

# 9TE / PO Informatique

## Logique Informatique

Programmation avec

# SCRATCH



*Version 0.7 du 20 février 2011*



Scratch your World

ÉDUCATION NATIONALE  
Luxembourg

9TE / PO Informatique  
**Logique Informatique**

Programmation avec



<b>Introduction</b>	<b>4</b>
<b>A. L'environnement Scratch</b>	<b>4</b>
<b>A.1. L'interface du programme</b>	<b>4</b>
<b>A.2. Éléments principaux d'un projet Scratch</b>	<b>5</b>
A.2.1 La scène	5
A.2.2 Les sprites	5
A.2.3 Les instructions	5
A.2.4 Les programmes	6
<b>A.3. Modifier les sprites</b>	<b>7</b>
<b>A.4. Les sprites ont des costumes</b>	<b>9</b>
<b>A.5. La scène</b>	<b>11</b>
<b>A.6. Documentation du projet</b>	<b>12</b>
<b>B. Développer un projet Scratch à l'aide de séquences</b>	<b>13</b>
<b>C. Réagir à des événements</b>	<b>15</b>
<b>D. La boucle infinie</b>	<b>17</b>
<b>E. Prendre des décisions</b>	<b>19</b>
<b>F. Projet intermédiaire</b>	<b>23</b>
<b>G. Boucles à condition</b>	<b>24</b>
<b>H. Envoyer et recevoir des messages</b>	<b>26</b>
<b>I. Contrôle à l'aide de la boucle</b>	<b>29</b>
<b>I.1. Les palettes de blocs utilisées</b>	<b>30</b>
<b>I.2. Dessiner le carré</b>	<b>30</b>
<b>I.3. Dessin du premier mandala</b>	<b>31</b>
<b>I.4. Exercices pour approfondir</b>	<b>33</b>
<b>I.5. Quelques mandalas composés simples</b>	<b>34</b>
<b>J. Projet intermédiaire</b>	<b>35</b>
<b>K. Variable et expressions mathématiques</b>	<b>36</b>
<b>K.1. Variables</b>	<b>36</b>
K.1.1 Créer et afficher des variables	36
K.1.2 Travailler avec des variables	37
K.1.3 Les variables prédéfinies	40
<b>K.2. Les expressions</b>	<b>40</b>
K.2.2 Vrai ou faux	42
K.2.3 Pour avancés : conditions combinées	45

## Introduction

Scratch est un langage de programmation qui facilite l'apprentissage de la programmation de l'ordinateur. Il te permet de créer des histoires interactives et de développer des jeux et des animations.

Tu peux télécharger le programme gratuitement à la page web [scratch.mit.edu](http://scratch.mit.edu). Tu trouves également sur ce site une multitude d'informations sur Scratch (des modes d'emploi, des tutoriels vidéo, des cartes Scratch, la foire aux questions etc.). Et finalement tu peux t'inscrire sur ce site et publier tes propres projets Scratch.

Ce manuel va t'introduire étape par étape dans la programmation avec Scratch.

## A. L'environnement Scratch

Ce chapitre te permet de découvrir l'environnement dans lequel tu développeras tes projets Scratch.

### A.1. L'interface du programme



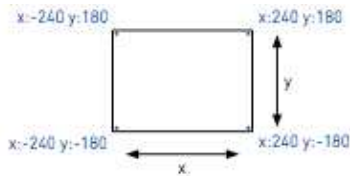
## A.2. Éléments principaux d'un projet Scratch

### A.2.1 La scène

La *scène* est le lieu où se déroulent les histoires, jeux et animations.

Les *sprites* (objets, voir plus loin) se déplacent sur une scène et agissent les uns avec les autres.

La scène a une largeur de 480 unités et une hauteur de 360 unités. Elle possède un système de coordonnées XY dont le point 0,0 se trouve au centre de la scène.



### A.2.2 Les sprites

Un projet Scratch est composé d'un certain nombre d'objets appelés *sprites* (r.d.t. : sprite veut dire littéralement ogre mais désigne ici un graphique numérique). On peut aussi appeler les sprites simplement **objets**.

Tu peux modifier l'aspect d'un sprite en lui donnant une apparence selon ton choix, p.ex. celle d'une personne, d'un train, d'un papillon etc.

Un sprite possède un nom. Scratch en donne automatiquement un (Objet1, Objet2...), mais pour que tu puisses mieux te repérer dans ton projet, il t'est conseillé de leur donner des noms significatifs.



### A.2.3 Les instructions

Tu peux faire exécuter des ordres à un sprite, lui dire de bouger, de jouer de la musique ou de le faire agir avec d'autres sprites. Pour communiquer une **instruction** à un sprite, tu dois faire glisser les **blocs graphiques** dans l'espace de programmation.

Ces blocs d'instructions sont regroupés dans des palettes d'instructions. Chaque instruction possède une couleur selon son type et se trouve dans la palette correspondante à son type. Ainsi toutes les instructions servant au déplacement d'un sprite ont la couleur bleu foncé et sont regroupées dans la palette **Mouvement**.

Voici les différentes palettes d'instructions :



En plus les différents blocs d'instructions se distinguent également par leur forme.

## A.2.4 Les programmes

Un programme est une suite d'instructions qui sont exécutées l'une après l'autre. Pour développer un programme Scratch tu dois assembler les blocs d'instruction nécessaires comme les pièces d'un puzzle. Un sprite peut avoir plusieurs programmes. Un autre mot pour un programme Scratch est **Script**.

Un double clic sur ton programme permet d'exécuter les blocs l'un après l'autre, du haut vers le bas.

Si tu n'as plus besoin d'un bloc d'instruction, il te suffit de le faire glisser de nouveau dans une palette d'instructions.

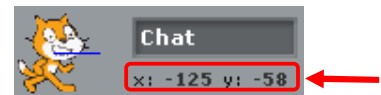
### Exercice A-01 Mon premier programme 🕒 15 min

#### Objectif :

Développer un programme Scratch.

#### Description

- Lance le programme Scratch.
- Renomme le sprite en lui donnant le nom de **Chat**.
- Fais glisser le bloc `avancer de 10 pas` vers l'espace de programmation. Effectue un double clic sur le bloc. Qu'est-ce qui se passe ? Observe la position du sprite sur la scène ! Explique !



---

---

---

- Développe le programme suivant :



- Lance le programme par un double clic.

- f) Arrête l'exécution du programme.
- g) Y a-t-il un autre moyen de lancer le programme ?

- 
- h) Modifie le mode de rotation du sprite et expérimente avec les différentes possibilités :

Explique les différents modes de rotation :



\_\_\_\_\_



\_\_\_\_\_



\_\_\_\_\_

Lequel des modes de rotation est le plus adapté pour ce projet ?

- 
- i) Sauvegarde le projet sous le nom de **Exercice A-01**.

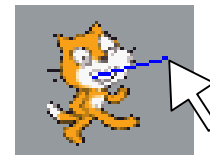
### A.3. Modifier les sprites

Pour **déplacer** un sprite à un autre endroit de la scène, il suffit de le glisser avec la souris en ayant le bouton de la souris appuyé.

Tu peux **dupliquer**, **supprimer** un sprite ainsi qu'**agrandir** et **réduire** sa taille : l'option choisie est effectuée par un clic dans la barre d'outils ci-contre.



La ligne bleue sur le sprite te permet de le faire **pivoter** à l'aide de la souris.



