Esempio

```
import java.util.*;
import java.io.*;

public class LetturaFile {
public static void main(String[] args)
{
try{File inputF= new File("input1.txt");
Scanner in= new Scanner(inputF);
PrintWriter out= new PrintWriter("output1.txt");
String riga= in.nextLine();
 out.println(riga);
 out.close();}
catch(FileNotFoundException exc){
System.out.println("Il file di input non esiste");}
}
}
```

Esercizio 9

Scrivere un programma che legge tutte le righe presenti in un file e le scrive in un altro file inserendo per ciascuna riga il corrispondente numero di riga.

Esercizio 10

Scrivere un programma che utilizzando la classe Impiegato crei un array di elementi di tale classe, e le memorizzi in un file, ed infine si rilegga il file e lo si stampi a video.

```
public class Impiegato2 {
private String nome;
private String cognome;
private double salario;

public Impiegato2(String n, String c, double s) {
nome = n;
cognome = c;
salario = s;}
public void incrementasalario(int percentuale) {
salario = salario + ((salario * percentuale) / 100);}
public String dettagli() {
return ("Nome " + nome + " Cognome " + cognome + " Salario " +salario);
}} }
```