TP de Transmission image multimédia

TP1 : les représentations vectorielles, le format Postscript

1 Introduction

Le langage Postscript permet de construire un graphique de manière générique et portable. L'idée consiste à décrire le dessin par un ensemble de tracés élémentaires de **type vectoriel**. Un fichier Postscript est donc constitué d'une succession de **commandes de tracé**, auxquelles s'ajoutent les **instructions arithmétiques** de base, nécessaires à la manipulation et à la gestion de paramètres.

Deux principes directeurs guident le langage Postscript :

1. les opérateurs s'utilisent en **notation postfixée**, c'est-à-dire qu'une commande `command` nécessitant n arguments `argi` ($1 \leq i \leq n$) sera appelée comme suit : `arg1 arg2` $\cdots$ `argn command` ;
2. la suite des instructions s'effectue dans une **pile**, et les arguments des commandes obéissent également à cette règle.

2 Un exemple : tracé d'un rectangle

Le code suivant permet de tracer un rectangle à partir des coordonnées d'un sommet et de celles des sommets consécutifs (1 point Postscript=$\frac{1}{72}$ pouce) :

```
% Les commentaires doivent être précédés d'un %
newpath         % Début de description du rectangle
100 100 moveto  % Premier sommet
100 200 lineto  % Segment entre le premier et le deuxième sommet
200 200 lineto  % ...
200 100 lineto  % ...
closepath       % Fermeture du rectangle
stroke          % Affichage de l'objet
```

Pour observer le résultat de ces instructions, taper le code dans un fichier `toto.ps` et visualiser le avec `gv` ou `ghostview`. Il est également possible de travailler en mode interactif sous `GS` (taper `gs`). Au format `A4`, la taille d'une image en Postscript est de `595`×`842`.

3 Utilisation de procédures et de variables

Un certain nombre de fonctionnalités permettent de simplifier l'écriture d'un fichier Postscript. Ainsi, une **variable** peut se définir par : `/nom_variable valeur_variable def`. Par analogie, une **procédure**, prenant ou non des arguments, s'écrit de la manière suivante :

```
/nom_procedure {
                 suite ... d'instructions
               } def
```

Les arguments des commandes utilisées dans la procédure peuvent être données directement lors de leur appel dans la procédure. Sinon, ils sont récupérés dans la pile.

4 Travail demandé

Exercice 1

En vous aidant de votre imagination et des informations disponibles à l'adresse *http://barthes.ens.fr/carto/docps/index.html*, dessiner un animal ou un objet de votre choix[1], dont vous remplirez certains éléments par des couleurs appropriées.
Il sera notamment intéressant de vérifier :

- la nécessité de la commande `stroke` pour valider la présence d'un objet ;
- l'effet de la commande `closepath` ;
- la possibilité d'entourer l'image d'un cadre noir d'épaisseur donnée ;
- la relation entre les commandes `lineto` et `rlineto`.

Appliquer le principe suivant : "Rien de mieux, pour apprendre, que d'expérimenter par soi-même".

Exercice 2

Reprendre l'**exercice 1** en utilisant au mieux l'avantage des procédures et des variables, pour construire des objets présents plusieurs fois dans le dessin. On pourra tirer profit du guide (en anglais) disponible à l'adresse URL *http://www.tailrecursive.org/postscript/postscript.html*. Pensez à utiliser les opérateurs arithmétiques.

Exercice 3

Reconstruire le dessin (ou contruire un dessin) utilisant des transformations affines (translation, rotation et homothétie) pour les objets qui le nécessiteraient. En particulier, on pourra utiliser successivement les deux commandes `rotate` et `translate`, puis les permuter pour voir l'effet de l'ordre d'appel.

Exercice 4

Centrer un échiquier dans une image Postscript et placer des objets sur certaines de ses cases (cf. Figure 1). On pourra, à l'occasion, utiliser des sauvegardes de contexte (commandes `gsave` et `grestore`).

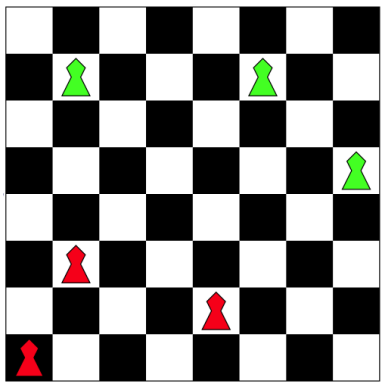

Figure 1: Exemple d'échiquier à construire en Postscript.

Exercice n

Tester les exemples donnés aux adresses URL citées ci-dessus, et créer de nouvelles figures.

[1] On demande un minimum de recherche graphique...