Université Nice SOPHIA ANTIPOLIS

Programmation Objet & Patrons de conception

F. Mallet

miage.m1@gmail.com

http://deptinfo.unice.fr/~fmallet/

Organisation du cours

- Chargés de TD
  - Michel Buffa
  - Frédéric Mallet
- Volume horaire :
  - 9*2h de cours
  - 11*3h de TD machine
  - Le mardi toute la journée
- Evaluation
  - Un projet et une soutenance orale individuelle
  - Un examen écrit

Bibliographie et Webographie

- *« Informatique Industrielle et Java »*
  - F. Mallet et F. Boéri, Dunod 2003

  http://www.iutenligne.net/ressources/informatique_indus/Mallet/JavaII/

- *« Conception objet en Java avec BlueJ: Une approche interactive »*, 4ème édition
  - D. Barnes et M. Kolling, Pearson Education 2009

- *« Penser en Java »*, 2ème Edition
  - B. Eckel, http://penserenjava.free.fr/

- Documentation *en ligne* (API et tutoriels)
  - Sun, http://java.sun.com/javase/reference/api.jsp

Plan du cours

- Rappels sur Java
  - Fichiers, tableaux
  - Réutilisation: héritage **et** composition
- Patrons de conception:
  - Décorateur, Écouteur, Visiteur, Observateur, MVC...
- Réflexivité et chargement dynamique
  - RTTI (Real-Time Type Information)
- Annotations
- Création de plugins Eclipse

La programmation impérative

- Programmation
  - concevoir (**algorithme**) et réaliser (**programme**) pour une exécution **automatique**
- Algorithme
  - Impératif: succession d'ordres
    - Pour calculer le maximum, je parcours les éléments un à un et je les compare à l'élément supposé maximum
  - Déclaratif
    - $max(T) = \{ m \mid \forall t \in T, t \leq m \}$

Example de la factorielle

- Impératif
  - Itératif

```
long factorielle(int n) {
  long res = 1;
  for(int i = 2; i<n; i++) {
    res = res * i;
  }
  return res;
}
```

  - Récursif

```
long factorielle(int n) {
  if(n==0) return 1;
  return n*factorielle(n-1);
}
```

- Déclaratif
  - Fonctionnel / Applicatif

```
(factorielle (n)
  ((zero? n) 1)
  (* n (factorielle (- n
  1)))
```

  - Logique

```
factoriel(0,1):-
  !.
factoriel(N,T):-
  N1 is N-1,
  factoriel(N1,T1),
  T is T1*N.
```

La programmation orientée-objet

- La POO guide la conception par
  - Un ensemble de concepts (abstraction, modularité, encapsulation, polymorphisme)
  - Des langages et des outils qui supportent ces concepts

- Ses forces (supposées)
  - Du code (plus) facile à maintenir
  - Reflète plus finement les objets du monde réel
  - Plus stable : un changement s'applique à un sous-système facile à identifié et **isolé** du reste du système

Historique

- Simula 61-67 : Dahl et Nygaard *http://www.simula.no*
- Smalltalk 72 : A. Kay *http://www.smalltalk.org*
- C++ 81-86 : Stroustrup
- Eiffel 85 : B. Meyer *http://www.eiffel.com*
- Java 94-95 : J. Gosling/Sun *http://java.sun.com*
- Ruby 95 : Y. "Matz" Matsumoto *http://www.ruby-lang.org*
- C# 2000 : Microsoft *http://www.learn-c-sharp.com*
- Scala 2001 : Martin Odersky *http://www.scala-lang.org*

- Turing Award 2001 à Dahl et Nygaard
  - 'for their role in the invention of OOP, the most widely used programming model today'.