Kids Programming with Smalltalk

Hilaire Fernandes

Department of Public Instruction
Geneva

November 2023

About me

- Educator in public school, Geneva, B.Math, Ma.Ed
- Computer scientist, PhD.CS
- Free software enthusiast and user since 1998
- And of course, Smalltalk user since 2002

Contents

Morphic 3

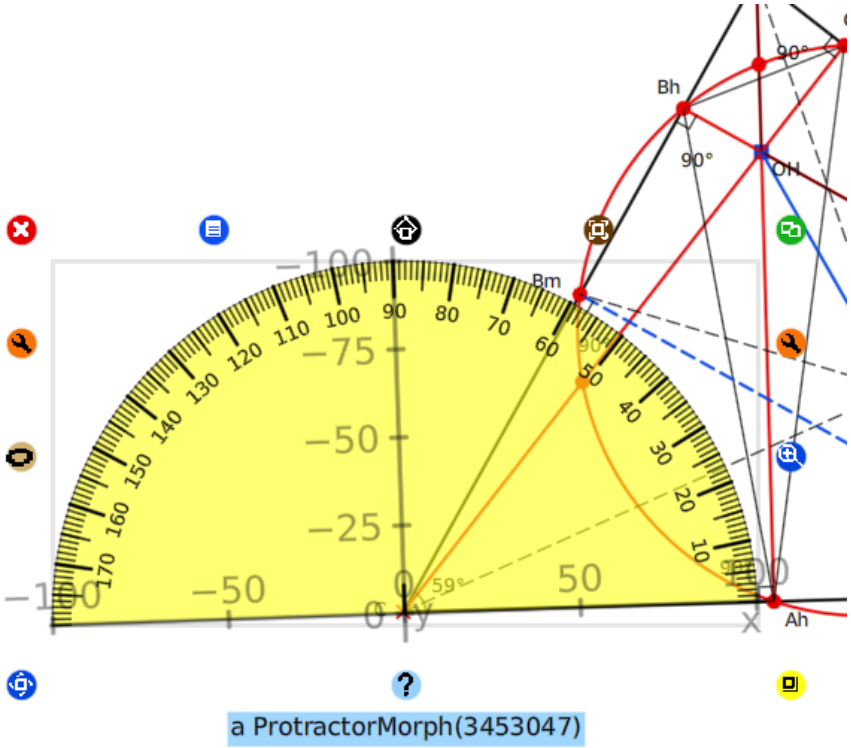
Mature, Simplified, Understandable and Vector quality !

```
DyViewerVisitor>>visitCourse: course
  | column |
  visitedModel := course.
  column := LayoutMorph newColumn.
  column
        addMorph: (self
            paneFor: course courseHour:
            label: 'Periods' translat
            browse: false);
        addMorph: (self
            paneFor: course teacher
            label: 'Teacher' translated
            browse: false);
        addMorph: (self
            paneFor: course topics
            label: 'Topics' translated
            browse: false).
  ↑ self plugView: column
```

Vector Graphics

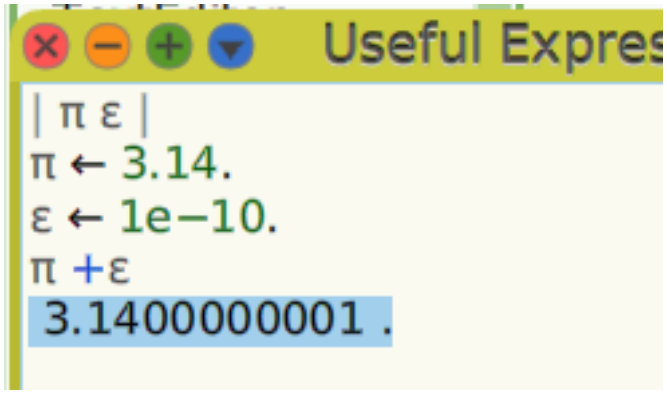
Mature too, API SVG compatible. Fast !

```
ProtractorMorph>>drawOn: canvas
|  p1 p2 |
canvas
    strokeWidth: 1
    color: Color black
    do: [
       canvas moveTo: 0 @ -0.5;
       lineTo: 0 @ -8].
-180 to: 0 do: [:degree |
   canvas strokeWidth: 0.5 color: Color b
   p1 := Point
      r: 100
      degrees: degree.
   p2 := Point
       r: 95
       degrees: degree.
   engine moveTo: p1 ; lineTo: p2]]
```



