COMMENT AFFECTER UNE VALEUR A CHAQUE ATTRIBUT ?

SOLUTION 1 : 1 SCANNER + 1 SETTER PAR ATTRIBUT

⇓

MAUVAISE SOLUTION

→ TOUT LES ATTRIBUTS VONT FAIRE PARTIE DE L'INTERFACE

CASSE L'ENCAPSULATION

TOUT LES ATTRIBUTS EN PUBLIC ≠ DES SETTERS SI CHANGEMENT ATTRIBUT

→ SI LE PROGRAMMEUR UTILISATEUR OUBLIE D'INITIALISER UN ATTRIBUT

TRANSFERT DE RESPONSABILITÉ AU PROGRAMMEUR

SOLUTION 2 : METHODES DEDIEES = CONSTRUCTEURS

⇓ DEBUT DE VIE OBJET

CONSTRUCTEUR

```
NomClasse ( parametres )
                └→ liste attributs
{
    attribut 1 = parametre
    attribut 2 = parametre
}
```

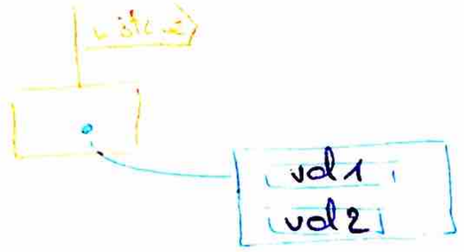
=> PAS DE TYPE DE RETOUR (PAS void !)
=> MEME NOM QUE SA CLASSE
=> INVOQUÉ A CHAQUE INSTANCIATION

CONSTRUCTEURS

INITIALISATION PAR CONSTRUCTEUR

```
NomClasse instance = new NomClone ( val1, val2, ..., valN )
```

↓ valeurs de la liste des arguments en parametre



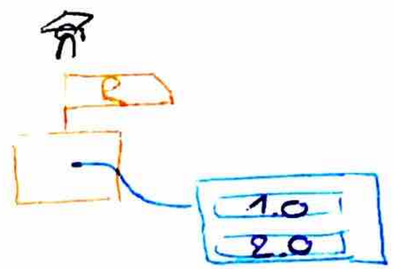
CONSTRUCTEUR PAR DEFAUT

VALEUR QUE L'ON SOUHAITE PAR DEFAUT

```
Rectangle () { h = 1.0 ; l = 2.0 ; }
```

↓ UTILISATION

```
Rectangle r = new Rectangle () ;
```

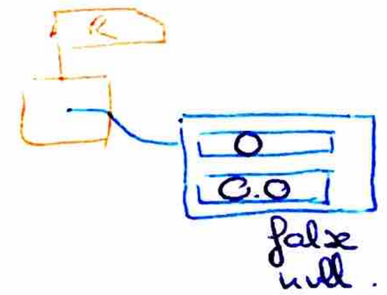


CONSTRUCTEUR PAR DEFAUT PAR DEFAUT

SI PAS DE CONSTRUCTEUR CRÉE

└→ `Rectangle 2 = new Rectangle()`

INITIALISATION DE TOUT LES ATTRIBUTS A 0 / NULL.



⚠ DES CREATION D'UN CONSTRUCTEUR PAR DEFAUT PAR DEFAUT N'EST PLUS FOURNI

APPEL AUX AUTRES CONSTRUCTEUR DE LA MEME CLASSE

```
public Rectangle ( double h, double l )
{
  hauteur = h ; largeur = l
}
public Rectangle () {
  this ( 0.0 , 0.0 ) ;
}
```

this(...)

1 SEULE PAR CONSTRUCTEUR PAS D'INSTRUCTION AVANT

APPEL DU CONSTRUCTEUR A 2 ARGUMENTS.

BONNE PRATIQUE :

→ NE PAS DONNER DE VALEUR PAR DEFAUT AUX ATTRIBUTS.

~~private double hauteur = 4.0;~~

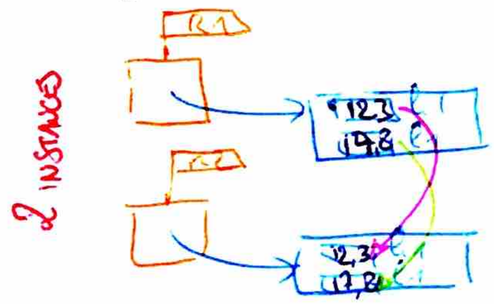
→ PRIVILEGIER L'UTILISATION DES CONSTRUCTEURS

⇓

ILS DOIVENT SE CHARGER DE L'INTEGRETÉ DES INITIALISATIONS.

