Comparaison avec Java (et C++, Smalltalk, Eiffel, etc)

Typage statique

Ces langages sont *statiquement typés*, ce qui veut dire que l'utilisateur est obligé de déclarer la classe des variables et des arguments à une fonction ou procédure.

Par conséquent, un type abstrait doit aussi être une classe. On introduit donc souvent une classe racine (parfois sans instances) dont le nom est le nom du type abstrait

Restrictions sur les opérations

Tous ces langages ont la *sélection simple*, à savoir qu'une la méthode d'une opération peut être déterminée uniquement par un seul argument (le premier).

Le premier argument est tellement différent que la syntaxe est spéciale:

CLOS: `(mount-engine my-car the-engine)`

Autres: `my_car.mount_engine(the_engine)`

ou: `the_engine.mount_engine(my_car)`

selon si la sélection est souhaité par le type de voiture ou le type de moteur.

Restrictions sur les opérations

Cette syntaxe spéciale est responsable de l'impossibilité de passer une opération en tant qu'argument à une autre opération.

C'est pourquoi ces langages traitent le premier argument différemment des autres.

Relations entre classe et méthodes

Avec ces langages, la relation entre une classe et un ensemble de méthodes est tellement forte que les méthodes sont physiquement à l'intérieur de la définition de la classe.

Héritage simple ou multiple

Certains de ces langages ont l'héritage simple uniquement, par exemple Smalltalk.

Certains autres ont l'héritage multiple, comme C++.

Le langage Java à l'héritage simple plus la notion d'interface presque équivalent à l'héritage multiple.

Terminologie CLOS

La structure d'une instance est définie par les *créneaux* (slots)

Chaque créneau a un nom et une valeur (similaires aux champs d'une structure) .

Un créneau est soit *local* (une valeur différente dans chaque instance) ou partagé (une valeur commune pour l'ensemble des instances).

Une classe peut être définie comme une extension d'une autre classe ou de plusieurs autres classes, appelées *superclasses*. La nouvelle classe est une *sous-classe* de ses superclasses.

Terminologie CLOS

Un ensemble d'objets est représenté par un *objet de type classe*.

Un objet dans l'ensemble est une *instance directe* de la classe.

Chaque instance directe d'une classe a la même structure et le même type.

Fonctions Génériques

Une fonction générique est une *définition d'interface* et non d'implémentation.

L'implémentation d'une fonction générique est faite par zéro, une ou plusieurs *méthodes*.

Le choix de la méthode à utiliser est déterminé par le types des arguments (mécanisme de *sélection* (dispatch)).

En CLOS, la sélection est déterminé par *l'ensemble des argument* (multiple dispatch, multi methods), alors que d'autres langages n'utilisent que le premier argument pour la sélection.