Il y a plusieurs possibilités pour **ajouter** un sprite au projet :



Dessiner soi-même le sprite

Charger un sprite enregistré

Sprite surprise

## Exercice A-02

## Créer d'autres sprites



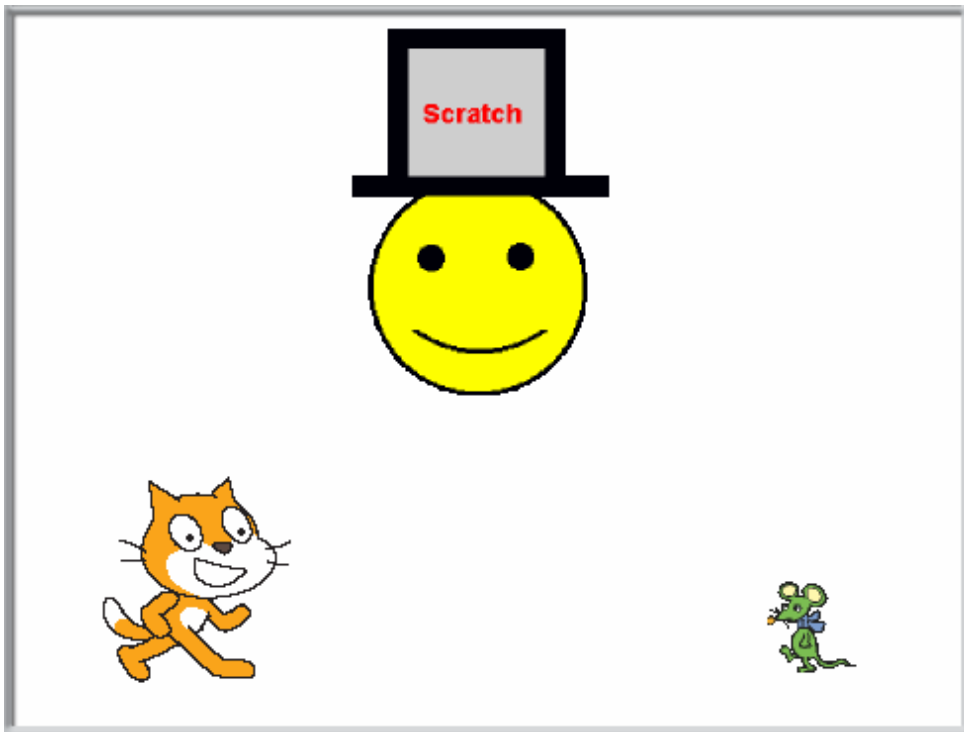
15 min

### Objectif :

Créer de nouveaux sprites dans un projet et les modifier.

### Description

- a) Ouvre le projet de l'**Exercice A-01**.
- b) Crée un nouveau sprite et place-le sur la scène comme indiqué sur le schéma ci-dessous. La souris peut être chargée d'un fichier, le smiley doit être dessiné dans l'éditeur graphique.
- c) Renomme les sprites.
- a) Sauvegarde le projet sous le nom de **Exercice A-02**.



## A.4. Les sprites ont des costumes

Un sprite peut posséder un ou plusieurs costumes qui lui permettent de changer d'apparence.

Tu peux voir les costumes d'un sprite quand tu sélectionnes d'abord le sprite et tu cliques ensuite sur l'onglet intitulé **Costumes**.



Le chat a p. ex. deux costumes. L'habit actuel est mis en évidence (**cat1-a**). Pour basculer vers un autre costume, il suffit de cliquer sur son aperçu.

Il y a trois possibilités pour créer un nouveau costume :

- Clique sur **Dessin** pour dessiner toi-même le costume dans l'éditeur Paint.
- Clique sur **Importer** pour charger une image graphique enregistrée sur le disque dur.
- Si ton ordinateur est équipé d'une webcam, alors clique sur **Caméra** pour photographier de nouveaux costumes.

Scratch est capable de traiter de nombreux formats graphiques : JPG, GIF, BMP et PNG.

Tu peux modifier l'ordre des costumes en les déplaçant à l'aide de la souris.

Enfin, tu peux retravailler un costume existant en cliquant sur **Édition**.

## Exercice A-03 Les costumes

🕒 10 min

**Objectif :** Modifier des costumes.

### Description

- Ouvre le projet de l'**Exercice A-01**.
- Charge le sprite **boy4-walking-a** à partir du disque dur et place-le sur la scène comme indiqué sur le schéma ci-dessous.
- Nomme le sprite **Garçon**.
- Combien de costumes le garçon possède-t-il ? \_\_\_\_\_
- Crée d'autres costumes en important les fichiers **boy4-walking-b**, **boy4-walking-c**, **boy4-walking-d** et **boy4-walking-e**.
- Copie le programme du sprite **Chat** dans le sprite **Garçon** (tire le programme de l'espace de programmation vers le sprite **Garçon** dans la liste des sprites comme indiqué sur le schéma).
- Lance maintenant les deux programmes (clic sur drapeau vert).
- Sauvegarde le projet sous le nom de **Exercice A-03**.



## A.5. La scène

De la même manière que les sprites changent d'apparence en basculant vers un autre costume, la scène peut changer d'apparence en basculant vers un autre **arrière-plan**.

Il est en plus possible de programmer la scène.

Pour modifier l'arrière-plan de la scène, clique sur l'icône scène qui se trouve à gauche de la liste des sprites.

### Exercice A-04 La scène – les arrière-plans

🕒 5 min

#### Objectif :

Modifier l'arrière-plan de la scène.

#### Description

- Ouvre le projet de l'**Exercice A-03**.
- Modifie l'arrière-plan comme affiché ci-dessous.
- Sauvegarde le projet sous le nom de **Exercice A-04**.



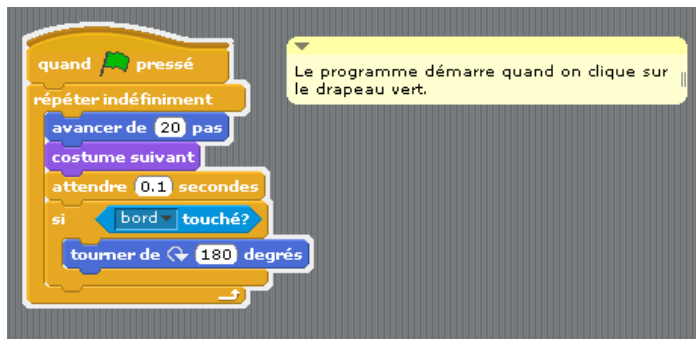
## A.6. Documentation du projet

Il est toujours conseillé de documenter son projet de manière suffisante. Ainsi quand tu voudras le modifier plus tard, une bonne documentation t'aidera à mieux t'y retrouver.

Il existe deux types de documentation.

- 1) Les **Notes** te permettent d'ajouter un petit texte à ton projet.  
Choisi le point menu **Fichier → Notes du Projet...** .  
Ici tu peux saisir une description globale de ton projet : par exemple ce que ton projet fait et comment l'utiliser.  
Remarque : Si tu veux publier ton projet alors ces notes de projet sont également affichées sur la page Internet de Scratch.
- 2) Les **Commentaires** te permettent d'annoter certaines parties de tes programmes. Cette documentation s'impose surtout pour des programmes complexes dans lesquels il est plus difficile de se retrouver.

Clique avec le bouton droit de la souris dans l'espace de programmation, choisis **ajouter un commentaire** dans le menu contextuel, puis saisis le commentaire.



### Exercice A-05 La documentation du projet 🕒 10 min

#### Objectif :

Documenter un projet

#### Description

- a) Ouvre le projet de l'**Exercice A-04**.
- b) Ajoute une note à ton projet.
- c) Essaie de comprendre comment fonctionne le programme du sprite **Chat** et saisis un commentaire pour chaque bloc d'instruction.
- d) Sauvegarde le projet sous le nom de **Exercice A-05**.

## B. Développer un projet Scratch à l'aide de séquences

La séquence est la forme la plus simple pour la structure d'un programme. Plusieurs instructions sont exécutées l'une après l'autre. Dans d'autres langages de programmation, ces instructions sont séparées p.ex. par un point-virgule. En plus, les instructions sont regroupées par des mots clés ou des symboles (p.ex. *begin* et *end* en Pascal).

Voici un exemple d'une séquence Scratch :

```

aller à x: -180 y: -81
pointer en direction -90
montrer
attendre 3 secondes
dire Ça va faire du travail ! pendant 2 secondes
attendre 3 secondes
penser à On l'espère... pendant 2 secondes
attendre 4 secondes
dire Je savais bien que ça allait faire du travail ! pendant 2 secondes
  
```

### Exercice B-01

#### Objectif :

Réaliser un projet à l'aide de séquences.

#### Description

Le chat doit :

- se rendre à la position  $x=0$ ,  $y=-100$
- regarder à gauche
- dire « Salut ! » pendant 2 secondes.
- Sauvegarde le projet sous le nom de **Exercice B-01**.

## Exercice B-02 « Le corbeau et le renard ! »

**Objectif :** Réaliser un projet à l'aide de séquences.

### Description

a) Développe un projet avec les sprites suivants (voir image). Tu peux dessiner le fromage toi-même, selon ton goût.

