

Programmation Orienté Objets en JAVA

Daniele Varacca
Département d'Informatique
Université Paris Est

2014

Un autre langage?!?

J'ai déjà appris a programmer en C,
pourquoi apprendre un nouveau langage?

Tout peut être fait en C
D'ailleurs tout peut être fait en langage machine

Un autre paradigme!

On n'apprendra pas juste un nouveau langage

- ▶ une façon différente de concevoir
- ▶ une façon différente d'organiser le travail

Développement

Phases de la création d'un programme:

- ▶ Prendre conscience d'un problème
- ▶ Décomposer en problèmes plus simples
- ▶ Analyser ces problèmes
- ▶ Concevoir une solution
- ▶ Réaliser la solution sous forme de programme
- ▶ Évaluer le programme

Abstraction

```
00000000 01 00 FF FF 00 00 00 00 00 00 00 00 00 40 00 CC 80 .....@
00000010 0C 00 00 00 00 00 26 01 8F 00 00 00 00 00 00 53 00 .....&...S
00000020 65 00 6C 00 65 00 63 00 74 00 20 00 52 00 75 00 elect.Ru
00000030 6C 00 65 00 00 00 08 00 00 00 01 4D 00 53 00 lect...HS
00000040 20 00 53 00 68 00 65 00 6C 00 6C 00 00 00 44 00 ...Sh...l...D
00000050 6C 00 67 00 00 00 00 00 00 00 00 00 02 00 00 lg
00000060 03 01 A1 50 53 00 3A 00 C3 00 36 00 32 25 00 00 ...PS...6.2%
00000070 FF FF 83 00 00 00 00 00 00 00 00 00 00 00 00 00 ...PVA...J&
00000080 03 00 01 50 0E 00 56 00 41 00 0A 00 4A 26 00 00 ...PVA...J&
00000090 FF FF 80 00 26 00 41 00 70 00 70 00 6C 00 79 00 ...&Apply
000000a0 20 00 74 00 6F 00 20 00 61 00 6C 00 6C 00 00 00 ...to...all...
000000b0 00 00 00 00 00 00 00 00 00 00 00 00 01 00 01 50 ...P
000000c0 7E 00 7D 00 32 00 0E 00 01 00 00 FF FF 80 00 ...}2...
000000d0 4F 00 4B 00 00 00 00 00 00 00 00 00 00 00 00 00 O.K.
000000e0 00 00 01 50 B4 00 7D 00 32 00 0E 00 02 00 00 00 ...P...}2...
000000f0 FF FF 80 00 43 00 61 00 6E 00 63 00 65 00 6C 00 ...Cancel
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 50 ...P
00000110 EA 00 7D 00 32 00 0E 00 09 00 00 FF FF 80 00 ...}2...
00000120 26 00 48 00 65 00 6C 00 70 00 00 00 00 00 00 00 &Help...
00000130 00 00 00 00 00 00 00 00 80 08 81 50 0E 00 3A 00 ...P
00000140 3E 00 0E 00 2F 25 00 00 FF FF 81 00 00 00 00 00 ...%
00000150 00 00 00 00 00 00 00 00 00 00 02 50 0E 00 30 00 ...P.O
00000160 1E 00 08 00 EE 25 00 00 FF FF 82 00 46 00 69 00 ...%...Fi
00000170 6C 00 65 00 20 00 54 00 79 00 70 00 65 00 00 00 le.Type...
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 02 50 ...P
00000190 54 00 30 00 2C 00 08 00 EF 25 00 00 FF FF 82 00 T.O...%
000001a0 50 00 61 00 72 00 73 00 69 00 6E 00 67 00 20 00 Parsing...
000001b0 52 00 75 00 6C 00 65 00 73 00 00 00 00 00 00 00 Rules...
000001c0 00 00 00 00 00 00 00 00 07 00 00 50 06 00 07 00 ...P
000001d0 1A 01 71 00 ED 25 00 00 FF FF 80 00 00 00 00 00 ...q...
000001e0 00 00 00 00 00 00 00 00 00 00 02 50 0E 00 11 00 ...P
000001f0 3E 00 08 00 EC 25 00 00 FF FF 82 00 53 00 65 00 ...%...Se
00000200 6C 00 65 00 63 00 74 00 20 00 52 00 75 00 6C 00 lect.Ru
00000210 65 00 20 00 46 00 6F 00 72 00 20 00 46 00 69 00 e.For...F
00000220 6C 00 65 00 00 00 00 00 00 00 00 00 00 00 00 le
00000230 80 08 81 50 0E 00 1B 00 08 01 0E 00 EB 25 00 00 ...P...%
00000240 FF FF 81 00 00 00 00 00 00 00 00 00 00 00 00 00 ...P.a.7...k&
00000250 00 02 50 19 00 61 00 37 00 08 00 6B 26 00 00 ...
00000260 FF FF 82 00 00 00 00 00 |
```

langage machine/assembleur \Rightarrow C

Abstraction

```
1  #include<stdio.h>
2  #include<conio.h>
3  void main()
4  {
5      int num,i,sum=0;
6      printf("Enter a number\n");
7      scanf("%d",&num);
8      for(i=0; i<=num; i++)
9      {
10         if((i%2)==0)
11             sum=sum+i;
12     }
13     printf("Sum of %d even number is %d",num,sum);
14     getch();
15 }
16
```

langage machine/assembleur \Rightarrow C

Abstraction

langage machine/assembleur \implies C

- ▶ un saut d'abstraction
- ▶ on modélise de façon plus abstraite le fonctionnement de l'ordinateur

De l'outil au problème

C est un langage qui modélise l'outil qu'on a pour résoudre les problèmes.