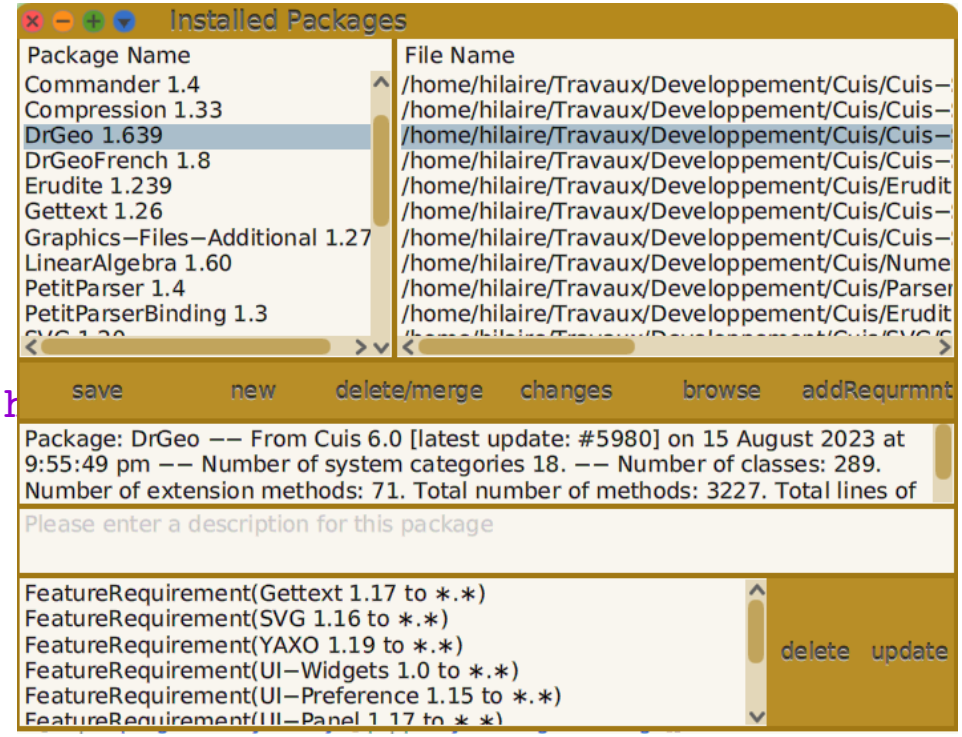
Unicode

- The default encoding for source code, text and text files
- Methods can be nammed with Unicode symbols
- Variables too !



Packaging System

```
"Install DrGeo code"
Feature require: #'DrGeo'.
Feature require: #'DrGeoFrench
```



Example of fine tuned end-user application with Smalltalk !

- Set up your development environment
- Spread your code in packages
- Use different code repository
- Localise your application
- Elaborate vector graphic user interface
- Develop your own widget
- Deliver end-user bundle

Interested to know more ?

⇒ Workshop Friday 16 :00 *Develop end-user GUI application with Cuis*

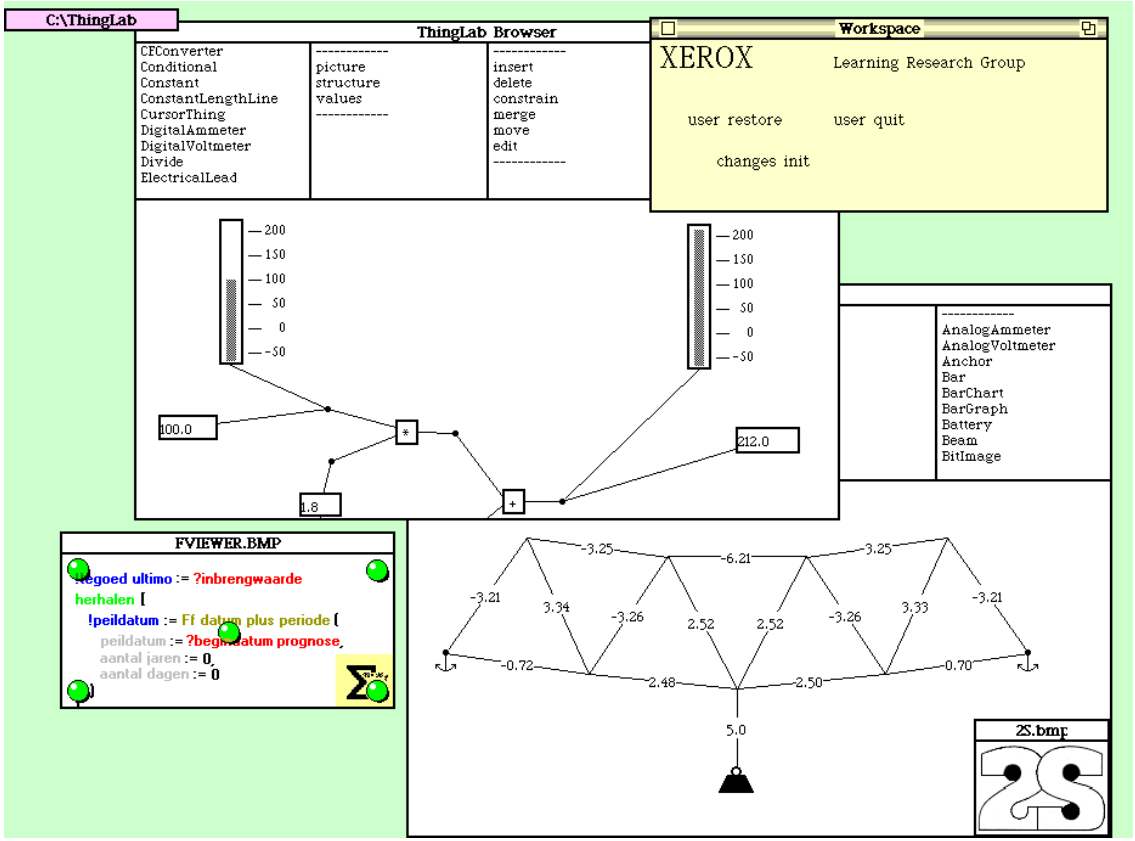
- Kids write code the old way
- Learning by the example
- Step-by-step introduction to programming concepts
- Do math as well !
- Smalltalk code in native language

Want to get Smalltalk back on track in school ?

