

IL CONCETTO DI PACKAGE

- Una applicazione è spesso composta di *molte classi* (eventualmente correlate)
- Un **package** è un **gruppo di classi** che costituiscono una **unità concettuale**.
 - un package può comprendere parecchie classi
 - anche definite in *file separati*
- Una **dichiarazione di package** ha la forma:
package <nomepackage> ;
Se presente, deve essere all'inizio di un file.

ESEMPIO

```
package pippo;
```

File Counter.java

```
public class Counter {  
    ...  
}
```

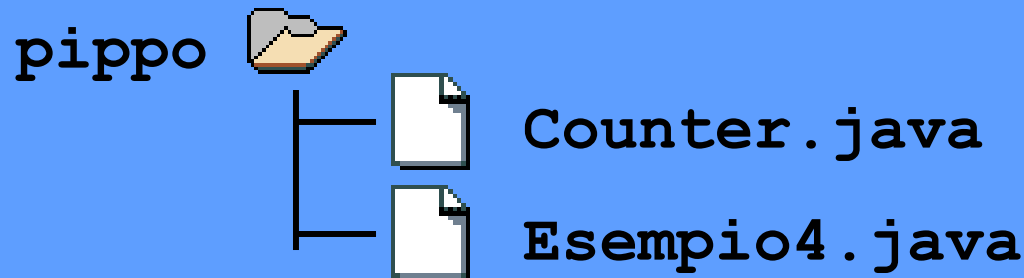
```
package pippo;
```

File Esempio4.java

```
public class Esempio4 {  
    public static void main(String args[]) {  
        ... }  
}
```

PACKAGE E FILE SYSTEM

- Esiste una *corrispondenza biunivoca* fra
 - *nome del package*
 - *posizione nel file system* delle classi del package
- Un package di nome **pippo** richiede che tutte le sue classi si trovino in una **cartella (directory)** di nome **pippo**

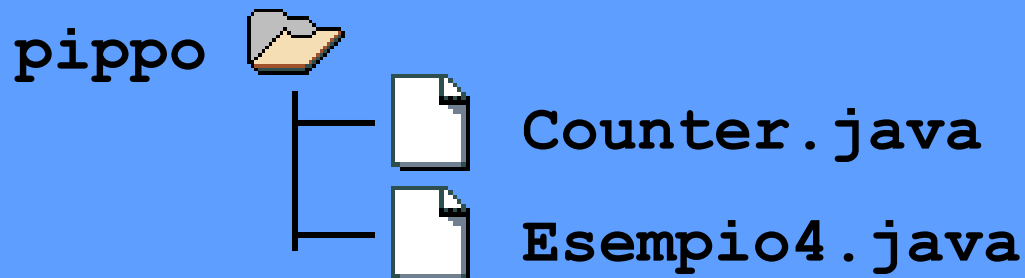


COMPILAZIONE ...

Per compilare una classe **Counter** che fa parte di un package **pippo** occorre:

- *porsi nella cartella superiore a pippo*
- e lì invocare il compilatore con il *percorso completo* della classe:

```
javac pippo/Counter.java
```



COMPILAZIONE ...

Per compilare una classe **Counter** che fa parte di un package **pippo** occorre:

