2I002
EGALITÉ ENTRE OBJETS, CLONAGE D'OBJET

Vincent Guigue & Christophe Marsala

SCIENCES SORBONNE UNIVERSITÉ
LIP6
MLIA Machine Learning & Deep Learning for Information Access

- Le signe = se comporte de manière spécifique avec les objets...
- Le signe == également spécifique avec les objets...

Vocabulaire (uniquement pour les opérations sur objets)

new Instanciation / Création d'instance


```
1  Point p = new Point(1,2);
```

- Le signe = se comporte de manière spécifique avec les objets...
- Le signe == également spécifique avec les objets...

Vocabulaire (uniquement pour les opérations sur objets)

new Instanciation / Création d'instance

= Création de référence


```
1  Point p = new Point(1,2);
2  Point q = p;
```

- Le signe = se comporte de manière spécifique avec les objets...
- Le signe == également spécifique avec les objets...

Vocabulaire (uniquement pour les opérations sur objets)

new Instanciation / Création d'instance

= Création de référence

== Egalité référentielle


```
1  Point p = new Point(1,2);
2  Point q = p;
3  Point r = new Point(1,2);
4  System.out.println((p==q) +" " + (p==r));// true false
```

Comment dupliquer une instance?

Idée (assez raisonnable somme toute)

Créer une nouvelle instance dont les valeurs des attributs sont identiques

- Exemple de code : clonage de `Point`

```
1 // Ou placer le code?
```

Comment dupliquer une instance?

Idée (assez raisonnable somme toute)

Créer une nouvelle instance dont les valeurs des attributs sont identiques

- Exemple de code : clonage de `Point`

```
1 // Ou placer le code?
2 public class Point{
3 // Quelle signature de methode?
```

Idée (assez raisonnable somme toute)

Créer une nouvelle instance dont les valeurs des attributs sont identiques

- Exemple de code : clonage de `Point`

```
1 // Ou placer le code?
2 public class Point{
3 // Quelle signature de methode?
4   public Point clone(){
5     return new Point(x, y);
6   }
7 }
```

- Usage:

```
1 // main
2 Point p = new Point(1,2);
3 Point p2 = p.clone();
```

NB: construction de nouvelle instance sans `new` explicite dans le `main`

Avant le JAVA, il y avait le C++

En C++ : la syntaxe standard = **constructeur de copie**

- Exemple de code dans la classe `Point`

```
1 // Constructeur de Point a partir d'un autre Point
2 public Point(Point p){
3   this.x = p.x; // this facultatif
4   this.y = p.y; // this facultatif
5 }
```

- Usage:

```
1 Point p = new Point(1,2);
2 Point p2 = new Point(p);
```

- Résultat ABSOLUMENT identique (depuis JAVA 1.5)
- Avantage du clone en JAVA: il s'agit d'une méthode standard (= + facile à lire)

Comment tester l'égalité structurelle?

Créer une méthode qui teste l'égalité des attributs

- **Solution 1** (simple mais pas utilisée)

```java
// Dans la classe Point
public boolean egalite(Point p){
  return p.x == x && p.y == y;
}
  // equivalent simplifie de
  // if(p.x == x && p.y == y)
  //   return true
  // else
  //   return false;
```

```java
// Dans le main

Point p1 = new Point(1.,2.);
Point p2 = p1;
Point p3 = new Point(1.,2.);
Point p4 = new Point(1.,3.);

p1.egalite(p2); // true
p1.egalite(p3); // true
p1.egalite(p4); // false
```

- `public boolean egalite(Point p)` produit le résultat attendu
- ⚠ ATTENTION à la signature:
  - la méthode retourne un booléen
  - la méthode ne prend qu'un argument

    (on teste l'égalité entre l'instance qui invoque la méthode et l'argument)

Méthode standard: `boolean equals(Object o)`

- `equals` existe dans tous les objets (comme `toString`)
  - mais teste l'égalité référentielle... Pas intéressant (comme `toString`)
- $\Rightarrow$ Redéfinition = faire en sorte de tester les attributs
  Un processus en plusieurs étapes:
  1. Vérifier s'il y a égalité référentielle / pointeur null
  2. Vérifier le type de l'`Object` o (cf cours polymorphisme)
  3. Convertir l'`Object` o dans le type de la classe (idem)
  4. Vérifier l'égalité entre attributs

```
1     public boolean equals(Object obj) {
2         if (this == obj) return true;
3         if (obj == null) return false;
4         if (getClass() != obj.getClass())
5           return false;
6         Point other = (Point) obj;
7         if (x != other.x) return false;
8         if (y != other.y) return false;
9         return true;
10    }
```