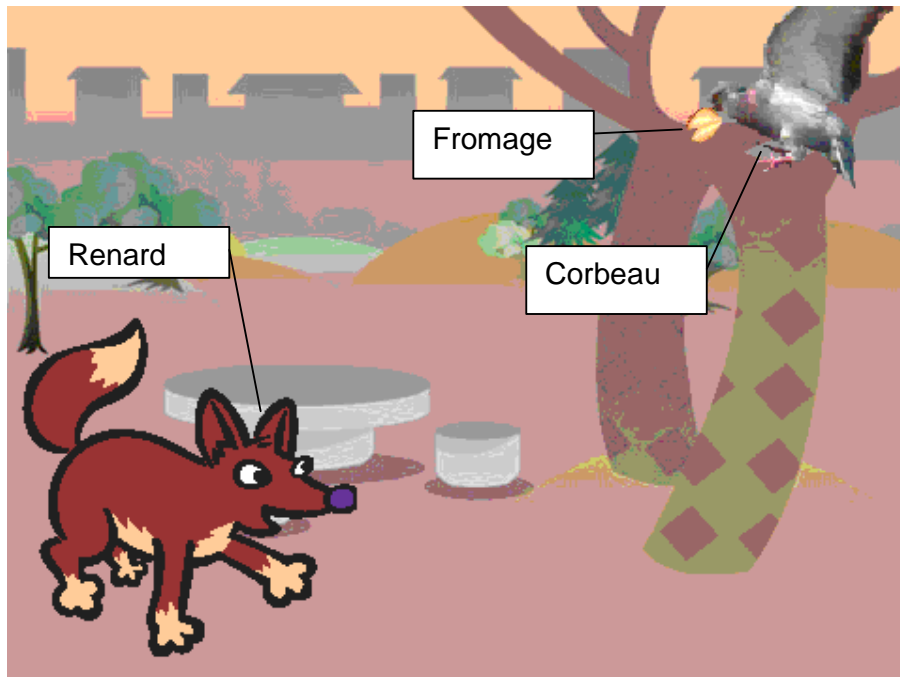
b) **Renard** doit réciter phrase par phrase le texte suivant :

*Et Bonjour Monsieur du Corbeau,*

*Que vous êtes joli ! Que vous me semblez beau !*

*Sans mentir, si votre ramage se rapporte à votre plumage*

*Vous êtes le phénix des hôtes de ces bois*



c) Le **Corbeau** dit ensuite le texte « Merci », après quoi le **Fromage** tombe par terre.

### Remarques :

- Réfléchis à l'avance comment tu peux réaliser l'ordre chronologique des actions des 3 sprites.
  - Le fromage tombe le mieux avec le mouvement ,glisse ...'.
  - Fais en sorte que le fromage se trouve de nouveau dans le bec du corbeau lors du prochain lancement du programme.
- d) Sauvegarde le projet sous le nom de **Exercice B-02**.

## C. Réagir à des événements

Les événements sont symbolisés par des blocs appelés 'chapeaux'. Ces blocs possèdent une bordure supérieure arrondie comme p.ex. :



Le programme accolé est exécuté, quand on clique sur le drapeau vert.



Le programme accolé est exécuté, quand on clique sur le sprite indiqué.



Le programme accolé est exécuté, quand on appuie sur la touche indiquée.

Un tel bloc forme le sommet d'une pile de blocs assemblés. Il attend qu'un certain événement ait lieu (p.ex. on appuie sur une touche indiquée). Lors de cet événement les blocs en-dessous du bloc chapeau sont exécutés.

### Exercice C-01

#### Objectif :

Réagir au clic de la souris

#### Description

- Crée un projet représentant une bouteille pleine et un verre vide. Vu que ce sprite n'existe pas dans la collection, il te faut le dessiner ou le charger d'Internet.
- Quand on clique sur le sprite, alors la bouteille se vide à fur et à mesure tandis que le verre se remplit. Une fois la bouteille vide, le tout recommence dès le début.



- Sauvegarde le projet sous le nom de **Exercice C-01**.

## Exercice C-02

### Objectif :

Réagir aux événements de clavier

### Description

- a) Crée un projet dans lequel le chat se déplace selon la touche flèche appuyée (gauche/droite/haut/bas).
- b) Sauvegarde le projet sous le nom de **Exercice C-02**.

## D. La boucle infinie

Dans cette unité, nous aborderons le bloc de contrôle intitulé „répéter indéfiniment. Il nous permet de répéter des actions indéfiniment. En termes de programmation, nous parlons de boucle infinie.



Si nous insérons les blocs d'instructions des exercices précédents dans cette boucle infinie, nous allons voir qu'ils seront exécutés jusqu'à ce qu'on clique sur l'icône stop rouge.

### Exercice D-01

#### Objectif :

Utiliser la boucle infinie.

#### Description

- a) Crée un projet avec le sprite suivant.



- b) La chauve-souris dispose de deux costumes qui permettent de la voir voler. Quand nous cliquons sur le drapeau départ nous pouvons l'observer voler sur la scène.
- c) Sauvegarde le projet sous le nom de **Exercice D-01**.

## Exercice D-02 La balle folle

### Objectif :

Utiliser la boucle infinie.

### Description

- a) Crée un projet avec un sprite qui représente une balle.



- b) La balle doit, de manière continue, tourner dans une direction aléatoire, avancer de 20 unités, rebondir du bord et attendre un 10<sup>e</sup> de seconde. Un clic sur la balle lui fait dire « OK » pendant une seconde.
- c) Sauvegarde le projet sous le nom de **Exercice D-02**.

## E. Prendre des décisions

Tu as certainement remarqué qu'il est souvent nécessaire, lors de la programmation de prendre une décision (comme dans la vie réelle). Ainsi ton sprite devra faire une chose ou une autre selon les circonstances qu'il rencontre.

Voici les blocs utilisés pour réaliser ce choix :



si une condition est vérifiée, exécute les instructions



si une condition est vérifiée, exécute les instructions,  
sinon exécute d'autres instructions

La condition doit être ou bien "vraie" ou bien "fausse" pour faire exécuter les instructions de la partie "si".

Voici deux types de conditions, il y en a encore d'autres types :



te permet de tester si un sprite ou le bord a été touché.



te permet de tester si une couleur a été touchée.

Tu trouves ces blocs dans la palette '**Capteurs**'.

Le sous-chapitre K.2.3 montre comment combiner des conditions.

## Exercice E-01 Cherche le repas !

🕒 10 min

### Objectif :

Le chat cherche son repas. Voici un exemple :



### Description :

- Le chat a très faim, (il est vorace). Pendant qu'il court dans la pièce il pense « Mais où est passé mon repas ? »
- Quand il trouve son repas le chat dit « Le voilà enfin ! » et le programme s'arrête.
- Sauvegarde le projet sous le nom de **Exercice E-01**.

### Extension :

Dès que tu maîtrises la technique de l'envoi de messages, tu peux laisser le chat manger son repas.

## Exercice E-02


## Free the crab !

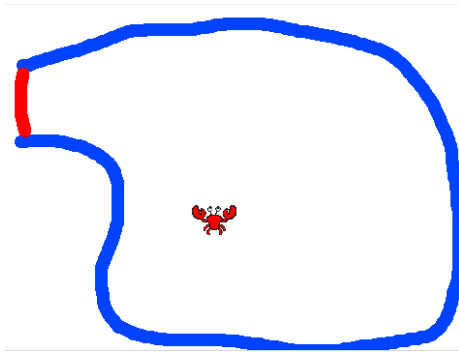
🕒 20 min

### Objectif :

Il faut libérer le crabe de sa cage.

### Description :

- Utilise le sprite **Crabe**  de la collection Scratch.
- Crée pour la scène un arrière-plan ("cage") qui possède un bord d'une couleur. La sortie est d'une deuxième couleur comme dans l'exemple ci-dessous :



- Le crabe essaie de s'échapper de sa cage. Quand il rencontre la couleur (bleue) du bord il pense « Hmmm... », se tourne un peu et essaie à nouveau. Quand il rencontre la couleur (rouge) de la sortie, il s'écrie « enfin libre » et le programme s'arrête.
- Sauvegarde le projet sous le nom de **Exercice E-02**.

### Extension pour avancés :

Au début du programme, le crabe doit se trouver au milieu de la cage et commencer sa recherche dans une direction aléatoire.

