Augustin De Vita
Matthieu Lemonnier
Yoann Moguer

Projet d'architecture logicielle- Un langage de dessin vectoriel

Contents

Augustin De Vita
Matthieu Lemonnier
Yoann Moguer

I Définition du langage de dessin

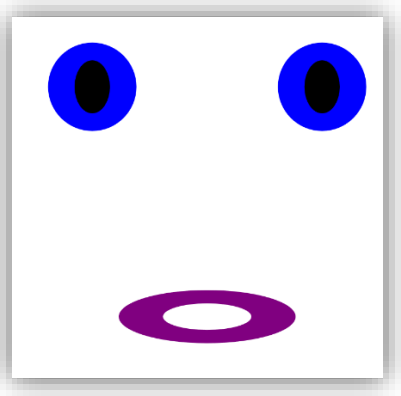
Le langage utilisé pour permettre à l'utilisateur de définir les figures qu'il souhaite représenter utilise dans un premier temps des mots clés, chaque mot clé décrivant la figure à dessiner. Ainsi les choix possibles sont CERCLE, ELLIPSE, LINE et RECT pour respectivement dessiner un cercle, une ellipse, une ligne et un rectangle. Afin de caractériser la fenêtre de dessin nous utilisons de plus le mot clé CANVAS.

En plus des mots clés qui représentent la nature de la figure, ce langage utilise des paramètres pour décrire chaque figure. Voici pour chaque figure les paramètres associés :

- CERCLE(x, y, r, color) où x et y sont les coordonnées du centre du cercle, r le rayon du cercle et color le nom d'une couleur en anglais. (Exemple : CERCLE (120, 120, 50, blue))
- ELLIPSE(x, y, r1, r2, color) où x et y sont les coordonnées du centre de l'ellipse, r1 et r2 les rayons de l'ellipse et color le nom d'une couleur en anglais. (Exemple : ELLIPSE (120, 120, 20, 30, black))
- LINE(x1, y1, x2, y2, w, color) où x1 et y1 sont les coordonnées du point de départ de la ligne, x2 et y2 les coordonnées du point d'arrivée de la ligne, w est la largeur du trait et color le nom d'une couleur en anglais. (Exemple : LINE (50, 50, 150, 150, 2, red))
- RECT(x,y,w,h,color) où x et y sont les coordonnées du point de départ du rectangle, w est la largeur du rectangle, h est la hauteur du rectangle et color le nom d'une couleur en anglais. (Exemple : RECT (150, 150, 20, 50, green))

Ainsi, chaque instruction décrivant une figure prend une ligne dans la page. Voici un exemple d'utilisation du langage décrivant la figure ci-contre :

```
Canvas(500,500)
CERCLE(120,120,50,blue)
ELLIPSE(120,120,20,30,black)
CERCLE(380,120,50, blue)
ELLIPSE(380,120,20,30,black)
ELLIPSE(250,380,100,30,purple)
ELLIPSE(250,380,50,15,white)
```



II Les interprétations du langage

On propose deux interprétations du langage définit précédemment.

II.1 L'interprétation SVG :

La première interprétation suit la norme SVG définie par W3C (http://www.w3.org/TR/SVG/).
Dans notre langage on utilise principalement les formes cercle, ellipse, ligne et rectangle. Ces formes sont définies en SVG par :

- Canvas(w,h) devient `<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="w" height="h">` en SVG
- CERCLE(x,y,r,color) devient `<circle cx="x" cy="y" r="r" fill="color" />` en SVG.
- ELLIPSE(x,y,r1,r2,color) devient `<ellipse cx="x" cy="y" rx="r1" ry="r2" fill="color" />` en SVG.
- LINE(x1,y1,x2,y2,w,color) devient `<line x1="x1" y1="y1" x2="x2" y2="y2" style="stroke:color;stroke-width:w" />` en SVG.
- RECT(x,y,w,h,color) devient `<rect x="x" y="y" width="w" height="h" fill="color" />` en SVG.

Le dessin écrit par l'utilisateur est alors créé dans un fichier au format SVG, un format vectoriel d'image.

