IUT2 Université Grenoble Alpes
Département STID

Programmation Objet

R3.VCOD.10

Martin LENTSCHAT

martin.lentschat@univ-grenoble-alpes.fr

IUT2 Université Grenoble Alpes
Département STID

La programmation Orientée Objet

Plan du jour :

- Le paradigme de la POO
- Les concepts fondamentaux de la POO
- Exemples sur machines

IUT2 Université Grenoble Alpes
Département STID

Le paradigme de la POO

- Qu'est-ce qu'un paradigme de programmation ?
  - Permet de décrire et classer les langages de programmation en fonction de plusieurs critères (fonctionnalités, exécution, données …)
- Pourquoi différents paradigmes ?
  - Il n'existe pas, en programmation, de langage magique qui puisse tout faire.
  - On a donc une diversité d'outils pour résoudre différents problèmes

IUT2 Université Grenoble Alpes
Département STID

La programmation Orientée Objet

- Permet au développeur de créer des briques logicielles, appelées *objets*.
  - Simula I (1972), C++ (1983)
- Un objet représente un concept, et possède une structure interne

Avantages de la POO :

- Permet de représenter des concepts du monde réel
- Facilite la conception et la maintenance de gros logiciels.

IUT2 Université Grenoble Alpes
Département STID

La programmation Orientée Objet

- Au fil du temps, les logiciels ont gagnés en complexité.
- La POO permet:
  - de faciliter la maintenance
  - de faciliter le développement collaboratif
  - de faciliter l'adaptation aux changements
  - de faciliter la réutilisabilité des briques logicielles.

IUT2 Université Grenoble Alpes
Département STID

La programmation Orientée Objet

Avantages de la POO :

- **Modularité**: Les objets sont faciles à manipuler.
- **Abstraction**: Les objets représentent des concepts du monde réel.
- **Réutilisabilité**: Les objets sont facilement réutilisables et extensibles.
- **Encapsulation**: Les objets peuvent gérer le niveau de visibilité de leurs composantes.

IUT2 Université Grenoble Alpes
Département STID

Les principales notions de la POO

- **Classe (avec ses Attributs et Méthodes) et Objet/Instance**
- **Envoi de messages**
- Associations entre classes
- Héritage
- Polymorphisme
- Encapsulation
- Agrégation
- Composition

IUT2 Université Grenoble Alpes
Département STID

Les concepts fondamentaux de la POO

Classe:

- Une classe est un modèle à partir duquel on fabrique des objets.
- On peut voir une classe comme un moule, ou une maquette.
- La classe est un modèle de la structure (attributs et méthodes) des objets associés à cette classe.

Une **classe** représente un ensemble de "choses" du monde réel (on appellera ces choses des "objets")

Ex : *Une classe Personne, une classe Etudiant, une classe Voiture, une classe PointGeometrique, …*

IUT2 Université Grenoble Alpes
Département STID

Les concepts fondamentaux de la POO

Objet:

- Un objet représente un **concept**
- Il possède trois composantes:
  - Son identité: généralement une **référence** à l'objet, ou son adresse mémoire.
  - Son état: une collection de variable appelées **attributs** contenant des informations sur
  - l'objet.
  - Son comportement: une collection de fonctions appelées **méthodes**.

IUT2 Université Grenoble Alpes
Département STID

Les concepts fondamentaux de la POO

Instance:

- Une instance d'une classe est un objet qui "appartient" à cette classe.
- Le mot vient de l'anglais instance.

IUT2 Université Grenoble Alpes
Département STID

Les concepts fondamentaux de la POO

Exemple :

Prenons le concept de vélo.

- On peut parler de la classe Vélo.
- Chaque vélo garé sur le campus est un objet, une instance de Vélo.
- Les vélos possèdent des points communs, sont constitués des mêmes composants.

  Ils ont un type, une marque, … sont composés d'un cadre, d'une paire de roues, ...

