

MITIC 5e

INSTITUT
florimont

01000001

01110101011101000110010000000111010001000000100001001
111010101110010011100110010000000111010001000000100001001
100101011011100110111101101001011101000010000001001110011
0000101100100011000010111010110010000100000011001010111
010000100000010100110110100101101011011110110111000100
0000101100110010101110010011001000110000101101110001011
100010000001001001011011100111001101110100011010010111010
00111010101101000010000001000110011011000110111101110010
011010010110110101101111011011100111010000101110000000000

Informatique 5^e – Fiches MITIC

Institut Florimont

Petit-Lancy (Suisse)

© Tout droit réservé. Crédit photographie couverture : Institut Florimont. Illustration des premières pages de chapitre issue de *Codex Leicester* de Leonardo da Vinci (domaine public).

2^{ème} édition, v2.0

juin 2021



Informatique 5^e
Fiches MITIC

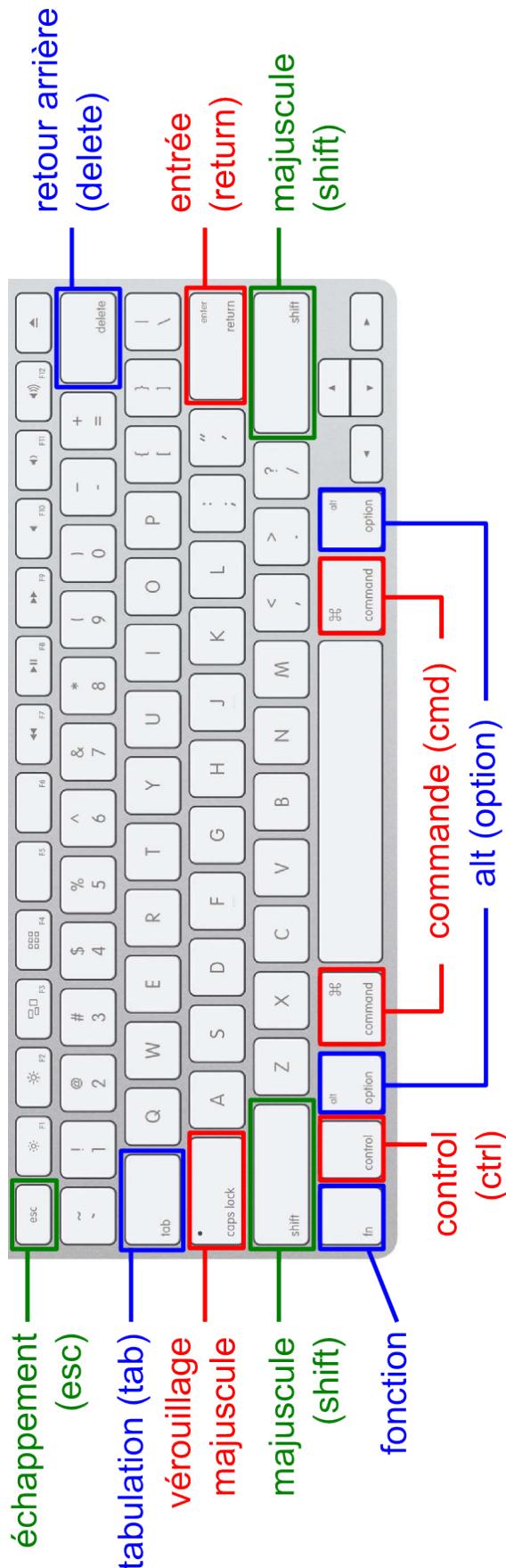
Institut Florimont

Ce livret appartient à

Table des matières

1	Programmation Scratch.....	1
1	Séance 1 : dessiner une spirale	2
1.1	Pour bien démarrer...	2
1.2	Sujet de l'activité...	3
1.3	Pour aller plus loin...	4
2	Séance 2 : un quiz de calcul mental	5
2.1	Pour bien démarrer...	5
2.2	Sujet de l'activité...	5
2.3	Pour aller plus loin...	5
3	Séance 3 : créer un jeu de « Pong » en Scratch	6
3.1	Pour bien démarrer...	6
3.2	Sujet de l'activité...	6
3.3	Pour aller plus loin...	6
4	Aide pour réaliser les activités	7
4.1	Aide pour la séance 1	7
4.2	Aide pour la séance 2	11
4.3	Aide pour la séance 3	13

Les touches spéciales du clavier



Pour sauvegarder son travail : cmd + S

Pour annuler la dernière opération : cmd + Z

Philosophie du document

Vous avez entre les mains le deuxième tome d'une série de quatre fascicules qui accompagneront les élèves des classes de 6^e, 5^e, 4^e et 3^e jusqu'au moment où ils recevront un ordinateur qu'ils seront en mesure d'exploiter au mieux pour leur travail.

Ce document se présente sous la forme d'un livret qui rassemble des fiches MITIC¹ permettant aux élèves d'apprendre à utiliser les logiciels et espaces numériques mis à leur disposition. Pour l'année de 5^e, sont traités les logiciels *Microsoft Word* (traitement de texte), *Microsoft Excel* (tableur grapheur), *Gimp* (retouche d'image), *Audacity* (traitement des fichiers son) et *Scratch* (programmation). Au début de chaque chapitre un lien permettant de télécharger le logiciel est fourni.