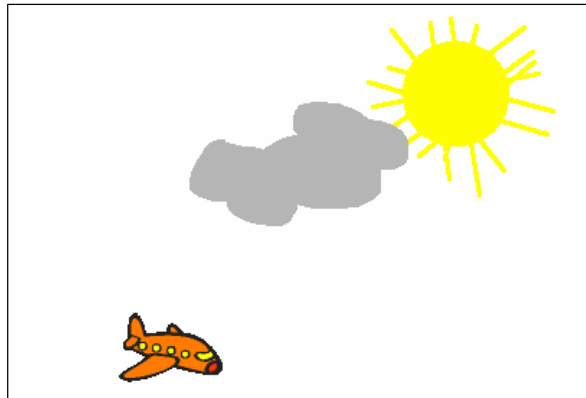
## Exercice E-03

## Chaud, plus chaud, brûlant...

🕒 20 min

### Objectif :

Un avion vole vers le haut et peut rencontrer un nuage ou le soleil et ... fondre.  
Voici un exemple :



### Description :

- Crée trois sprites : un (petit) **Avion**, un **Nuage** (gris) et un **Soleil** (jaune).
- L'avion pointe dans une direction quelconque et monte. Il rebondit du bord sans se tourner sur le dos.
- Lorsqu'il entre en collision avec le nuage, il atterrit et décolle à nouveau en changeant de direction (Il peut même changer de couleur).
- Lorsqu'il entre en collision avec le soleil, il fond et disparaît<sup>1</sup>. Le programme devrait alors s'arrêter.
- Sauvegarde le projet sous le nom de **Exercice E-03**.

### Extension :

Sachant comment programmer le déplacement des objets tu pourrais faire bouger le nuage devant le soleil. Programme le nuage de manière à ce qu'il se déplace lentement vers le soleil.

### Pour les experts uniquement :

Fais bouger le soleil sur l'arrière-plan dans un mouvement de demi-cercle.

---

<sup>1</sup> Pour réaliser cela tu peux modifier la taille de l'avion en étapes de 10 points, puis dire « Poufff ! » à la fin avant de disparaître complètement.



## F. Projet intermédiaire

## G. Boucles à condition

Jusqu'à ce point, tu t'es déjà beaucoup exercé en Scratch et tu as appris beaucoup de choses. Tu sais programmer des séquences d'instructions et des boucles infinies et tu sais utiliser des conditions.

Savais-tu qu'on pouvait parfois combiner les deux : boucles infinies et conditions ? En le faisant tu obtiens ceci :



La séquence intérieure sera exécutée tant qu'une condition est vérifiée. Si la condition n'est plus vérifiée, la boucle n'est plus exécutée. Dès qu'elle est de nouveau vérifiée, les instructions sont de nouveau exécutées. Il s'agit d'une boucle infinie, c.-à-d. le programme ne sort jamais de ce bloc.



Dans ce cas, le programme exécute d'abord la séquence intérieure et regarde ensuite si la condition est vérifiée.

Si c'est le cas, la boucle s'arrête et la programmation se poursuit avec l'instruction qui suit la boucle. Sinon, la boucle sera exécutée à nouveau, et ainsi de suite.

### Exercice G-01 Tourner en rond

🕒 5-10 min

**Objectif :**

Le chat doit courir pendant que la touche 'espace' est appuyée. Voici un exemple :



**Description :**

- Tant que la touche 'espace' est appuyée, le chat court de gauche à droite et de droite à gauche.
- Sauvegarde le projet sous le nom de **Exercice G-01**.

Que se passe-t-il si tu lâches la touche 'espace' ?

Que se passe-t-il si tu appuies la touche à nouveau ?

## Exercice G-02 Tourner en rond 2

🕒 10 min

### Objectif :

Le chat doit courir jusqu'à ce que la touche 'espace' soit appuyée.

### Description :

- Le chat court de gauche à droite et de droite à gauche jusqu'à ce que la touche 'espace' soit appuyée.
- Sauvegarde le projet sous le nom de **Exercice G-02**.

Que se passe-t-il si tu appuies la touche 'espace' ?

---

Que se passe-t-il quand tu lâches la touche ?

---

## Exercice G-03 Liberté au chat !

🕒 20 min

### Objectif : Libérer le chat de sa prison



### Description :

- Le chat court de gauche à droite et de droite à gauche (c.-à-d. il court tant qu'il touche la prison). Il change de direction en touchant le bord gris de la prison. Dessine toi-même le sprite de la prison.
- Pour avancés : Toutes les 10 secondes, il pense « Comment sortir d'ici ? » (Utiliser des expressions mathématiques, voir chapitre K)
- Quand on appuie sur la touche 'espace', la prison s'ouvre et le chat te remercie pour sa liberté.
- Sauvegarde le projet sous le nom de **Exercice G-03**.

### Extension :

Fais en sorte que la prison soit positionnée en avant-plan au lancement du programme.

Fais en sorte que le chat se trouve à l'intérieur de la prison au lancement du programme.

## H. Envoyer et recevoir des messages

Dans certaines situations, un sprite aimerait exécuter une action (programme) à un autre sprite.

Scratch le rend possible par l'envoi et la réception de messages.

Il existe deux instructions qui servent à envoyer un message :



Envoie un message à tous les sprites et la scène, ce qui peut les amener à exécuter une action. Le programme continue tout de suite son exécution sans attendre que les autres sprites aient fini leurs actions.

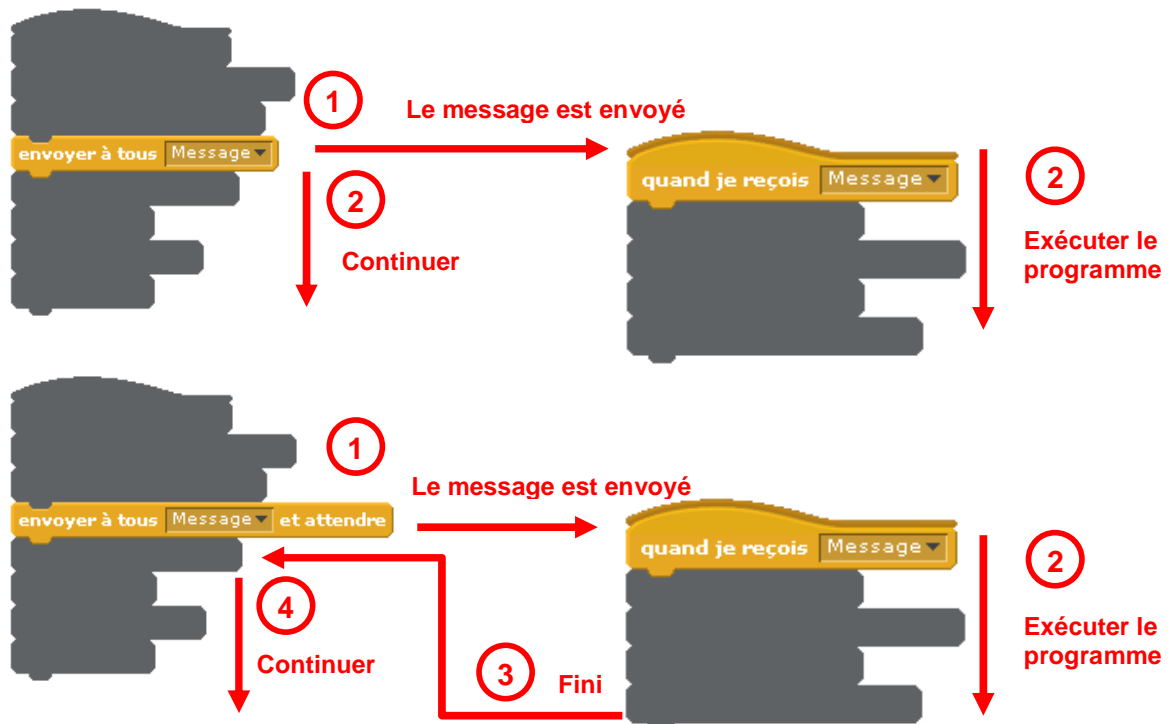


Envoie un message à tous les sprites et la scène, ce qui peut les amener à exécuter une action. Le programme attend de continuer son exécution jusqu'à ce que les autres sprites aient fini leurs actions.

Lorsqu'un un sprite ou la scène reçoit le message alors la séquence d'instructions

qui se trouve sous le bloc chapeau est exécutée.

Les schémas suivants illustrent la différence entre les deux instructions d'envoi :



## Exercice H-01

### « Vite dans la corbeille ! »

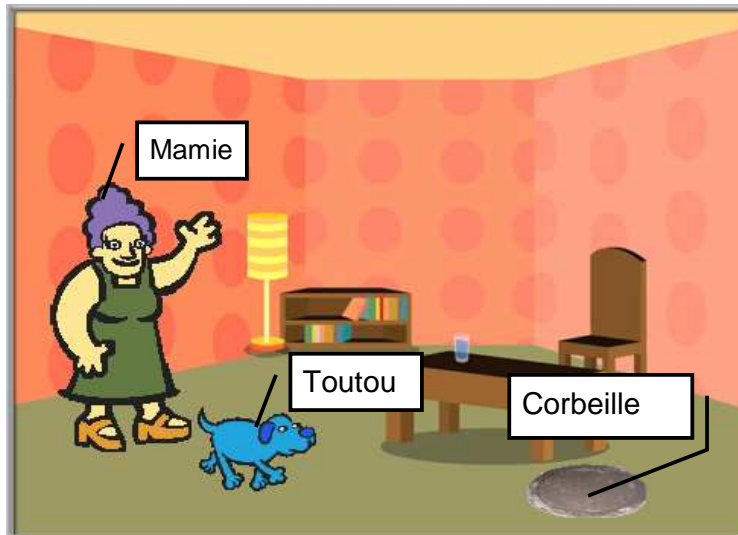
🕒 15 min

#### Objectif :

Envoyer et recevoir des messages.

