

Programmation objet



Différents types de programmation

- Programmation procédurale
 - Les instructions se déroulent les unes à la suite des autres
- Programmation objet
 - Des objets sont créés et manipulés pour réaliser les traitements
- Programmation événementielle
 - L'exécution dépend des actions de l'utilisateur

Classe et objet

Programmation objet

- S'appuie sur l'existence de classes d'objets
- Les classes permettent de définir la structure des objets que l'application pourra gérer
- Un objet possède les propriétés de sa classe

Classe / Objet



Intérêts de la programmation objet

- Organisation structurée de l'application
- Réutilisabilité des programmes
- Encapsulation des informations

Programmation objet

Création d'une classe

- Création de la structure des objets d'une application
- Ne s'exécute pas

Utilisation d'une classe

- Création d'un objet d'une classe existante
- Appel de méthodes sur l'objet pour exécuter des opérations

Contenu d'une classe

- Une classe définit la structure qui pourra être utilisée pour créer des objets
 - Constructeur
 - Attributs
 - Méthode

Constructeur

- Une classe contient au minimum un constructeur
- Le constructeur est une fonction ayant le même nom que la classe et permettant de créer un objet de la classe
- Le constructeur initialise les propriétés définies dans la classe lors de la création d'un objet

Attributs

- Les attributs ou propriétés de la classe correspondent à des variables permettant de caractériser la classe
- Les variables / attributs ont une valeur spécifique pour chaque objet

Exemple d'attributs



Landship



Circuit Special



Méthodes

- Les méthodes sont des fonctions ou procédures indiquant les actions pouvant être réalisées sur les objets de la classe

Exemple de méthodes



- Accélérer
- Freiner
- Déraper
- Tourner à droite
- Tourner à gauche
- Sauter
- Utiliser une option
- ...

Le framework java

Les framework

- Existent dans la majorité des langages
- Ensemble de classes outils assurant diverses fonctionnalités
 - Utilitaires (gestion de date, de liste...)
 - Graphique
 - Tests (JUnit)
 - Gestion de la persistance
 - ...

LA CLASSE STRING

Etude d'une classe

- La classe String permet de gérer une chaîne de caractère
- [Documentation String](#)

Création d'un objet


- Pour créer un objet, il faut
 - définir une variable, son type sera le nom de la classe d'objet à utiliser
 - Initialiser l'objet à l'aide d'un constructeur de la classe

Création d'un objet

- Syntaxe

TypeObjet nomObjet = intialisation grâce à un constructeur;

NomClasse nomObjet = new NomClasse();



Classe : nom commence par une majuscule



Objet : nom commence par une minuscule

Constructeurs de la classe

Constructors

Constructor and Description

`String()`

Initializes a newly created `String` object so that it represents an empty character sequence.

`String(byte[] bytes)`

Constructs a new `String` by decoding the specified array of bytes using the platform's default charset.

`String(byte[] bytes, Charset charset)`

Constructs a new `String` by decoding the specified array of bytes using the specified `charset`.

`String(byte[] ascii, int hibyte)`

Deprecated.

This method does not properly convert bytes into characters. As of JDK 1.1, the preferred way to do this is via the `String` constructors that take a `Charset` argument.

`String(byte[] bytes, int offset, int length)`

Constructs a new `String` by decoding the specified subarray of bytes using the platform's default charset.

`String(byte[] bytes, int offset, int length, Charset charset)`

Constructs a new `String` by decoding the specified subarray of bytes using the specified `charset`.

`String(byte[] ascii, int hibyte, int offset, int count)`

Deprecated.

This method does not properly convert bytes into characters. As of JDK 1.1, the preferred way to do this is via the `String` constructors that take a `Charset` argument.

`String(byte[] bytes, int offset, int length, String charsetName)`

Constructs a new `String` by decoding the specified subarray of bytes using the specified charset.

`String(byte[] bytes, String charsetName)`

Constructs a new `String` by decoding the specified array of bytes using the specified `charset`.

`String(char[] value)`

Allocates a new `String` so that it represents the sequence of characters currently contained in the character array argument.

Exercice

- Créer un objet de la classe String à partir de la chaîne de caractères « test de classe »
- `String st = new String("test de classe");`

Appel d'une méthode

- Une fois l'objet créé, on peut appeler les méthodes de la classe pour réaliser des actions sur l'objet
- Syntaxe :
 - `nomObjet.nomMethode(...);`
 - Vérifier la valeur de retour de la méthode

Exercice 1

- Afficher la position de la sous-chaine « de » ou un message d'erreur si elle n'est pas trouvée
- Afficher le contenu de la chaine en majuscule
- Afficher le nombre de caractères de la chaine

PROJET GEOMETRIE

Références d'objet

Projet géométrie

Carre	
- longueur	: double
+ Carre(double l) {}	
+ perimetre ()	: double
+ aire ()	: double
+ affiche ()	: void
+ modifLongueur(double l) {}	: void

Rectangle	
- longueur	: double
- largeur	: double
+ Rectangle(double l, double larg) {}	
+ perimetre ()	: double
+ aire ()	: double
+ affiche ()	: void
+ modifRectangle(double l, double la) {}	: void