CONSTRUCTEUR DE COPIE :

```
Rectangle r1 = new Rectangle (12.3, 17.8);
Rectangle r2 = new Rectangle (r1)
```




```
NomClasse (NomClasse autreClasse)
{
    att1 = autreClasse.att1;
    att2 = autreClasse.att2;
}
```

⚠ IL FAUT FAIRE LE CONSTRUCTEUR POUR FAIRE DES COPIES SINON CA NE FONCTIONNE PAS

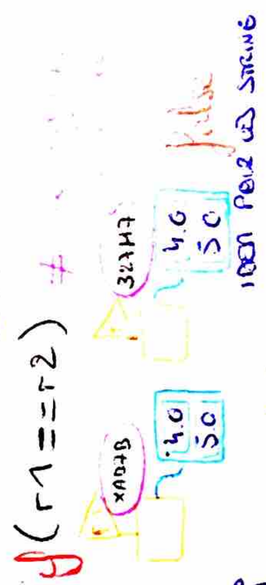
ou

r2 = r1.clone() ← METHODE CLONE.

D'OBJETS

UTILISER LA METHODE A FAIRE DANS LA CLASSE PUIS.. if (r1.equals(r2)) true

COMPARAISON



FIN DE VIE D'UN OBJET

```
METHODE() {
    INSTANCIATION D'UN OBJET
    UTILISATION, FIN D'UTILISATION
}
```

= FIN DE VIE D'UN OBJET → LA REFERENCE N'EST PAS UTILISÉE AILLEURS DANS LE CODE.

AFFECTATION ET COPIES D'OBJET

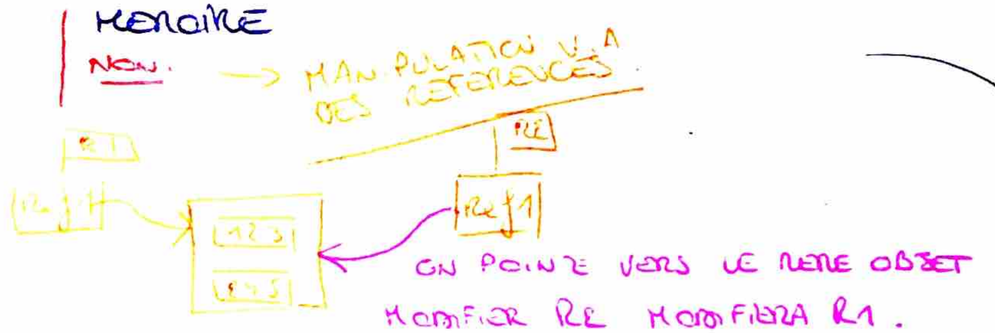
```
Rectangle r1 = new Rectangle (12.3, 24.5);
Rectangle r2 = r1;
```

→ ds certains langage ça marche !

⇓

PROBLEME : DE VRAI 2 OBJETS DISTINCTS EN MEMOIRE

NON. → MANIPULATION VIA DES REFERENCES

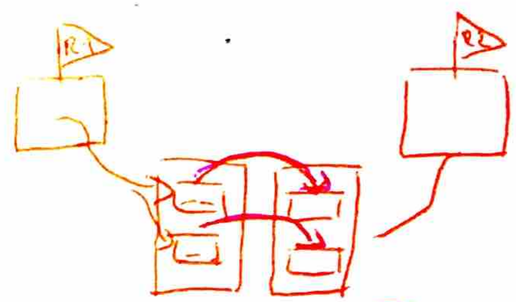


ON POINTE VERS LE MEME OBJET MODIFIER R2 MODIFIERA R1.

SOLUTION

⇓

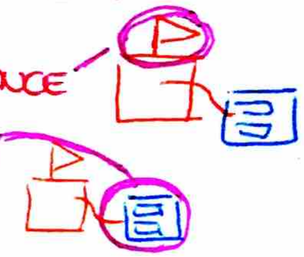
- CONSTRUCTEUR DE COPIE
- CLONE()



AFFICHAGE

Syso(objet) → AFFICHE LA REFERENCE

Syso(objet) → AFFICHE LES ATTRIBUTS



↑

PLACER LA METHODE toString() ds la classe d'objet