Chapitre 1

Figure Smalltalk de Dr. Geo

Les *Figures Smalltalk de* Dr. Geo – (FSD) – sont des figures écrites en langage Smalltalk. Il ne s'agit donc plus de construire une figure à l'aide de l'interface graphique de Dr. Geo mais plutôt de décrire une figure dans le langage Smalltalk. Nous avons apporté le plus grand soin afin que la syntaxe utilisée soit facile et légère.

1.1 Figure Smalltalk par l'exemple

En lui-même Smalltalk est un langage de très haut niveau. Lorsqu'une figure est définie dans ce langage, nous disposons également de toute sa puissance pour par exemple définir récursivement telle partie de la figure, ou bien pour placer aléatoirement certains objets de telle sorte qu'à chaque ouverture de la figure, celle-ci est légèrement différente. Bref, les FSD sont libérées du carcan de l'interface graphique tout en étant renforcées par le langage Smalltalk.

Une FSD est un code source Smalltalk à exécuter dans un espace de travail – *workspace*. C'est une fenêtre texte depuis laquelle du code Smalltalk est écrit et exécuté. Pour ouvrir un tel espace, faire Ctrl-k lorsque aucune fenêtre n'est sélectionnée. On peut aussi y coller du texte par Ctrl-v.

Nous allons étudier plusieurs exemples, chacun d'eux sera écrit dans un workspace et exécuté en sélectionnant le code puis la séquence de touches Ctrl-d pour *Do-it !*

Commençons par étudier un exemple simple de FSD :

```
DrGeoCanvas new
```

C'est la plus petite FSD que nous puissions définir. Lors de son exécution, celle-ci va simplement créer une nouvelle figure vide. Le canevas Dr. Geo affiché est simplifié puisqu'il ne comporte ni barre de menu, ni barre d'outils. En revanche un menu contextuel s'affiche par un clic bouton droit ou gauche sur le fond du canevas ; il donne accès aux outils pour construire et éditer la figure. du canevas.

Pour déplacer ce canevas, l'attraper sur les bords en bas ou à droite ; pour le fermer, presser le bouton du milieu de la souris et utiliser l'icone fermer en haut à gauche.

Abordons un deuxième exemple :

```
| c item |
c := DrGeoCanvas new.
item := c point: 1.2 @ -2.
item name: 'A'.
```

Cette FSD définit une figure avec un point libre A de coordonnées initiales $(1,2\ ;\ -2)$. Les objets sont ajoutés à la figure à partir du canevas, ici `point:` crée un point libre à partir

de deux coordonnées. Le résultat est un objet point, qu'il est possible de modifier, ici il est renommé 'A'.

Poursuivons avec un troisième exemple :

```
| c triangle hasard m n p |

triangle := [:p1 :p2 :p3 |
c segment: p1 to: p2.
c segment: p2 to: p3.
c segment: p3 to: p1].

hasard := [5 - 10 atRandom].

c := DrGeoCanvas new.
m := c point: hasard value @ 0.
n := c point: 5 @ 0.
p := c point:  hasard value @ 3.
triangle value: m value: n value: p.
```

Cet exemple est particulièrement intéressant, il nous montre trois choses importantes :

1. L'introduction d'une construction de plus haut niveau, non prévue au départ par DR. GEO. Ici nous avons défini le bloc de code `triangle` qui, à partir de trois points, construit le triangle passant par ces trois points. Nous pouvons comparer ceci avec les macro-constructions mais avec un degré de liberté beaucoup plus important.
2. La définition d'un bloc de code associé, ici nous avons défini `hasard` qui retourne un nombre entier compris entre -5 et 5. Nous utilisons ce bloc pour placer au hasard certains points de notre figure, ainsi à chaque exécution la figure est légèrement différente.
3. L'affectation du résultat d'une construction à une variable n'est pas obligatoire, nous l'utilisons lorsque nous souhaitons garder une référence de l'objet créé. Par exemple dans le bloc de code `triangle`, nous ne gardons pas de référence des segments créés, en revanche lorsque nous définissons nos trois points nous avons besoin de garder une référence dans des variables temporaires. Ainsi, lors de l'utilisation du bloc de code `triangle`, nous passons en paramètre les variables `m`, `n` et `p`.

