

Programmation objet (Cours 3)

1. Constantes
2. Variables et constantes de classe
3. Méthodes de classe

1. Constantes

Constante = variable non modifiable

Une fois initialisée une variable dont le type est précédé du mot
{ clé *final* n'est plus modifiable

final int N = 10 ;

On doit initialiser une constante dès sa déclaration. La valeur fournie est non modifiable.

N = N + 1 ;

rejet du compilateur : impossible de modifier la valeur d'une constante.

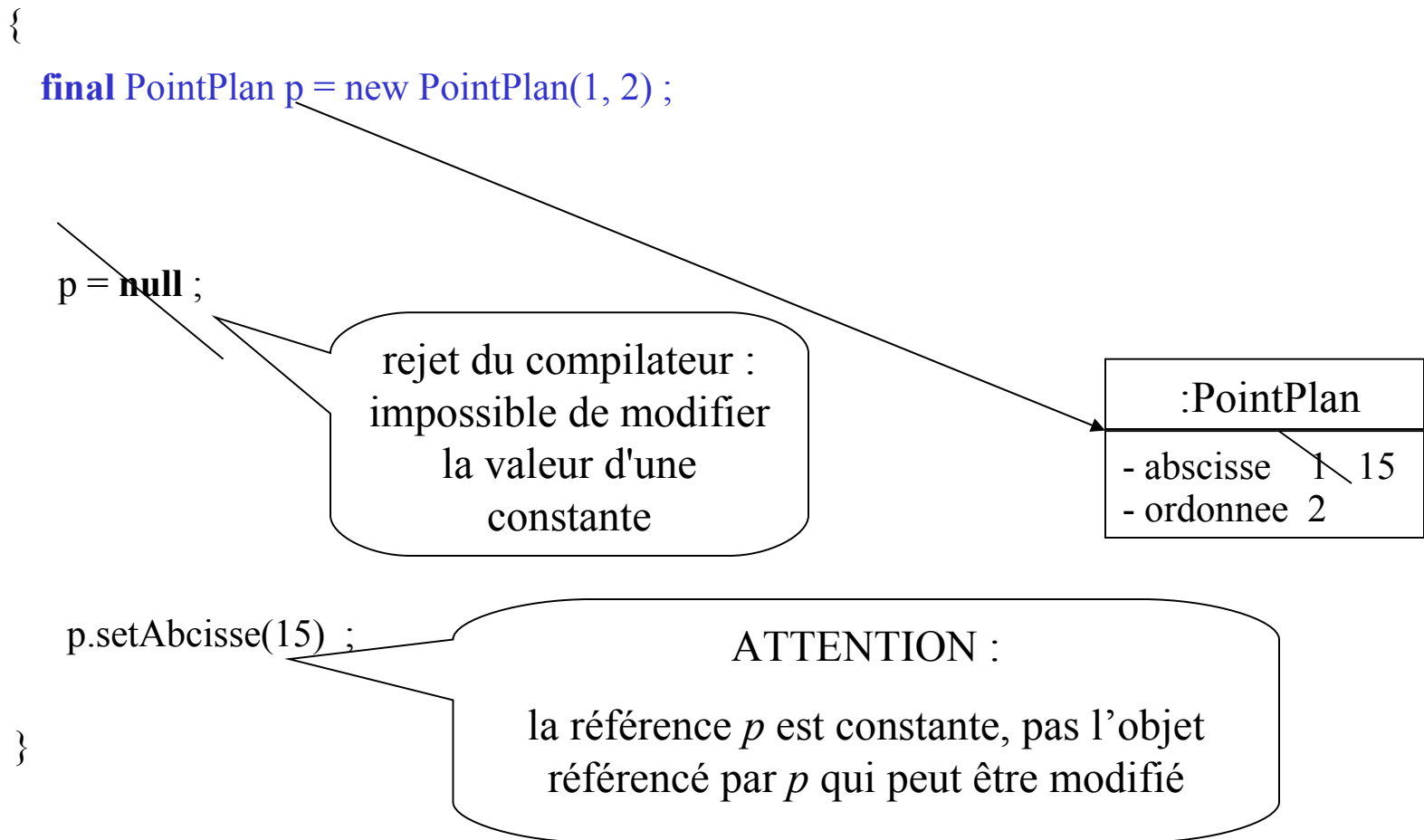
final double GRAVITE = 9.81 ;

}

L'usage est d'écrire les constantes primitives entièrement en majuscules

Référence constante

une référence précédée du mot clé **final** devient non modifiable.