⇒ Conference Wednesday 10 :00 *Revisiting the Dynabook concept*

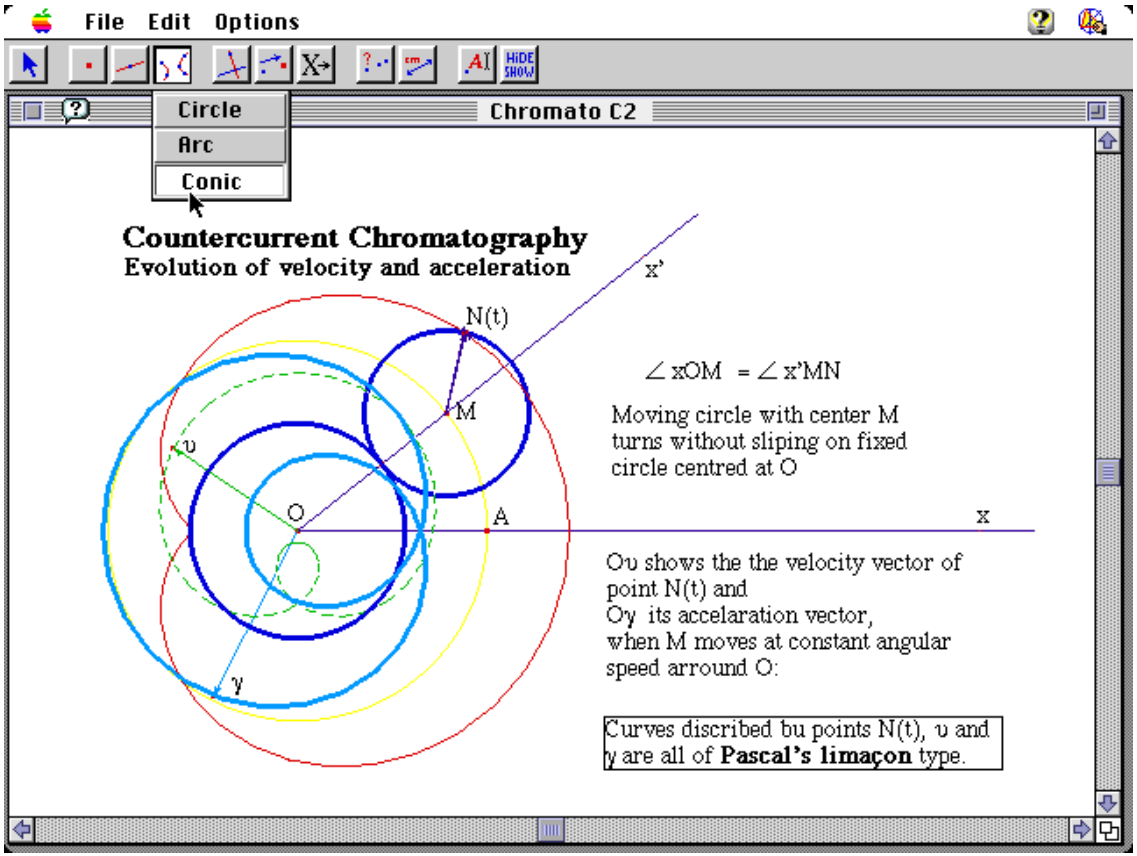
Thinglab, 1979[1]

A Smalltalk system that provides an object-oriented environment for building simulations [..] constraints are employed as a way of describing the relations among its parts.



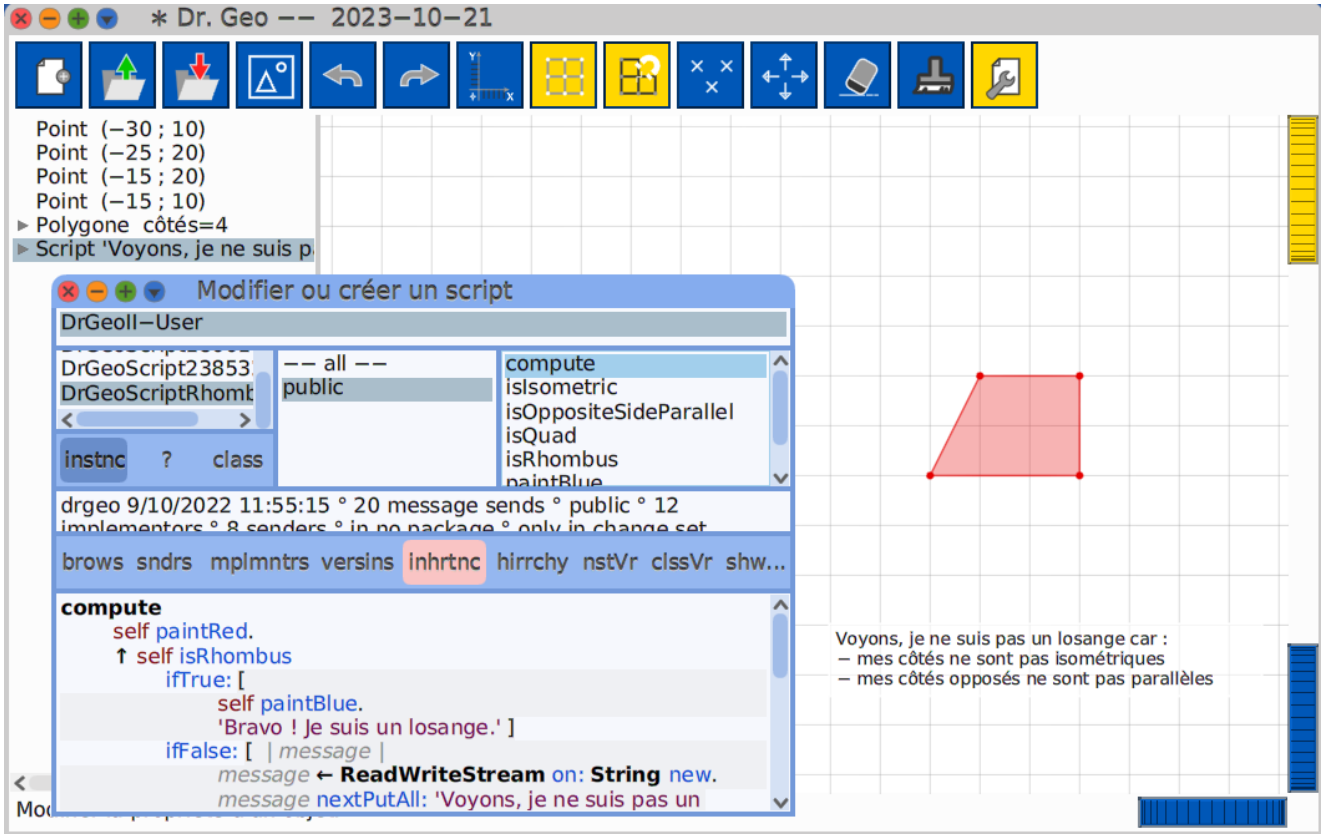
Cabri Geometer, 1986[4]

First interactive geometry system dedicated to education in mathematics.
A top-down approach to describe the relations among the parts.