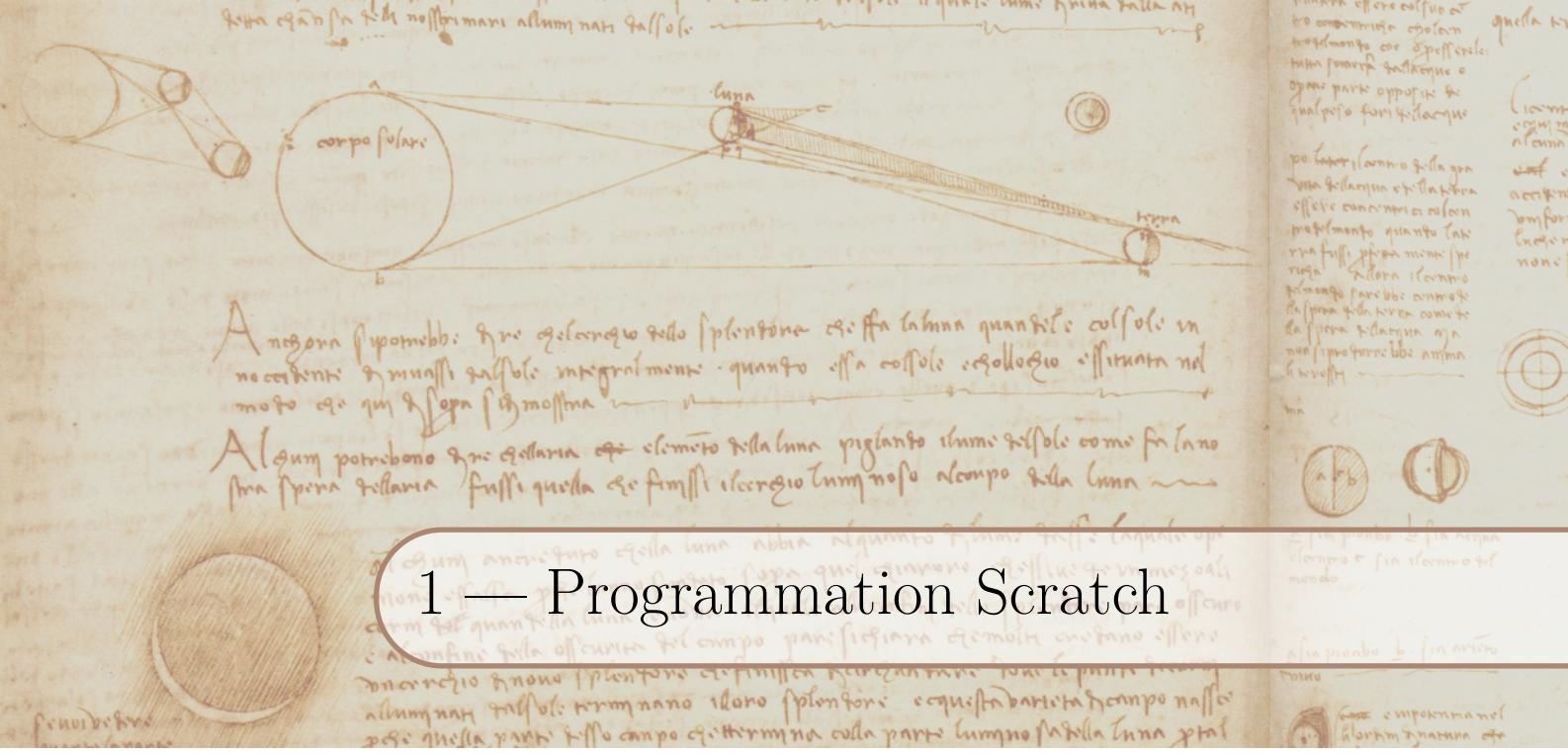
Chaque fiche est conçue pour être exploitée à plusieurs occasions et dans des matières différentes, à chaque fois lors d'une séance de 45 minutes. La fiche sur le tableur, par exemple, est découverte en physique-chimie (*Séance 1*), exploitée à nouveau en mathématiques (*Séance 2*) puis en histoire-géographie (*Séance 3*) selon un calendrier proposé en début de fiche. Nous avons à chaque fois essayé de faire coïncider les notions abordées dans la fiche avec le programme de la matière concernée.

Professeurs, c'est à vous que revient la tâche délicate d'inclure le contenu de ces fiches dans votre progression. À vous de le faire vivre : arriver en salle informatique et demander aux élèves de remettre en forme un texte de Molière ne présente que peu d'intérêt pédagogique. Donnez du sens à ces fiches et profitez-en pour diversifier votre enseignement. N'hésitez pas à exploiter dans vos cours les techniques présentées dans ce fascicule afin que les élèves utilisent plusieurs fois leurs nouvelles compétences et, par là-même, les pérennisent.

Merci d'avance à tous pour votre implication.

L'équipe de rédaction.

1. MITIC : Médias, Images et Technologies de l'Information et de la Communication.



1 — Programmation Scratch

Les ordinateurs sont des machines qui exécutent des programmes. On peut écrire des programmes dans différents *langages de programmation*, par exemple *Python*, *C++*, *Java*... ou encore *Scratch*.

- Logiciel¹ : *Scratch 3.0*
- Prérequis (se reporter si nécessaire aux *Fiches Mitic 6^e*) :
 - choisir et paramétriser l'objet sprite et l'objet scène ;
 - créer/insérer un nouvel objet ;
 - écrire un code comprenant mouvements, réponses à événement, boucles et son ;
 - associer un code à un objet ;
 - utiliser la structure conditionnelle if (bloc *si ..*) ;
 - écrire un programme simple qui réponde à une problématique donnée.
- Matière concernée : mathématiques.
- Compétences :
 - créer une variable et modifier sa valeur ;
 - utiliser la boucle for (bloc *répéter n fois*) ;
 - utiliser la structure if .. then .. else (bloc *si .. alors .. sinon*) ;
 - utiliser la boucle infinie (bloc *répéter indéfiniment*) ;
 - lire un algorithme écrit sous la forme d'un *flowchart* ;
 - écrire un programme à partir d'un *flowchart*.

Scratch est un langage de programmation visuelle (on place des blocs d'**instructions** pour créer des programmes composés de **codes**) et événemmentielle (le programme réagit à des **événements** comme le clic de souris ou l'appui sur une touche). Il contient des **objets** : le sprite est un objet, l'arrière plan de la scène est un autre objet. On peut modifier les propriétés des objets, leur associer des **codes**, des **costumes** ou des **sons**.

1. Le logiciel *Scratch* est librement téléchargeable : <https://scratch.mit.edu/>

1 Séance 1 : dessiner une spirale

1.1 Pour bien démarrer...

Passer Scratch en langue française

Avant de commencer, vous pouvez si nécessaire choisir la langue de l'interface en cliquant sur l'icone en haut à gauche . Nous choisirons par exemple **Français**