2. Variables de classe

- ✓ Une variable de classe appartient à une classe et non à une instance (objet)
- ✓ Une variable de classe décrit une propriété de la classe et non une propriété d'une instance (objet) particulière
 - ✓ Une variable de classe existe en 1 exemplaire, une variable d'instance possède autant d'exemplaires qu'il y a d'instances (objets)

On déclare une variable de classe en utilisant le mot clé *static*

ex) **public static** int n ;

Exemple (1/2)

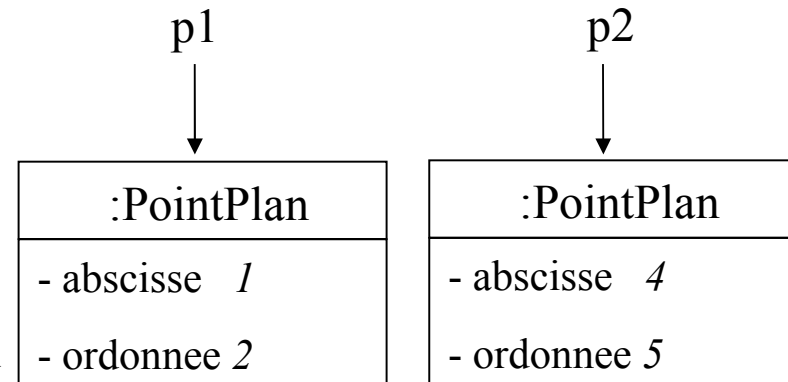
Compter le nombre d'objets *PointPlan* créés

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;
    public static int nbPointsCrees = 0 ;
}
// fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        PointPlan p1 = new PointPlan(1, 2) ;
        PointPlan p2 = new PointPlan(4, 5) ;
    }
} // fin classe TestPointPlan
```

La classe *PointPlan* possède 1 exemplaire de la variable de classe *nbPointCrees*

Chaque instance (objet) *PointPlan* possède 1 exemplaire des variables d'instances *abscisse* et *ordonnée*



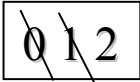
Exemple (2/2)

Compter le nombre d'objets *PointPlan* créés

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    public static int nbPointsCrees = 0 ;

    public PointPlan(float x, float y )
    {
        this.abscisse = x ;
        this.ordonnee = y ;
        PointPlan.nbPointsCrees++ ;
    }
} // fin classe PointPlan
```

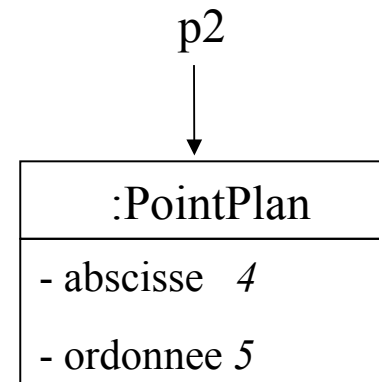
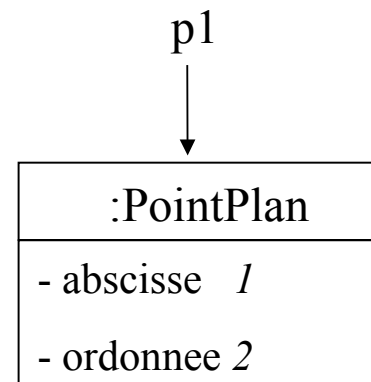

nbPointCrees

On accède à une variable de classe en la faisant précéder du nom de la classe qui la contient

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        PointPlan p1 = new PointPlan(1, 2) ;
        PointPlan p2 = new PointPlan(4, 5) ;

        System.out.println(PointPlan.nbPointsCrees) ;
    }
} // fin classe
```

2



Une variable de classe est indépendante de tout objet créé

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    public static int nbPointsCrees = 0 ;
```

0

nbPointCrees

```
public PointPlan(float x, float y )
{
    this.abscisse = x ;
    this.ordonnee = y ;
    PointPlan.nbPointsCrees++ ;
}
```

```
} // fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        System.out.println(PointPlan.nbPointsCrees) ;
    }
}
```

0

Aucun objet *PointPlan* n'a été créé. La variable de classe n'en est pas moins utilisable puisqu'elle appartient à la classe et non à un objet

Une variable de classe peut être *public* ou *private*

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    private static int nbPointsCrees = 0 ;

```

0

nbPointCrees

```

} //fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        System.out.println(PointPlan.nbPointsCrees) ;
    }
}
```

Erreur à la compilation : la variable est privée.

Constante de classe = variable de classe non modifiable

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    private static int nbPointsCrees = 0 ;

    public static final float ORIGINE_X = 0.0 f ;

    public static final float ORIGINE_Y = 0.0 f ;

} // fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        System.out.println(PointPlan.ORIGINE_X) ;
    } // fin main
} // fin TestPointPlan
```

Les constantes de classes à vocation publique (accessible depuis l'extérieur de la classe) possède l'accès **public**. Elles sont initialisées dès leur déclaration et sont non modifiables : il est inutile de protéger leur contenu

