

Initiation à la programmation objet avec Java

Intervenant : Jean-Frédéric VINCENT



The background of the slide is a close-up photograph of numerous long, narrow green leaves, likely from a plant like an iris, which are layered and slightly curved. The lighting is soft, creating a natural, organic feel. A semi-transparent dark green horizontal band is positioned across the middle of the image, serving as a backdrop for the text.

Module 8

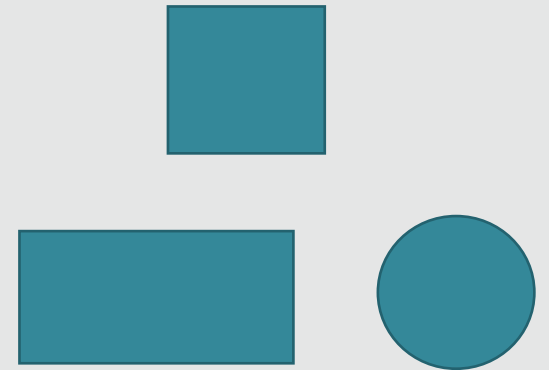
POO : Héritage

La Programmation orientée objet

Démonstration

Cahier des charges : Projet Dessin

- Nous devons placer des formes sur un repère orthonormée en X et Y
 - Un rectangle, un cercle et un carré
 - Un rectangle possède une longueur et une largeur
 - Il est possible de calculer la superficie



POO : l'héritage

Problème

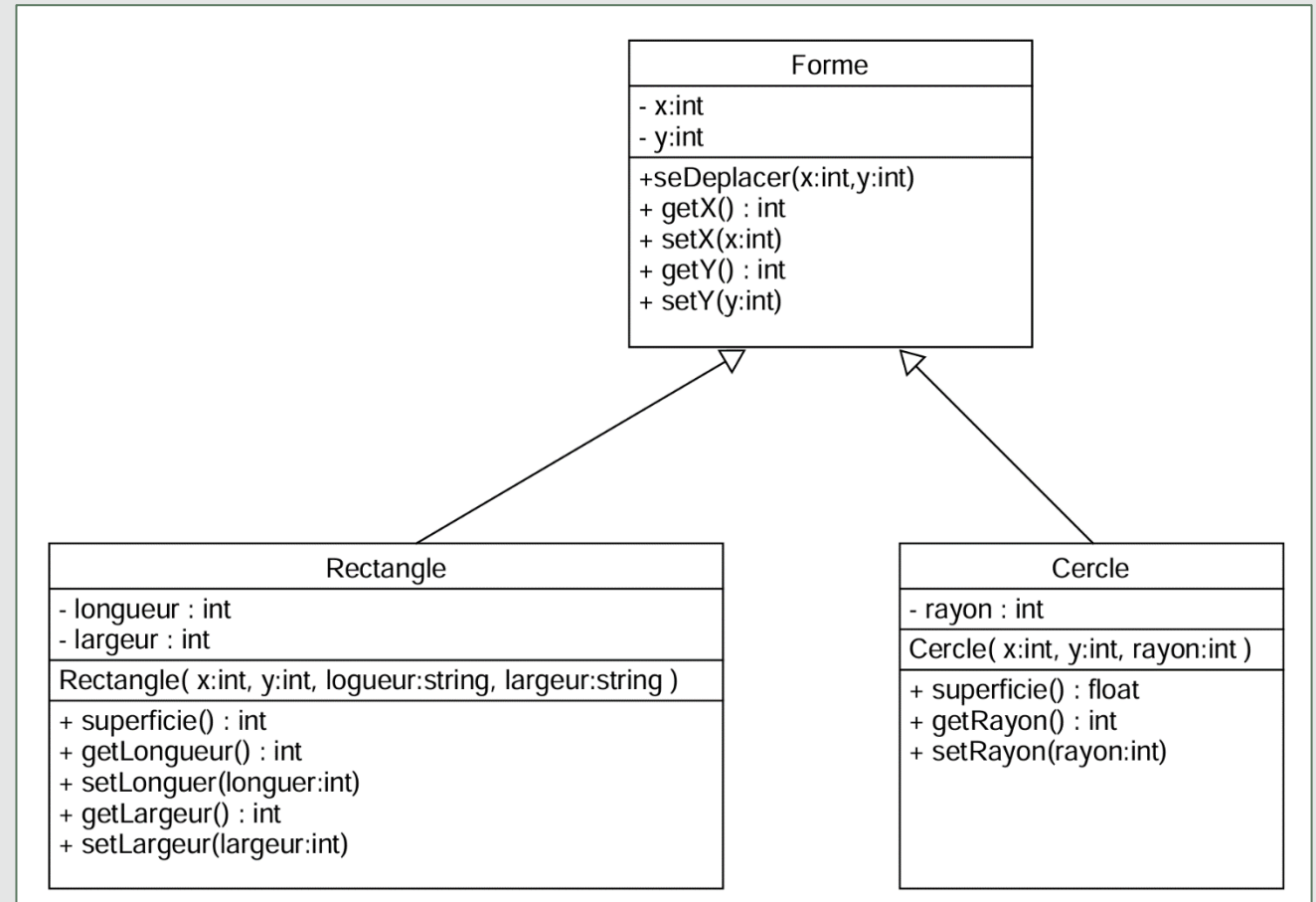
- Le code de la fonction seDeplacer() est dupliqué à 2 endroits
- Les getter et setter de x et y se répète