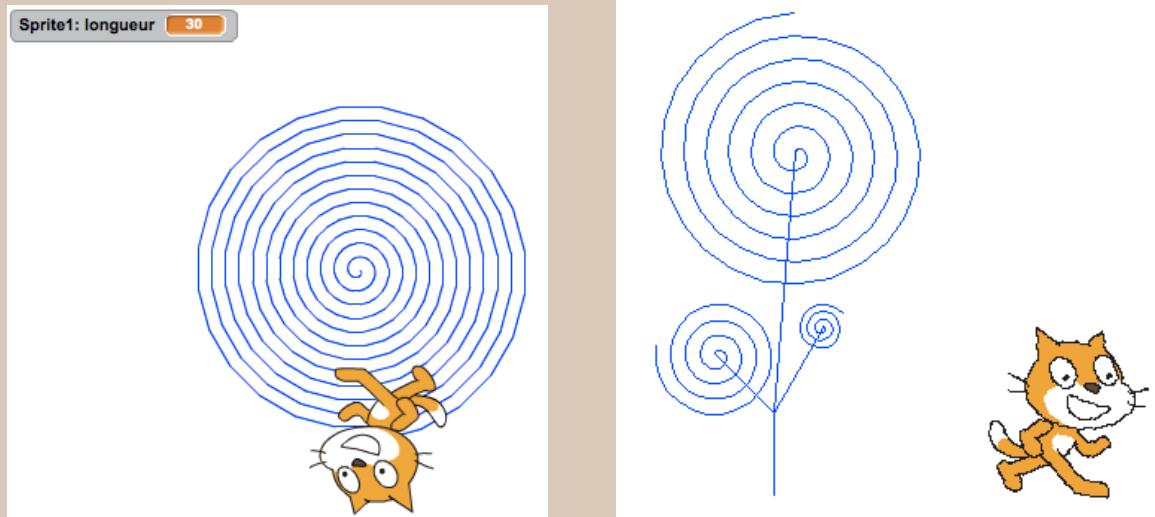


Penser à enregistrer régulièrement

Dès que vous avez ouvert un nouveau programme dans *Scratch*, sauvegardez-le au format Nom-seance1.sb3 : dans le menu **Fichier**, choisir **Sauvegarder sur votre ordinateur**. Pendant que vous travaillez, pensez à sauvegarder régulièrement votre travail.

1.2 Sujet de l'activité...

Le but de cette séance est de dessiner une spirale, comme montré sur la figure ci-dessous. Il faudra ensuite dessiner une fleur en réutilisant le code conçu pour dessiner la spirale.

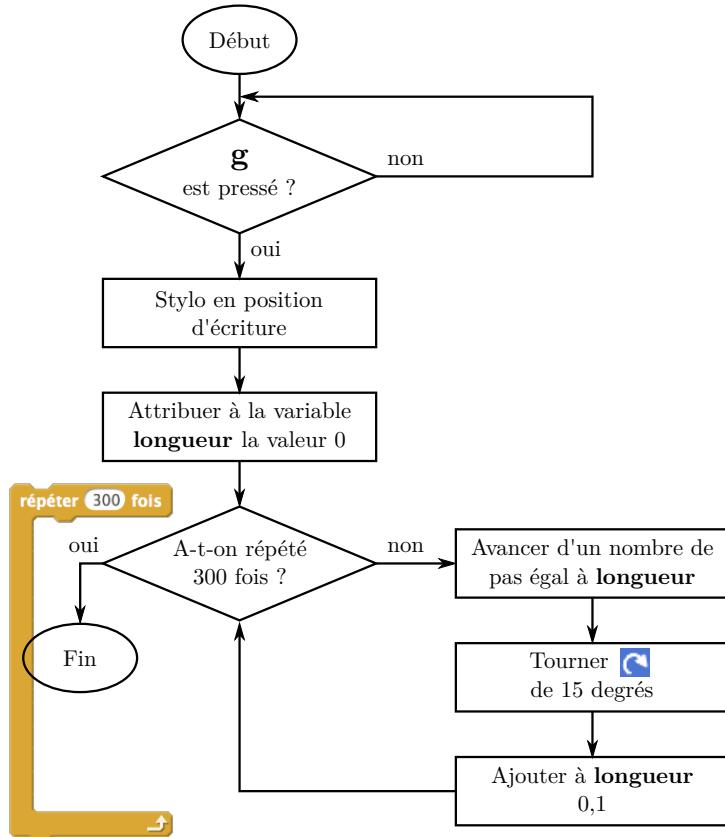


Les étapes pour réaliser ce programme sont les suivantes :

- création d'une variable utile pour dessiner
- dessin de la spirale
- dessin de la fleur

Une fois le programme terminé, vous enregistrerez votre fichier au format .sb3 (le fichier sera nommé à partir de votre nom : Nom-seance1.sb3), puis vous le rendrez sur *Teams* à l'endroit indiqué par votre enseignant (si nécessaire, se reporter à la fiche méthode *Remettre son devoir*, page ??).

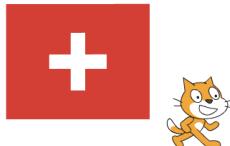
Indication : pour créer la spirale, on pourra par exemple suivre une séquence d'instructions qui effectue les actions suivantes :



Pour obtenir de l'aide, rendez-vous à la page 7

1.3 Pour aller plus loin...

Écrire un programme en *Scratch* qui dessine le drapeau suisse. Ce programme devra utiliser des boucles.



Indication : on peut créer une variable *PositionY* qui contient la position en *y* du sprite, puis l'utiliser dans une boucle qui se répète tant que *PositionY* n'a pas atteint la valeur souhaitée. Avant de terminer la boucle, il faudra bien entendu modifier la valeur de *PositionY*.



2 Séance 2 : un quiz de calcul mental

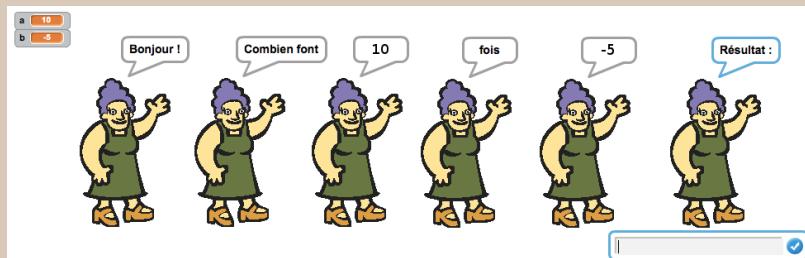
2.1 Pour bien démarrer...

Dès que vous avez ouvert un nouveau programme dans *Scratch*, sauvegardez-le au format Nom-seance2.sb3 : dans le menu **Fichier**, choisir **Sauvegarder sur votre ordinateur**. Pendant que vous travaillez, pensez à sauvegarder régulièrement votre travail.

2.2 Sujet de l'activité...

Le but de cette séance est d'écrire un programme qui demande à l'utilisateur le résultat d'une multiplication. L'utilisateur donne alors sa réponse, et le programme lui indique si sa réponse est juste ou non.

Par exemple, le programme pourra poser la question « *Bonjour, combien font* $10 \times (-5)$? ». L'utilisateur devra alors répondre -50 (cette réponse est entrée au clavier). La figure ci-dessous montre différentes étapes de l'exécution du programme. Essayez de trouver par vous-mêmes comment obtenir ce résultat.