Dr. Geo, 1998[2, 3]

First interactive geometry system for GNU/Linux, enhanced with a touch of end-user programming.



- A collection of math objects – items – in a child-parents relation
- A child depends on its parents.
  Example :
  1. **Child**. A point, *middle* of a segment
  2. **Parent**. A segment with liberty of movement.
  3. **Dragging child.** ✗ child stuck as a middle
  4. **Dragging parent.** ✓ child updated accordingly to keep its property of middle of the segment

DEMO

Demo contents

1. segment, middle → drag segment
2. segment (one extremity constrained), middle → drag segment, observe
3. segment, perpendicular bisector → reverse drag the line, observe
4. triangle with one vertex "A" on a circle → drag circle
5. elaborate previous with constructed altitude (3), construct H → what are the positions of "H"
6. elaborate previous with locus of "H" when "A"
7. Locus and script

Why describing a sketch with code ?

Point & Click is cool, it hides complexity.

Nevertheless :

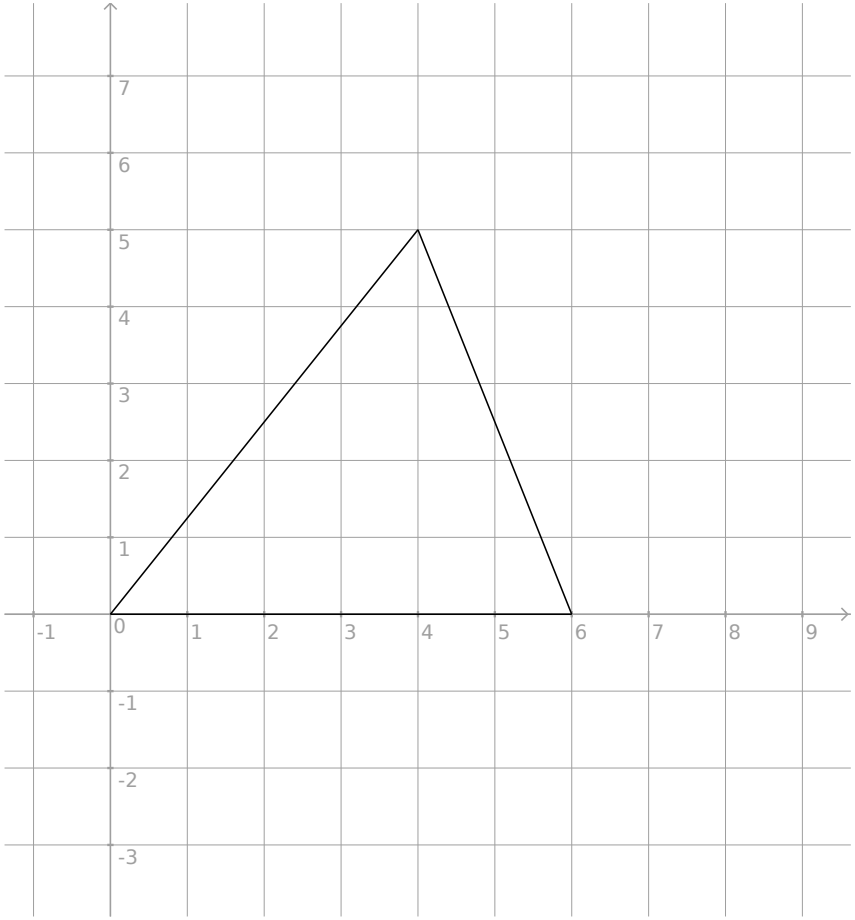
- We may want it (complexity) back, to elaborate on the math underneath i.e. coordinates system.
- Describe a sketch as a text. How can be described a segment ? Think of its mathematical nature.
- Capitalize on the programming features
- Growing in complexity, difficult to achieve with *Point & Click*.
  - Constructing hundred of items
  - Grouping items in collection to apply arbitrary transformation
  - ...

Declarative with sent messages[5]

Type-in program...

```
DrGeoSketch new
    axesOn;
    gridOn;
    segment: 0 @ 0 to: 6 @ 0;
    segment: 6 @ 0 to: 4 @ 5;
    segment: 4 @ 5 to: 0 @ 0.
```

...to get this sketch



Challenges & Glossary

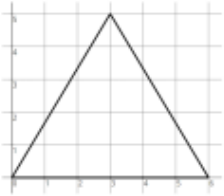
Leçon 1

Lorsque tu es satisfait de ton résultat, sauvegarde le code source du programme, coche la case et passe à la figure.

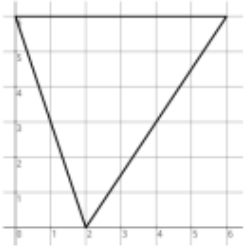
☐ **Triangle rectangle**

Lorsque tu es satisfait de ton résultat, coche la case et sauvegarde le code source sous le nom « Triangle rectangle ».

