INTRODUCTION A LA PROGRAMMATION OBJET

Baccalauréat STI2D - Systèmes Informatiques et Numériques

Centre d'interêt :

- Traitement de l'information (ET 3.1.4) : Programmation objet : structures élémentaires de classe, concept d'instanciation

Objectifs

A la fin de la séquence, l'élève doit être capable

- de définir ce qui distingue une programmation classique d'une programmation objet
- de définir les notions de classe, d'attributs et de méthodes
- de définir les notions d'héritage

Pré-requis

- Structures algorithmiques

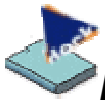
Extrait du document d'accompagnement STI2D

La programmation objet se limite à l'utilisation d'objets possédant une représentation visuelle et donc une interface facilement identifiable. L'objectif étant de montrer la

constitution d'un objet à travers notamment les méthodes et variables internes, leurs réutilisations, et l'intérêt de la programmation objet pour un projet « complexe ».

Mettre en oeuvre des outils graphiques pour l'édition/instanciation d'objets, la programmation associée à un simulateur.

Programmation procédurale



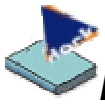
Définition

Avec des langages tels que le C ou le Pascal, la résolution d'un problème informatique passe généralement par l'analyse descendante qui consiste à décomposer un problème en sous-problème jusqu'à descendre à des actions primitives.
On décompose ainsi un programme en un ensemble de sous-programmes appelés procédures qui coopèrent pour la résolution d'un problème.

Les procédures et fonctions sont généralement des outils qui produisent et/ou modifient des données.

Ainsi :

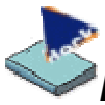
PROGRAMME = ALGORITHME + DONNEES



Inconvénients

- L'évolution d'une application développée suivant ce modèle n'est pas évidente car la moindre modification
  des structures de données d'un programme conduit à la révision de toutes les procédures manipulant ces données.
- Pour de très grosses applications, le développement peut être très long

Programmation Orientée Objet



Définition

Les inconvénients de la programmation procédurales, et en particulier les temps de programmation trop longs, ont contribués à développer une nouvelle méthode de programmation appelée POO.

Le concept objet logiciel est né du besoin de modéliser des objets du monde réel.

Tout objet du mode réel peut être représenter par un objet, qu'il s'agisse d'un objet vivant,

ou d'un objet construit.

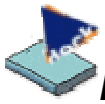
Par exemple, prenons un véhicule industriel à automatiser.

- Un véhicule possède plusieurs comportements : avancer, reculer, charger des éléments, ...
- Un véhicule possède des caractéristiques propres : Vitesse, poids, taille, résistance, ...

Pour représenter ce véhicule dans l'approche objet :

- Les comportements seront représentés par des méthodes.
- Les caractéristiques seront représentées par des variables.

L'objet



L'OBJET

Un objet est un "paquet" logiciel qui contient une série de procédures et de données reliées. Dans l'approche orientée objet les procédures sont des méthodes et les données des variables ou attributs.

L'approche objet consiste à mettre en relation directement les données avec les algorithmes qui les manipulent. Un objet regroupe à la fois des données et des algorithmes de traitement de ces données. Au sein d'un objet, les algorithmes sont généralement appelés des méthodes les données des propriétés ou des attributs.

OBJET = ATTRIBUTS + METHODES

En informatique, un objet est une entité virtuelle possédant

- des attributs (*properties*)
- des méthodes (*methods*)

Exemple : un point (un pixel) sur un écran est un objet.

Ses attributs peuvent être :

