

Il linguaggio Java

I packages

Concetti base



- Un package è una collezione di classi ed interfacce *correlate* che fornisce uno **spazio dei nomi** ed un **controllo sugli accessi**
- Un package
 - **facilita** il reperimento e l'utilizzo delle classi/interfacce
 - **evita** conflitti sui nomi
 - **controlla** l'accesso alle classi/interfacce

Il package



Un package è una collezione di classi *correlate*

- Il package **java.lang** contiene le classi fondamentali del linguaggio
- Il package **java.util** contiene classi di utilità
- Il package **java.io** contiene le classi relative all'I/O
- Il package **java.net** contiene le classi relative alla rete
- Il package **mio.tesi** contiene le classi relative alla mia tesi

Nome di un package



- Il nome di un package deve essere *univoco*
 - **it.unipi.iet.dini**
- I package permettono di definire nomi unici per le classi
 - **it.unipi.iet.dini.didattica.MiaClasse**
 - **it.unipi.iet.dini.ricerca.MiaClasse**

Utilizzo delle classi nei package (1)



- Una classe può utilizzare tutte le classi del suo package e tutte le classi **public** di un altro package
- La classe di un package può essere specificata attraverso il suo **nome qualificato**

```
java.util.Date today = new java.util.Date();
```

- La classe di un package può essere **importata**
- ```
import java.util.Date;
```

...

```
Date today = new Date();
```

## Utilizzo delle classi nei package (2)



- Un intero package può essere **importato**
- ```
import java.util.*;
```
- ...
- ```
Date today = new Date();
```
- Il package **java.lang** è importato di *default*

## Collisione tra i nomi



- La classe `Date` è contenuta nei package `java.util` e `java.sql`

```
import java.util.*;
import java.sql.*;
...
Date today; // compile-time error
```

- Si deve fare uso del *nome qualificato...*

```
import java.util.*;
import java.sql.*;
...
java.util.Date today;
```

## Collisione tra i nomi



- ... oppure si *importa* una classe specifica

```
import java.util.*;
import java.sql.*;
import java.util.Date;
...
Date today; // quella di java.util
```

## Aggiungere una classe ad un package



La parola chiave **package** consente di aggiungere una classe ad un package

```
package it.unipi.iet.dini;
public class MiaClasse {
 // corpo
}
```

MiaClasse.java

- Il file **MiaClasse.java** deve essere posto nella directory **it/unipi/iet/dini**

A partire da quale radice? Lo vediamo più avanti...

## Il package di default



- Se non si inserisce lo statement **package** in un file sorgente, le sue classi sono inserite automaticamente nel *package di default*
- Il package di default non ha nome

## Esempio



```
com/foo/bar/
```

```
 abc/
```

```
 A.java
```

```
 A.class
```

```
 xyz/
```

```
 B.java
```

```
 B.class
```

```
import com.foo.bar.xyz.B;
package com.foo.bar.abc.A
```

```
public class A {
 // ...
 B b = new B();
 // ...
}
```

```
> javac com/foo/bar/abc/A.java
> java com.foo.bar.abc.A
```

oppure

```
package com.foo.bar.xyz;
```

```
public class B {
 // corpo di B
}
```

```
> cd com/foo/bar/abc
> javac A.java
> java A
```

TIGA

11

## Percorso delle classi



- Il percorso delle classi (**class path**) è una collezione di directory a partire dalle quali l'interprete cerca le classi
- L'opzione **-classpath** e la variabile di ambiente **CLASSPATH** permettono di specificare il percorso delle classi

TIGA

12

## Percorso delle classi



- Unix

```
javac -classpath /home/classes:./home/archives
```

- Windows

```
javac -classpath %home%\classes;.;%home%\archives
```

- Unix/Linux, C shell (csh; .cshrc)

```
setenv CLASSPATH /home/classes:./home/archives
```

- Unix/Linux, Bourne again shell (bash; .bashrc)

```
export CLASSPATH=/home/classes:./home/archives
```

- Windows NT/2000/XP

Avvio – Impostazioni - Pannello di controllo - Sistema – Variabili d'ambiente

## Percorso delle classi

(II)