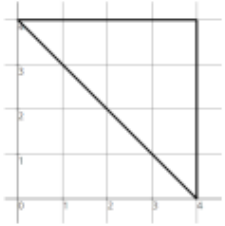
☐ **Triangle isocèle**



☐ **Triangle sur la tête**



☐ **Triangle rectangle isocèle**



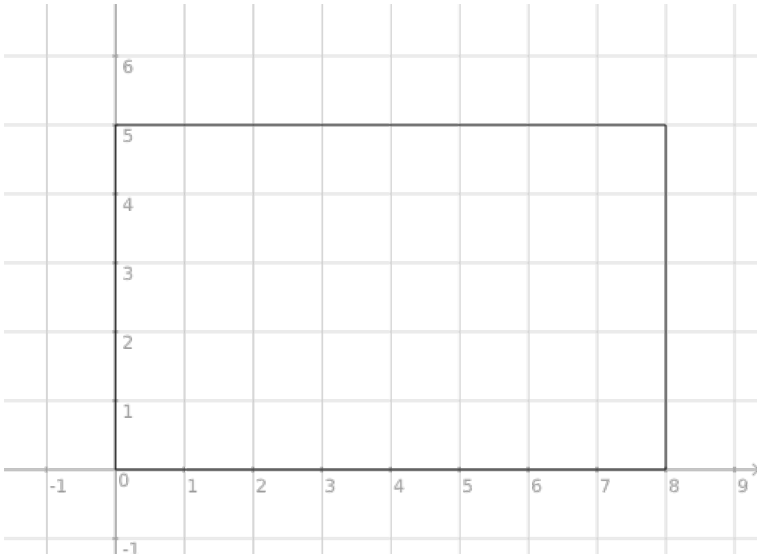
3. Glossaire

`DrGeoFigure nouveau`	→	Crée une nouvelle figure vide.
`afficherAxes` et `afficherGrille`	→	Messages envoyés à la figure pour afficher les axes et une grille.
`2 @ 3`	→	Point de coordonnées (2 ; 3), utilisé comme paramètre d'un message.
`segmentDe:à:`	→	Message envoyé à une figure pour créer un segment dont les extrémités sont des

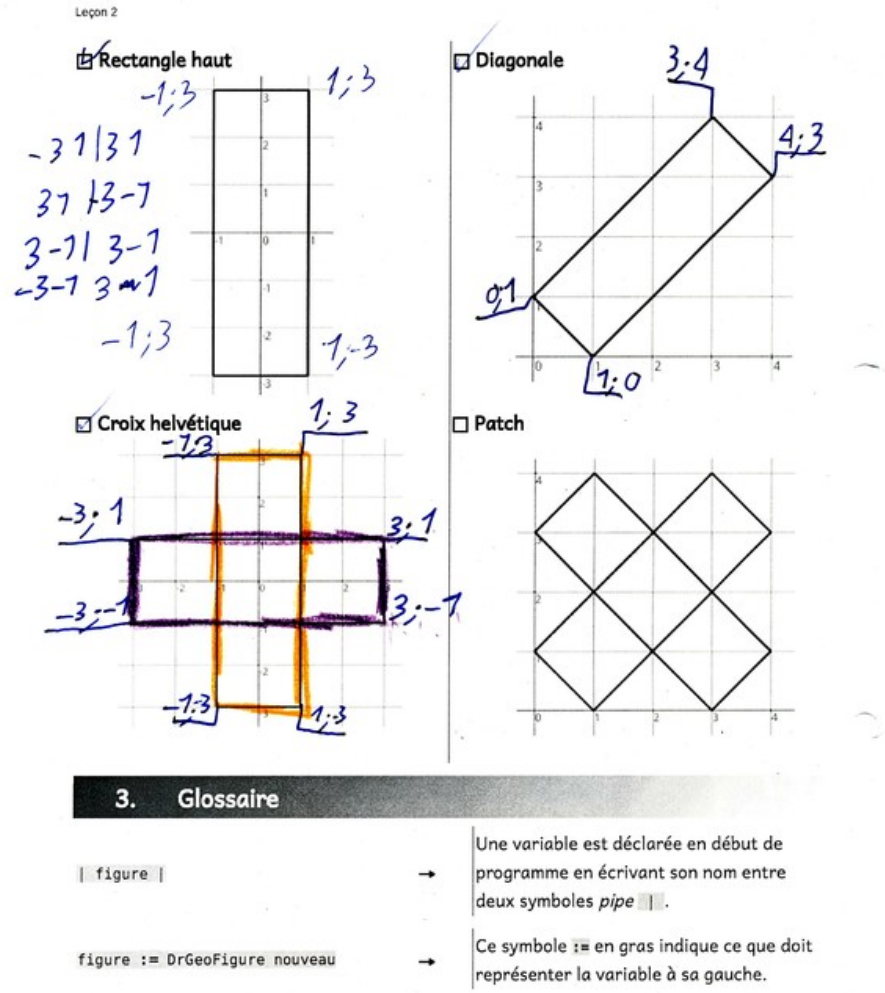
Introduce variable[6]

Type-in program...

```
| sketch |
sketch := DrGeoSketch new.
sketch axesOn; gridOn.
sketch segment: 0 @ 0 to: 8 @ 0.
sketch segment: 8 @ 0 to: 8 @ 5.
sketch segment: 8 @ 5 to: 0 @ 5.
sketch segment: 0 @ 5 to: 0 @ 0
```