  On peut les garer, les utiliser, les réparer …

IUT2 Université Grenoble Alpes
Département STID

Les concepts fondamentaux de la POO

Des caractéristiques appelées **Attributs**

- *Une personne a un nom, un prénom, un âge, une adresse….*
- *Une voiture a une couleur, un modèle, une puissance fiscale…*
- *Un point géométrique a une abscisse x, une ordonnée y, une couleur…*

- Un comportement représenté par un ensemble de **Méthodes**
  - *Une personne change d'adresse, d'âge, se marie, …*
  - *Une voiture roule, change de propriétaire….*
  - *Un point géométrique change de couleur, bouge…*

Exemple de classe, avec attributs et méthodes

Nom de la classe

3 attributs

6 méthodes

1 méthode particulière, le **constructeur**

```python
class Personne:
    def __init__(self, nom : str, prenom : str, adresse : str):
        self.firstname = prenom
        self.lastname = nom
        self.address = adresse

    def get_firstname(self):
        return self.firstname

    def get_lastname(self):
        return self.lastname

    def get_address(self):
        return self.address

    def change_firstname(self, nouv_prenom : str):
        self.firstname = nouv_prenom

    def change_lastname(self, nouv_nom : str):
        self.lastname = nouv_nom

    def change_address(self, nouv_adresse : str):
        self.address = nouv_adresse
```

personne.py

Rq : En Python, le premier argument de toutes les méthodes d'une classe est *self*, qui est une référence à l'objet lui-même (*this* dans d'autres langages).

Le constructeur d'une classe

- Toute classe possède un **constructeur** : en Python, le constructeur s'appelle *__init__*
- Le constructeur est une méthode particulière qui sert à :
  - créer un objet instance de la classe
  - initialiser les attributs de l'objet (attributs d'instance)

3 **attributs d'instance** de la classe Personne

```python
class Personne:
    def __init__(self, nom : str, prenom : str, adresse : str):
        self.firstname = prenom
        self.lastname = nom
        self.address = adresse
```

personne.py

Appel au **constructeur** pour créer 2 objets p1 et p2 instance de la classe Personne

p1 et p2 s'appellent les **identificateurs** (**oid**) des objets créés

```python
import personne

p1 = personne.Personne("Durand", "Paul", "Grenoble")
p2 = personne.Personne("Dupont", "Jeannine", "SMH")
```

main.py

Objet / Instance d'une classe

- Un **objet** est **instance** d'une classe.
  - Il représente une vraie "chose" du monde réel
  - Il est représenté par son **identificateur** (oid : object identifier)
  - Il admet une valeur pour chaque attribut de sa classe conforme au type.

Objet / Instance d'une classe

- Les valeurs des attributs d'instance définissent l'**état** de l'objet. Un objet peut changer d'état suite à l'appel d'une (ou plusieurs méthodes) sur l'objet

p2 change d'état suite à l'appel de méthodes

```python
import personne

p1 = personne.Personne("Durand", "Paul", "Grenoble")
p2 = personne.Personne("Dupont", "Jeannine", "SMH")

print(p2.firstname, p2.lastname, p2.get_address())

#p2 se marie avec p1, elle prend le nom de p1 et démanage pour habiter avec p1
p2.change_lastname(p1.get_lastname())
p2.change_address(p1.get_address())

print("Après le mariage : ")
print(p2.firstname, p2.lastname, p2.get_address())
```

main.py

```
Jeannine Dupont SMH
Après le mariage :
Jeannine Durand Grenoble
```

À l'exécution

Envoi de message : vocabulaire

- Pour demander à un objet d'exécuter une méthode, il faut lui envoyer un **message**.

Dans l'exemple précédent : p2.get_address()

receveur	**sélecteur**	**paramètres**
p2	get_address	()

receveur : identificateur de l'objet à qui le message est envoyé

sélecteur : nom de la méthode à exécuter

paramètres : le ou les valeurs de paramètres effectifs (ou aucun)

IUT2 Université Grenoble Alpes
Département STID

Exercice

- Définissez une classe *Etudiant* avec :
  - Son constructeur, permettant d'initialiser les attributs *firstname*, *lastname* et *age*.
  - Les méthodes *get* permettant d'accéder aux attributs.
  - Les méthodes *set* permettant de modifier la valeur des attributs.