#### Description

- a) Crée un projet avec les sprites suivants.



- b) Toutou devrait avoir au moins deux costumes pour le voir se dandiner.  
 c) Quand on clique sur la Mamie, alors elle dit « Vite dans la corbeille ! » et Toutou doit se rendre à sa corbeille.

Programme la **Mamie** et **Toutou** en utilisant les instructions suivantes.



- d) Sauvegarde le projet sous le nom de **Exercice H-01**.  
 e) Ajoute l'action suivante :  
 Quand Toutou arrive à sa corbeille alors la mamie dit « Bon Toutou ! »

## Exercice H-02

## Le chat qui court dans la maison

🕒 30 min

### Objectif :

Envoyer et recevoir des messages.

### Description

- Crée un projet avec le chat et importe pour la scène plusieurs arrière-plans représentant les différentes pièces de la maison.
- Le chat doit parcourir la maison, d'une pièce à l'autre. Il traverse chaque pièce de gauche à droite. Lorsqu'il arrive au bord droit d'une pièce, alors il apparaît dans la pièce suivante, au bord gauche.



- Sauvegarde le projet sous le nom de **Exercice H-02**.

## I. Contrôle à l'aide de la boucle

Dans ce chapitre, tu apprendras comment tu peux contrôler l'exécution répétée d'instructions à l'aide d'une boucle.

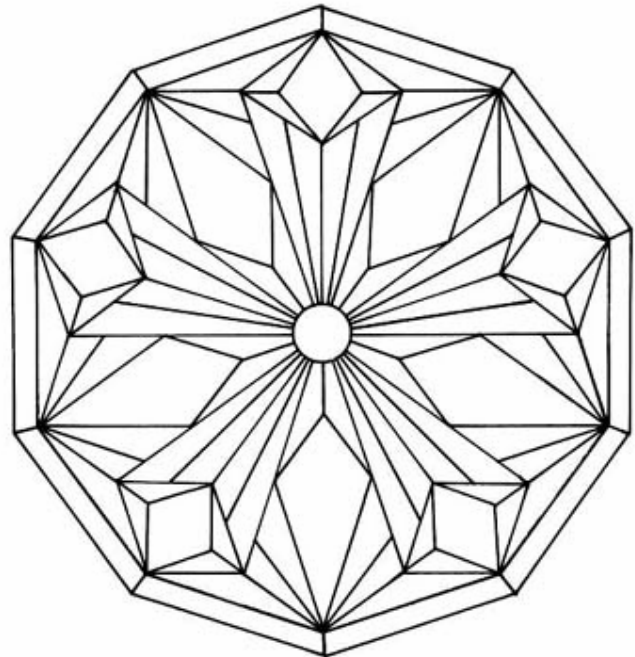
Pour réaliser cette Unité, il est important de rafraichir nos notions de géométrie (*calcul d'angles ; symétrie centrale ; rotation ; système de coordonnées, quatre quadrants*).

### Mandala

Le mot **Mandala** (du sanscrit) veut dire 'cercle' et désigne une forme symbolique circulaire ou carrée avec un centre. A l'origine il a été utilisé uniquement dans un contexte religieux (Wikipedia).

Aujourd'hui le mandala sert plutôt pour les feuilles de coloriage pour enfants et adolescents.

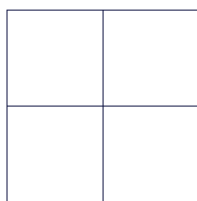
Dans cette unité tu programmeras un mandala... plus simple que l'exemple ci-contre.



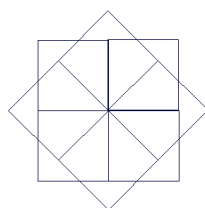
Le mandala ci-dessus est composé de parties égales (ici 5 ou plutôt 2x5) autour du centre.

Le mandala que nous allons réaliser maintenant est composé d'un bon nombre de carrés qui vont tourner autour du centre selon un angle à calculer

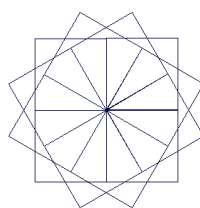
Exemple :



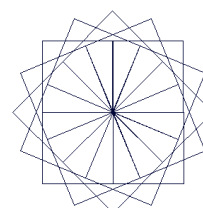
4 carrés



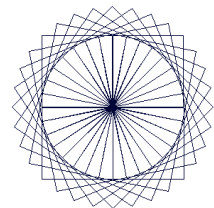
8 carrés



12 carrés



16 carrés



32 carrés

## I.1. Les palettes de blocs utilisées

---



## I.2. Dessiner le carré

---

Le carré est dessiné autour du centre de la scène (point 0;0).

Un carré possède ..... côtés ..... et ..... angles .....  
 ..... La somme des angles d'un carré est de ..... Un angle mesure  
 exactement ....., il s'agit d'un angle .....

Avant de dessiner un des carrés du mandala il est très important de calculer les angles qui sont utilisés pour ce carré. Dans ce but, étudie d'abord la feuille „Caractéristiques des polygones réguliers“ de l'**Exercice I-3**.

**Exercice I-01**
**Un carré**
🕒 **20 min**

- **Blocs d'instruction utilisés :**



Les arguments des blocs d'instruction doivent être adaptés à nos besoins !  
 Chaque bloc peut être utilisé plusieurs fois.

Condition de départ : le carré est dessiné quand on appuie sur la touche '1'.  
 La longueur d'un côté du carré est de 100.

Le stylo doit être positionné à la position (0;0).

Toutes traces d'un dessin précédent doivent être effacées.

## I.3. Dessin du premier mandala

Le mandala est composé de 8 carrés (voir l'exemple au début du chapitre).

Il faut d'abord calculer l'angle qui se trouve entre deux carrés qui se suivent afin de rendre le mandala symétrique.

- Pour mieux comprendre : mesure l'angle entre deux éléments de la mandala ci-dessus et vérifie la thèse suivante :

$$\text{Angle} = 360^\circ / \text{nombre d'éléments égaux}$$

Calcule maintenant l'angle pour ton mandala.

$$\text{Angle} = \dots\dots^\circ / \dots\dots = \dots\dots^\circ$$

### Exercice I-02

### Mon premier mandala

🕒 25 min

- Blocs d'instruction utilisés :



Les arguments des blocs d'instruction doivent être adaptés à nos besoins !

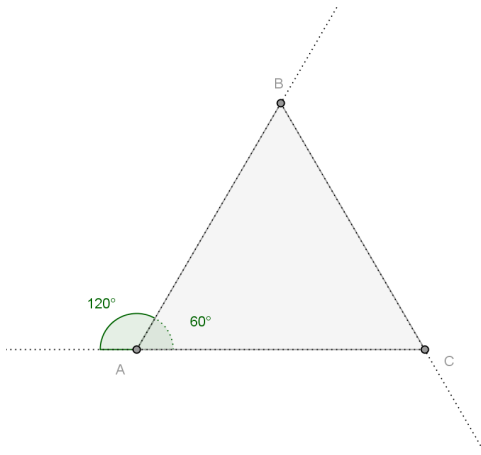
Condition de départ : le carré est dessiné quand on appuie sur la touche '8'.

Le stylo doit être positionné à la position (0;0).

Toutes traces d'un dessin précédent doivent être effacées.