Une fois le programme terminé, vous enregistrerez votre fichier au format .sb3 (le fichier sera nommé à partir de votre nom : Nom-seance2.sb3), puis vous le rendrez sur *Teams* à l'endroit indiqué par votre enseignant (si nécessaire, se reporter à la fiche méthode *Remettre son devoir*, page ??).

Pour obtenir de l'aide, rendez-vous à la page 11

2.3 Pour aller plus loin...

Écrire l'algorithme correspondant au jeu décrit ci-dessous, puis écrire le programme :

Scratch choisit un nombre compris entre -100 et 100 et le joueur essaie de le deviner. Il faut utiliser une variable « nombre » qui stocke le nombre choisi aléatoirement. Chaque fois que le joueur propose un nombre, on lui indique soit « bravo c'est gagné », soit « le nombre cherché est plus petit », soit enfin « le nombre cherché est plus grand ».

Pour améliorer le jeu, ajouter une variable qui compte le nombre de coups dont le joueur a eu besoin pour deviner le nombre et l'afficher à la fin du jeu.

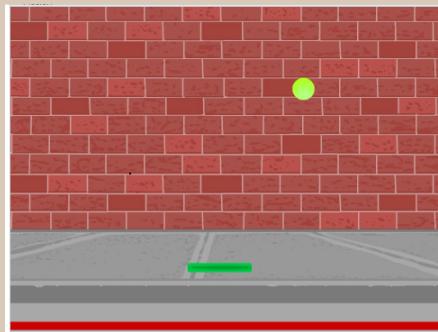
3 Séance 3 : créer un jeu de « Pong » en Scratch

3.1 Pour bien démarrer...

Dès que vous avez ouvert un nouveau programme dans *Scratch*, sauvegardez-le au format Nom-seance3.sb3 : dans le menu **Fichier**, choisir **Sauvegarder sur votre ordinateur**. Pendant que vous travaillez, pensez à sauvegarder régulièrement votre travail.

3.2 Sujet de l'activité...

Le but de cette séance est d'écrire un jeu de « Pong » dans lequel le joueur doit faire rebondir une balle avec une raquette et éviter que la balle ne touche le bas de l'écran. La figure ci-dessous montre à quoi ressemblera le jeu une fois terminé.



Une fois le programme terminé, vous enregistrerez votre fichier au format .sb3 (le fichier sera nommé à partir de votre nom : Nom-seance3.sb3), puis vous le rendrez sur *Teams* à l'endroit indiqué par votre enseignant (si nécessaire, se reporter à la fiche méthode *Remettre son devoir*, page ??).

Pour obtenir de l'aide, rendez-vous à la page 13

3.3 Pour aller plus loin...

Pour améliorer le jeu, on peut :

- changer la couleur de la balle à chaque fois qu'elle touche la raquette ;
- ajouter un son quand la balle touche la ligne du bas ;
- ajouter un compteur de points, par exemple en ajoutant 1 point chaque fois que la balle touche la raquette ou en ajoutant une ligne en haut de la scène et en ajoutant 1 point chaque fois que la balle touche cette ligne ;
- augmenter la vitesse de la balle quand le nombre de points augmente.

Et si on essayait de passer en mode deux joueurs ? En effet, il est possible de créer une deuxième raquette et ainsi de pouvoir jouer à deux. La deuxième raquette peut par exemple être déplacée à l'aide des flèches de direction.

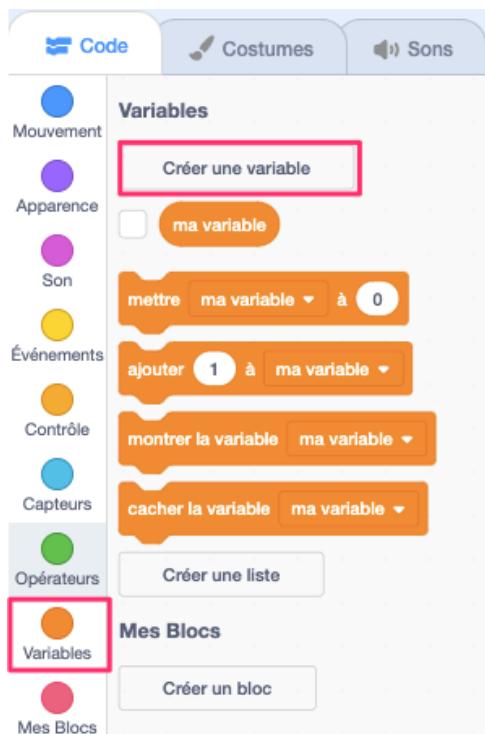
4 Aide pour réaliser les activités

4.1 Aide pour la séance 1

Première étape : création d'une variable

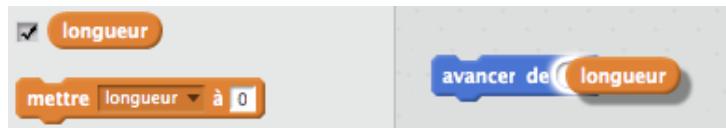
Pour dessiner la spirale, il faut utiliser une *variable* : c'est une case dans la mémoire de l'ordinateur qui permet d'enregistrer une valeur et de la modifier par la suite. La case mémoire porte un nom (ici *longueur*), et c'est ce nom qui est utilisé pour accéder à la valeur.

Pour créer une variable, il faut sélectionner **Variables** dans la colonne de gauche, puis **Créer une variable** :



Une boîte de dialogue s'ouvre alors (ci-dessous, à gauche) : il faut y inscrire le nom de la variable, puis choisir une *variable locale* en sélectionnant **Pour ce sprite uniquement**, et enfin cliquer sur **OK**. Le menu **Variables** possède alors de nouveaux blocs de commande associés à notre variable **longueur** (ci-dessous à droite).

La nouvelle variable **longueur** peut alors être utilisée, par exemple pour faire avancer le sprite d'un nombre de pas égal à la valeur de la variable **longueur** :



Il est également possible de modifier la valeur de la variable **longueur** en lui ajoutant une valeur (par exemple sur la figure ci-dessous, on ajoute 0,1). Le contenu de la case mémoire **longueur** est augmenté de la valeur indiquée. Par exemple, si la case mémoire **longueur** contenait la valeur 14,5, après cette instruction elle contient la valeur 14,6.



À retenir...

