Package di I/O

- I/O in Java e’
  - basato su stream
  - sincrono
- contiene le classi che implementano i tipi di stream comunemente usati
- Classi base astratte
  - InputStream, OutputStream, Reader, Writer
  - defniscono i metodi base di accesso agli stream
  - le classi derivate devono defnirne l’implementazione e quindi la semantica

OutputStream

- write(int n)
- write(byte[] b)
- write byte[] b, int offset, int length)
- flush()
- close()

La classe File

- definisce un oggetto file in modo indipendente dalla piattaforma
- definisce azioni quali
  - test di esistenza
  - test di abilitazione alla scrittura
  - directory parent
  - pathname assoluto
  - pathname della directory parent
  - generazione dell'URL corrispondente al file
  - etc.

Classi Derivate di OutputStream

- FileOutputStream
  - gestione di uno stream su file
- ByteArrayOutputStream
  - gestione di uno stream su un buffer (byte[])
- PipedOutputStream
  - in coppia con PipedInputStream per implementare il concettoUnix di pipe
- FilterOutputStream
  - classe base per filtri
- Sono a loro volta base per ulteriori estensioni

Classe DataOutputStream

- realizza la codifica XDR per i tipi primitivi
- permette la scrittura di file platform independent
- permette la scrittura di file language independent
  - attenzione ai caratteri: UNICODE != ASCII

Output bufferizzato

- per efficienza e’ possibile effettuare output bufferizzato
- classe BufferedOutputStream
- si costruisce su un parametro di tipo OutputStream
- estende FilterOutputStream
  - puo’ essere usata in ogni posto ove puo’ essere usato un OutputStream

Esempio di uso di filtri

- Classe ZipOutputStream
  - estende DeflaterOutputStream
    - implementazione di compressore di dati indipendente dalla compressione
  - DeflaterOutputStream estende a sua volta FilterOutputStream
- permette di generare files nel formato zip
- si scrivono dati in uno stream di output
- i dati si possono dividere in “entries”

Stream di caratteri

- Converte automaticamente i caratteri nel formato nativo della macchina locale
- Classi derivate dalla classe astratta Writer
- tutti gli stream visti precedentemente trattavano i caratteri come dati binari e mantenevano la codifica UNICODE
- stessi tipi di stream visti precedentemente

La classe PrintWriter

- si puo’ costruire passando un parametro di tipo OutputStream

  PrintWriter pw =

  new PrintWriter(OutputStream o);

- permette di fare output formattato di stringhe (e ogni altro tipo Java)

  pw.print() e pw.println()

Stream di Input

- sono la simmetrica controparte di quelli di output
  - scrivi usando un XOutputStream
  - leggi usando un XInputStream
  - se a caratteri scrivi con un writer e leggi con un reader

```
File f = new File(“pippo”);
PrintWriter pw = new PrintWriter(new FileOutputStream(f));
int i = 0;
for(i=0;i<10;i++)
        pw.println(i);
pw.flush();
pw.close();
BufferedReader br = new BufferedReader(new FileInputStream(f));
for(i=0;i<10;i++)
        System.out.println(br.readLine());
```