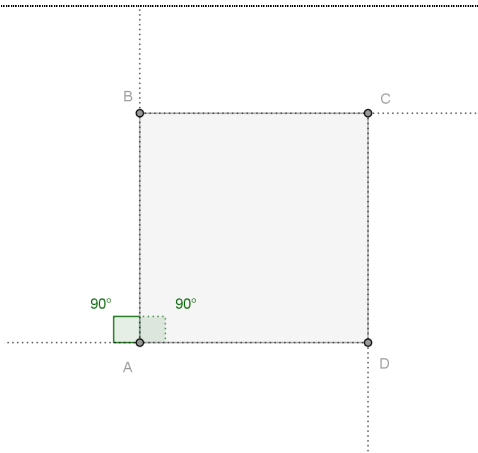
## Exercice I-03      Caractéristiques des polygones réguliers      🕒 20 min

Un polygone régulier est un polygone convexe<sup>2</sup> dont tous les angles ont la même mesure et tous les côtés la même longueur. (Wikipedia)



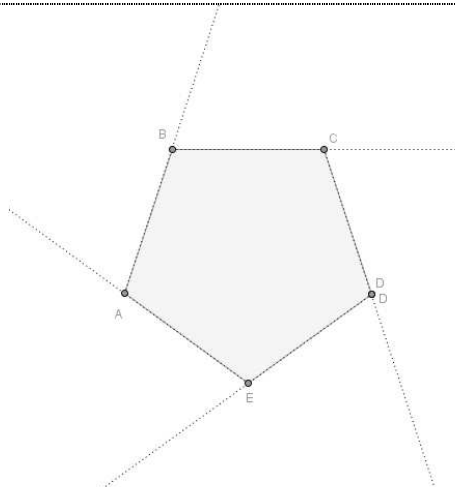
### Triangle équilatéral

1. Identifie les angles qui sont nécessaires pour tracer la figure. Marque-les en rouge !
2. Combien de fois le curseur doit-il changer de direction pour tracer la figure (nombre d'angles) ?
3. Que peut-on dire de la somme des angles ?
4. Comment peut-on calculer la valeur d'un des angles ?



### Carré

1. Identifie les angles qui sont nécessaires pour tracer la figure. Marque-les en rouge !
2. Combien de fois le curseur doit-il changer de direction pour tracer la figure (nombre d'angles) ?
3. Que peut-on dire de la somme des angles ?
4. Comment peut-on calculer la valeur d'un des angles ?



### Pentagone

Utilise les informations que tu as identifiées pour tracer les deux figures précédentes.

1. Identifie les angles qui sont nécessaires pour tracer la figure. Marque-les en rouge !
2. Combien de fois le curseur doit-il changer de direction pour tracer la figure (nombre d'angles) ?
3. Que peut-on dire de la somme des angles ?
4. Comment peut-on calculer la valeur d'un des angles ?

<sup>2</sup> La mesure de tout angle intérieur d'un polygone convexe est inférieure à 180 degrés.

## I.4. Exercices pour approfondir

### Exercice I-04 Dessiner un cadre 🕒 10 min

Le mandala est mis en évidence s'il est muni d'un cadre double.

- Point du coin en haut à droite du cadre intérieur : 150 ; 150
- Taille du stylo 1
- Point du coin en haut à droite du cadre extérieur : 160 ; 160
- Taille du stylo 2

Condition de départ : le cadre est dessiné quand on appuie sur la touche 'r'.

### Exercice I-05 Mon deuxième mandala 🕒 25 min

Mon mandala est composé de 16 carrés. Le calcul des angles se fait à l'aide des opérateurs proposés.

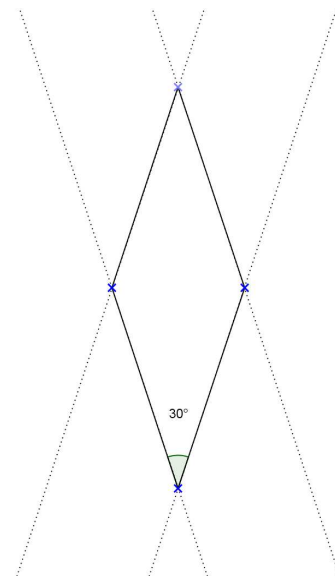
Condition de départ : le mandala est dessiné quand on appuie sur la touche 'm'.

### Exercice I-06 Un mandala encore plus beau 🕒 30 min

Le mandala devient bien plus joli s'il est dessiné avec des losanges au lieu de carrés.

Condition de départ : le mandala est dessiné quand on appuie sur la touche 't'.

Le losange : définis par un calcul les angles sur le dessin ci-contre.

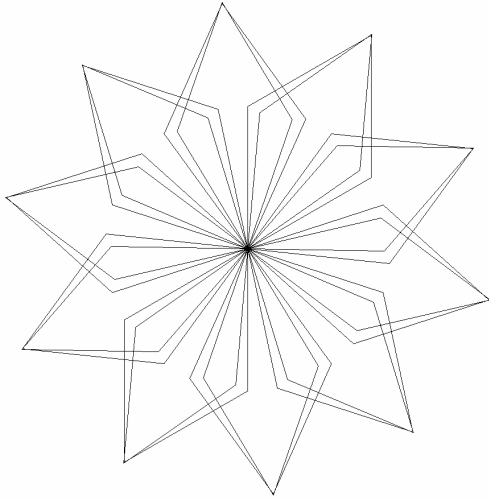


### Exercice I-07 Mandalas composés 🕒 30 min

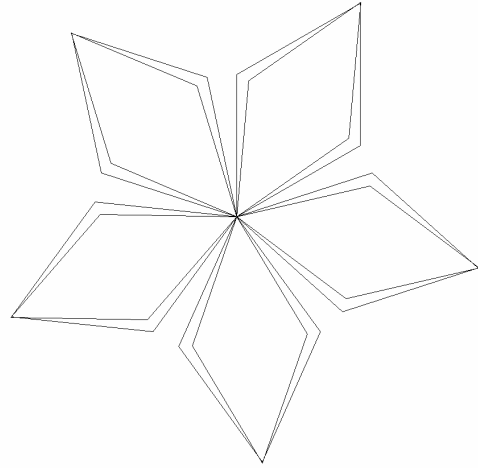
On peut réaliser un mandala en combinant deux ou plusieurs motifs.

## I.5. Quelques mandalas composés simples

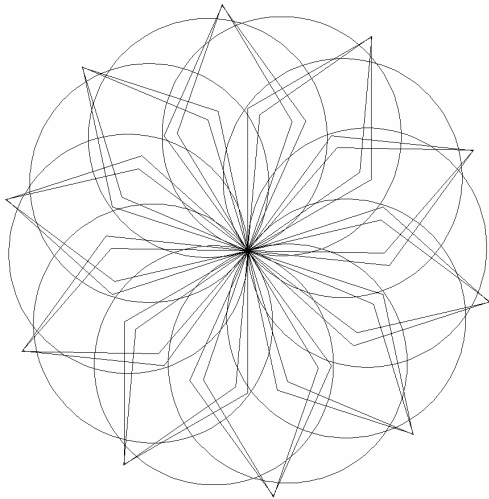
---



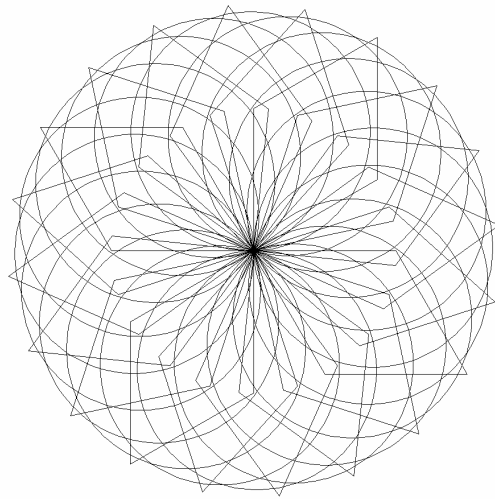
Grand losange :  $s = 82$ , angle :  $130 / 50$   
 Petit losange :  $s = 80$ , angle :  $140 / 40$   
 10 rotations



Grand losange :  $s = 82$ , angle :  $130 / 50$   
 Petit losange :  $s = 80$ , angle :  $140 / 40$   
 5 rotations



Cercle :  $s = 8$ , angle =  $6$   
 10 rotations



Cercle :  $s = 8$ , angle =  $6$   
 20 rotations



## J. Projet intermédiaire

## K. Variable et expressions mathématiques

### K.1. Variables

Nous sommes très souvent amenés à devoir nous souvenir de certaines choses : un numéro de maison, le code PIN d'un téléphone, un mot de passe, le nom d'un interlocuteur au téléphone, combien d'argent il nous reste dans la poche, ce qu'il faut encore faire l'après-midi, etc. En programmation, nous utilisons à cette fin les 'variables' au lieu d'un bout de papier ou d'un nœud dans un mouchoir.

Les variables nous permettent de mémoriser des valeurs, de les relire et de les changer. En Scratch, des variables de types différents sont utilisés : les nombres entiers, les nombres décimaux, ou des textes.

