

# OOP in JAVA



Programmazione  
orientata agli Oggetti

Giulio Angiani  
I.I.S. "Blaise Pascal" - Reggio Emilia





# OOP

Object Oriented Programming



# Metodi Set e Get

Come accedere e modificare **attributi privati**?

Un pattern di programmazione standard è l'uso di

- metodi **setter** per assegnare valori agli attributi **privati**
- metodi **getter** per leggere i valori degli attributi **privati**

Avendo

```
private <paramtype> paramname;
```

Il corrispondente metodo **getter** ha la forma

```
public <paramtype> getParamname() {  
    return this.paramname;  
}
```

NOTA: gli attributi **public** non hanno bisogno di metodi per essere letti o scritti



# Metodi Set e Get (2)

Riprendiamo l'esempio della classe PuntoCartesiano

Aggiugiamo i metodi **get** per i due attributi privati x e y;

```
public class PuntoCartesiano {  
    private double x; // <-- attributo privato  
    private double y; // <-- attributo privato  
  
    public PuntoCartesiano(double ics, double ipsilon) {  
        this.x = ics;  
        this.y = ipsilon;  
    }  
  
    public double getX() {  
        return this.x;  
    }  
  
    public double getY() {  
        return this.y;  
    }  
}
```

JAVA



# Metodi Set e Get (3)

Struttura standard dei metodi **set** per la modifica dei valori di attributi privati.

L'uso dei **set** ci permette anche di **controllare** il valore prima di assegnarlo effettivamente ad un attributo privato

- realizziamo il pattern di **controllo dell'input**

Avendo

```
private <paramtype> paramname;
```

JAVA

Il corrispondente metodo **setter** ha la forma

```
public void setParamname(<paramtype> p) {  
    this.paramname = p;  
}
```

JAVA



# Metodi Set e Get (4)

Torniamo alla classe PuntoCartesiano

Se vogliamo permettere la modifica dei valori degli attributi anche **dopo** l'instanziazione dell'oggetto, aggiungiamo anche i metodi **set**

```
public class PuntoCartesiano {  
    private double x; // <-- attributo privato  
    private double y; // <-- attributo privato  
  
    public PuntoCartesiano(double ics, double ipsilon) {  
        this.x = ics;  
        this.y = ipsilon;  
    }  
  
    public void setX(double p) {  
        this.x = p;  
    }  
  
    public void setY(double p) {  
        this.y = p;  
    }  
}
```

JAVA



# Metodi Set e Get (5)

Immaginiamo di voler impedire di avere valori di **x** e **y** al di fuori del primo quadrante del piano cartesiano:

Possiamo modificare i metodi **set** come indicato

```
public void setX(double p) {  
    if (p>0) this.x = p;  
}  
  
public void setY(double p) {  
    if (p>0) this.y = p;  
}
```

JAVA



# Rappresentazione Stringa di un Oggetto (1)

Un altro metodo molto utilizzato è il metodo **toString()** per avere una rappresentazione stringa di un oggetto

Il metodo **deve** essere definito come

```
public String toString() {  
    return <una stringa>  
}
```

JAVA



# Rappresentazione Stringa di un Oggetto (2)

```
public String toString() {  
    return "[" + this.x + ":" + this.y + "]";  
}
```

JAVA

L'output di

```
PuntoCartesiano p0 = new PuntoCartesiano(5,3);  
System.out.println("p0 = " + p0);
```

JAVA

diventa

```
p0 = [5.0:3.0]
```

OUTPUT





Giulio Angiani  
I.I.S. "Blaise Pascal" - Reggio Emilia