Le programmeur doit réfléchir comment utiliser cet outil

On pourrait directement modéliser le problème!

On fera un autre saut d'abstraction en utilisant un langage qui modélise le problème

Autres questions

D'autres questions se présentent également

- ▶ Modularité
- ▶ Portabilité
- ▶ Lisibilité
- ▶ Extensibilité
- ▶ Réutilisation

Approche à objets

Plusieurs approches possibles, pourquoi objets?

- ▶ Populaire
- ▶ Efficace
- ▶ Intuitif

Objets

Dans la réalité c'est quoi un objet?



- ▶ Il a une identité
- ▶ Il a un état qui change dans le temps
- ▶ Il peut faire (ou subir) des actions
- ▶ Parmi ses actions, un objet peut demander à d'autres objets de faire (ou subir) des actions

Objets



- ▶ Identité: Comment un objet dont l'état change reste le même objet
- ▶ En quoi consiste l'état d'un objet: en d'autres objets!
- ▶ Les actions modifient l'état des objets

Exemples

Un objet: un bibliothèque.

- ▶ elle contient des objets: des livres, un bibliothécaire, une salle de lecture, etc
- ▶ la salle de lecture contient des utilisateurs
- ▶ les utilisateurs peuvent entrer dans la salle, parler au bibliothécaire, emprunter des livre

Exemples

Un objet: une agence immobilière.

- ▶ elle contient des agents
- ▶ chaque agent a un portfolio de biens
- ▶ les biens ont un prix, les agents peuvent le changer

Exemples

La A86:



- ▶ une voiture qui a un vitesse qui peut accélérer et ralentir et sonner le klaxon
- ▶ une personne qui a de l'argent, qui peut monter sur une voiture, conduire la voiture, descendre de la voiture, payer une somme (s'il l'a)
- ▶ un gendarme qui peut faire payer une personne si la voiture va trop vite

Exemples

Un autre exemple plus informatique:

- ▶ une file d'objets
- ▶ elle permet d'y ajouter en tête, enlever, vérifier si elle est vide

Objets

Le monde consiste en un tas d'objets qui interagissent l'un l'autre.

Un programme orienté objets de même

Programmer à objets consiste à écrire du code pour chaque objet

Questions

Questions:

- ▶ Et si un objet *a* demande à un objet *b* de faire une action que *b* ne peut pas faire?
- ▶ Comment peut on créer des centaines d'objets?
- ▶ Quand il y a de nombreux objets, comment peut-on comprendre leur interaction?

Classe

- ▶ Avoir un moyen de savoir ce qu'un objet peut faire
- ▶ Avoir un moyen de créer plusieurs objets
- ▶ Avoir une taxonomie des objet pour simplifier la compréhension et la conception

Cela se fait à l'aide du concept de **classe**

Classe

- ▶ Une classe est une famille d'objets du même "type"
- ▶ Les objets d'une classe savent faire les mêmes actions
- ▶ La liste d'actions que les objet d'une classe savent faire s'appelle la **signature**
- ▶ Si on connaît la signature d'un objet, on ne lui demandera pas de faire ce qui n'est pas dans la signature

Classe

- ▶ Chaque classe permet de générer plusieurs objets.
- ▶ Une classe est comme un "modèle" pour la création automatique d'objets.

Classe

- ▶ En spécifiant les relations entre classes on comprend aussi les relation entre les objets.
- ▶ Les classes permettent de représenter des données, des concepts abstraits, de donner de la structure aux programmes

Objets et classes

Un bon programme à objets

- ▶ une bonne conception des classes et des relations entre elles
- ▶ des objets qui interagissent en respectant la signature

Objets et classes

Programmation objets selon Alan Kay



- ▶ Tout est un objet
- ▶ Un programme et un tas d'objet qui se disent l'un l'autre ce qu'ils doivent faire
- ▶ Tout objet peut inclure d'autres objets
- ▶ Les objets ont un type (classe)
- ▶ Les objets d'une même classe savent faire les mêmes choses

Langages à objets

Plusieurs langages de programmation

- ▶ Smalltalk
- ▶ Simula
- ▶ C + +
- ▶ Java
- ▶ Python
- ▶ Scala
- ▶ Javascript
- ▶ C#
- ▶ Objective C
- ▶ ...

Langages à objets

Chaque langage a ses caractéristiques, mérites, limites. De plus, pas tous les langages à objets utilisent une notion de classe.

Java

Dans ce cours on choisi Java. Au fur et à mesure on verra les caractéristiques les mérites les limites de Java.
On commencera à modéliser des petits scénarios et à voir comment cela se réalise en Java.

Déroulement du cours

- ▶ 12 semaines (avec pause pour la Toussaint)
- ▶ 18 heures de cours d'amphi (12 séances de 1h30)
- ▶ 21 heures de TD (14 séances de 1h30)
- ▶ 24 heures de TP (8 séances de 3h)
- ▶ Un projet en commun avec
Développement de Programmes
- ▶ Examen 67%, projet 33%

Responsables:

- ▶ Daniele Varacca (CM)
- ▶ Gaétan Hains (TD)
- ▶ Julien Cervelle, Luidnel Maignan (TP)