...then do the challenges

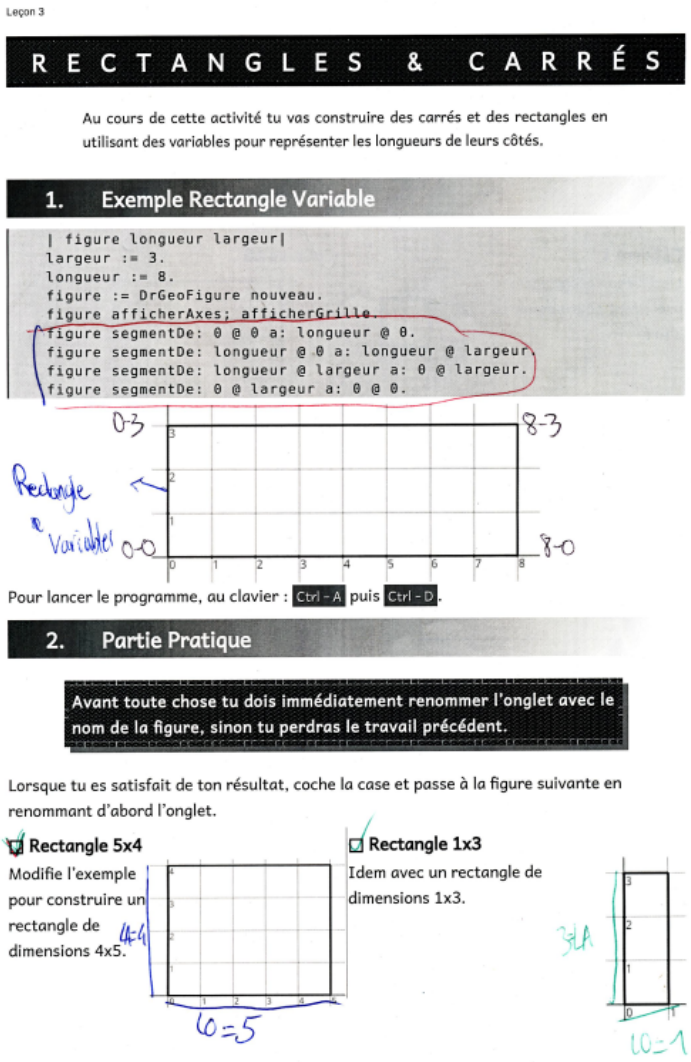


Use variables

Type-in program...

```
| sketch w h |
h := 3.
w := 8.
sketch := DrGeosketch new.
sketch axesOn ; gridOn.
sketch segment: 0 @ 0 to: w @ 0.
sketch segment: w @ 0 to: w @ h.
sketch segment: w @ h to: 0 @ h.
sketch segment: 0 @ h to: 0 @ 0.
```

...and kid analysis



Leçon 3

RECTANGLES & CARRÉS

Au cours de cette activité tu vas construire des carrés et des rectangles en utilisant des variables pour représenter les longueurs de leurs côtés.

1. Exemple Rectangle Variable

```
| figure longueur largeur|
largeur := 3.
longueur := 8.
figure := DrGeoFigure nouveau.
figure afficherAxes; afficherGrille.
figure segmentDe: 0 @ 0 a: longueur @ 0.
figure segmentDe: longueur @ 0 a: longueur @ largeur.
figure segmentDe: longueur @ largeur a: 0 @ largeur.
figure segmentDe: 0 @ largeur a: 0 @ 0.
```

Pour lancer le programme, au clavier : Ctrl - A puis Ctrl - D.

2. Partie Pratique

Avant toute chose tu dois immédiatement renommer l'onglet avec le nom de la figure, sinon tu perdras le travail précédent.

Lorsque tu es satisfait de ton résultat, coche la case et passe à la figure suivante en renommant d'abord l'onglet.

☑ Rectangle 5x4

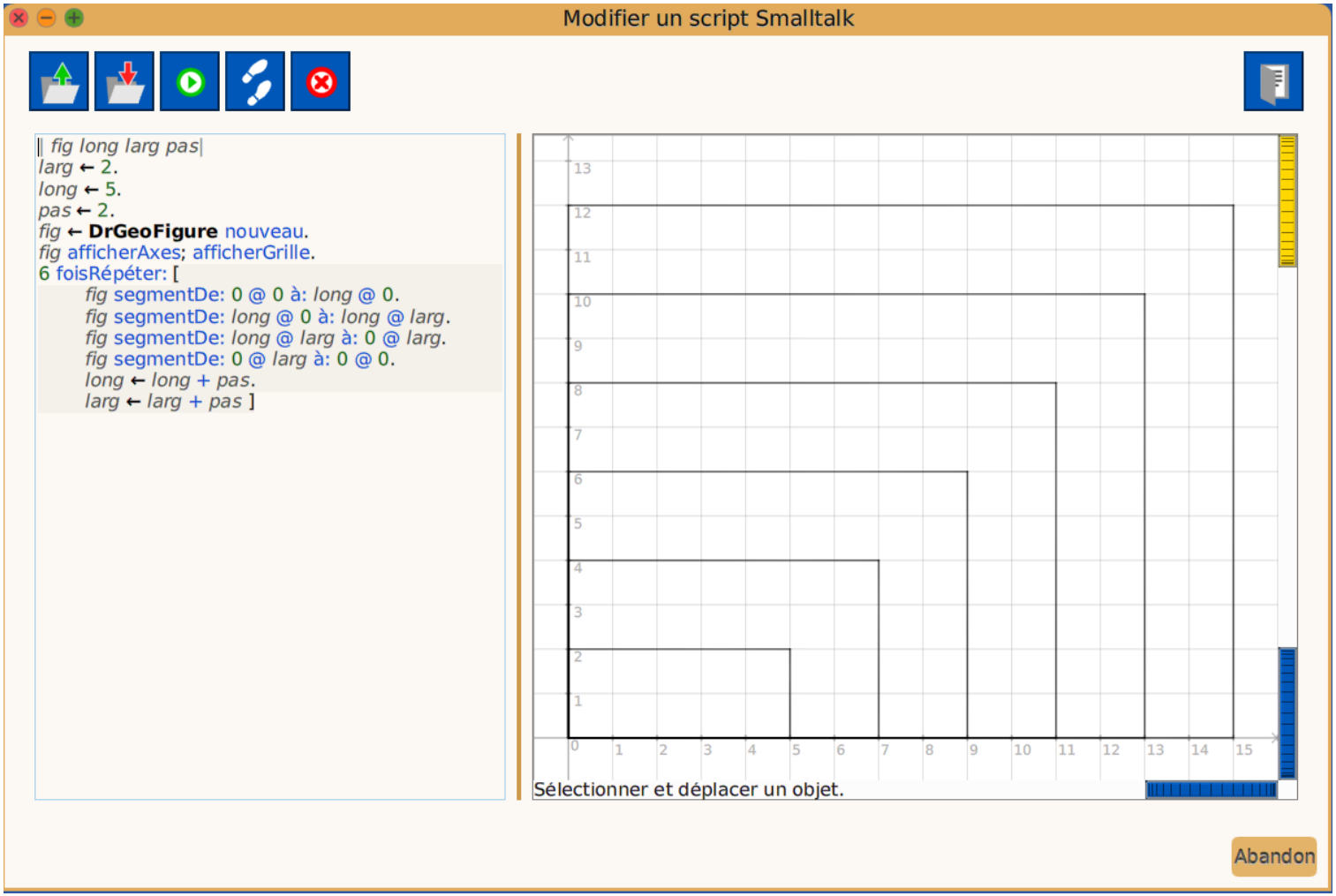
Modifie l'exemple pour construire un rectangle de dimensions 4x5.