Classe Carre

```
public class Carre {  
    private double longueur;  
    //constructeur vide  
    public Carre()  
    {  
        longueur=0;  
    }  
    // constructeur qui initialise la longueur d'un côté du carré  
    public Carre(double l)  
    {  
        longueur = l;  
    }  
    //méthode qui retourne le périmètre du carré  
    public double perimetre()  
    {  
        return longueur*4;  
    }  
}
```

Classe Rectangle

```
public class Rectangle {  
    private double longueur;  
    private double largeur;  
    //constructeur vide
```

```
public Rectangle()
```

```
{  
    longueur=0;  
    largeur=0;  
}
```

```
// constructeur qui initialise la longueur d'un côté du rectangle
```

```
public Rectangle(double l, double larg)
```

```
{  
    longueur = l;  
    largeur = larg;  
}
```

Exercice 2

- Créer un objet de la classe Carré de 10 cm de côté
- Créer un objet de la classe Rectangle de 6 cm de largeur et 8 cm de longueur
- Afficher les informations de vos deux objets pour vérifier leur création.
- Afficher le périmètre du carré et l'aire du rectangle

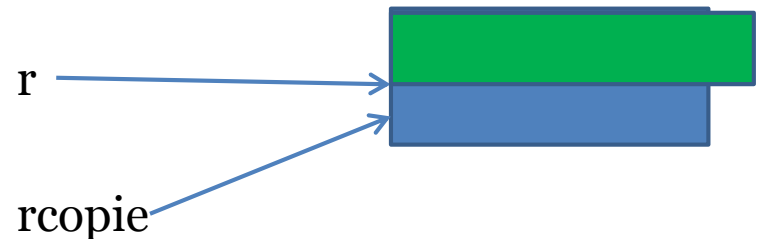
Exercice 3

- Créer un objet Carré à partir de l'objet précédent.
- Modifier la longueur d'un côté et afficher les informations des deux objets.
- Que remarquez-vous ?

Référence d'objets

- L'affectation d'un objet consiste à créer une nouvelle référence sur le même objet
- Pour créer une copie d'un objet, on utilisera la méthode clone (à définir)

- `Rectangle r = new Rectangle(8,6);`
- `Rectangle rcopie = r;`
- `rcopie.modifRectangle(10,3);`



Référence null

- null est un mot clé Java
- Peut être utilisé pour créer un objet ou pour tester sa validité
- Un objet qui n'a pas été initialisé par un constructeur est null
- L'appel d'une méthode sur un objet null crée une erreur

Exercice 4

- Créer un objet Rectangle avec la ligne suivante
 - `Rectangle r = null;`
- Appeler la méthode affiche sur l'objet
- Corriger le programme en ajoutant un test pour lancer la méthode uniquement si l'objet n'est pas null ou afficher un message d'erreur

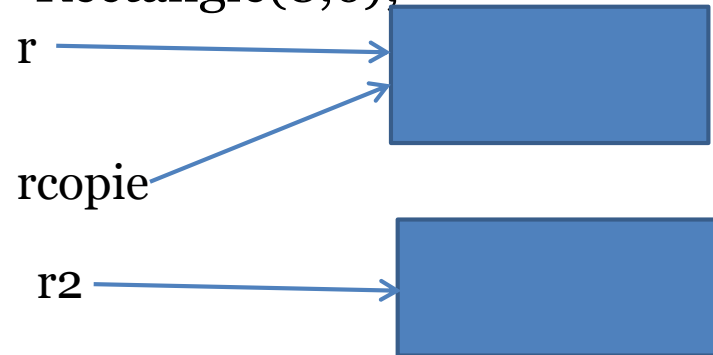
Exercice 5

- Créer un rectangle r1 de 12*4 cm
- Créer un rectangle r2 de 12*4 cm
- Tester l'égalité des objets avec l'opérateur ==
- Que remarquez-vous ?

Egalité des objets

- L'opérateur == entre des objets teste l'égalité des références d'objet
- Pour tester si deux objets sont égaux, on utilisera la méthode equals (à définir)

- Rectangle r = new Rectangle(8,6);
- Rectangle rcopie = r;
- Rectangle r2 = new Rectangle(8,6);



r == rcopie **true**

r == r2 **false**

Encapsulation d'objets

Carre	
- longueur	: double
+	
Carre(double l)	()
perimetre()	: double
aire()	: double
affiche()	: void
modifLongueur(double l)	: void

Rectangle	
- longueur	: double
- largeur	: double
+	
Rectangle(double l, double larg)	()
perimetre()	: double
aire()	: double
affiche()	: void
modifRectangle(double l, double la)	: void

Dessin	
- c	: Carre
- r	: Rectangle
- color	: String
+	
Dessin(Carre c1, Rectangle r1, String col)	()
aire()	: double
affiche()	: void

Exercice 6

- Créer un objet Dessin contenant une couleur, un carré et un rectangle de votre choix
- Afficher l'aire du dessin
- Afficher le contenu du dessin

Exercice 7

- A partir de l'objet Dessin,
 - Afficher le périmètre du carré
 - Afficher l'aire du rectangle
 - Vérifier la valeur de l'aire du dessin (aire du carré + aire du rectangle)