Les **variables** sont très importantes pour la programmation. Une variable correspond à une case dans la mémoire de l'ordinateur où l'on peut stocker une valeur. Pour rappeler cette valeur, il suffit d'utiliser le nom de la variable. Des opérations peuvent être effectuées avec les variables :

- On peut **créer** une nouvelle variable. Si on choisit *Pour tous les sprites*, la variable est **globale**, c'est-à-dire qu'elle peut être utilisée partout dans le programme. Si on choisit *Pour ce sprite uniquement*, la variable est **locale**, c'est-à-dire qu'elle ne peut être utilisée que pour le sprite pour lequel elle a été créée.
- On peut **assigner** une valeur à la variable, c'est-à-dire ranger une valeur dans la case mémoire désignée par le nom choisi (ici, on range la valeur 0 dans la case mémoire *longueur*) :



- On peut **ajouter** un nombre à la valeur de la variable, ce qui fait changer la valeur stockée dans la case mémoire désignée par le nom choisi (ici, on ajoute 0,1 à la valeur stockée dans la case mémoire *longueur*) :



- Enfin, ce nom de variable peut être utilisé à tout moment dans le programme (ici, on demande que le sprite avance d'un nombre de pas égal à la valeur stockée dans la case mémoire *longueur*) :

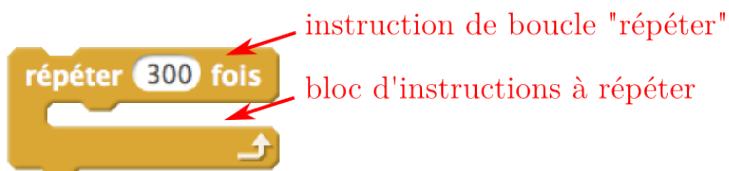


Deuxième étape : le code qui dessine la spirale

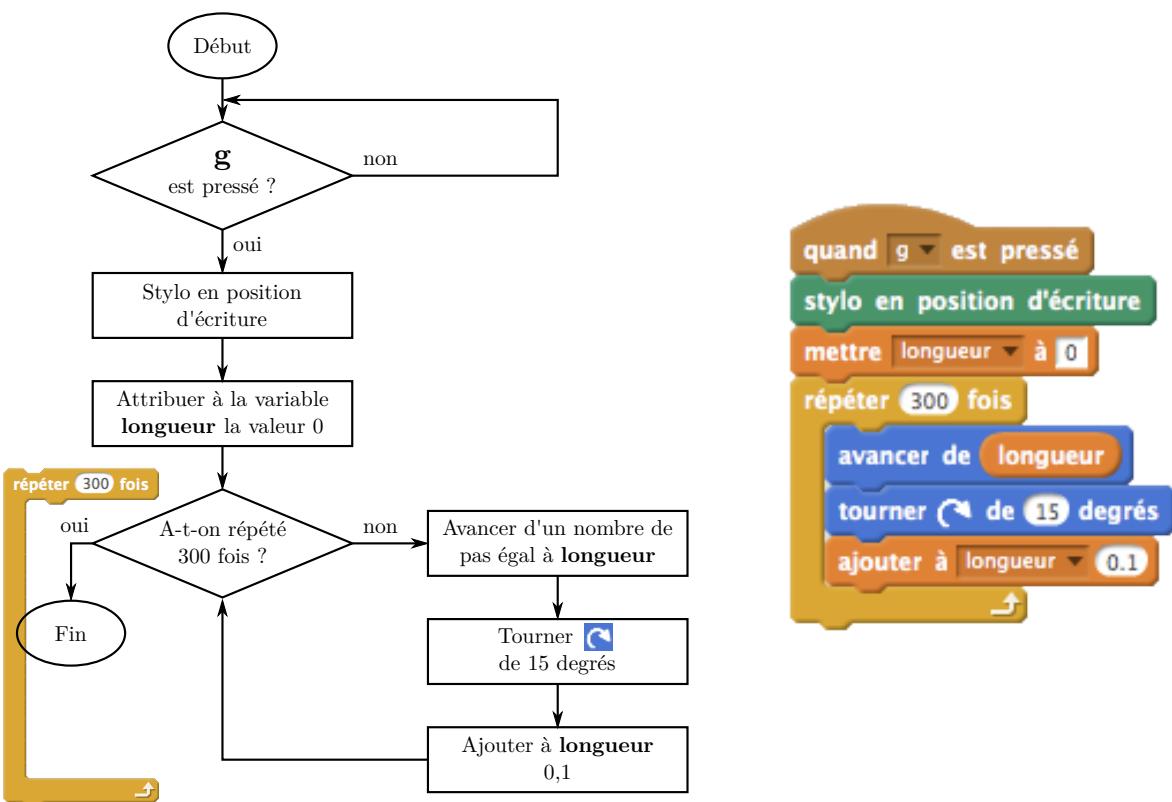
Pour dessiner une spirale, il faut répéter plusieurs fois le même bloc d'instructions « *Avancer de x pas – Tourner un peu – Augmenter la valeur de x* ». Nous avons vu l'année passée que ceci était possible grâce à une **boucle** qui permet de répéter un certain nombre de fois un bloc d'instructions.

À retenir...

La **boucle** est une structure importante en programmation : elle permet de répéter un bloc d'instructions plusieurs fois, tant qu'une condition est vérifiée ou même indéfiniment. Dans notre programme, nous utilisons une boucle « répéter 300 fois ».



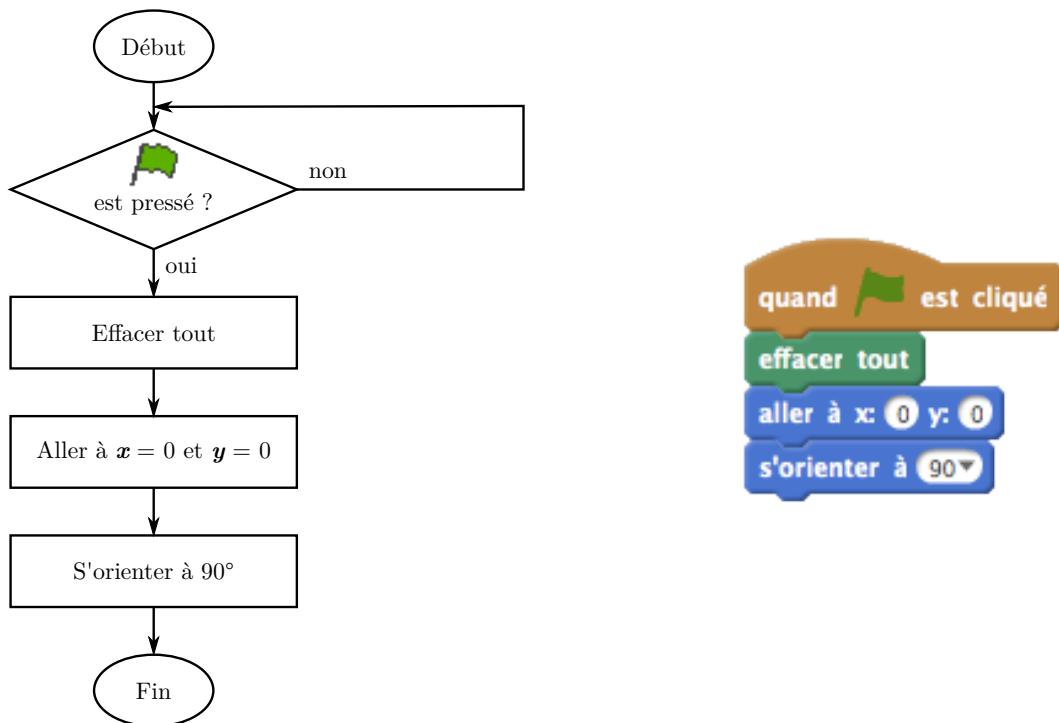
Créer maintenant le code suivant, qui doit être associé au sprite (à gauche, ci-dessous, l'*algorithme* qui correspond au code) :



