POO

en Java 8

Objet : concepts - vocabulaire

Classe et Objet

Une classe est un "patron", un "gabarit" qui permet de construire des objets.

- classe des êtres vivants
- classe des véhicules

Un objet a une existence : il naît grâce à new; il est possible de changer ses caractéristiques puis il meurt.

- EtreVivant martin = new EtreVivant();
- martin.changeLook(Cheveux blonds, Tenue négligée);

Classe

La **classe** factorise les caractéristiques communes aux objets.
Les classes ont un double rôle :

- décrire et **classer** de façon abstraite des objets
- servir de « moule » à objets (mécanisme **d'instanciation**).

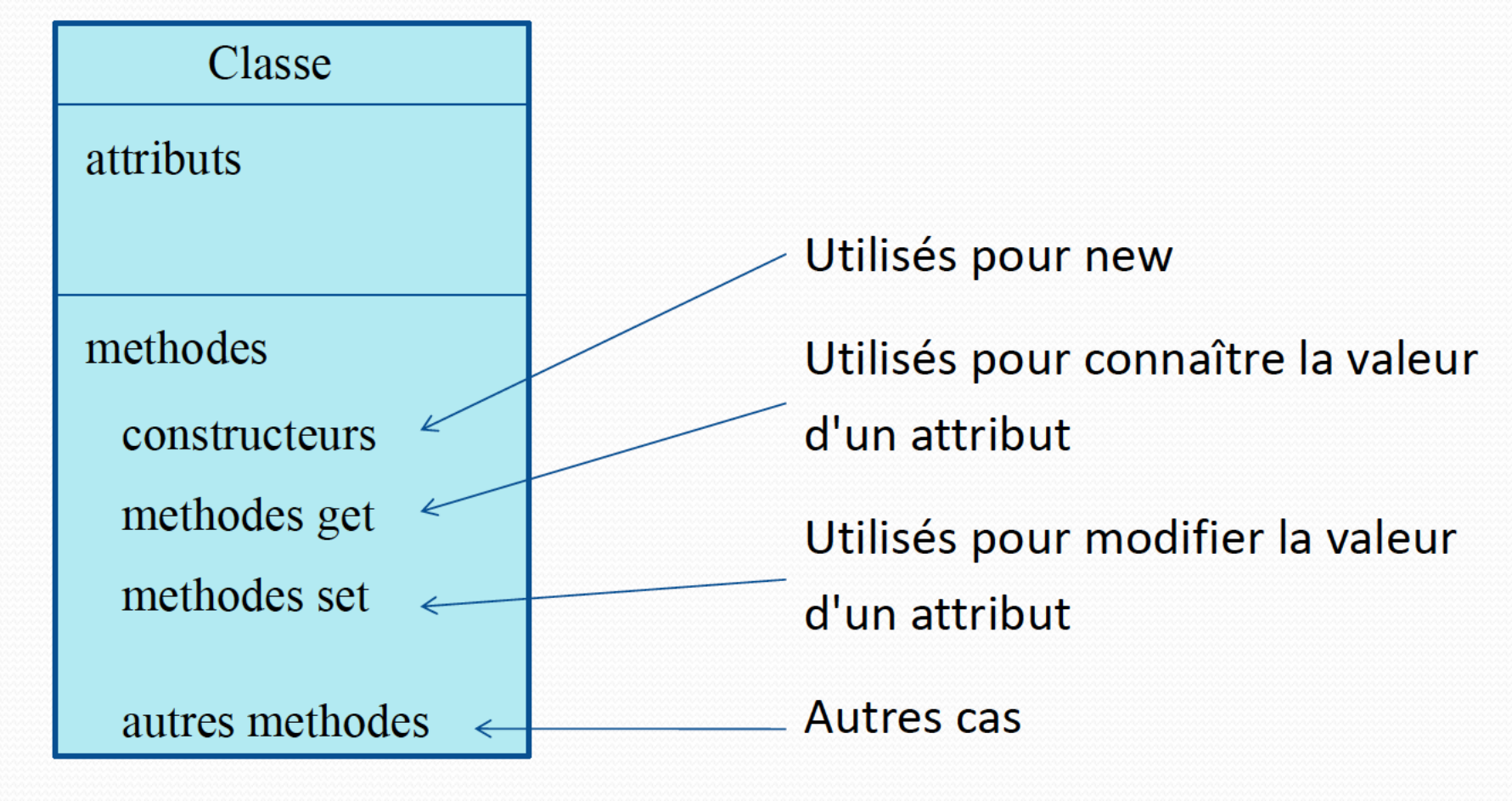
Une classe désigne un **type générique** dont les objets sont **des instances**

Une application est structurée sous forme d'un ensemble de classes qui se partagent les traitements à effectuer

Classe : exemple

```
class Voiture {
    Embrayage embrayage;
    Moteur moteur;
    BoiteVitesse boite;
    public void demarrer() {
        embrayage.enfoncer();
        moteur.faireTourner(.);
        boite.passerVitesse(1);
        embrayage.relacher();
    }
}
```

Classe : structure



Les **membres** ou **champs** d'une classe peuvent être les **attributs** ou les **méthodes**

Classe : structure

```
public class User {
public String nom;
public String prenom;
…..
public String calculPw(){
…..
}
}
```

La classe a :

- Une visibilité : public
- Un nom : il est identique au nom du source qui contient le code lié à la classe.

