INSA Centre Val de Loire
Département STI

Année 2024-2025

# Programmation Orientée Objet

# TD 3 – Héritage, Polymorphisme et Liaison

**NB :** le franąis et l'anglais sont parfois un peu mélangés. Ainsi, FigureGeometrique et GeometricalShape, par exemple, font bien référence au même concept. **Code de départ :** Ce TD reprends le projet `Point`, `Segment`, `Quadrilateral`, `Rectangle` etc enrichi en exercices d'application du poly héritage. Une version de dÃ©part est disponible sur celene.

**À faire** :

1. Du code .java
2. Un compte rendu dans lequel vous répondez aux questions posées dans le sujet. Le symbole ◇ signale qu'une question nécessite un commentaire dans le compte-rendu (il est possible que certaines questions nécessitant commentaire n'aient pas de ◇).

## Exercice 1 : Polymorphisme, liaison

**a :** Créer une méthode `draw` dans la classe `GeometricalShape` qui se contente d'afficher un message, déclinée en :

— `draw(int zone)`, qui affiche simplement *Drawing the *valeur de zone*th zone of a shape.*

— `draw()`, qui affiche *Drawing a geometrical shape*

**b :** ◇ (Affichage lors de l'exécution puis commentaire) Ajouter une méthode `draw()` à la classe `Rectangle` qui affiche simplement le message *Drawing a rectangle.* Instancier un rectangle et vérifier le comportement de chacune des méthodes `draw` sur l'objet de type `Rectangle` en utilisant le code suivant :

```
1 Rectangle r = new Rectangle();
  r.draw(4);
3 r.draw();
```

**c :** ◇ De quels types de polymorphisme s'agit-il (qu'a-t-on fait à la méthode) ?

**d :** ◇ (Question) Tester le code suivant :

```
1 GeometricalShape [] listFig = new GeometricalShape[3];
  listFig[0] = new Rectangle();
3 listFig[1] = new Quadrilateral();
  listFig[2] = new Ellipse();
```

Pourquoi ce code est-il acceptable malgré les différences de type déclaré de listFig et des objets auxquels les éléments du tableau font référence ? De quel type de transtypage s'agit-il ?

**e :** ◇ Tester le code suivant :

```
1 Rectangle rec = new Quadrilateral();
```

Expliquer le message d'erreur à la compilation.

**f :** ◇ (Affichage lors de l'exécution puis commentaire + questions) Tester le code suivant :

```
1 GeometricalShape [] listFig = new GeometricalShape[3];
  listFig[0] = new Rectangle();
3 listFig[1] = new Quadrilateral();
  listFig[2] = new Ellipse();
5 for (int i=0;i<=2;i++) {
    listFig[i].draw();
7 }
  listFig[0].getPerimeter();
```

Commentez la méthode *getPerimetre()* de FigureGeometrique. Expliquez l'erreur, puis décommentez.

En ne modifiant que la dernière ligne, comment faire pour utiliser la méthode `getPerimeter()` de `Rectangle` sur listFig[0] ?

**g :** ◇ Sans modifier l'implémentation des classes ni l'objet référencé par listFig[0], et en modifiant uniquement le *main*, est-il possible de déclencher la méthode `getPerimeter()` de `Quadrilateral` pour listFig[0], plutôt que la versions redéfinie dans `Rectangle` ?

**h :** Expliquez le principe de la liaison tardive.

**i :** Dans la classe `GeometricalShape`, déclarer une variable d'instance `code`, de type `int` et de visibilité `public`. L'initialiser à la valeur 0 dans les constructeurs. Dupliquer (c'est à dire **re-déclarer**) cette variable dans les classes `Quadrilateral`, `Rectangle` et `Ellipse`, en l'initialisant, respectivement à 1, 2 et 3.

**j :** ◇ (Affichage lors de l'exécution puis commentaire + questions) À la suite de l'initialisation de listFig, tester le code suivant :

```
  for (int i=0;i <=2;i++)
2   System.out.println(listFig[i].code);
```

Expliquer le principe de la liaison statique propre aux attributs.

À l'aide du transtypage, proposer un moyen d'atteindre, pour l'objet de type déclaré `GeometricalShape`, la valeur des codes des sous-objets `Ellipse`, `Quadrilateral` et `Rectangle`.

Sans transtypage, comment récupérer la bonne valeur ?

**Exercice 2 : Visibilité, abstract, final**

**a :** ◇ Ajouter le mot-clé **final** à la déclaration de la classe `Conical`. Constater et expliquer le message d'erreur à la compilation. Annuler la modification.

**b :** ◇ Restreindre la visibilité de la méthode `draw` de `Quadrilateral` à **protected**. Constater et expliquer le message d'erreur à la compilation. Corriger.

**c :** ◇ Ajouter le code suivant à votre méthode de test :

```
GeometricalShape gs = new GeometricalShape();
```

Que se passe-t-il ? Supprimer la ligne ci-dessus de votre méthode de test.