Pour clore cette section, voici un dernier exemple :

```
| c a b d |

c := DrGeoCanvas new.
a := c point: 1@0.
b := c point: 5@0.
d := c line: a to: b.
a color: Color yellow;
   round;
   large.
b hide.
d dashed.
```

Deux points et une droite sont créés. Ensuite des commandes sont utilisées pour modifier l'aspect des objets, voire pour en cacher.

Nous avons terminé notre petite visite guidée des *Figures Smalltalk Dr. Geo*. Dans les sections suivantes nous exposons l'ensemble des commandes disponibles pour définir des FSD.

1.2 Méthodes de référence pour les Figures Smalltalk Dr. Geo

La définition d'objets dans un document FSD se fait dans un canevas Dr. Geo. Les objets ainsi créés sont modifiables comme nous le verrons par la suite.

Cependant, avant toute définition d'objets d'une figure, cette dernière doit être créée avec la commande `DrGeoCanvas new`.

1.2.1 Commandes générales

`<canvas> DrGeoCanvas new`

⟶ Retourne une référence vers un canevas et affiche celui-ci. Cette référence est nécessaire pour créer des objets dans ce canevas, il est donc important de la placer dans une variable.
Exemple :

```
| canvas |
canvas := DrGeoCanvas new.
```

`canvas do: bloc`

`bloc` : bloc de code Smalltalk contenant des instructions de construction et/ou d'animation de la figure interactive.
Action : Exécute le bloc de code dans un processus en tâche de fond. A utiliser lorsque la construction doit se faire sous les yeux de l'utilisateur ou bien lorsque la figure est animée.
Exemple :

```
| canvas point |
canvas := DrGeoCanvas new.
point := canvas point: 0@0.
canvas do: [
   -5 to: 5 by: 0.1 do: [:x |
      point moveTo: x@(x cos * 3).
      (Delay forMilliseconds: 100) wait.
      canvas update]
]
```

`canvas update`

Action : Mise à jour du canevas après modification d'attributs de quelques items. La plupart du temps ce n'est pas nécessaire.

`canvas fullscreen`

Action : La fenêtre du canevas est étendue pour couvrir tout l'écran.

`canvas gridOn`

Action : Affiche la grille unitaire du canevas.

`canvas centerTo: aPoint`

`aPoint` : Coordonnées du point au centre de la fenêtre du canevas

Action : L'origine du canevas est modifiée afin d'afficher le point donné en argument au centre de la fenêtre du canevas.
Exemple :
`canvas centerTo: 5@0.`

`canvas scale: anInteger`

`anInteger` : Échelle du canevas. Une unité représente approximativement 1 pixel.
Action : modifie l'échelle du canevas
Exemple :
`canvas scale: 10.`

Point.

`<point> canvas point: aPoint`

`aPoint` : un couple de coordonnées (x,y)
⟶ référence d'un point libre du plan de coordonnées initiales `aPoint`
Exemple :
`canvas point: 5@2.`

`<point> canvas pointX: v1 Y: v2`

`v1` : un objet valeur
`v2` : une objet valeur
⟶ référence d'un point contraint par ses coordonnées
Exemple :
`canvas pointX: (canvas freeValue: 2) hide Y: (canvas freeValue: 5) hide.`

`<point> canvas pointOnCurve: curve at: abscissa`

`curve` : référence d'une ligne (droite, demi-droite, segment, etc.)
`abscissa` : abscisse curviligne du point libre, la valeur appartient à l'intervalle [0 ; 1]
⟶ référence d'un point libre sur une ligne
Exemple :
`myPoint := canvas pointOnCurve: s1 at: 0.5.`

`<point> canvas middleOf: p1 and: p2`

`p1` : référence d'un point ou d'un couple de coordonnées
`p2` : référence d'un point ou d'un couple de coordonnées
⟶ référence du milieu des deux points
Exemple :

```
| a i |
a := canvas point: 1@1.
i := canvas middleOf: a and: 4@4.
```

`<point> canvas middleOf: s`

`s` : référence d'un segment
⟶ référence du milieu du segment
Exemple :
`canvas middleOf: s.`

`<point> canvas intersectionOf: l1 and: l2`

`l1` : référence d'une ligne
`l2` : référence d'une ligne
⟶ référence du point d'intersection des deux lignes
Exemple :

```
canvas intersectionOf: droite and: segment
```