II.2 L'interprétation « Description »

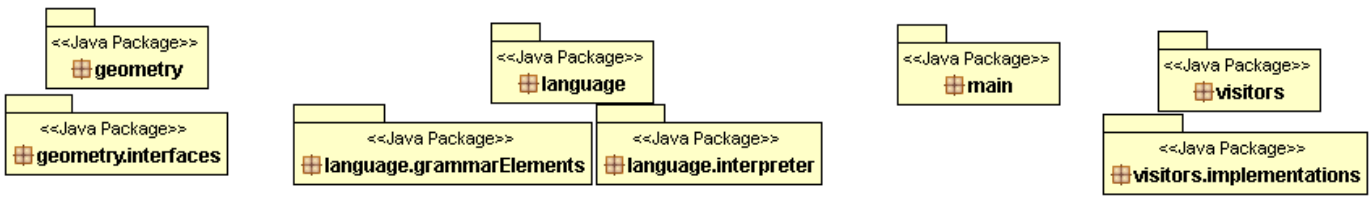
En tant que deuxième interprétation du dessin, nous avons décidé de créer une interprétation dite de description. Cette interprétation décrit le dessin écrit par l'utilisateur de la manière suivante (en reprenant les notations du dessus) :

```
It is a drawing with a size of w on h which contains:
- a color circle with a center (x, y) and a radius of r
- a color ellipse with a center: (x, y), with a horizontal radius of r1,
and a vertical radius of r2
- a color line going from (x1, y1) to (x2, y2) with a thickness of w
- a color rectangle with this origin :(x, y), a width of w, a height of h
And that's it
```

III Architecture logicielle

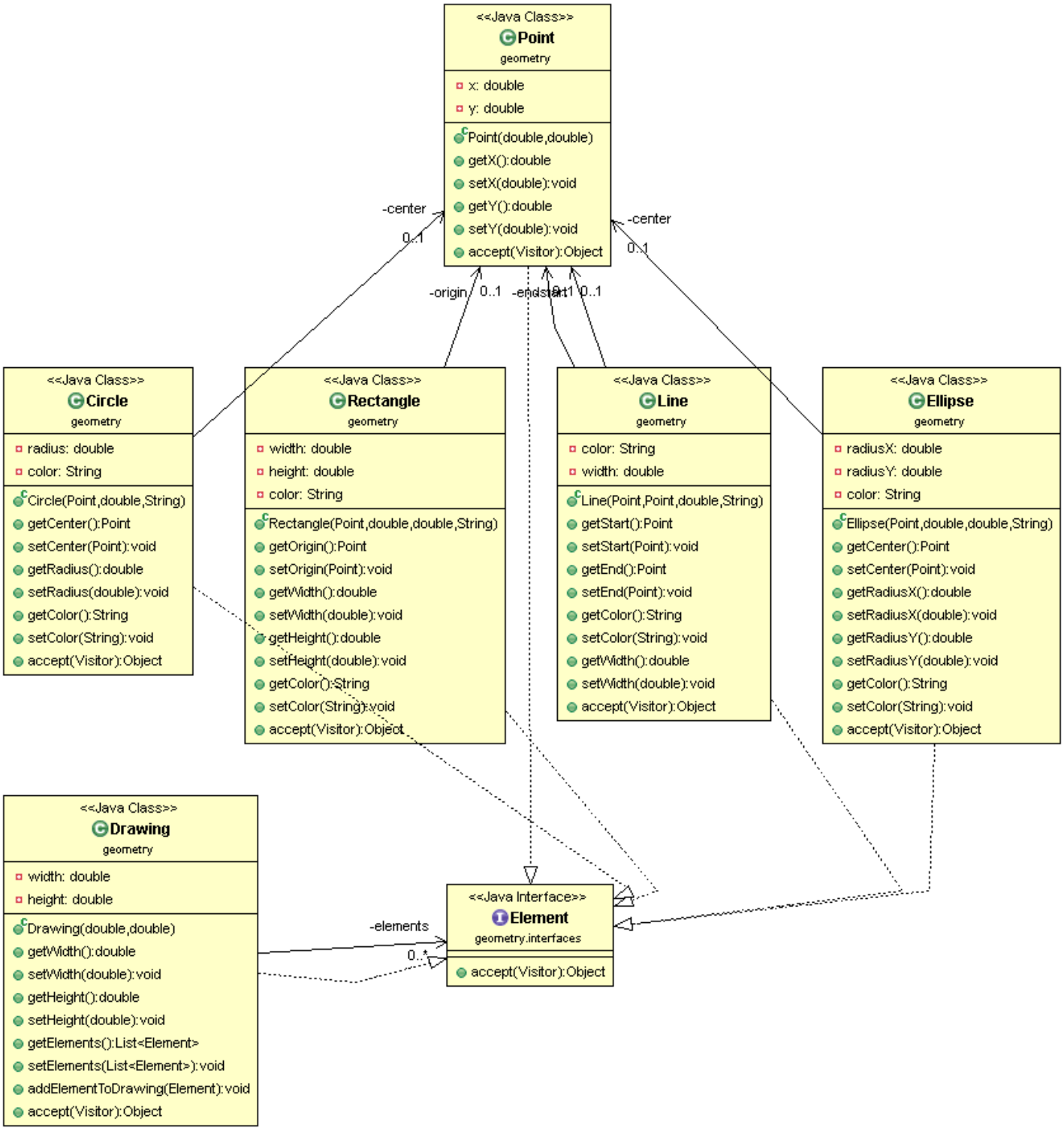
III.1 Schéma général

Le projet est décomposé en 4 packages principaux et de 4 sous-packages comme vue si dessous :

