Chapitre 1
Les concepts fondamentaux de la programmation à objets

Jean Privat

Université du Québec à Montréal

INF7845 — Principes avancés des langages à objets
Hiver 2017

Objectif du cours

Étudier la programmation par objets

- Concepts fondamentaux
- Concepts avancés

Présenter l'application de ces concepts dans des langages à objets

- Existants
- Qui ont existé
- Ou existerons

Objectif du cours

Trouver les réponses aux questions

- Quelle est la différence entre un "langage à objets" et un "langage pas à objets" ?
- Pourquoi le paradigme objet est-il actuellement dominant ?
- Quelles sont les problématiques auxquelles font face les langages à objets ?

De quoi on va parler ?

Domaines abordés

- Spécification des langages
- Modélisation
- Programmation
- Implémentation (un peu)

De quoi on va parler ?

Problématiques étudiées

- Héritage, héritage multiple
- Typage statique, généricité
- Envoi de message, appel de méthode
- Méta-programmation, réflexivité
- Typage dynamique
- Modules et raffinement de classes

Prérequis

Connaissance d'un ou plusieurs langage à objets

- Typage statique C++, Java, C#
- Typage dynamique Python, Ruby, JavaScript

Bonne maîtrise de la programmation

- Algorithmique
- Modélisation
- Patrons de conceptions

Concepts de base

- Objet
- Propriété
- Envoi de message
- Classe
- Spécialisation et héritage

Objet

Concept central de l'approche objet

- Capsule
- Données et procédures
- Identité

Questions

- A-t-on besoin d'objets pour faire de la programmation par objets ?
- Tout est-il objet dans les langages à objets ?

Propriété

Méthode

- Fonction
- Procédure
- Routine
- Opération

Attribut

- Champ
- Variable d'instance
- Slot

D'autres trucs aussi

- Constructeurs, invariants, etc.

Envoi de message

Exécution d'une propriété d'un objet

```
Animal a;
a = new Vache();
a.cri() // affiche "meuh !"
```

- 'a' est le receveur, 'cri' le message

Polymorphisme

- Le receveur « décide » du comportement
- On parle aussi de liaison tardive
- Permet de discriminer les langages à objets des autres

Classe

Concept central des langages à classes

- Capsule
- Regroupe des objets similaires (ses instances)
- Factorise leur propriétés

Questions

- A-t-on besoin de classes pour faire de la programmation par objets ?
- Les classes ont-elles d'autres rôles ?
  Modélisation
  Espace de noms
  Unité de compilation

Héritage et Spécialisation

Objectif

- Factoriser les propriétés des classes

Structure des classes

- Hiérarchie de classes : spécialisation-généralisation
- On parle de super-classe et de sous-classe

Règles

- Une classe hérite les propriétés des classes qu'elle spécialise.
- Un classe peut étendre ses super-classes et redéfinir les propriétés héritées.

Raison du succès de l'approche à objets

Adéquation avec le mode de pensé humain

- Classification et catégorisation
- Logique Aristotélicienne
- Exemple : classification des espèces naturelles

Adéquation avec les exigences du génie logiciel

- Expressivité, lisibilité, fiabilité, évolutivité, réutilisabilité, etc.

Catégorisation des langages à objets

- Exécution : compilé, interprété
- Typage : statique, dynamique
- Héritage : simple, multiple
- Méta-niveau : aucun, introspectif, réflexif
- Sélection : simple, multiple
- Paradigme : multiple, pur objet