`<point> canvas altIntersectionOf: l1 and: l2`

`l1` : référence d'une ligne
`l2` : référence d'une ligne
⟶ référence de l'autre point d'intersection des deux lignes, lorsqu'il existe
Exemple :

```
canvas altIntersectionOf: droite and: circle.
```

`<point> canvas point: bloc parent: item`

`bloc` : bloc de code retournant un point
`item` : référence d'un item géométrique
⟶ référence d'un point dont les coordonnées sont calculées avec le bloc de code ayant comme argument item
Exemple :

```
| figure s mobile c block |
figure := DrGeoCanvas new.
s:=figure
   segment: (figure point: -5@0)
   to: (figure point: 5@0).
mobile := figure pointOnCurve: s at: 0.1.
block := [:mathItem | |x|
   x := mathItem point x.
   x @ (x * x * x / 25 - x)].
c := figure point: block parent: mobile.
figure locusOf: c when: mobile.
```

`<point> canvas point: bloc parents: itemCollection`

`bloc` : bloc de code retournant un point
`itemCollection` : une collection d'items géométriques
⟶ référence d'un point dont les coordonnées sont calculées avec le bloc de code ayant comme argument itemCollection
Exemple :

```
| figure a b d m p |
figure:=DrGeoCanvas new.
a:=figure point: (-2)@1.
b:=figure point: 3@3.
d:=figure line: a to: b.
d color: Color blue.
m:=figure point: 1@(-1).
p:= figure
   point: [:parents | parents first closestPointTo: parents second point]
   parents: {d . m}.
```

Droite.

`<line> canvas line: p1 to: p2`

`p1` : référence d'un point ou d'un couple de coordonnées
`p2` : référence d'un point ou d'un couple de coordonnées
$\longrightarrow$ référence d'une droite passant par deux points
Exemple :

```
| p1 |
p1 := canvas point: 0@0.
canvas line: p1 to: 1@2.
```

`<line> canvas parallel: d at: p`

`p` : référence d'un point ou d'un couple de coordonnées
`d` : référence d'une direction (droite, segment, vecteur, ...)
$\longrightarrow$ référence d'une droite parallèle à la direction de `d` et passant par `p`
Exemple :

```
| a |
a := canvas point: 1@5.
canvas parallel: d at: a.
```

`<line> canvas perpendicular: d at: p`

`p` : référence d'un point ou d'un couple de coordonnées
`d` : référence d'une direction (droite, segment, vecteur, ...)
$\longrightarrow$ référence d'une droite perpendiculaire à la direction de `d` et passant par `p`
Exemple :

```
canvas perpendicular: d at: 1@5.
```

`<line> canvas perpendicularBisector: s`

`s` : référence d'un segment
$\longrightarrow$ référence de la médiatrice au segment `s`
Exemple :

```
canvas perpendicularBisector: (canvas segment: 0@0 to: 4@4)
```

`<line> canvas perpendicularBisector: p1 to: p2`

`p1` : référence d'un point ou d'un couple de coordonnées
`p2` : référence d'un point ou d'un couple de coordonnées
$\longrightarrow$ référence de la médiatrice du segment d'extrémités `p` et `p2`
Exemple :

```
canvas perpendicularBisector: 0@0 to: 4@4)
```

`<line> canvas angleBisector: a`

`a` : référence d'un angle géométrique défini par **trois points**
$\longrightarrow$ référence de la bissectrice de l'angle `a`
Exemple :

```
canvas angleBisector: angle
```

`<line> canvas angleBisectorSummit: a side1: b side2: c`

`a,b,c` : points définissant l'angle géométrique $\widehat{bac}$

⟶ référence de la bissectrice de l'angle $\widehat{bac}$
Exemple :
```
canvas angleBisectorSummit: 0@0 side1: 1@0 side2: 0@1
```

Demi-droite.

`<ray> canvas ray: o to: p`
`o` : référence d'un point ou d'un couple de coordonnées, origine de la demi-droite
`p` : référence d'un point ou d'un couple de coordonnées, point de la demi-droite
⟶ référence d'une demi-droite définie par son origine et un point
Exemple :
```
| a |
a := canvas point: 1@5.
canvas ray: 0@0 to: a.
```

Segment.