☑ Rectangle 1x3

Idem avec un rectangle de dimensions 1x3.

Compute with loop

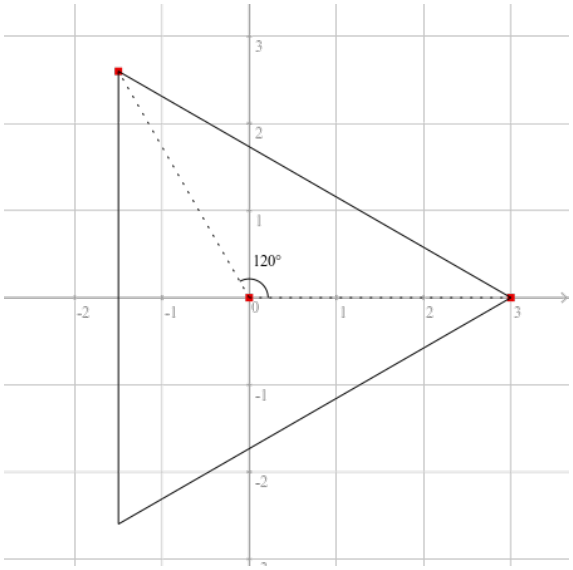
The Dr. Geo kid IDE



Explore mathematics

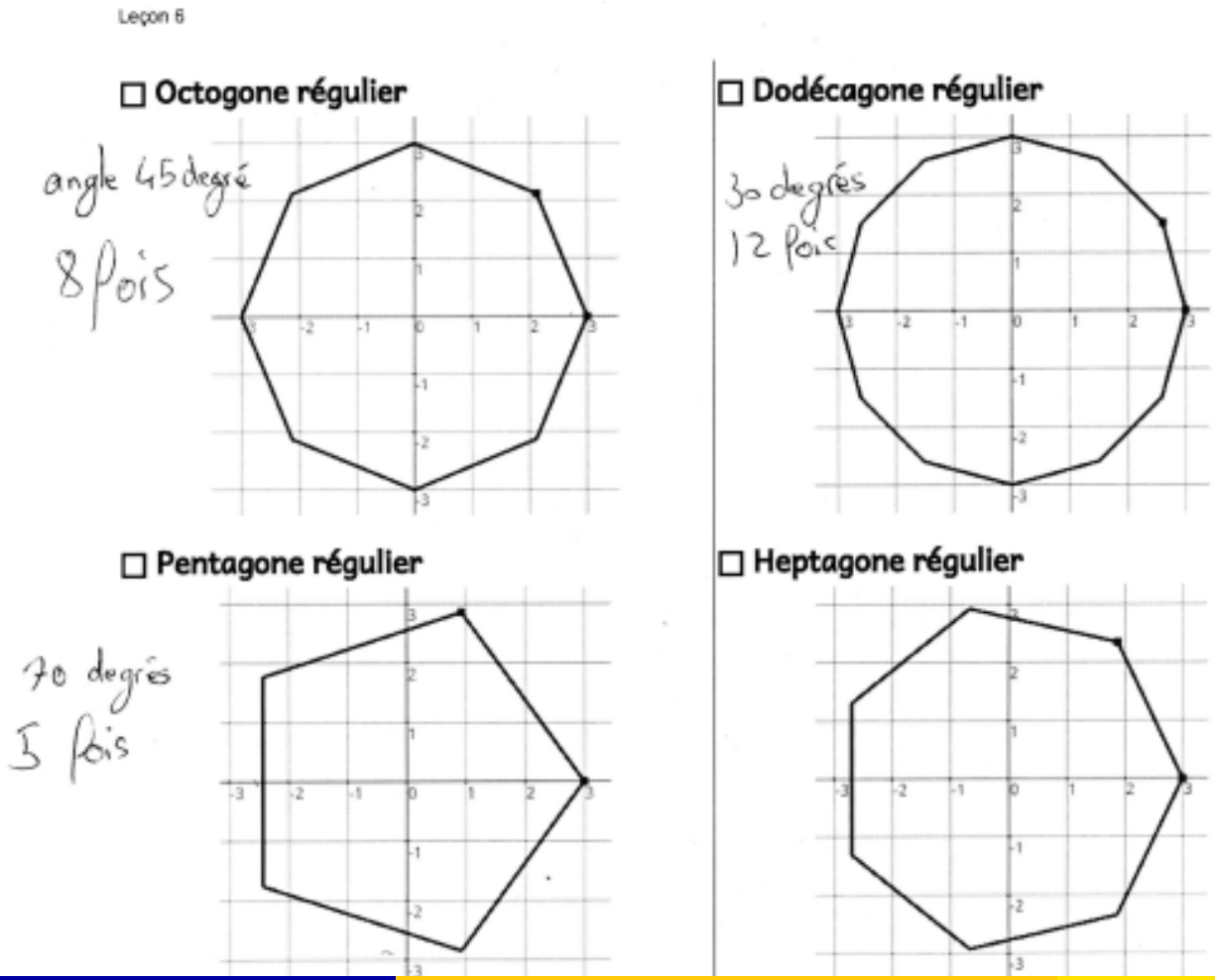
Regular polygon & transformation

```
| sketch ptA ptB seg angle |
sketch := DrGeoSketch new.
sketch axesOn; gridOn.
angle := 120.
ptA := sketch point: 3 @ 0.
ptB := sketch
    rotate: ptA
    center: 0 @ 0
    angleDegrees: angle.
seg := sketch segment: ptA to: ptB.
2 timesRepeat: [
    seg := sketch
        rotate: seg
        center: 0 @ 0
        angleDegrees: angle].
(sketch segment: 0@0 to: ptA) small; dotted.
(sketch segment: 0@0 to: ptB) small; dotted
```