Attention ! Comprenez bien l'algorithme ci-dessus, car lors de la prochaine séance, seul l'algorithme sera donné.

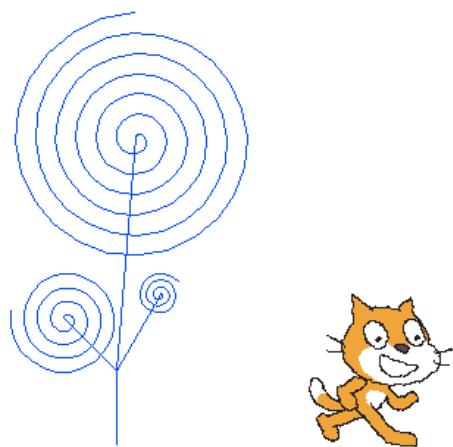
Troisième étape : un code qui efface l'écran

Créer enfin ce petit code, associé au sprite, qui permet d'effacer l'écran :



Quatrième étape : à vous de jouer !

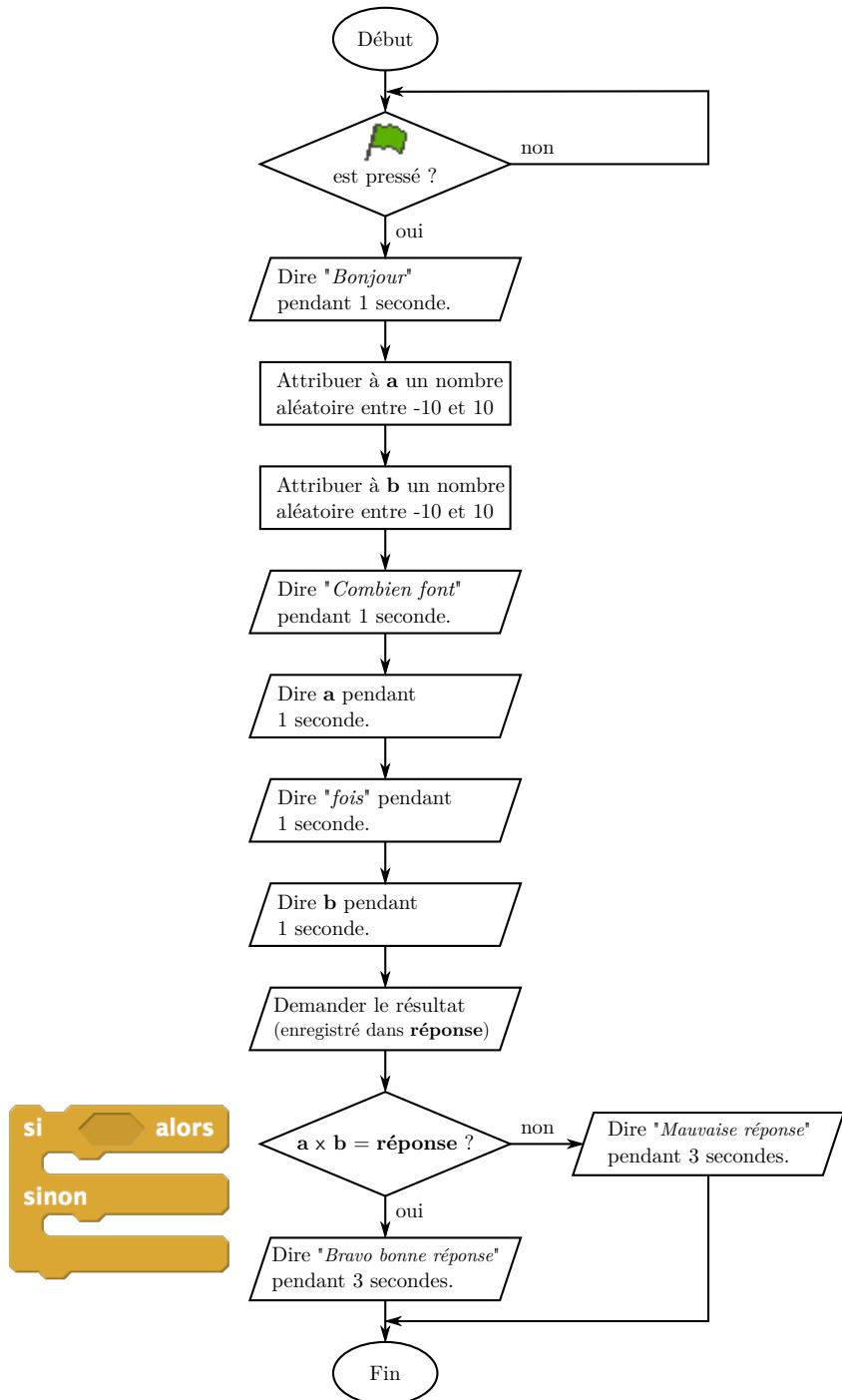
Compléter le programme afin de dessiner une fleur qui ressemble à celle montrée ci-dessous.



4.2 Aide pour la séance 2

Algorithme du programme

L'algorithme du programme est le suivant :



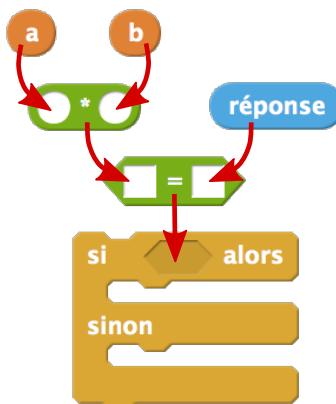
Aide pour l'écriture du programme

Pour écrire ce programme, il faudra :

1. Créer deux variables **a** et **b** (voir si nécessaire la section 4.1 page 7) ;
2. Supprimer le sprite par défaut et en choisir un autre en cliquant sur une des icônes **Nouveau lutin:**  , comme par exemple celui de l'image ci-dessous.