Rectangle	Cercle
- x : int - y : int - longueur : int - largeur : int	- x : int - y : int - rayon : int
Rectangle(x:int, y:int, longueur:string, largeur:string)	Cercle(x:int, y:int, rayon:int)
+ superficie() : int + seDeplacer(x:int,y:int) + getX() : int + setX(x:int) + getY() : int + setY(y:int) + getLongueur() : int + setLonguer(longuer:int) + getLargeur() : int + setLargeur(largeur:int)	+ superficie() : float + seDeplacer(x:int,y:int) + getX() : int + setX(x:int) + getY() : int + setY(y:int) + getRayon() : int + setRayon(rayon:int)

POO : l'héritage

La solution : l'héritage

- Création d'une classe mère Forme



POO : l'héritage

L'héritage

protected

Dans la classe mère *Forme* :

`protected` donne l'accès aux attributs aux classes enfants

```
public abstract class Forme {  
    → protected int x;  
    protected int y;  
  
    public Forme(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
    // [...]  
}
```

POO : l'héritage

L'héritage

abstract

bloque la création d'objet de type `Forme` dans le controller



```
public abstract class Forme {  
    protected int x;  
    protected int y;  
  
    public Forme(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
    // [...]  
}
```

```
Forme f = new Forme (10,10);
```

✖ Cannot instantiate the type Forme
Press 'F2' for focus

La solution : l'héritage

Extends

Pour définir qu'une classe hérite d'une classe mère

Nous allons hériter des méthodes et des attributs



```
public class Rectangle extends Forme{  
    private int longueur;  
    private int largeur;  
  
    public Rectangle(int x, int y, int longueur, int largeur) {  
        super(x, y);  
        this.longueur = longueur;  
        this.largeur = largeur;  
    }  
}
```


La solution : l'héritage

super

- Pour appeler le constructeur de la classe mère

```
public class Rectangle extends Forme{  
    private int longueur;  
    private int largeur;  
  
    public Rectangle(int x, int y, int longueur, int largeur) {  
        → super(x, y);  
        this.longueur = longueur;  
        this.largeur = largeur;  
    }  
}
```

POO : l'héritage

La solution : l'héritage


final

final bloque l'héritage : on ne peut pas hériter de cercle



```
public final class Cercle extends Forme{  
    private int rayon;  
    public Cercle(int x, int y, int rayon) {  
        super(x, y);  
        this.rayon = rayon;  
    }  
    // [...]  
}
```

```
public class Rond extends Cercle{  
    }  
}
```

 The type Rond cannot subclass the final class Cercle
1 quick fix available:
[Remove 'final' modifier of 'Cercle'](#)
Press 'F2' for focus