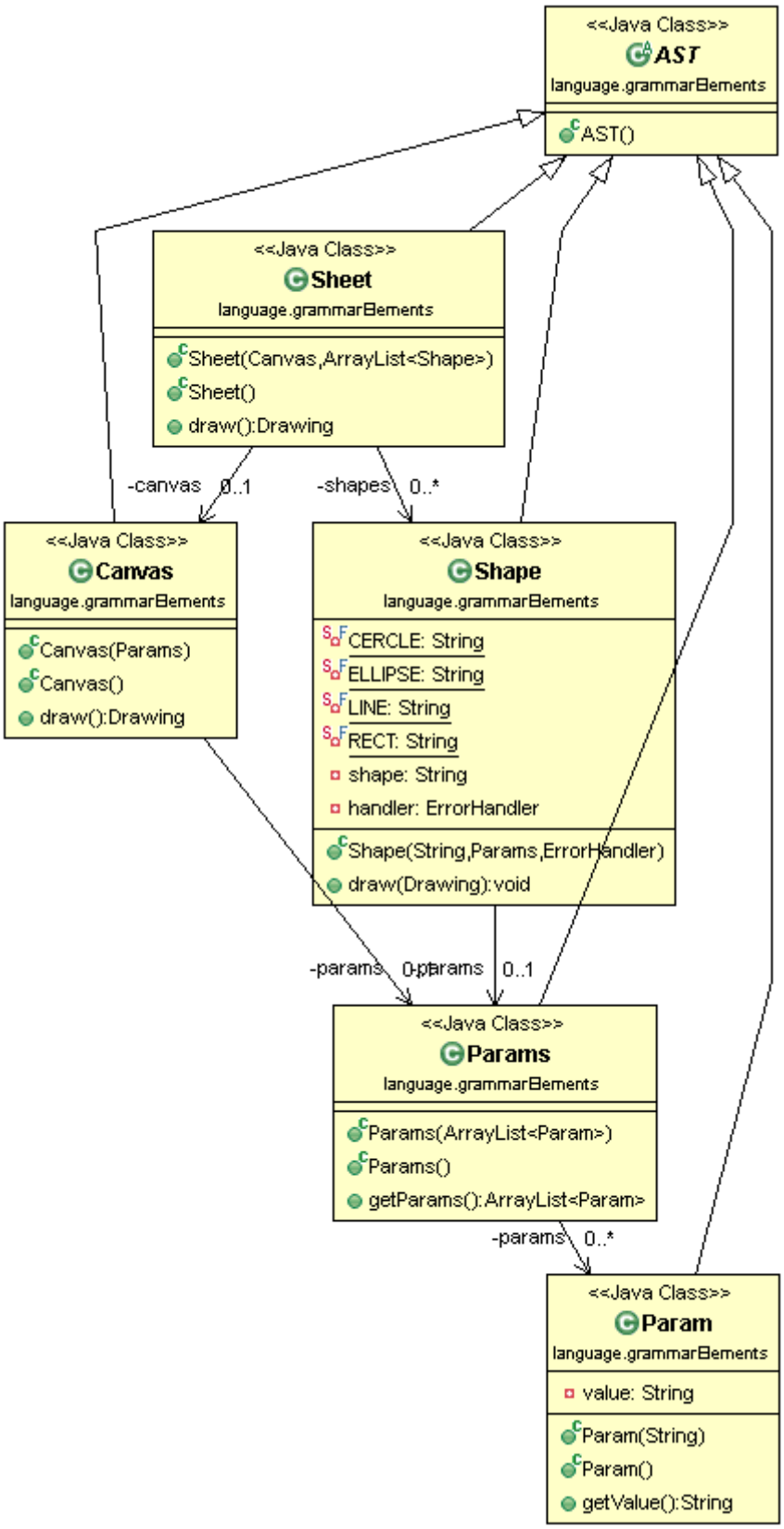


Augustin De Vita
Matthieu Lemonnier
Yoann Moguer

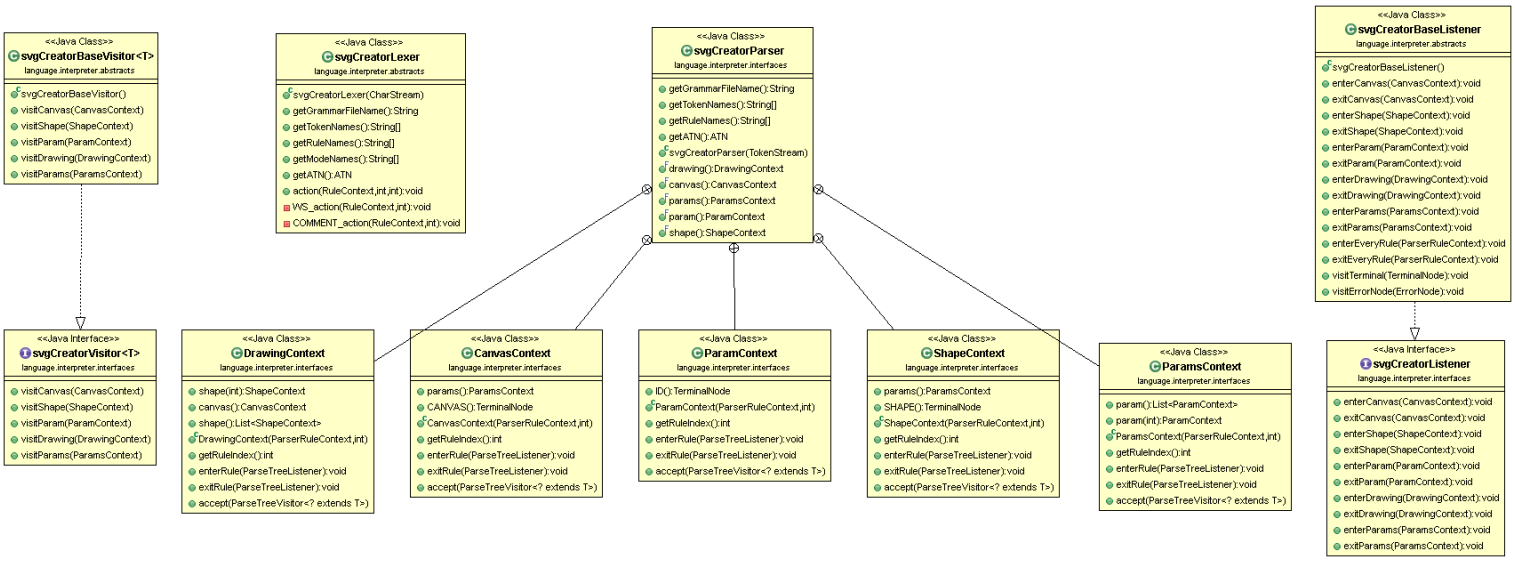
Le premier package *geometry* est composé de 6 classes et d'une interface donc voici le diagramme :