`<segment> canvas segment: p1 to: p2`
`p1` : référence d'un point ou d'un couple de coordonnées
`p2` : référence d'un point ou d'un couple de coordonnées
⟶ référence d'un segment défini par ses extrémités
Exemple :
```
| a |
a := canvas point: 5@5.
canvas segment: 10@10 to: a.
```

Cercle.

`<circle> canvas circleCenter: c to: p`
`c` : référence d'un point ou d'un couple de coordonnées, centre du cercle
`p` : référence d'un point ou d'un couple de coordonnées, point du cercle
⟶ référence d'un cercle défini par son centre et un point
Exemple :
```
| a |
a := canvas point: 1@5.
canvas circleCenter: a to: 10@4.
```

`<circle> canvas circleCenter: c radius: r`
`c` : référence d'un point ou d'un couple de coordonnées, centre du cercle
`r` : référence d'un item numérique ou d'une valeur numérique, rayon du cercle
⟶ référence d'un cercle défini par son centre et son rayon
Exemple :
```
| a r |
a := canvas point: 1@5.
r := canvas freeValue: 4.
canvas circleCenter: a radius: r.
canvas circleCenter: 4@4 radius: 5
```

Arc de cercle.

`<arc> canvas arc: p1 to: p2 to: p3`

`p1` : référence d'un point ou d'un couple de coordonnées, 1^{ere} extrémité de l'arc
`p2` : référence d'un point ou d'un couple de coordonnées de l'arc
`p3` : référence d'un point ou d'un couple de coordonnées, 2^{eme} extrémité de l'arc
$\longrightarrow$ référence d'un arc de cercle défini par ses extrémités et un point
Exemple :

```
| a b |
a := canvas point: 1@5.
b := canvas point: 0@5.
canvas arc: a to: b to: -1 @ -2.
```

`<arc> canvas arcCenter: O from: A to: B`

`O` : référence d'un point ou d'un couple de coordonnées, centre de l'arc
`A` : référence d'un point ou d'un couple de coordonnées, origine de l'arc
`B` : référence d'un point ou d'un couple de coordonnées, tel que l'angle de l'arc est $\widehat{AOB}$
$\longrightarrow$ référence d'un arc de cercle défini par son centre et l'angle $\widehat{AOB}$
Exemple :

```
| a b |
a := canvas point: 1@5.
b := canvas point: 0@5.
canvas arcCenter: a from: b to: -1 @ -2.
```

Polygone.

`<polygon> canvas polygon: collection`

`collection` : une liste de références de points ou de couples de coordonnées ; sommets du polygone
$\longrightarrow$ référence d'un polygone défini par ses sommets
Exemple :

```
| b |
b := canvas point: 1@3.
canvas polygon: {1@2. b. 0@0. d.}.
```

`<polygon> canvas regularPolygonCenter: center vertex: summit sides: number`

`center` : référence d'un point ou d'un couple de coordonnées, centre du polygone
`summit` : référence d'un point ou d'un couple de coordonnées, sommet du polygone
`number` : référence d'une valeur ou valeur, nombre de sommets du polygone
$\longrightarrow$ référence d'un polygone régulier défini par son centre, un de ses sommets, et son nombre de sommets
Exemple :

```
| b |
b := canvas point: 1@3.
canvas regularPolygonCenter: b vertex: 1@1 sides: 7.
```

Les transformations géométriques.
Les transformations géométriques permettent la construction des tranformés d'objets. Elles s'appliquent à des références d'objets de type point, segment, droite, demi-droite, vecteur, cercle, arc de cercle et polygone.

`<transformed item> canvas rotate: item center: centre angle: angle`

`item` : point, segment, droite, demi-droite, vecteur, cercle, arc-cercle, polygone
`objet` : référence de l'objet à transformer
`centre` : référence d'un point ou d'un couple de coordonnées, centre de la rotation
`angle` : référence d'un item valeur ou d'une valeur, angle de la rotation
⟶ référence de l'objet transformé
Exemple :

```
| c k l |
c := canvas point: 5@5.
k := 3.1415.
l := canvas line: 0@0 to: 5@5.
canvas rotate: l center: c angle: k.
canvas rotate: l center: 0@0 angle: Float pi / 3.
```

