

Approches Objet de la Programmation

- Numero: 1
- Prof: Didier Verna
- Date: 9 Octobre 2017

Introduction

Mail du prof: didier@lrde.epita.fr

Site du prof

Slides

Il y a aussi des tarballs avec des programmes complet a tester.

On va etudier les **Paradigmes**. * Approches classiques de la programmation Objet. (Ce cours) * Approches de la programmation fonctionnelle. * Approches autres de la programmation Objet (lisp & askell).

La plus part des langages Oriente Objet (C++, Java...) ne sont pas propre en realite d'un point de vue theorique.

Les notions de base de la programmation objet: * Reutilisation du code. * Polymorphisme d'inclusion.
* Chaque objet est construit comme un moule (sa classe).

Sommaire

- Contexte
 - Programmation Imperative
 - Programmation Procedural
- Paradigmes de Programmation
 - Notions de Paradigme
 - Limitations de l'Imperatif / Procedural
- L'approche Orientee-Objet
 - Programmation Imperative
 - Programmation Procedural
 - Evolution

Contexte

Programmation Imperative

Imperatif signifie donner des ordres. La programmation imperative est une suite d'instruction. **Instruction**: Ordre destiné à produire un effet de bord. Effet de bord ou **Side effect** signifie modifier l'environnement.

```
1 Instruction 1
2 Instruction 2
3 ...
4 Instruction N
```

L'ordre des exécutions est important !

Programmation Procedural

Dans les années 50. C'est une extension logique de la programmation imperative. Elle est née d'une frustration de la programmation imperative.

Elle permet de factoriser une suite d'instruction souvent répétées en procédure que l'on peut appeler autant de fois que l'on veut.

```
1 ...
2 fais ceci
3 fais cela
4 ...
5 fais ceci
6 fais cela
7 ...
```

C'est de la merde car: * Manque de lisibilité. * Bug duplique. * Plus dur à maintenir.

Il faut donner des noms aux choses.

```
1 fais tout:
2   fais ceci
3   fais cela
4 ...
5 fais tout
6 ...
7 fais tout
8 ...
```

Origine

Modele *Von Neumann*. L'idée c'est que les donnees et les instruction sont la meme chose et peuvent se stocker au meme endroit. C'est cette idee qui nous permet de parler de memoire / ordinateur etc...

Les langages machine font: * Manipulation de registres * Assurer les echanges avec la memoire

Deux familles de langages

Bottom-Up, langages construits en partant du bas niveau et en remontant. Il construit en voulant ajouter de l'abstraction.

Top-Down, prend le contre pied de l'approche historique. On part de l'abstraction et une fois que l'on est satisfait de la theorie alors on l'implemente. (Ex: HTML, Lisp)