3. Méthodes de classe

Une méthode de classe ne s'applique pas à un objet

Une méthode de classe décrit un service rendu par la classe qui n'est lié à aucun objet

On déclare une méthode de classe en utilisant le mot clé ***static***

ex) **public static** int getNbPointCrees() {...}

Exemple 1

Une méthode retournant le nombre d'objets *PointPlan* créés

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;
    private static int nbPointsCrees = 0 ;
    ...
}
```

0

nbPointCrees

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        PointPlan p1 = new PointPlan(1, 2) ;
        PointPlan p2 = new PointPlan(4, 5) ;
        int n = PointPlan.getNbPointsCrees() ;
    }
}
```

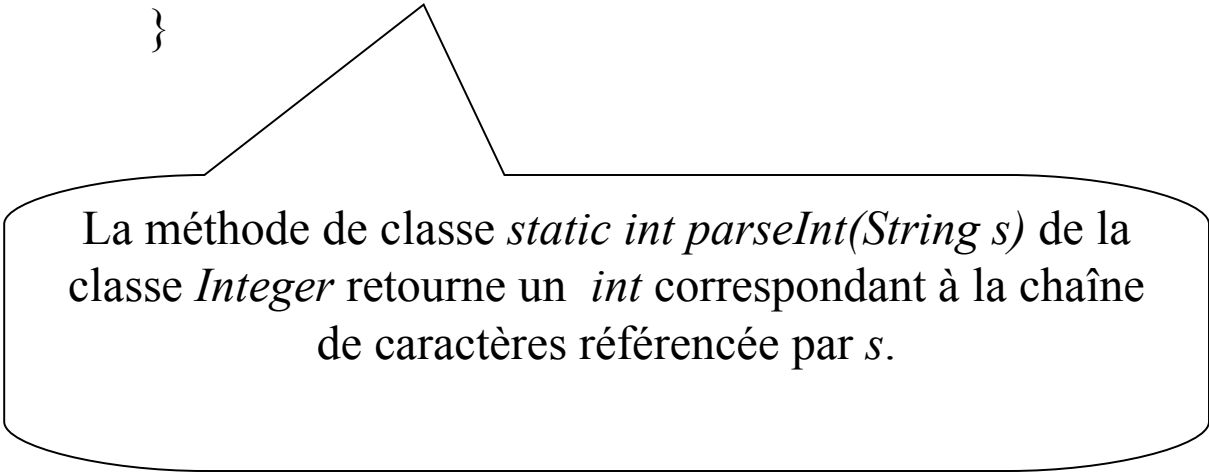
```
public static int getNbPointsCrees()
{
    return PointPlan.nbPointsCrees ;
}
// fin classe PointPlan
```

On applique la méthode *getNbPointCrees* à la classe en faisant précéder le nom de la méthode par le nom de classe.

Exemple 2

Conversion d'une chaîne de caractères en entier (int)

```
{  
    int n = Integer.parseInt("234" ) ;  
}
```



La méthode de classe *static int parseInt(String s)* de la classe *Integer* retourne un *int* correspondant à la chaîne de caractères référencée par *s*.

la référence *this* n'a pas de sens dans une méthode de classe (1/2)

Il n'y a pas d'objet courant

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;
    private static int nbPointsCrees = 0 ;
    ...
    

0

nbPointCrees

    public static void affiche()
    {
        System.out.println(this.abscisse) ;

    }
} // fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        PointPlan.affiche( ) ;
    }
}
```

rejet du compilateur :

La méthode s'appliquant à une classe.

La référence **this** (contenant l'adresse de l'objet courant) n'a pas de sens : il n'y a pas d'objet courant.

la référence *this* n'a pas de sens dans une méthode de classe (2/2)

Il n'y a pas d'objet courant !

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    private static int nbPointsCreés = 0 ;

    ...

```

0

nbPointCreés

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        float x = PointPlan.retourneAbscisse( ) ;
    }
}
```

```
public static float retourneAbscisse()
{
    return this.getAbcisse( ) ;

}
```

```
} //fin classe PointPlan
```

rejet du compilateur

main est une méthode de classe

java TestPointPlan ↵

revient implicitement à faire l'appel *TestPointPlan.main*

```
public class PointPlan
{
    private float abscisse ;
    private float ordonnee ;

    ...
}
```

```
} // fin classe PointPlan
```

```
public class TestPointPlan
{
    public static void main(String[ ] args)
    {
        PointPlan p = new PointPlan( ) ;
        p.init( ) ;
    }
}
```

Une méthode de classe peut créer et utiliser des objets locaux, mais il n'y a pas d'objet courant car la méthode s'applique à la classe.