- Dov'è che l'interprete va a cercare **com.foo.bar.xyz.B.class**?
- Per prima cosa prova tra i class file di sistema contenuti in **jre/lib** e **jre/lib/ext**.
- Poi, prova *in sequenza* le directory specificate nel class path
  - **/home/classes/com/foo/bar/xyz**
  - **./com/foo/bar/xyz**
  - **/home/archives/com/foo/bar/xyz**
- Se non lo trova produce un errore

## Reperimento delle classi



- Il compilatore reperisce le classi contenute in un package e specificate con **import** e le compila (se necessario)
- Il compilatore non verifica che la directory in cui si trova effettivamente una classe sia consistente con (il nome de) il package di cui fa parte
- Nel bytecode, per il riferimento a classi si usano sempre i nomi completi di tali package

## Reperimento delle classi



- Il reperimento delle classi è più complesso per il compilatore che per JVM
- Nel caso peggiore compilatore deve cercare una classe in tutti i package ed in tutti i percorsi specificati dal class path
- Esempio
  - il sorgente contiene
    - **import java.util.\*;**
    - **import it.unipi.iet.dini.tiga.\*;**
  - e riferisce la classe
    - **Employee** (continua)



# Reperimento delle classi

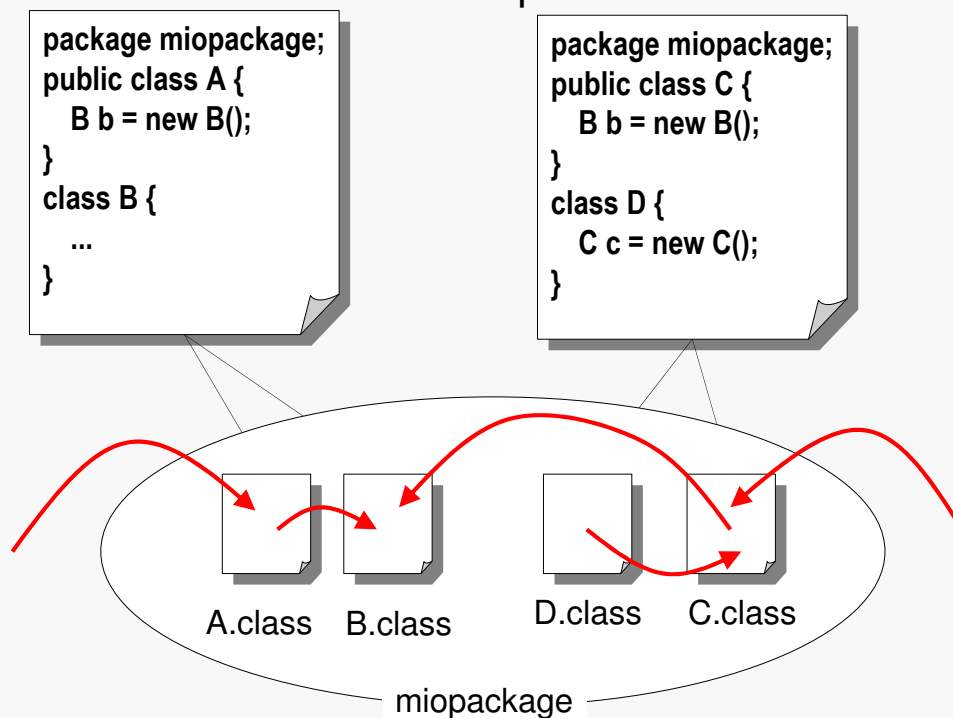


- Esempio
  - nel caso peggiore, il compilatore cerca
    - `java.lang.Employee`
    - `java.util.Employee`
    - `it.unipi.iet.dini.tiga.Employee`
  - nel
    - package corrente
    - ed in tutte le locazioni specificate dal class path

# Accesso alle classi



unità di compilazione





- *Accesso pubblico.*
  - Il modificatore di accesso **public** specifica che la classe è accessibile da qualunque parte del programma
- *Accesso a livello di package.*
  - L'assenza del modificatore di accesso **public** specifica che la classe è accessibile solo all'interno del package  
In questo caso è buona norma
    - definire **private** i campi membro
    - dare alle funzioni membro l'accessibilità a livello di package
- Non è possibile dichiarare **private** o **protected** una classe