Elle commence par un symbole {.

Elle se termine par un symbole }.

La classe a des attributs.

Objet ou instance de classe

Une **instance** est un représentant particulier d'une classe.
Les attributs prennent des valeurs distinctes dans chacune des instances.

L'**instanciation** est le processus par lequel on crée de nouveaux objets à partir du modèle défini par une classe. C'est utiliser une classe qui sert de modèle pour créer un objet.

```
Voiture voiture = new Voiture();
```

Objet ou instance de classe

L’objet créé par l’instanciation d’une classe contient :

- des données qui définissent son **état** (**attributs** ou **propriétés**)
- des fonctions (**méthodes**).

- L’objet est alloué dynamiquement en mémoire grâce à l’opérateur **new.**

- La variable de l’objet déclaré dans un programme ne contient pas l’objet, mais la localisation en mémoire de l’objet : **Référence d’objet.**

Etat

C’est la valeur des **attributs** considérés dans leur ensemble, qui reflète l'**état** dans lequel se trouve l'objet à un instant donné.

```
class Personne {
        String nom;
        String prenom;
        int age;
}
```

Personne unePersonne = new Personne("Martin", "Pierre", 35);

Méthode

Une méthode est un « petit programme » qui exécute une opération.
Les méthodes sont décrites au niveau des classes.
Chacune à un nom (**sélecteur**) qui permet de l'activer.
L'ensemble des méthodes permet d'assurer le **comportement** des objets.

```
class Voiture {
        public void demarrer() {
        embrayage.enfoncer();
        moteur.faireTourner(.);
        .....
        }
}
```

Méthode : constructeur

C’est une méthode particulière que tout objet possède et qui sert à sa construction et à son initialisation.

-Il doit porter obligatoirement le nom de sa classe.

-Il peut posséder des paramètres d’entrée, mais ne peut retourner aucun type, même pas void.

```
class Voiture {
    public Voiture () {
        nbRoues = 4;
        ……
    }
}
```

Constructeur et instance

Quand on veut créer un nouvel objet à partir d'une classe, on l'instancie c'est à dire qu'on utilise une méthode "constructeur "

```
class Voiture
{
      public Voiture ()  {
        nbRoues = 4;
      }
}
```

```
public class TestVoiture
{
      public static void main(String[] args)
      {
            Voiture v1 = new Voiture();
      }
}
```

Le mot **new fait appel à la méthode constructeur de la classe nommée.**
Si aucun constructeur n'est défini, il existe toujours le constructeur par défaut qui est vide.

Constructeur et instance

Un constructeur, comme n'importe quelle méthode peut être **surchargé,** ce qui signifie qu'il peut être déclaré plusieurs fois avec des **signatures différentes.**

La signature est dans ce cas le nombre, le type, et l'ordre de passage des paramètres à la méthode .

Voir Demo : Personne.java

Constructeur et instance

Un objet est référencé lorsqu'il est utilisé dans un programme pour accéder à l'un de ses attributs ou à l'une de ses méthodes.
Dés qu'un objet n'est plus référencé, sa place en mémoire est récupérée par le « garbage collector » ou ramasse-miettes.
Tout objet possède une référence sur lui même : le mot clé **this**

```
public class Personne {
	String strNom;
	public Personne (String strNom) {
		this.strNom= strNom;
	}
}
```

Attributs d'instance – attributs de classe

Les variables d’instance sont allouées dynamiquement en mémoire lors de la création de l’objet.
Les variables de classe sont communes pour l'ensemble des instances de la classe. (**static**)

Voir démo : PersonneStatic.java

Message

La partie *dynamique* des objets est assurée par la notion **d'envoi de messages**.
Envoyer un message à un objet, c'est lui dire ce qu'il doit faire. On utilise les **méthodes** pour mettre en place ce mécanisme.

```
class Voiture {
    // Format : visibilité récepteur sélecteur (Paramètres)
    public void demarrer() {
        embrayage.enfoncer();
        boite.passerVitesse(1); // Un paramètre
        ……
    }
}
```

Méthode de classe :

Une méthode peut être définie au niveau de la classe.
Il n'est pas nécessaire d'instancier une classe pour utiliser une méthode de classe : on utilise directement la classe.
Une méthode de classe ne peut travailler que sur des variables de classes.

Voir démo : TestMethodStatic.java

Fin …

… exercice
N° 024