Ogni linguaggio ha un proprio stile, che è definito in modo da essere il più comprensibile possibile da qualsiasi persona che prende mano a un qualche pezzo di codice. In Java, alcuni di questi esempi sono:

- Usare SEMPRE le graffe, in qualsiasi blocco (evitare quei blocchi con istruzione singola senza graffe)
- Usare lo stile 1TBS per le graffe (la graffa di apertura si mette sulla stessa linea del blocco che lo apre, mentre quella di chiusura va nella stessa colonna dello statement)
- I nomi di package con lettere minuscole, senza underscore _
- I nomi di campi, metodi e variabili in camelCase: quindiCosì
- I nomi di classi vanno in PascalCase: QuindiCosì
- I campi static final usano SNAKE_CASE, QUINDI_COSI

Opzioni di compilazione avanzate

E' possibile specificare la destinazione dei file Java che vengono compilati con `javac` tramite il flag `-d`.
Divideremo i file in due cartelle:

- `src` conterrà i file sorgente (`.java`)
- `bin` conterrà i file compilato (`.class`)

Il `-cp`, utilizzato come flag di `java` o `javac` permette di specificare la classpath di una specifica classe. Se per esempio utilizzo una classe dichiarata in un altro file `.class`, essa va specificata tramite la classpath (ossia la directory in cui si trova il package di cui abbiamo bisogno) prima di eseguire il file che la utilizza.

Esempio: supponiamo che in una file `Yo.java` ci serva la classe `Ciao` che si trova nel file `vulpi/progetto/bin/Ciao.class` (supponiamo di trovarci in `vulpi/altropercorso`). In compilazione utilizzeremo il seguente comando:

```
javac Yo.java -d bin -cp ../progetto/bin
```

Non specifichiamo quindi il file .class di cui necessitiamo, ma la directory nel quale è contenuto.

Compilare tramite `-d` manterrà la struttura dei package: se compilo un file in `/ciao/dir2/Ciao.java` nella directory `dest`, esso verrà compilato in `/dest/ciao/dir2/Ciao.class`.

```
javac C1.java -d dest -cp b/lib/foo/bar
```