`<transformed item> canvas scale: item center: centre factor: k`

`item` : point, segment, droite, demi-droite, vecteur, cercle, arc-cercle, polygone
`objet` : référence de l'objet à transformer
`centre` : référence d'un point ou d'un couple de coordonnées, centre de l'homothétie
`k` : référence d'un item valeur ou d'une valeur, facteur de l'homothétie
⟶ référence de l'objet transformé
Exemple :

```
| c k l |
c := canvas point: 5@5.
k := -3.
l := canvas line: 0@0 to: 5@5.
canvas scale: l center: c angle: k.
canvas scale: l center: 0@0 angle: 5.
```

`<transformed item> canvas symmetry: item center: centre`

`item` : point, segment, droite, demi-droite, vecteur, cercle, arc-cercle, polygone
`centre` : référence d'un point ou d'un couple de coordonnées, centre de la symétrie
⟶ référence de l'objet transformé
Exemple :

```
| a |
a := canvas point: 4@2.
canvas symmetry: a center: 0@0.
```

`<transformed item> reflect: item axe: axe)`

`item` : point, segment, droite, demi-droite, vecteur, cercle, arc-cercle, polygone
`axe` : référence d'une droite, axe de la réflexion
⟶ référence de l'objet transformé
Exemple :

```
canvas reflect: polygon axe: d1.
```

`<transformed item> canvas translate: item vector: vecteur`

`item` : point, segment, droite, demi-droite, vecteur, cercle, arc-cercle, polygone

`vecteur` : référence d'un vecteur ou d'un couple de coordonnées
⟶ référence de l'objet transformé
Exemple :

```
| u a|
u := canvas vector: (canvas point: 1@1) to: (canvas point: 3@2).
a := canvas translate: (canvas point: 2@1) vector: u.
```

Exemple :

```
| u a|
a := canvas translate: (canvas point: 2@1) vector: 2@1.
```

Lieu d'un point.

`<locus> canvas locusOf: c when: m`

`m` : référence d'un point mobile sur une ligne
`c` : référence d'un point fixe dépendant du point `m`
⟶ référence d'un lieu
Exemple :

```
canvas locusOf: p when: mobile.
```

Vecteur.

`<vector> canvas vector: o to: e`

`o` : référence d'un point ou d'un couple de coordonnées, origine du vecteur
`e` : référence d'un point ou d'un couple de coordonnées, extrémité du vecteur
⟶ référence d'un vecteur
Exemple :

```
| b |
b := canvas point: 0@5.
canvas vector: b to: -1 @ -2.
```

`<vector> canvas vector: p`

`p` : référence d'un point ou d'un couple de coordonnées, coordonnées du vecteur
⟶ référence d'un vecteur
Exemple :

```
| p |
p := canvas point: 5@5.
canvas vector: p.
canvas vector: -5 @ -5
```

Nombre.

`<point> pointItem coordinates`

`pointItem` : un point géométrique
⟶ coordonnées (statiques) de pointItem
Exemple :

```
| c p |
p := canvas pointOnCurve: segment at: 0.5.
```

```
c := p coordinates.
c x
```

`<value> canvas abscissaOf: pointOrVector`

`pointOrVector` : un point ou un vecteur
⟶ abscisse (dynamique) de `pointOrVector`
Exemple :

```
| m x |
m := canvas middleOf: 10@5 and: 7@8.
x := canvas abscissaOf: m
```

`<value> canvas ordinateOf: pointOrVector`

`pointOrVector` : un point ou un vecteur
⟶ ordonnée (dynamique) de `pointOrVector`
Exemple :

```
| m x |
m := canvas middleOf: 10@5 and: 7@8.
x := canvas ordinateOf: m
```

`<value> canvas freeValue: v`

`v` : la valeur initiale du nombre
⟶ référence d'un nombre libre
Exemple :

```
canvas freeValue: (-1 arcCos).
```

`<value> canvas lengthOf: segment | circle | arc | vector)`

`segment|circle|arc|vector` : référence d'un segment, cercle, arc ou vecteur
⟶ référence d'un nombre, longueur de l'item
Exemple :

```
canvas lengthOf: v1.
```

`<value> canvas distance: line | point to: point2`

`line|point` : référence d'une ligne ou d'un point
`point2` : référence d'un point
⟶ référence d'un nombre, distance entre deux points ou un point et une droite
Exemple :

```
canvas distance: l1 to: a.
```

`<value> canvas slopeOf: line`

`line` : référence d'une droite
⟶ référence d'un nombre, pente de la droite
Exemple :

```
| p |
p := canvas slopeOf: d.
```

Angle.