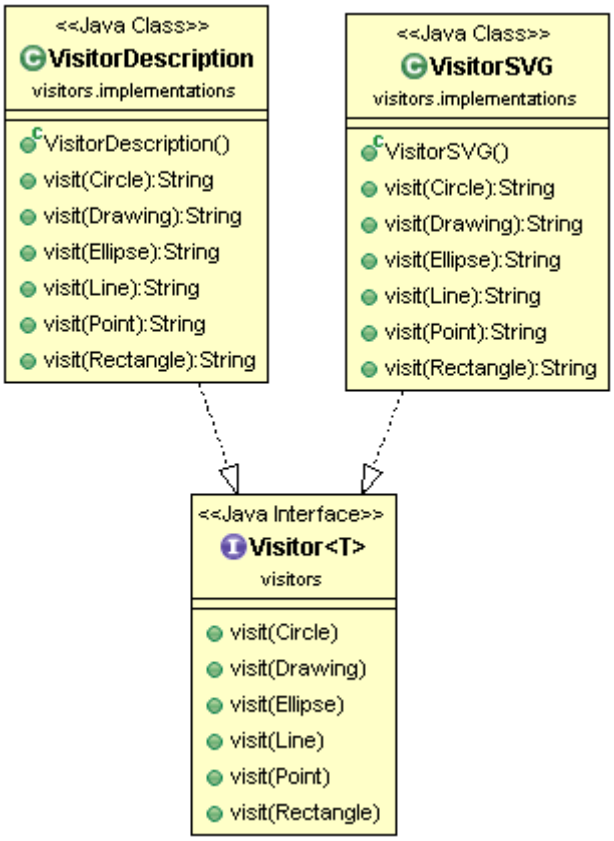
Le deuxième package *language* est composé de 6 classes principales en plus des classes de l'interpréteur qui ne sont pas représentées dans les diagrammes de classe suivant :