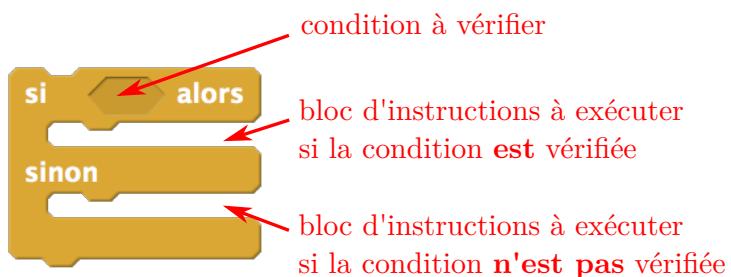


3. Utiliser des blocs **dire [text v] pendant [durée s] secondes** pour poser la question à l'utilisateur ;
4. Utiliser un bloc **demandeur [text v] et attendre** pour attendre la réponse de l'utilisateur. Elle est alors enregistrée dans une variable **réponse** que l'on trouve déjà prête dans les blocs sous la catégorie **capteur** : **réponse** .
5. Construire le bloc **si ... alors .. sinon** comme indiqué ci-dessous.



À retenir...

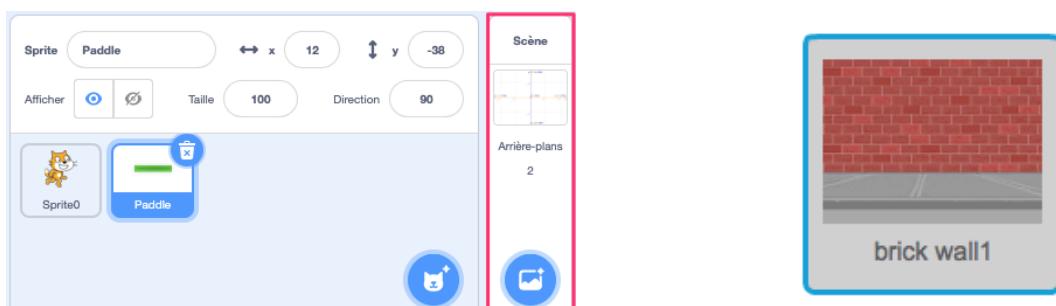
La structure conditionnelle **si .. alors .. sinon** est une structure importante en programmation : elle permet d'exécuter un bloc d'instructions **si** une condition est vérifiée, et **sinon**, elle exécute un autre bloc d'instructions.



4.3 Aide pour la séance 3

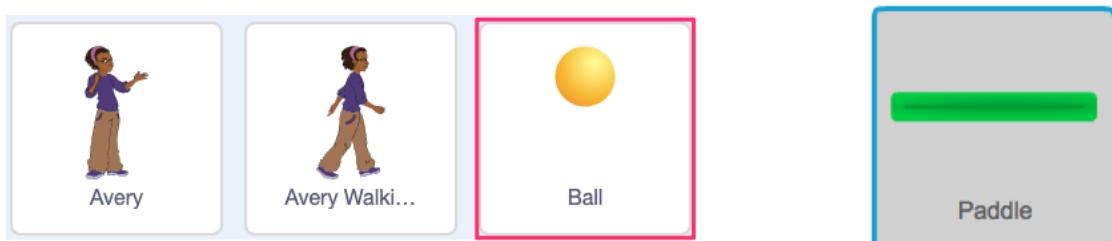
Première étape : création de l'arrière plan et choix des sprites

La première étape consiste à choisir un arrière plan :



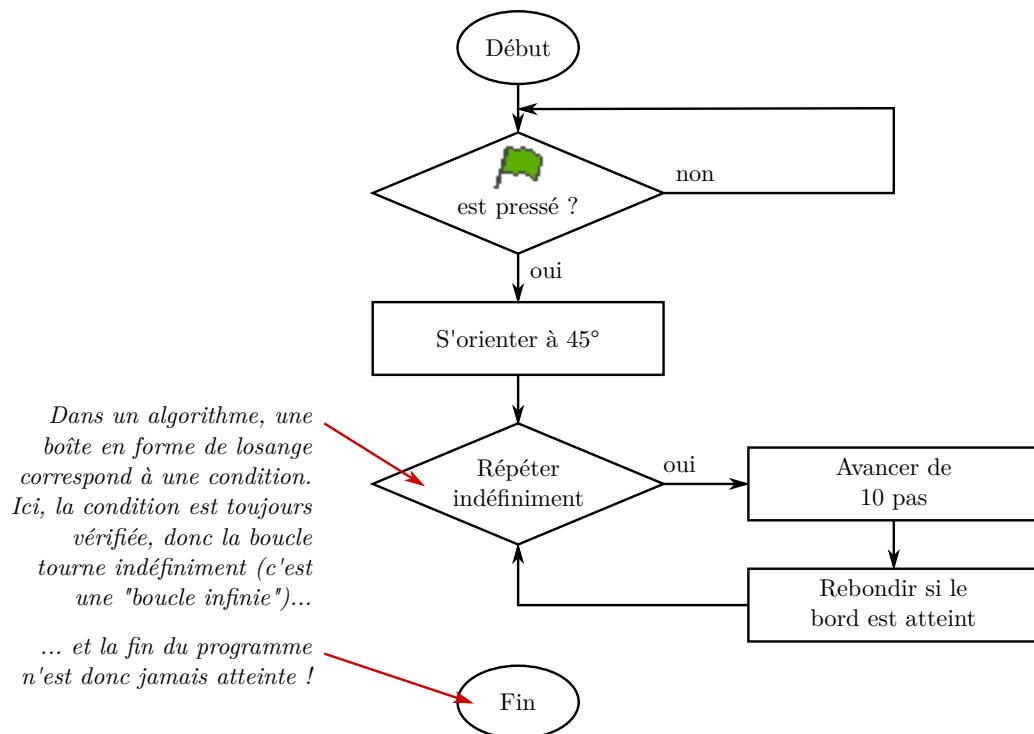
Supprimer ensuite le sprite existant par défaut, puis choisir les deux sprites nécessaires

au jeu, la raquette et la balle, que l'on redimensionne si nécessaire.