`<value> canvas angle: a to: b to: c`

`a` : référence d'un point
`b` : référence d'un point, sommet de l'angle
`c` : référence d'un point
$\longrightarrow$ référence d'un angle géométrique $\widehat{abc}$
Exemple :

```
canvas angle: a to: b to: c
```

`<value> canvas angle: v1 to: v2`

`v1` : référence d'un vecteur
`v2` : référence d'un vecteur
$\longrightarrow$ référence d'un angle orienté formé par les deux vecteurs
Exemple :

```
| v1 v2 a |
v1 := canvas vector: a to: b.
v2 := canvas vector: a to: c.
a := canvas angle: v1 to: v2.
```

Équation.

`<equation> canvas equationOf: lineOrCircle`

`lineorCirlce` : référence d'une droite ou d'un cercle
$\longrightarrow$ référence d'une équation de la droite ou du cercle
Exemple :

```
| e d |
d := canvas line: 0@0 to: 15@13.
e := canvas equationOf: e
```

1.2.2 Modification d'attributs d'objets

Pour modifier les attributs d'un objet déjà créé, nous utilisons un système de messages envoyés directement à l'objet. La modification des attributs se fait donc toujours à posteriori.

`item color: aColor`

`item` : référence d'un objet
`aColor` : une instance de `Color`, voir ses méthodes de classe pour des définitions existantes : `Color black, Color red, Color blue, Color orange, Color yellow,...`
Action : modifie la couleur d'un item
Exemple :

```
pointA color: Color green.
```

`item name: aString`

`aString` : une chaîne de caractères
Action : renomme un item
Exemple :

```
segment name: '[AB]'.
```

`item hide`
Action : masque un item

`item show`
Action : montre un item

`line small`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne une épaisseur fine à une ligne
Exemple :
`circle small.`

`line normal`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne une épaisseur normale à une ligne
Exemple :
`arc normal.`

`line large`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne une épaisseur large à une ligne
Exemple :
`polygon large.`

`line plain`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne un style ligne continue
Exemple :
`polygon plain.`

`line dashed`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne un style en tirets à une ligne
Exemple :
`polygon dashed.`

`line dotted`
`line` : référence d'une ligne (droite, demi-droite, cercle, lieu, etc.)
Action : donne un style en pointillés à une ligne
Exemple :
`arc dotted.`

`point cross`
`point` : référence d'un point
Action : donne une forme en croix à un point

Exemple :
```
a cross.
```

`point round`
`point` : référence d'un point
Action : donne une forme en rond à un point
Exemple :
```
a round.
```

`point square`
`point` : référence d'un point
Action : donne une forme carrée à un point
Exemple :
```
a square.
```

`point small`
`point` : référence d'un point
Action : donne une petite taille à un point
Exemple :
```
a small.
```

`point large`
`point` : référence d'un point
Action : donne une taille large à un point
Exemple :
```
a large.
```

`item moveTo: point`
`item` : référence d'un point ou d'une valeur
`point` : couple de coordonnées
Action : déplace l'item à la position donnée, pour peu que cela ait un sens
Exemple :
```
| a |
a := canvas point: 0@0.
a moveTo: 5@5.
canvas update
```

1.2.3 Méthodes complémentaires

La classe `DrGeoCanvas` propose dans la catégorie `helpers` des méthodes supplémentaires pour faciliter la réalisation de figures interactives complexes.

`canvas plot: aBlock from: x0 to: x1`
`aBlock` : un bloc de code à un argument décrivant une fonction
`x0` : nombre, abscisse inférieure de la courbe
`x1` : nombre, abscisse supérieure de la courbe
Action : affiche la courbe représentative de la fonction décrite par le bloc de code de x0 à x1
Exemple :

```
| canvas |
canvas plot: [:x| x * x] from: -3 to: 3.
```

`<block> canvas float: float1 at: aPoint from: float2 to: float3 name: aString`

`float1` : valeur initiale
`aPoint` : position du bord gauche de la réglette
`float2` : valeur minimum
`float3` : valeur maximum
`aString` : nom de la valeur
⟶ un bloc de code retournant la valeur courante de la réglette
Action : construis une réglette à la position indiquée avec une plage de valeur dans [float2 ; float3]
Exemple :

```
A := canvas float: 1 at: -10@4 from: 0 to: 10 name: 'A'.
F := canvas integer: 3 at: -10@3 from: 0 to: 10 name: 'F' showValue: true.
A value + F value.
```

Il existe d'autres variantes, dont certaines pour des nombres entiers.