- Sa position horizontale
- Sa position verticale
- Sa couleur

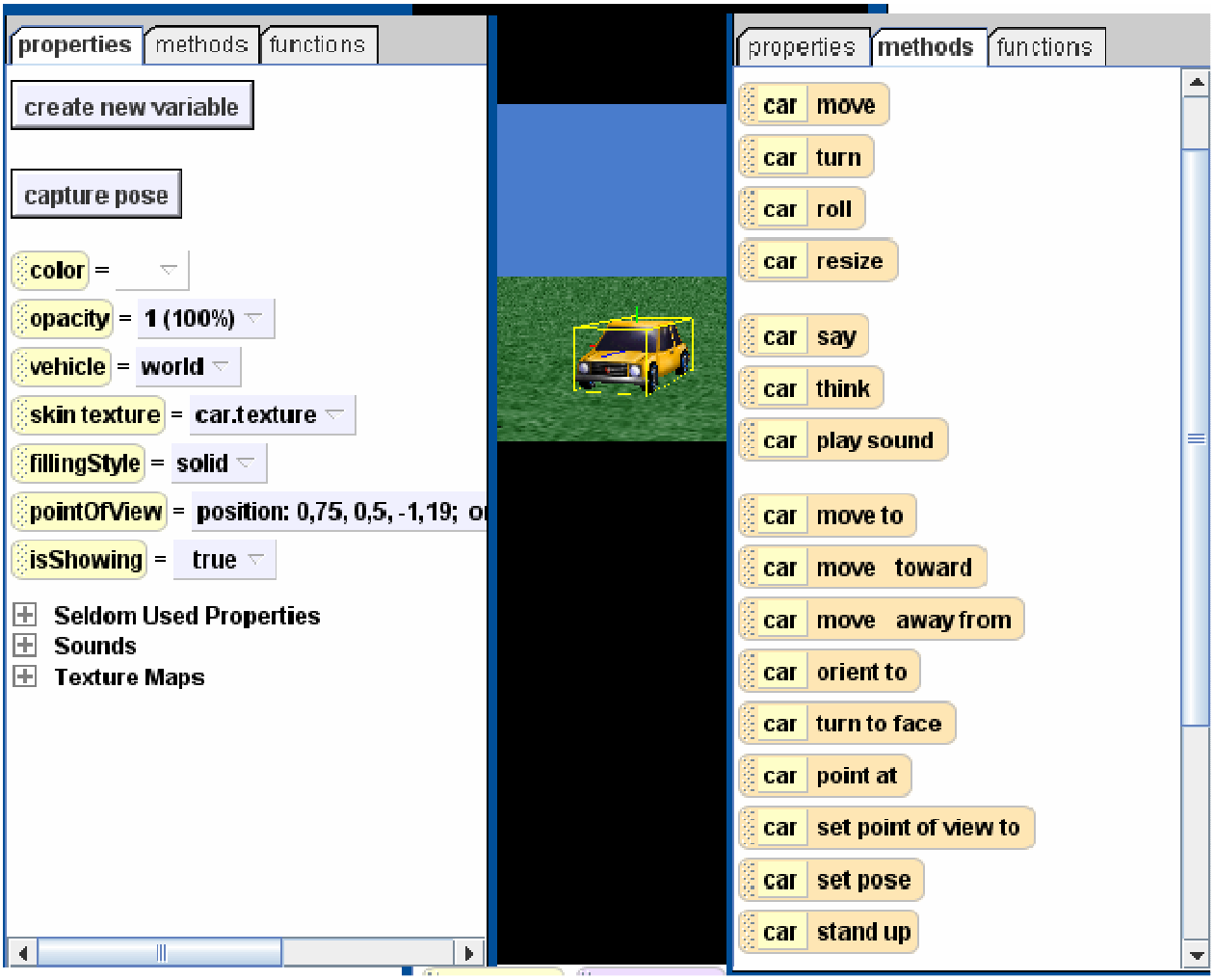
Des méthodes :

- positionner en x,y
- Afficher le point

Exemple d'un objet

Voiture (car) dans le logiciel Alice

Dans l'onglet de gauche, les attributs de l'objet et dans l'onglet de droite les méthodes.



Fondements

Les classes (Class)

Une classe est un moule d'objet, elle décrit les attributs et les méthodes de toute une famille d'objets.

Une classe définit en particulier un type d'objet.