Définition du mouvement de la balle

Avant de mettre la balle en mouvement, il faut la positionner (au centre du jeu par exemple) et l'orienter (à 45° par exemple si on veut que la balle parte vers le haut) avant de définir la boucle qui fait avancer la balle et la fait rebondir sur les murs.



Pour vérifier que cette partie du jeu est bien programmée, cliquer sur le drapeau vert : la balle doit avancer sans arrêt et rebondir sur les murs.

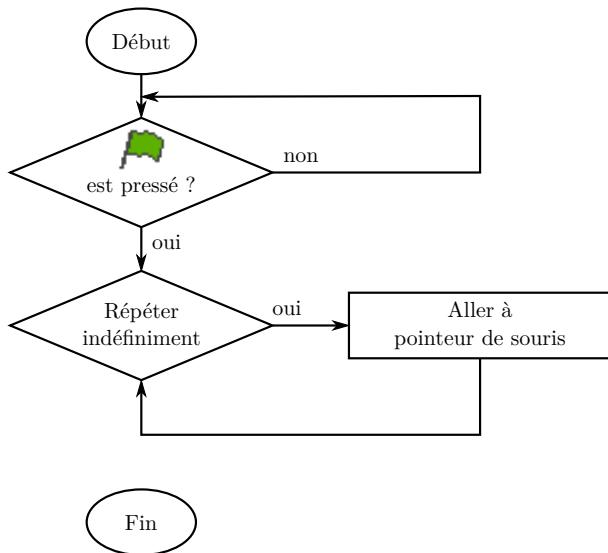
À retenir...

La structure de boucle infinie **Répéter indéfiniment** est une structure importante en programmation : elle permet d'exécuter un bloc d'instructions sans jamais se terminer.



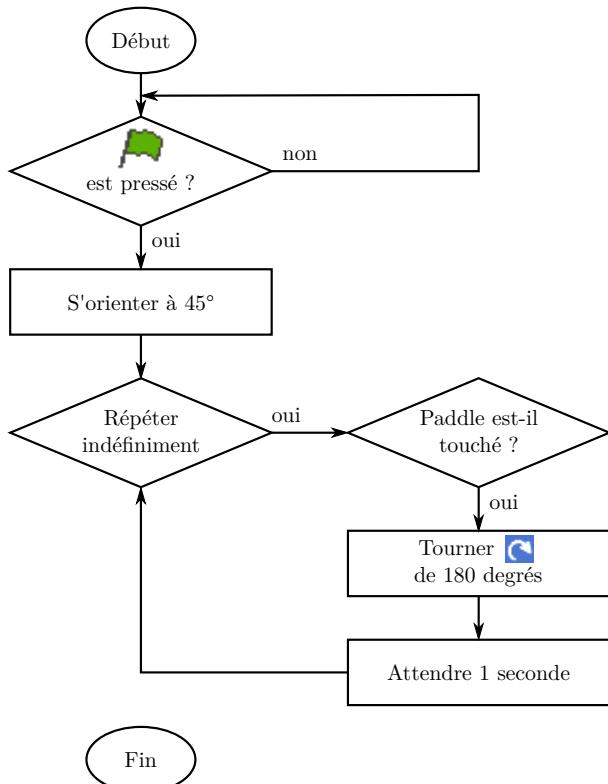
Mouvements de la raquette et rebond de la balle sur la raquette

Pour pouvoir contrôler les mouvements de la raquette avec la souris, il faut utiliser l'instruction **aller à pointeur souris**. Écrire la boucle correspondante en cliquant bien, au préalable, sur le sprite raquette.



Vérifiez que votre code est correct : quand vous cliquez sur le drapeau vert, la raquette suit la souris.

Revenir au sprite balle pour programmer ce qui se passe quand il touche la raquette. Pour cela, il faut utiliser la condition `si paddle touché..alors` et l'instruction `tourner de 180 degrés` qui fait que la balle repart dans l'autre sens quand elle touche la raquette.

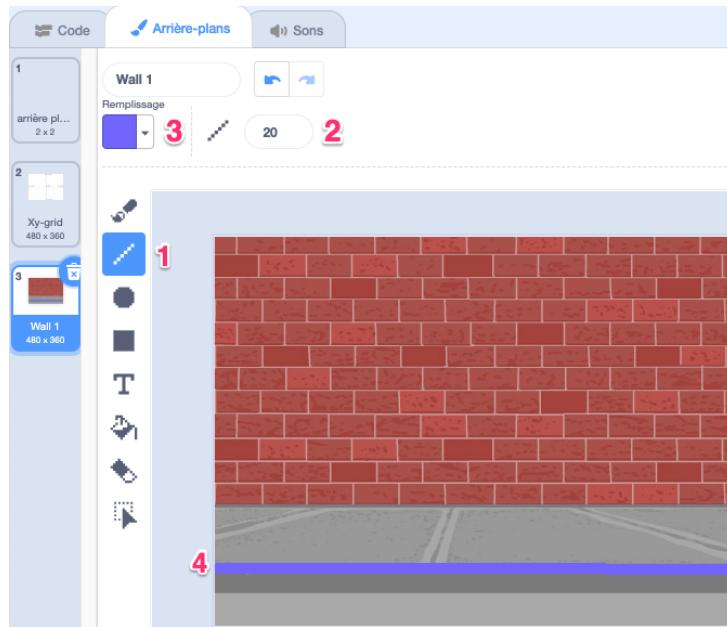


Vérifiez votre code : quand vous cliquez sur le drapeau vert et que la balle touche la raquette, elle rebondit.

Fin du jeu

Pour arrêter le jeu si la balle touche le bas, le plus simple est de tracer une ligne horizontale d'une couleur spécifique et d'utiliser le bloc **couleur touchée**.

Cliquer sur l'arrière-plan puis cliquer sur l'onglet **Arrière-plan**.



Sélectionner l'outil ligne ① puis choisir l'épaisseur ② et la couleur ③ du trait. Dessiner un trait ④ en bas de la scène.

Remarques :

- pour tracer un trait parfaitement horizontal, maintenir la touche **majuscule** (**shift**) enfoncee pendant que le trait est tracé ;
- si le trait tracé ne convient pas, il est possible de l'effacer en utilisant la touche

d'annulation de la dernière action .

Cliquer ensuite sur le sprite balle et ajouter un code avec le bloc **couleur touchée** pour arrêter la balle si elle touche la couleur de ligne. Une fois le bloc **couleur touchée** inséré, il faut cliquer sur le carré de couleur puis cliquer sur la ligne du bas de la scène pour sélectionner la bonne couleur. L'algorithme du code à construire est détaillé ci-dessous.

