Corso di Laurea in Ingegneria Informatica e Automatica
A. A. 2016/2017

Programmazione orientata agli oggetti

Luca Iocchi, Massimo Mecella

L3.1
Definizione di classi in Java

Sommario

- Definizione di classi
  - Campi dati
  - Campi operatori
- Modificatori di accesso
- Costruttori
- Riferimento oggetto this

Classi Java

La definizione di una classe è caratterizzata da:

- nome della classe che identifica la classe stessa, e quindi identifica il tipo delle sue istanze
- variabili di istanza (detti anche campi dati) che rappresentano le proprietà degli oggetti
- metodi (detti anche campi operazione), cioè metodi degli oggetti della classe, che consentono di specificare le operazioni invocabili sugli oggetti della classe.

Mediante opportuni modificatori si può imporre quali campi sono visibili all'esterno della classe.

Classi Java

Sintassi

```
public class Nome {
    campo-1
    ...
    campo-n
}
```

- I campi dati (o variabili d'istanza) servono a rappresentare la struttura interna e lo stato a run-time degli oggetti appartenenti alla classe.
- I campi operazione (o metodi) servono a realizzare le funzionalità della classe.

Classi Java

Definizione di campi dati (variaibli di istanza)

```
modificatore_accesso tipo nome;
```

Definizione di campi operazione (metodi)

```
modificatore_accesso tipo nomeMetodo ( parametri formali )
blocco_istruzioni
```

Nota:

- le variabili di istanza sono visibili all'interno dei metodi della classe.

Esempio: classe Persona

```
public class Persona {
   // variabili di istanza (campi dati)
   private String nome;
   private int eta;

   // metodi (campi operazione)
   public String getNome() {  return nome;  }
   public int getEta() {  return eta;  }
   ...
}
```

Persona
nome: String età: int

Modificatori di accesso

I modificatori di accesso consentono di definire la visibilità (accesso) dei campi (dati e operazioni) definiti all'interno di una classe.

In Java esistono quattro tipi di modificatori di accesso

- public – accesso da qualsiasi altra classe
- modificatore di default – accesso da tutte le classi del package
- protected – accesso da tutte le classi derivate
- private – accesso solo dalla classe in cui il campo è definito

Modificatori di accesso

Regola generale

Campi dati (variabili di istanza)	non public tipicamente private
Campi operazioni (metodi) di servizio	public
Campi operazioni (metodi) di supporto	private

Costruttori

Metodi che creano (allocano memoria per) gli oggetti della classe e inizializzano le variabili di istanza.

- Sono metodi (non static) di una classe
- Hanno lo stesso nome della classe
- Non hanno un tipo di ritorno esplicito (neanche void).

Note:

- una classe può avere più metodi costruttori
- se non viene specificato nessun costruttore, viene assegnato un costruttore di default senza argomenti

Esempio di metodo costruttore

```
public class Persona {
    // variabili di istanza (campi dati)
    private String nome;
    private int eta;

    // costruttore
    public Persona(String n, int e) {
        nome = n;
        eta = e;
    }
    ...
}
```

Riferimento all'oggetto this

this è un riferimento all'oggetto di invocazione del metodo in esecuzione

In caso di omonimia delle variabili di istanza e delle variabili o dei parametri formali dichiarati nel metodo, this consente di discriminare una variabile di istanza da una variabile locale.

Esempio di metodo costruttore con this

```
public class Persona {
    // variabili di istanza (campi dati)
    private String nome;
    private int eta;

    // costruttore
    public Persona(String nome, int eta) {
        this.nome = nome;
        this.eta = eta;
    }
    ...
}
```

Uso di oggetti di classi definite

```
public static void main(String[] args) {
      Persona p;
      p = new Persona("Mario Rossi",32);
      System.out.println(p.getNome());
      System.out.println(p.getEta());
}
```

Errore NullPointerException

Il tentativo di applicare metodi di istanza ad una variabile che non si riferisce ad un oggetto (quindi che contiene il valore null) genera un errore a run-time di tipo NullPointerException.

Esempio

```
public static void main(String[] args) {
      Persona p=null;
      // la variabile p non si riferisce ad alcun oggetto
      System.out.println(p.getNome());
      System.out.println(p.getEta());
}
```

Uso di oggetti (singolo file)

file Persona.java

```
public class Persona {
      // variabili di istanza (campi dati)
      ...
      // metodi (campi operazione)
      …

      // metodo statico main
      public static void main(String[] args) {
          Persona p = new Persona("Mario Rossi",32);
          System.out.println(p.getNome());
          System.out.println(p.getEta());
      }
}
```

Uso di oggetti (più file)

file Persona.java

```
public class Persona  {
      // variabili di istanza (campi dati)
      ...
      // metodi (campi operazione)
      …
}
```

file ProvaPersona.java

```
public classe ProvaPersona {
      // metodo statico main
      public static void main(String[] args) {
          Persona p = new Persona("Mario Rossi",32);
          System.out.println(p.getNome());
          System.out.println(p.getEta());
      }
}
```

Classi con soli campi dati pubblici

Classi con soli campi dati pubblici equivalgono ai record (o strutture)

Esempi

```
class NumeroComplesso {
   public double re, im;
}

class Data {
   public int giorno, mese, anno;
}
```

Classi non pubbliche

Classi non publiche possono essere definite in qualsiasi file che contiene una (e una sola) classe pubblica.

Esempio

```
// file Persona.java
class Data {
   public int giorno, mese, anno;
}

public class Persona {
   ....
}
```

Esempio: classe Moneta

Definiamo una classe Moneta con il campo dati (proprietà):
faccia, variabile int che rappresenta la faccia corrente della moneta (assume i valori TESTA e CROCE, costanti di tipo int)

Il comportamento della classe è definito dalle seguenti operazioni:

- costruttore Moneta, per costruire un oggetto
- metodo lancia, per lanciare la moneta
- metodo leggiFaccia, per riportare la faccia corrente
- metodo stampa, per stampare una descrizione dello stato della moneta

Esempio: Moneta

```
public class Moneta {

   // definizione di costanti
   public static final int TESTA=0;
   public static final int CROCE=1;

   // variabile di istanza
   private int faccia;

   ....
```

Esempio: Moneta

```
public class Moneta {

   ....

   // costruttore
   public Moneta() {
      lancia();
   }

   public void lancia() {
      faccia = Math.random()<0.5?TESTA:CROCE;
   }
```

Esempio: Moneta

```
public class Moneta {

   ....

   public int leggiFaccia() {
      return faccia;
   }

   public void stampa() {
      System.out.println("Lo stato della moneta è "
                         +((faccia==TESTA)?"TESTA":"CROCE"));
   }
```

Esempio: Moneta

```
public static void main(String[] args) {
    Moneta m = new Moneta();
    int nt=0, nc=0; int n=10;
    for (int i=0; i<n; i++) {
       m.lancia();  m.stampa();
       if (m.leggiFaccia()==Moneta.TESTA)
          nt++;
       else
          nc++;
    }
    System.out.println("Con "+n+" lanci della moneta abbiamo ottenuto "
                                    +nt+" TESTA e "+nc+" CROCE");
}
```

Esercizio 3.1.1

Scrivere una classe Libro con le proprietà titolo e prezzo e le seguenti operazioni:

- metodo costruttore con argomento titolo
- metodi di lettura delle proprietà
- metodo di modifica del prezzo

Scrivere un programma di prova che crea alcuni oggetti della classe Libro, inserisca e stampi i relativi valori.