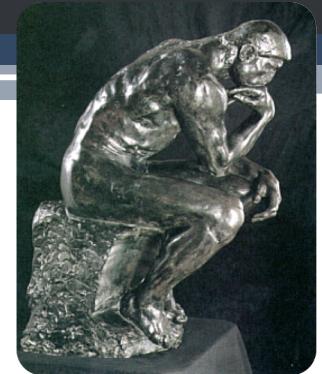


Réflexivité, Introspection

La Réflexivité

- Capacité d'un programme à **examiner** et à éventuellement **modifier** sa structure, son état et/ou son comportement
- 2 catégories :
 - L'introspection
 - L'intercession



L'Introspection (1)

du latin « *introspectus* », action de regarder à l'intérieur

Utilité :

- Analyse des classes, découverte d'attributs, de méthodes
- Chargement dynamique de classes, création d'instances
- Inspecter des objets dynamiquement
- Manipuler des tableaux génériques, créer des méthodes génériques
- Débogage, profiling, complétion automatique
- Etc.

L'Introspection (2)

- Java maintient l'Identification de Type au Run-Time (RTTI) sur tous les objets
 - Permet de connaître la classe réelle d'un objet
 - Permet d'implémenter la liaison dynamique (quelle méthode est réellement appelée) :

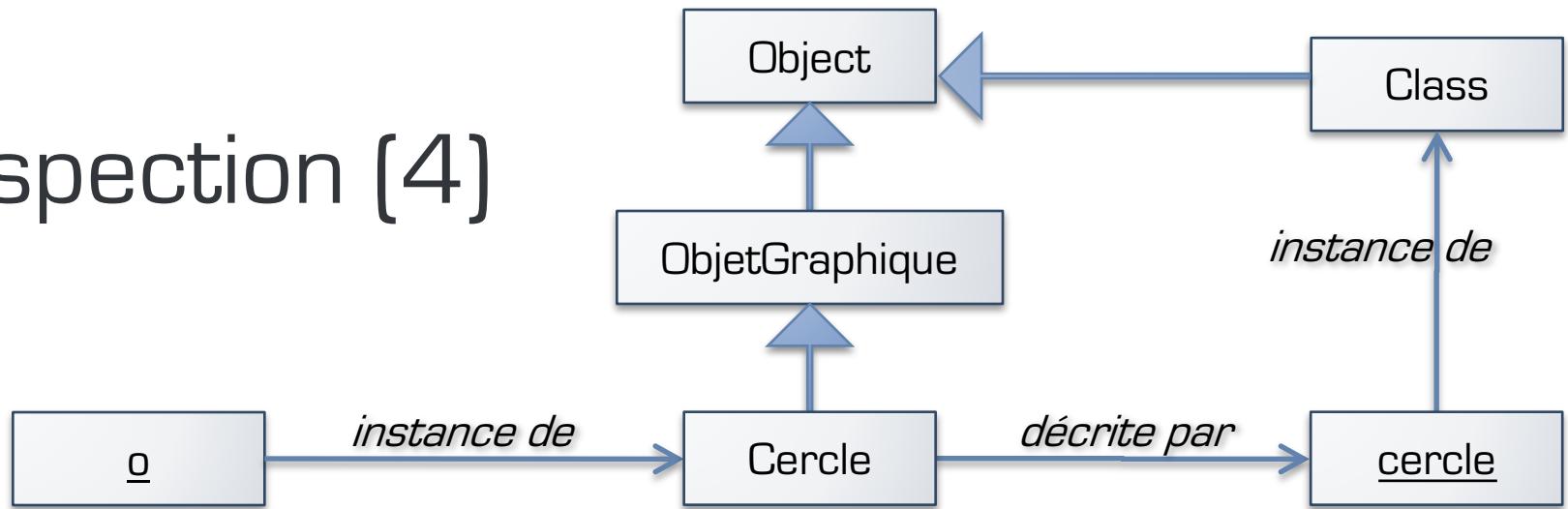
```
ObjetGraphique o = new Cercle();  
o.draw(); // de Cercle ou de ObjetGraphique ?;
```

L'Introspection (3)

- Accès au RTTI grâce à l'API :
`java.lang.reflect`
- On peut ainsi connaître la classe d'un objet :

```
Class c11 = o.getClass();
Class c12 = Cercle.class;
Class c13 = int.class;
```
- Un objet de type Class décrit un type, pas forcément une classe...

L'Introspection (4)



- Les représentations des classes sont des instances de `Class` et sont donc ... des objets !!
 - Possèdent des méthodes, des attributs
 - On peut les appeler
- La représentation de la classe `Object` est aussi une instance de `Class` (puisque `Object` est une classe...)
 - On peut donc l'utiliser de la même façon

L'Introspection (5)

- Exemple d'utilisation :

Utile pour instancer des classes dont le nom n'est connu qu'à l'exécution

```
// Sans utiliser l'introspection  
Foo foo = new Foo();  
foo.hello();
```

```
// En utilisant l'introspection  
Class cl = Class.forName("Foo");  
Method method = cl.getMethod("hello", null);  
method.invoke(cl.newInstance(), null);
```

Appel de méthode

Création d'une nouvelle instance

L'Intercession

- (ou Méta-programmation)
- Capacité d'un programme à modifier son code
- Non disponible en Java

Ex. en Python :

```
>>> class A:  
...     def hello(self):  
...         print "hello"  
...  
>>> A().hello()  
hello  
  
>>> def hello2(self):  
...     print "hello arno"  
...  
>>> A.hello = hello2  
>>> A().hello()  
hello arno
```