- *porsi*

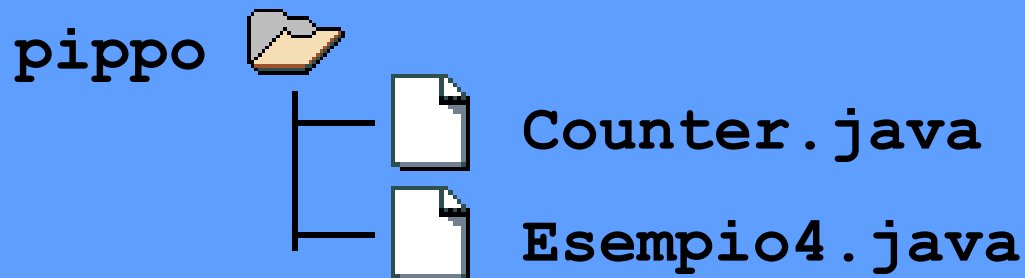
- *e lì i*

compilatore

corso

Non c'è scelta: ogni altro modo di invocare il compilatore è **errato!**

`javac pippo/Counter.java`



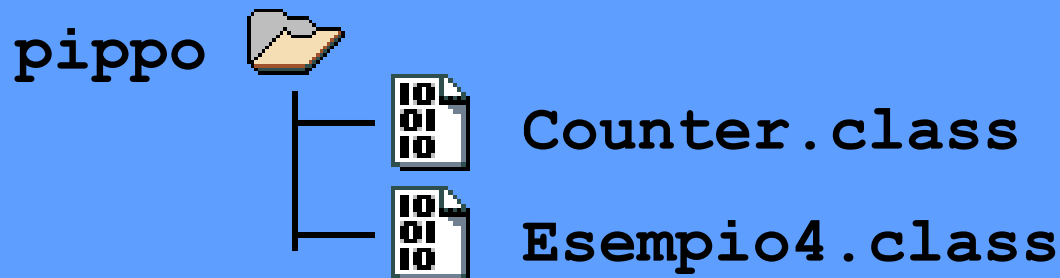
... ED ESECUZIONE

Per eseguire una classe **Esempio4** che fa parte di un package **pippo** occorre:

- *porsi nella cartella superiore a pippo*
- e lì invocare l'interprete con il **nome assoluto** della classe:

```
java pippo.Esempio4
```

Anche qui, nessuna alternativa



PACKAGE DI DEFAULT

- Se una classe non dichiara di appartenere ad alcun package, è automaticamente assegnata al *package di default*
- Per convenzione, questo package fa riferimento alla *cartella (directory) corrente*
 - è l'approccio usato in tutti i precedenti esempi
 - si possono compilare ed eseguire i file *nella cartella in cui si trovano*, senza premettere percorsi o nomi assoluti

SISTEMA DEI NOMI DEI PACKAGE

- Sono possibili *nomi di package strutturati*, come:
`java.awt.print`
`pippo.pluto.papero`
- **ATTENZIONE:** *questo non significa che un package ne contenga un altro!*
 - `pippo.pluto.papero` è un package *indipendente* da un eventuale package `pippo.pluto`
- Le classi di tali package hanno perciò un *nome assoluto strutturato*:
`java.awt.print.Book`
`pippo.pluto.papero.Counter`

SISTEMA DEI NOMI: DIFETTO

- Ogni volta che si usa una classe, Java richiede che venga denotata con il suo *nome assoluto*:

```
java.awt.print.Book b;  
b = new java.awt.print.Book();
```

- Questo è chiaramente scomodo se il nome è lungo e la classe è usata frequentemente.
- Per tale motivo si introduce il concetto di *importazione di nome*.

IMPORTAZIONE DI NOMI

- Per evitare di dover riscrivere più volte il nome assoluto di una classe, si può *importarlo*:

```
import java.awt.print.Book;
```

- Da questo momento, è possibile scrivere semplicemente **Book** invece del nome completo `java.awt.print.Book`

- Per importare in un colpo solo *tutti i nomi pubblici* di un package, si scrive

```
import java.awt.print.*;
```

IMPORTAZIONE DI NOMI

Attenzione:

*l' **import** è ben diversa dalla **#include** del C*

- *in C, il pre-processor gestisce la **#include** copiando il contenuto del file specificato nella posizione della **#include** stessa*
- *in Java non esiste alcun pre-processor, e non si copia assolutamente nulla*
- *si stabilisce solo una “scorciatoia” per scrivere un nome corto al posto di uno lungo.*

PACKAGE E VISIBILITÀ

- Oltre a pubblico / privato, in Java esiste un terzo tipo di visibilità: **la visibilità package**
- È l'impostazione predefinita per classi e metodi
non esiste quindi una parola chiave per indicarla
- Significa che dati e metodi sono **accessibili solo per le altre classi dello stesso package**
in qualunque file siano definite
- Altre classi, definite *in altri package*, non possono accedere a dati e metodi di questo package qualificati a “visibilità package”, esattamente come se fossero privati.

PACKAGE E VISIBILITÀ

- Oltre a *public* e *private* esiste un terzo tipo di visibilità: *default*.
 - A differenza del C, il file rimane solo un contenitore fisico, non definisce più uno scope di visibilità.
- È importante notare che *default* non esiste quindi una parola chiave per indicarla.
- Significa che dati e metodi sono accessibili solo per le altre classi dello stesso package in qualunque file siano definite.
- Anche se *default* non è una parola chiave, non è quindi possibile, *né sensato*, pensare di definire una classe *visibile in un solo file*: la visibilità si esprime solo con riferimento ai package.

IL PACKAGE `java.lang`

- Il nucleo centrale del linguaggio Java è definito nel package `java.lang`
- È sempre importato automaticamente:
la frase `import java.lang.*` è sottintesa
- `java.lang` definisce i tipi primitivi e buona parte delle classi di sistema
- Molte altre classi standard sono però definite altrove: *ci sono più di 50 package*
 - `java.awt`, `java.util`, `java.io`, `java.text`,...
 - `javax.swing`, ...