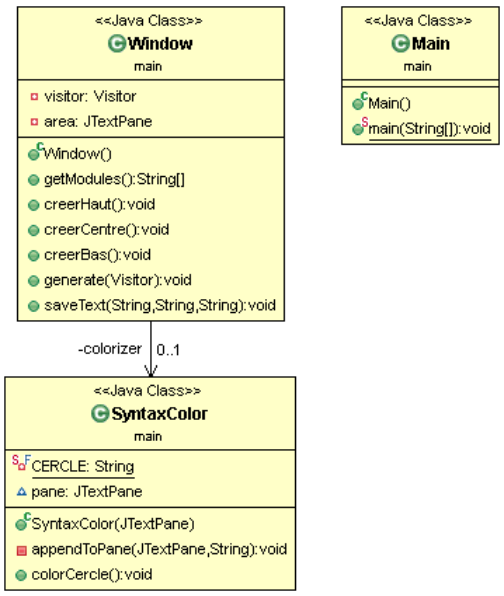
Augustin De Vita
Matthieu Lemonnier
Yoann Moguer



Le package *visitors* contient l'interface *Visitor* qui est ensuite implémenté en 2 interprétations *VisitorDescription* et *VisitorSVG* (dans le sous package *implementations*). Le digramme du prackage est décrit ci-dessous :



Enfin, le package Main qui permet de lancer l'IHM et le programme contient 3 classes dont le diagramme est le suivant :



III.2 Principes modulaires suivis et choix des patrons de conception

Le programme est découpé en trois packages :

- Le premier interprète un langage spécifique au projet, tout en restant basique il permet de créer des descriptions ou des fichiers au format svg depuis une interface graphique.
- Le deuxième est la partie embarquée du langage, le packet "geometry". Il définit les attributs spécifiques à chaque forme, et chaque forme implémente l'interface "Element". Cette interface fournit une méthode "accept", qui servira dans l'interprétation.
- Le troisième et dernier paquet, contient les différentes interprétations.

Un dessin étant constitué d'un élément "Drawing" qui lui-même agrège des "Element", la méthode "accept" de ces derniers permet l'utilisation du patron Visiteur sur des éléments dont on ne connait pas le sous-type (on sait qu'ils sont tous du type "Element").

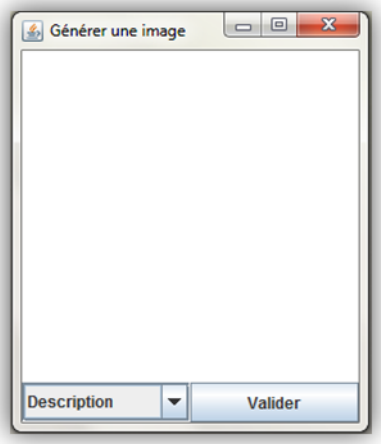
Grâce au patron de conception Visiteur, une interprétation est donc définit en une seule classe implémentant l'interface générique Visiteur<T>.

III.3 Méthode pour étendre le langage

Pour étendre le langage et donc rajouter une interprétation, il suffit d'ajouter une implémentation de l'interface *Visitor* dans le package *visitors*. Ensuite il faut mettre à jour l'interface graphique pour donner le choix à l'utilisateur d'utiliser cette nouvelle implémentation.

IV Guide d'utilisation

Pour utiliser le programme, il faut lancer la classe Main.java, présente dans le dossier main des sources. La fenêtre ci-contre s'ouvre alors :



Dans cette fenêtre il faut d'abord spécifier la taille du dessin vectoriel que l'on souhaite réaliser en utilisant la syntaxe Canvas(x,y), où x et y représente respectivement la largeur et la hauteur du dessin à réaliser (ne pas utiliser de mettre un C majuscule). Puis, il faut entrer les différentes figures que l'on souhaite réaliser en utilisant le langage décrit précédemment (mots clés tous en majuscule).

Il faut ensuite spécifier si on souhaite obtenir le fichier SVG, qui, en l'ouvrant dans un navigateur représente les figures décrites. Dans ce cas, le programme enregistre le fichier en .svg à l'endroit et au nom souhaité. Sinon, on peut choisir l'option « description » qui enregistre un fichier .txt à l'endroit et au nom voulu, décrivant le dessin et les figures qui la compose.