1.3 Galerie d'exemples

Pour illustrer l'utilisation des Figures Smalltalk Dr. Geo, nous vous proposons une petite série d'exemples. Ceux-ci vous montrent leurs importantes possibilités et nous espérons qu'ils seront également une source d'inspiration. Pour chacun de ces exemples, nous donnons le code source Smalltalk de la figure puis son résultat. Le code source doit être copié dans un workspace de l'environnement de Dr. Geo puis exécuté.

1.3.1 Animer une figure

Ces exemples s'appuient sur la gestion du temps et celle des processus programmés en Smalltalk.

Un premier exemple simple pour comprendre le principe :

```
| figure p pause |
figure:=DrGeoCanvas new.
p := figure point: 0@0.
pause := Delay forSeconds: 0.2.
figure do: [
   100 timesRepeat: [
      p mathItem moveTo: (p mathItem point + (0.1@0)).
      figure update.
      pause wait]]
```

Un deuxième exemple avec une figure plus élaborée :

```
| figure s r u pause |
figure := DrGeoCanvas new fullscreen.
s := figure segment: 0@ -1 to: 4@ -1.
r := figure pointOnCurve: s at: 0.8.
s := figure segment: 0@0 to: 0@1.
u := figure pointOnCurve: s at: 0.7.
u round small; color: Color blue.
1 to: 100 do: [:n|
   u := figure
```

```
      point: [:parents| |y t|
         y := parents first point y.
         t := parents second point x.
         (n / 5) @ t * y * (1 - y)]
      parents:  {u . r}.
   u round small; color: Color blue].
pause := Delay forSeconds: 0.1.
figure do: [
   0 to: 1 by: 0.05 do: [:x |
      r mathItem setCurveAbscissa: x.
      figure update.
      pause wait]]
```

1.3.2 Triangle de Sierpinski

Cet exemple s'appuie largement sur un bloc de code récursif.

```
| triangle c |
triangle := [:s1 :s2 :s3 :n |
    c segment: s1 to: s2;
        segment: s2 to: s3;
        segment: s3 to: s1.
    n > 0 ifTrue:
        [triangle
            value: s1
            value: (c middleOf: s1 and: s2) hide
            value: (c middleOf: s1 and: s3) hide
            value: n-1.
        triangle
            value: (c middleOf: s1 and: s2) hide
            value: s2
            value: (c middleOf: s2 and: s3) hide
            value: n-1.
        triangle
            value: (c middleOf: s1 and: s3) hide
            value: (c middleOf: s2 and: s3) hide
            value: s3
            value: n-1.]].

c := DrGeoCanvas new.
triangle
    value: (c point: 0 @ 3)
    value: (c point: 4 @ -3)
    value: (c point: -4 @ -3)
    value: 3.
```

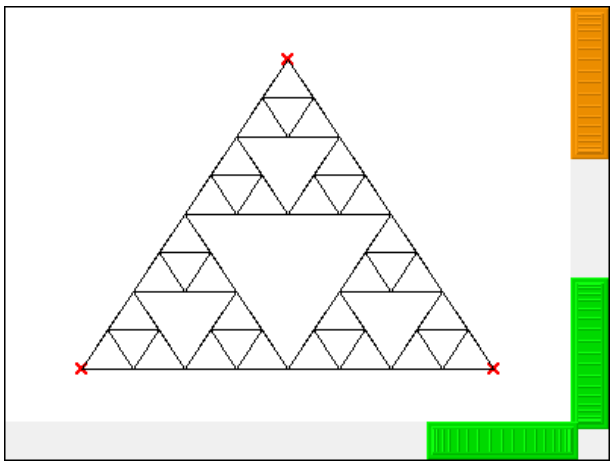

FIGURE 1.1 – Triangle de Sierpinski