#### K.1.1 Créer et afficher des variables

La palette **Variables** nous permet de créer de nouvelles variables, de les modifier et de les supprimer. Lors de la création d'une variable nous devons d'abord lui donner un nom significatif :



Nous définissons également quels objets ont le droit d'utiliser les variables, voire de les modifier.

#### Principe :

En général, une variable devrait toujours rester 'privée' ( → 'réservée' à cet objet). Aucun autre objet n'a le droit de modifier volontairement ou involontairement notre variable. (Nous ne laissons pas non plus trainer notre carnet d'adresses au milieu de la salle de classe.)

Dans quelques cas d'exception, nous mettons notre variable à disposition des autres objets.

Dès sa création, la variable est présentée elle-même sous forme d'un bloc :

**High Score** (le crochet indique que la variable est visible sur la scène)

L'affichage de la variable sur la scène peut se faire de plusieurs manières :



sortie normale



sortie large



potentiomètre

L'affichage (la 'sortie') de la variable peut être modifié par un double clic ou par un clic droit de la souris.

### Cas spécial du potentiomètre :

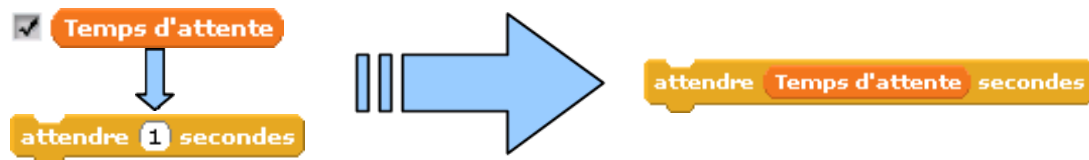
- Le potentiomètre ne fonctionne que pour les nombres entiers.
- Il est possible de définir également par un clic droit son intervalle de validité, c.-à-d. qu'on peut définir le minimum et le maximum du potentiomètre.

### K.1.2 Travailler avec des variables

Après avoir créé la première variable, nous voyons apparaître quelques blocs qui nous permettent de modifier les variables dans nos programmes :

	Sélectionner une variable puis entrer une valeur (il est possible d'entrer des nombres négatifs, des décimales ou du texte)
	Sélectionner une variable puis entrer une valeur (il est possible d'entrer des nombres négatifs ou des décimales – mais <b>pas de texte</b> )
	Sélectionner une variable : Ces deux blocs nous permettent de rendre une variable visible sur la scène - ou de la cacher

Dans un programme, tu peux utiliser les variables partout où jusqu'à présent tu as saisi directement des nombres ou des textes. Il suffit de les tirer avec la souris au bon endroit du bloc d'instruction.



**Attention : il n'est pas possible de mélanger les textes et les nombres !**

- Les variables numériques peuvent être utilisées partout.
- Les variables alphanumériques (de type texte) devraient être utilisées aux endroits prévus : (dit [ ]... , pense [ ] ... , demande [ ] ...)

**Exercice K-01      Free the crab - reloaded !**  **5 min**

**Objectif :**

Le crabe doit tenter de se libérer de la cage et compter combien de fois il rentre dans le mur de la cage.

**Description :**

- a) Ouvre l'**Exercice E-02** ("*Free the crab*").
- b) Ajoute une variable '*Collisions*', qui sera incrémentée (+1) à chaque fois que le crabe rentre dans le mur.
- c) Relance le programme quand le crabe a trouvé la sortie et a été replacé au milieu de la cage. Qu'est-ce que tu observes ? Résous ce problème !
- d) Sauvegarde le projet sous le nom de **Exercice K-01**.

**Exercice K-02      Free the crab - revisited !**  **5 min**

**Objectif :**

Le crabe doit tenter de se libérer de la cage et compter combien de fois il rentre dans le mur de la cage. La vitesse du crabe peut être saisie.

**Description :**

- a) Ouvre l'**Exercice K-01** ("*Free the crab - reloaded !*").
- b) Ajoute une variable '*Temps d'attente*' représentée comme potentiomètre. Règle le potentiomètre à un intervalle entre 0 et 0.5. Incorpore maintenant la variable dans ton programme afin de régler avec elle la vitesse du crabe.
- c) Sauvegarde le projet sous le nom de **Exercice K-02**.

## Exercice K-03

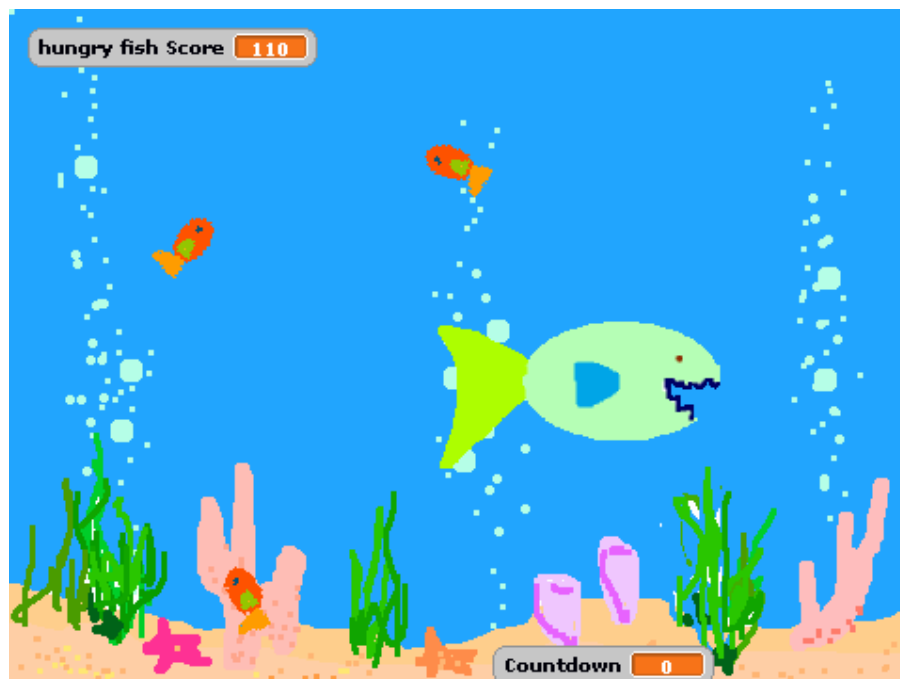
## Fishchomp - II

🕒 20 min

**Objectif :** Ajouter à un jeu existant un score et une limitation de temps.

**Description :**

- Ouvre le jeu **FishChomp** qui est inclus dans le paquet Scratch (*Exemples* → *Games* → *Fishchomp*). Teste le jeu (brièvement !).  
Principe : le gros poisson suit la position de la souris et tente de manger les petits poissons.  
 Analyse les scripts (séquences d'instructions) des différents objets et essaie de comprendre comment le programme fonctionne.
- Crée une nouvelle variable 'Score' qui est incrémentée de 10 points pour chaque petit poisson que le gros poisson attrape. Le score doit bien-entendu être remis à zéro au lancement du programme.
- Crée une variable 'Countdown', qui est mis à 30 au début du programme, puis décrétementée (diminuée) de 1 à chaque seconde. Après l'écoulement de 30 secondes, le jeu s'arrête.
- N'affiche la règle du jeu (le texte affiché en haut à droite) que pendant les 5 premières secondes du jeu.
- Sauvegarde le projet sous le nom de **Exercice K-03**.



## K.1.3 Les variables prédéfinies

Il existe pour chaque objet quelques variables prédéfinies qu'on peut utiliser de la même manière que les variables créées par nous-mêmes :

- dans la palette **Mouvement** : position x,y, direction (angle)
- dans la palette **Apparence** : costume n°, taille
- dans la palette **Sons** : volume, tempo
- dans la palette **Capteurs** : réponse, chronomètre, volume sonore

## K.2. Les expressions

Comme tout autre langage de programmation, Scratch permet de comparer des valeurs (les nombres et les textes) et de calculer avec elles. On n'a pas besoin de beaucoup de blocs d'opération pour obtenir finalement un puissant langage de programmation.