Challenges of regular polygons

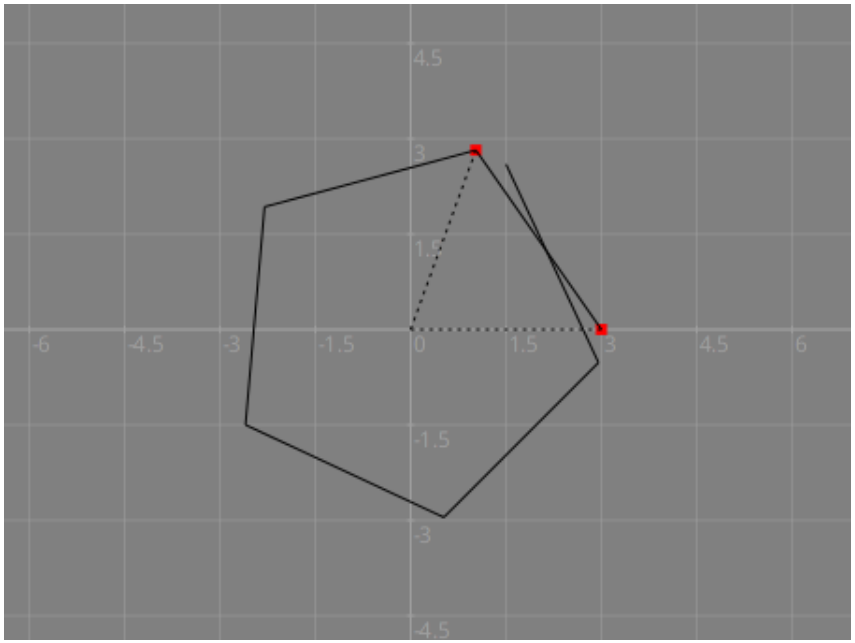
Measures with protractor...



Houston, we've had a problem here !

...mathematics to the rescue

```
| sketch ptA ptB seg angle |
sketch := DrGeoSketch new.
sketch axesOn; gridOn.
angle := 360 / 5.
ptA := sketch point: 3 @ 0.
```

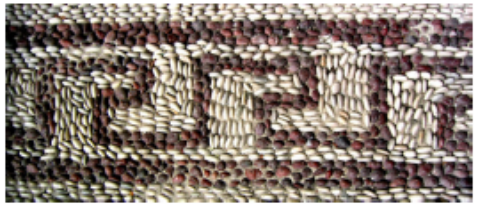


The benefit of collection

Leçon 9

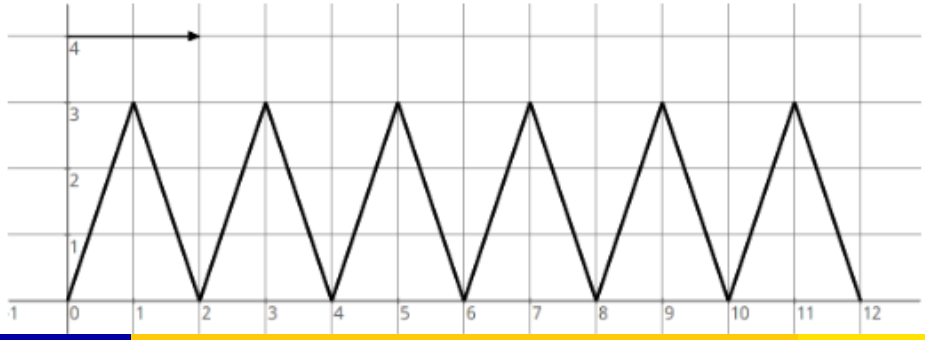
F R I S E 1

Connais-tu les frises ? Ce sont des motifs décoratifs qui se répètent, comme cette frise trouvée dans une rue de l'île grecque de Rhodes. A l'aide de translation, tu vas apprendre à les programmer.



1. Exemple Frise Triangle

```
| figure collection v |
figure ← DrGeoFigure nouveau.
figure afficherAxes; afficherGrille.
v ← figure vecteurOrigine: 0 @ 4 extrémité: 2 @ 4.
collection ← {figure segmentDe: 0 @ 0 à: 1 @ 3.
___figure segmentDe: 1 @ 3 à: 2 @ 0}.
5 foisRépéter: [
___collection ← collection collecter: [ :forme |
_____figure translationDe: forme parVecteur: v ] ]
```



- Design a DSL related to a taught domain, *Dr. Geo* DSL

  ⇒ vocabularies of the taught domain

- Makes DSL closes to the learner representations, *geometric idioms*

  ⇒ DSL in native language. Easy with Smalltalk.

- Learn from examples, *Human copies by-design !*

  ⇒ learner type-in code, do not elude this part

- Conceive challenges

  ⇒ progressive, challenge the learner domain knowledge (i.e pentagon and heptagon)

- Alan Borning. *Thinglab – A constraint Oriented Simulation Laboratory.* Xerox PARC, 1979.
- Hilaire Fernandes *GNU Dr. Geo.* Free Software Foundation, 1998-2013
- Hilaire Fernandes *A Brief History of GNU Dr. Geo.* Free Software Foundation, 1998
- Jean-Marie Laborde. Cabri history. Cabrilog, 2007
- Hilaire Fernandes *Programmer Géométrie - Leçon 1.* 2020, 2023
- Hilaire Fernandes *Programmer Géométrie - Leçon 2.* 2020, 2023