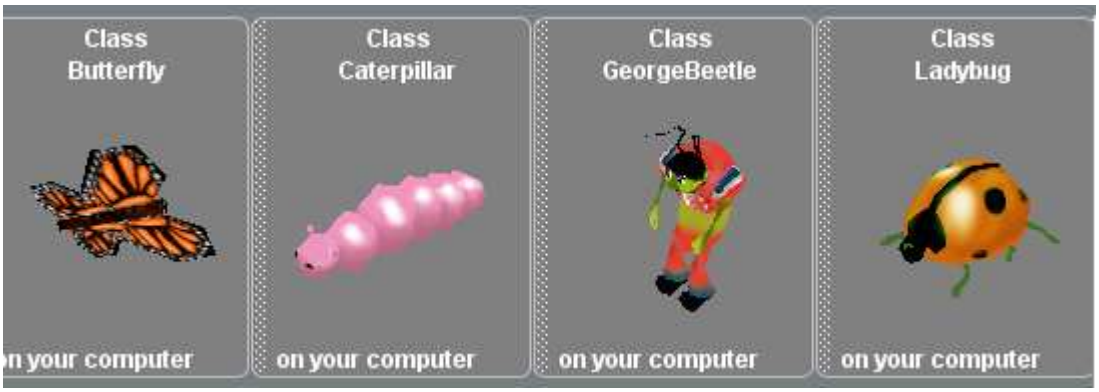
Les objets d'une même classe auront des attributs de même type, mais avec des valeurs différentes.

Une classe est la description générale d'une famille d'objet et un objet est une instance de classe. On parle d'***instanciation***.

Par exemple, si une classe ***Voiture*** décrit une voiture avec comme attributs une couleur, un type de moteur et un nombre de chevaux, une instance de cette classe, un objet, désigne donc une voiture particulière, "la voiture de Monsieur Jean", qui est rouge et qui a un moteur diesel de 90 chevaux.

Dans l'application Alice, une coccinelle, un papillon sont des classes.

En les sélectionnant, on créé des objets dérivés de ces classes et dont on peut changer les attributs.



Le polymorphisme

Le mot *Polymorphisme* vient du grec et veut dire *de formes différente*s.

Le polymorphisme permet d'avoir des méthodes (polymorphisme ad-hoc), des attributs (polymorphisme paramétrique) de même nom, avec des fonctionnalités similaires, dans des classes sans aucun rapport entre elles (si ce n'est bien sûr d'être des filles de la classe objet).

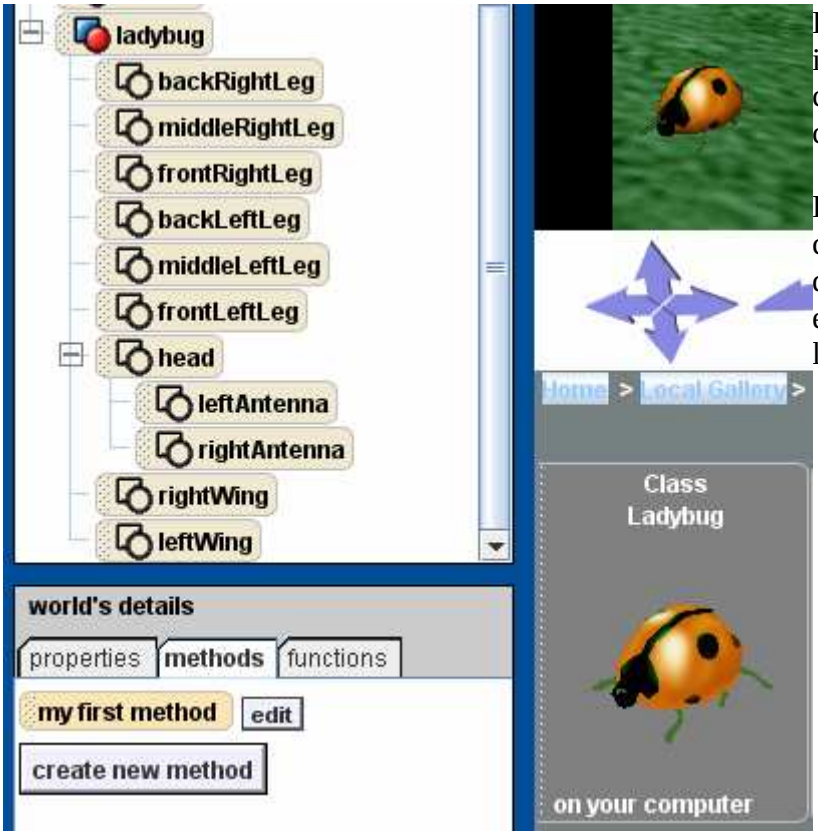
Par exemple, la classe texte et la classe image dans un logiciel de traitement de texte peuvent avoir chacune une méthode "afficher". Cette méthode à la même action alors qu'elle manipule des objets différents.

