Découvrir le langage Linotte

Le langage de programmation entièrement en français, simple, puissant.

Introduction au langage

2012 v0.7

Plan

- Découvrir le langage
- Le livres et les fonctions
- Les acteurs et les rôles
- Les actions
- L'affectation de valeur
- Interaction homme / machine
- Les boucles FOR classiques
- Les boucles FOR simplifiées
- Et bien plus encore !
- Les conditions
- Les boucles WHILE
- Les blocs
- Les mathématiques
- Paradigme impératif
- Paradigme fonctionnel
- Récursivité

Découvrir le langage Linotte

- Très simple à apprendre : une syntaxe proche du français;
- Pédagogique : on écrit ce que l'on pense;
- Notion de variables, de boucles, d'objets, de clonages, d'héritage, de graphisme, de réseaux, de fichiers, d'évènements, de traitements parallèles, etc.
- Gratuit et même plus : libre !
- Logiciel disponible sous la licence GPL V3.

Le livre et les fonctions

- Un livre regroupe les instructions qui vont permettre de créer votre programme.
- Un livre est structuré en fonctions.
- Les fonctions permettent d'architecturer un programme ou de construire des algorithmes (récursifs par exemple).
- Par comparaison, en langage Java, un livre est un programme, une fonction est une méthode.
- Une fonction peut avoir des paramètres.

Le livre et les fonctions

```
principale :
  prénom :: texte
  début
    affiche "Quel est ton prénom ?"
    demande prénom
    affiche "Bonjour, " + prénom
```

Les acteurs et les rôles

- Les acteurs sont les entités qui vont stocker les valeurs de votre programme.
- Les acteurs sont identifiés par un nom.
- La sémantique d'un acteur est déterminée par son rôle : un texte, un nombre, un casier, un drapeau ou une espèce.
- Les acteurs doivent être déclarés au début d'une fonction.
- Un acteur ne peut changer de rôle.

Les acteurs et les rôles

```
principale :
  message :: texte <- "Quel est ton âge ?"
  âge :: nombre
  début
    affiche message
    demande âge
    affiche "Tu as " + âge
```

Les actions

- Elles donnent les ordres à l'interprète Linotte (demander, afficher, terminer, etc.)
- Un verbe est une action;
- Les actions peuvent interagir avec un ou des acteurs;
- Le verbier est l'ensemble des actions reconnues par l'interprète : la liste exhaustive est présentée dans le tutoriel du langage (plus de 50 actions !)

L'affectation de valeur

- Lors de l'initialisation de l'acteur avant le mot clé *début* :

```
nom :: texte <- "nicolas"

âge :: nombre <- 16

âge frère :: nombre <- âge + 15
```

- En utilisant l'action *Valoir :*

```
âge vaut 16

âge vaut âge frère

âge vaut âge frère + 16
```

- Autre syntaxe possible avec l'action *Copier :*

```
copie âge frère dans âge
```

Interaction homme / machine

- Interroger l'utilisateur :
  - `Demande âge`
  - `Questionne âge sur "Quel est ton âge ?"`
  - `Âge ?`
- Envoyer un message à l'utilisateur :
  - `Affiche âge`
  - `Affiche "ton âge est " + âge`
  - `"Ton âge est " + âge !`

Les boucles classiques

- On précise l'acteur à incrémenter;
- On peut préciser le *pas* de la boucle;
- Boucles *FOR* avec un *pas* statique :
  - `Pour a de 1 à 10, affiche a`
  - `Pour a de 10 à 1, affiche a`
- Boucles *FOR* avec un *pas dynamique* :
  - `Pour n de 1 à 2 suivant n + 0.1, affiche n`
  - `Pour n de 1 à 100 suivant n * b , affiche n`

Les boucles simplifiées

- Le joker est un acteur pré-rempli par l'interprète;
- L'incrémentation ou la décrémentation est automatique;
- Les boucles FOR simplifiées :
  - `Pour chaque 3, affiche joker`
  - `Pour chaque âge, affiche "Tu as eu " + joker`

  - `De 1 à 10, affiche joker`
  - `De 10 à 1, affiche joker`

Les conditions

- Permet de brancher / débrancher une partie d'un livre :
  - `si <condition>, <action>`
  - `sinon si <condition>, <action>`
  - `sinon <action>`
- Exemples :
  - `si a < b, affiche "a est plus petit !"`
  - `si a <= c, affiche "a est plus petit ou égal à c"`
  - `si "Chloë est une fille" contient "fille", affiche "C'est une fille !"`

Les boucles WHILE

- La boucle *Tant Que* s'utilise avec une condition :
  - `tant que <condition>, <action>`
- Exemples :
  - `tant que a < b, a vaut a + 1`
  - `tant que a != b, a vaut (b + a) / 2`

Les blocs

- Ils regroupent un ensemble d'actions au sein d'une fonction;
- Ils s'utilisent avec les conditions et les boucles.
  - ```
    principale :
        n :: nombre
        b :: nombre
        début
            pour n de 1 à 2 suivant n + b, lis
                b vaut b + 0.1
                affiche n
            ferme
    ```
    ```

Les mathématiques

- La liste des opérations reconnues par Linotte est riche :
  - cosinus, sinus, valeur absolue, arc cosinus, arc sinus, puissance, racine carrée, logarithme décimale, etc...
  - La liste exhaustive est disponible dans le tutoriel du langage Linotte ou dans le menu Verbier de l'Atelier Linotte;
  - Supporte les décimaux très longs;
  - Quelques valeurs prédéfinies : pi, euler;
  - Exemples :
    - `n vaut cos (x) + log ( b + a )`
    - `affiche "Entier de pi = " + entier ( pi )`

Paradigme impératif

- Façon la plus simple pour apprendre à construire un programme.
- S'emploie avec le verbe *Aller* :

```
Spaguetti :
    âge :: nombre
    début
        questionne âge sur "Quel est ton âge ?"
        si a > 30, va vers cas 1
        si a < 31, va vers cas 2
Cas 1 :
    début
        affiche "Tu es vieux pour ton âge !"
Cas 2 :
    début
        affiche "Tu es un jeune !"
```

Paradigme fonctionnel

- L'utilisation du verbe *parcourir* associée au verbe *revenir* constitue la base pour écrire une fonction en Linotte :

```
principale :
    n :: nombre <- 2
    début
        parcours calcul avec n
        affiche n
```

```
Calcul :
    *t :: nombre
    début
        t vaut carré t
        reviens
```

- L'acteur *t* est un paramètre, il est le représentant de l'acteur *n* dans la fonction *calcul* (passage des valeurs par référence).

Récursivité

- Elle s'utilise naturellement avec le verbe *retourner.*
  - Exemple avec la factorielle :
  - ```
    principale :
        début
            affiche factorielle (5)
    factorielle :
        *a :: nombre
        début
            si a == 0, retourne 1
            sinon retourne a * factorielle (a-1)
    ```
- L'utilisation du verbe *retourner* offre la possibilité de définir des fonctions mathématiques.
    ```

Et bien plus encore !

- Ce document ne présente qu'une petite partie des fonctionnalités du langage !
- Linotte est un langage de programmation recommandé par le ministère de l'éducation nationale :
  - Programme pour la classe de seconde
  - Doc_ress_algo_v25_109178.pdf
- Un forum est disponible sur le site :
  - http://langagelinotte.free.fr/forum
- Pour télécharger la dernière version et obtenir les dernières nouveautés, le blog :
  - http://langagelinotte.free.fr/wordpress