La plupart de ces blocs se situent dans la palette '**Opérateurs**' et traitent des nombres :

	addition, soustraction multiplication, division	pour les nombres entiers et décimaux
	donner un nombre aléatoire	réservé aux nombres entiers
 	le reste de la division  arrondir un nombre vers l'entier le plus proche	réservé aux nombres entiers  pour les nombres décimaux, le résultat est un nombre entier
	plusieurs fonctions mathématiques	La fonction la plus importante : <i>racine carrée</i> <i>valeur absolue</i> Autres fonctions : <i>sin, cos, tan, asin, acos, atan,</i> <i>ln, log, e^, 10^</i>

Il est possible d'imbriquer les blocs les uns dans les autres afin de réaliser des calculs plus complexes. On peut comparer cela avec les parenthèses en mathématiques :

Exemple :

$(12+7)*(6-4)$   
correspond à



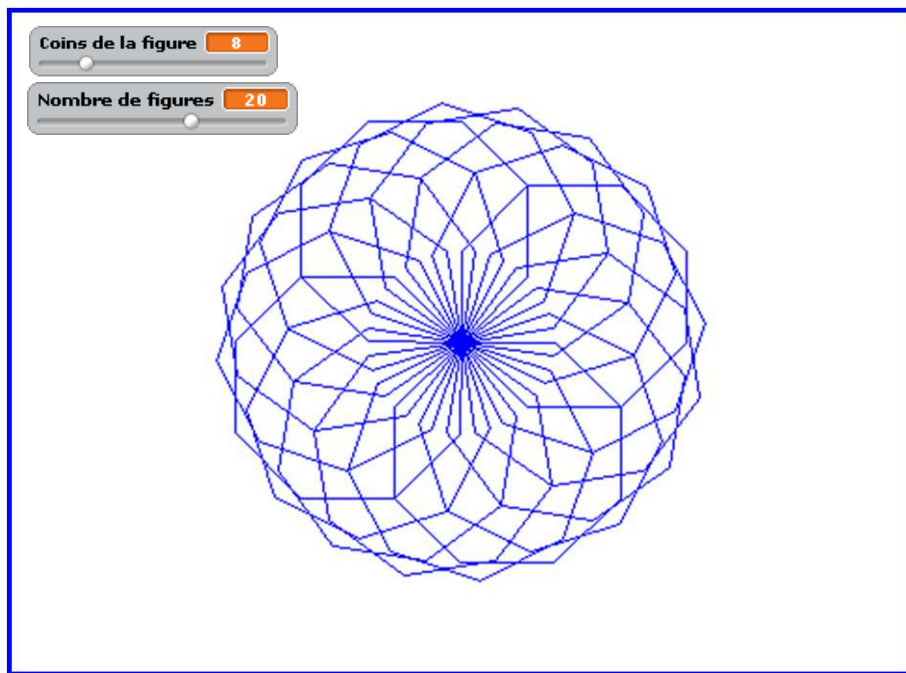
Il faut bien sûr veiller à imbriquer les blocs dans le bon ordre.

**Principe :** Le système procède en commençant avec le bloc le plus haut et en finissant avec le bloc le plus bas.

**Exercice K-04      Générateur de mandalas** 🕒 30 min

**Objectif :**

Calculer automatiquement des mandalas



**Description :**

- a) Définis 2 variables '**Coins de la figure**' et '**Nombre de figures**', affichés sous forme de potentiomètres. L'intervalle de validité pour les deux variables est [3...20].
- b) Quand on appuie sur la touche 'espace', le programme dessine un mandala composé d'autant de figures que la valeur de la variable '**Nombre de figures**', et chaque figure possède d'autant de coins que la valeur de la variable '**Coins de la figure**'.  
La longueur d'un côté doit être calculée automatiquement de manière à ce que la figure ne dépasse pas la scène.
- c) Affine les mandalas selon ton goût (couleur, taille et intensité du stylo, repeindre)
- d) Sauvegarde le projet sous le nom de **Exercice K-03**.

## K.2.2 Vrai ou faux

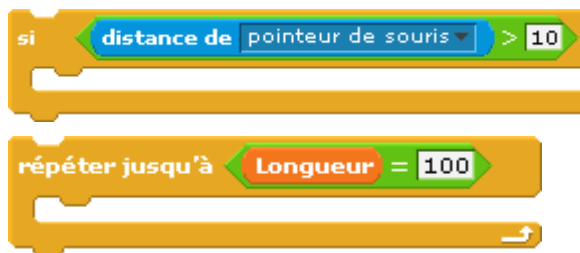
Comme nous avons vu dans le chapitre E (Prendre des décisions), les blocs aux extrémités pointues servent à formuler des **conditions**.

Plus précisément, un bloc aux extrémités pointues donne ou bien le résultat '**vrai**' (condition vérifiée) ou bien le résultat '**faux**' (condition non vérifiée). De tels blocs peuvent être utilisés comme sous-blocs p.ex. dans les blocs "si <...> ...sinon...", "attendre jusqu'à <...>" ou "répéter jusqu'à <...>".

Les blocs opérateurs suivants nous permettent de comparer des valeurs (des nombres ou des textes) et d'utiliser leur résultat dans le contrôle de nos scripts.

Opérateurs de comparaison : 

Exemples :



<b>Exercice K-05</b>	<b>Fishchomp - III</b>	🕒 <b>10 min</b>
----------------------	------------------------	-----------------

**Objectif :** Retenir et afficher le score maximal (Highscore)

**Description :**

- a) Ouvre ton jeu *FishChomp* – II (**Exercice K-03**).
- b) Modifie le compte à rebours afin de pouvoir utiliser la boucle '*répéter jusqu'à*'. (Ceci donne une solution plus flexible puisqu'il ne faut changer qu'à un seul endroit la valeur de départ de la variable '**Countdown**' pour modifier la durée du jeu.)
- c) Crée une variable '**Highscore**' qui vérifie à la fin du jeu si 'Score' dépasse le Highscore actuel. Dans ce cas, le score actuel devient le nouveau Highscore et un son (de ton choix) se fait entendre.
- d) Sauvegarde le projet sous le nom de **Exercice K-05**.

**Exercice K-06**

**Devinette**

 >60 min

**Objectif :**

**Deviner un nombre secret aléatoire calculé par l'ordinateur**

**Contexte :**

Il faut absolument que tu puisses téléphoner, mais les piles de ton portable sont vides. Cassy a son portable sur elle mais elle ne veut pas que tu l'utilises. Elle te lance le défi suivant : « *Si tu arrives à deviner mon code PIN à 4 chiffres en moins de 15 essais alors tu peux utiliser mon portable autant que tu veux. A chaque essai, je ne te dis que si mon code PIN est plus grand ou plus petit que le nombre de ton essai.* »



...

## Description :

Développe un programme qui simule Cassy et sa devinette :

- Au début du programme, il faut enregistrer dans une variable (invisible) **Pin** un code aléatoire entre 1000 et 9999. C'est le nombre que le joueur doit deviner.
- Tu peux également présenter le contexte de l'histoire. (P.ex. Cassy téléphone au début et dit « *Ah bon, tu veux utiliser mon téléphone. Eh bien, je te le laisse si tu arrives à deviner mon code à 4 chiffres en moins de 15 essais !* »)
- Utilise le bloc **demander** `Quel est mon code PIN ?` **et attendre** pour attendre la saisie du nombre suivant. Tu trouves ce bloc dans la palette **Capteurs**. La valeur saisie par le joueur se trouve alors automatiquement dans la variable prédéfinie **réponse**.
- Une deuxième Variable (visible) **Essais** mémorise le nombre d'essais du joueur.
- Si le joueur arrive à deviner le code PIN en moins de 15 essais, Cassy **se fâche** et donne le portable 😊
- Si, par contre, il n'y est pas arrivé après 14 essais, elle **danse de joie** pendant une seconde...
- Tu trouves des images pour Cassy dans la bibliothèque de Scratch (→ costumes → Importer → People). Pour la danse tu peux importer des sons de la bibliothèque ou utiliser tes propres sons. Pour rendre la danse plus rigolote encore, jette un coup d'œil sur les effets de la palette **Apparence** :

modifier l'effet `tournoyer` par `5`

mettre l'effet `oeil de poisson` à `15`

**Amuse-toi bien pour la programmation et la devinette !**

## K.2.3 Pour avancés : conditions combinées

Nous avons souvent besoin de plusieurs conditions pour vérifier un état. Ces conditions doivent alors être combinées.

En mathématiques p.ex. il faut souvent vérifier si un nombre se trouve à l'intérieur d'un intervalle :  $0 < X < 1000$

Une condition combinée (ou complexe) peut être réalisée en utilisant les blocs Scratch suivants :



<et> veut dire que les deux conditions doivent être vérifiées

<ou> veut dire qu'au moins une des deux conditions doit être vérifiée

<non> veut dire que la condition ne doit pas être vérifiée

Pour vérifier si  $0 < X < 1000$  on peut écrire :



Pour vérifier si  $X \leq 0$  ou  $X \geq 1000$  on peut écrire en Scratch :



Mais on peut également écrire :

