Introduction à la programmation avec un interprète graphique

Commencer par le mode de fonctionnement le plus simple.

C'est quoi, programmer ?

Programmer, c'est commander, donner des ordres à la machine.
Par exemple, on peut prendre une comparaison. Un programme est comme une recette de cuisine, c'est une suite d'ordres donnés à l'ordinateur : Faire ceci, puis faire cela, ensuite faire ça jusqu'à tel moment …

A la base du commandement : l'ordre, l'instruction.

L'ordinateur est conçu pour exécuter un ordre

A la base l'ordinateur exécute l'ordre qu'on lui donne : Il est conçu pour ça.

Syntaxe et grammaire

Encore faut-il parler le langage de l'ordinateur, car il est loin de parler comme nous.
Normalement, c'est à la machine de s'adapter à l'homme, mais ici ce n'est pas le cas, c'est à vous de vous adapter.
A ce niveau intervient la notion de grammaire du langage : le programmeur doit respecter une syntaxe : Construire des phrases, des ordres, selon des règles précises, au moyen de points, virgules, points virgules, parenthèses, crochets, points d'exclamation …

Commander avec un verbe d'action à l'impératif

- Pour commander l'ordinateur, il faut utiliser un verbe d'action à l'impératif (car c'est parlant, concret) : Lève le crayon, vide l'écran.
- Mais l'instruction peut aussi être un nom : ligne, point, liste.

Notion de syntaxe :

Une instruction est suivie par un point virgule.

La notion de modalité :

Quelques exemples concrets :
Faire fondre 50g de beurre.
Peser 100g de sucre.
Laisser refroidir 1h : Refroidir(60);

Complément de manière du verbe d'action : av(123);

- Pour exprimer la modalité, i.e. de quelle façon se fait l'action, on débouche sur la notion de paramètres d'une fonction
  . Encadré par des parenthèses
  . Séparé par des virgules
  Attention au niveau syntaxique, 'séparé' ne veut pas dire " suivi par des virgules ".
  . Si la fonction n'a pas de paramètre, en C, il faut encore les parenthèses : ve( ) ; ct( ) ; .

Un programme, c'est une suite d'ordres donnés à l'ordinateur.

- La notion de " suite d'ordres " débouche immédiatement sur un aspect traitement séquentiel des ordres.
- Par la suite on verra que les ordres peuvent se regrouper selon un aspect thématique et on débouchera sur la notion de sous-programme.

C'est une relation homme→machine,

. Les ordres proviennent de l'utilisateur et vont à la machine.
. Le but serait d'obtenir une relation conviviale à l'ordinateur.
. Mais elle se passe à un niveau technique, donc elle est rigide.
. Et c'est à l'utilisateur de se mettre au niveau de la machine.
. A terme évidemment le but est de développer des interfaces qui se mettent au service, au niveau de l'utilisateur.

Interprète graphique : ce qui est le plus simple pour l'utilisateur.

Environnement graphique de Seymour Papert (Logo 1968) :

- Environnement de la tortue graphique.
- Dans ce cas, les ordres sont tellement simples qu'ils tiennent sur une ligne.
- Interprète : l'ordinateur traite des ordres simples qui tiennent sur une ligne.

Note :

J'ai quelques idées pour d'autres environnements très simples, inspirés de la vie artificielle

La boucle d'évaluation en mode direct

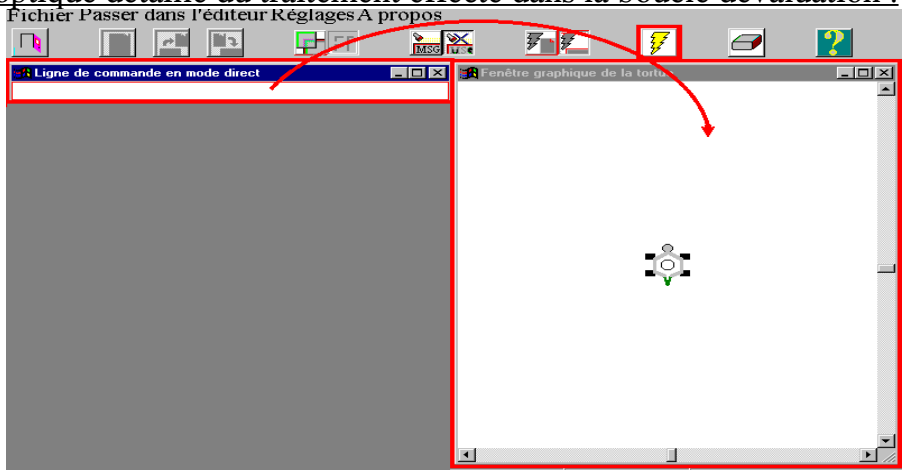
Description de la boucle :

Historiquement : Imprime la valeur de la lecture.

En C : Evalue/exécute le programme lu.

- Le programmeur saisit le programme dans la ligne de commande.
- En cliquant sur l'éclair, il demande au logiciel de l'interpréter.
- L'interprétation de cette commande dessine dans la fenêtre graphique de la tortue.

Synoptique détaillé du traitement effecté dans la boucle dévaluation :



Lecture	: Ajout d'espaces pour encadrer les séparateurs
Analyse lexicale.	: Voir si les mots utilisés appartiennent au lexique.
Analyse syntaxique.	: Vérification des règles de grammaire.
Traitement sémantique.	: Traitement rationnel de l'information (respect du sens)

Affichage des erreurs :

A chaque étape, il peut y avoir des erreurs.

L'interprétation des messages d'erreur envoyés par la machine, est souvent difficile pour le programmeur.

Aspect incrémental du mode direct

Au lancement du logiciel :

Tout est remis à zéro : lieu, orientation, couleur de la tortue ...

Action :

Chaque action modifie l'univers graphique (ici principalement la tortue).

Aspect incrémental :

L'effet de chaque action est gardé, mémorisé.

Aspect contextuel :

En mode direct, la même action (par exemple av(123);) peut donner des résultats différents selon :

- l'épaisseur du crayon.
- La couleur du crayon.
- Si le crayon est levé.
- La position de la tortue.

- Sa direction.

Il est important de comprendre que c'est très gênant.

Introduction de l'aspect hors-contexte

Nous avions le mode le plus simple : le mode contextuel.

Le mode contextuel est dangereux

- Le contexte c'est ce qui dépend de ce qui s'est passé avant.
- Selon le contexte (ce qui s'est passé avant) il ne donne pas le même résultat.

Exemple d'une anaphore :

. " Donne lui la clé " : Cette phrase, cet ordre induit une scène différente selon son contexte d'utilisation.
. Papa arrive, donne lui la clé.
. Maman arrive, donne lui la clé.
. Toto arrive, donne lui la clé.

Il faut bien voir qu'ici, selon le contexte, le pronom 'lui' réfère à 3 personnes possibles et la même phrase possède 3 sens différents, donne 3 résultats dissemblables.
Inutile de dire qu'en informatique, on a horreur de ça.

La boucle d'évaluation en mode éditeur

Passons à un niveau plus élaboré : Le mode hors contexte.

Le but :

C'est que la même séquence d'instructions donne toujours le même résultat : Son fonctionnement cesse d'être contextuel, mais devient hors contexte, i.e. invariant quel que soit le contexte précédant.

Comment :

. En mode 'éditeur', avant d'évaluer le programme, on efface (on RAZe – Remet A Zéro) tout le contexte.
. On évalue d'un seul coup toutes les commandes qui sont dans l'éditeur.
. Par conséquent, le programme sera toujours exécuté avec le même contexte, donc il donnera le même résultat.