L'héritage est une forme particulière de polymorphisme.

L'héritage (Herited)

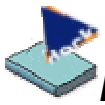
L'héritage est un mécanisme par lequel on définit une classe d'objet comme étant un cas particulier d'une classe plus générale.
Les classes peuvent être imbriquées à l'infini, et l'héritage s'accumule automatiquement.
Cette structure est arborescente et s'appelle une ***hiérarchie de classe***.
C'est grâce à l'héritage que l'on obtient une réelle "réutilisation" du code, une maintenance facilité et surtout une grande évolutivité. Toutes les sous-classes héritent d'une nouvelle fonctionnalité écrite dans la super-classe.

Dans l'application Alice les personnages sont des classes qui intègrent des sous-classes :



La classe ***Ladybug*** (coccinelle) intègre des sous-classes qui correspondent aux membres de la coccinelle.

Le principe même de l'héritage c'est que lorsqu'une méthode est décrite sur une classe parent elle est automatiquement héritée par les classes enfants.



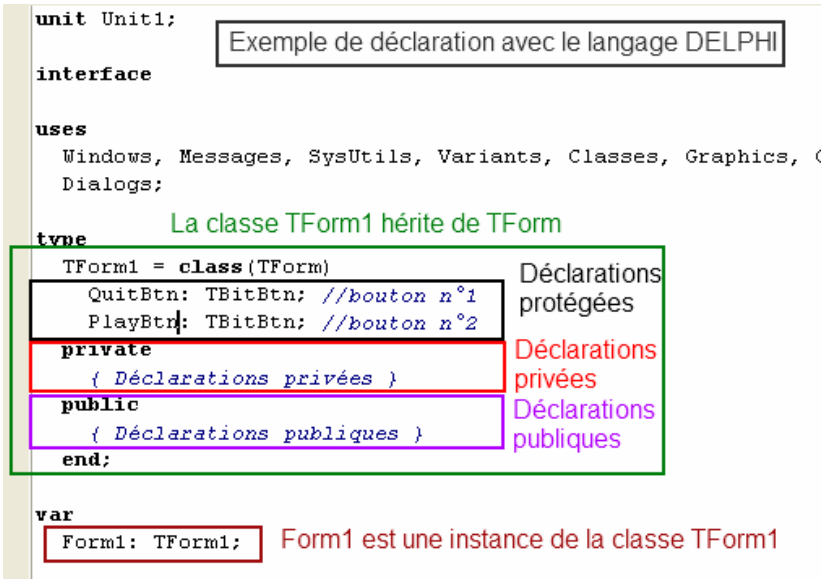
Encapsulation des données

L'encapsulation est un mécanisme consistant à rassembler les données et les méthodes au sein d'une structure en cachant l'implémentation de l'objet, c'est-à-dire en empêchant l'accès aux données et aux codes informatiques par un autre moyen que les services proposés.

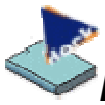
L'encapsulation permet donc de garantir l'intégrité des données contenues dans l'objet.

Pour ce faire, lors de l'écriture des classes on dispose de trois niveaux de lisibilité :

- **publique:** les fonctions de toutes les classes peuvent accéder aux données ou aux méthodes d'une classe définie avec le niveau de visibilité public.
- **protégée:** l'accès aux données est réservé aux fonctions des classes héritières, c'est-à-dire par les fonctions membres de la classe ainsi que des classes dérivées
- **privée:** l'accès aux données est limité aux



méthodes de la classe elle-même. Il s'agit du niveau de protection des données le plus élevé



Les messages

En programmation objet, les objets s'animent grâce aux méthodes.

Les méthodes sont activées grâce à des messages qui leur sont envoyés.

Les messages peuvent provenir

- d'un élément extérieur : clavier, souris, etc....
- de mécanismes internes au programme orienté objet : temporisation, signal provenant d'un autre objet.....