Exercices d'introduction à la programmation informatique

Michel Di Salvo

28 juin 2016

Ces exercices sont prévus pour être réalisés sur le site internet de Scratch :

`scratch.mit.edu`

Scratch est un projet du Lifelong Kindergarten Group au sein du MIT Media Lab.

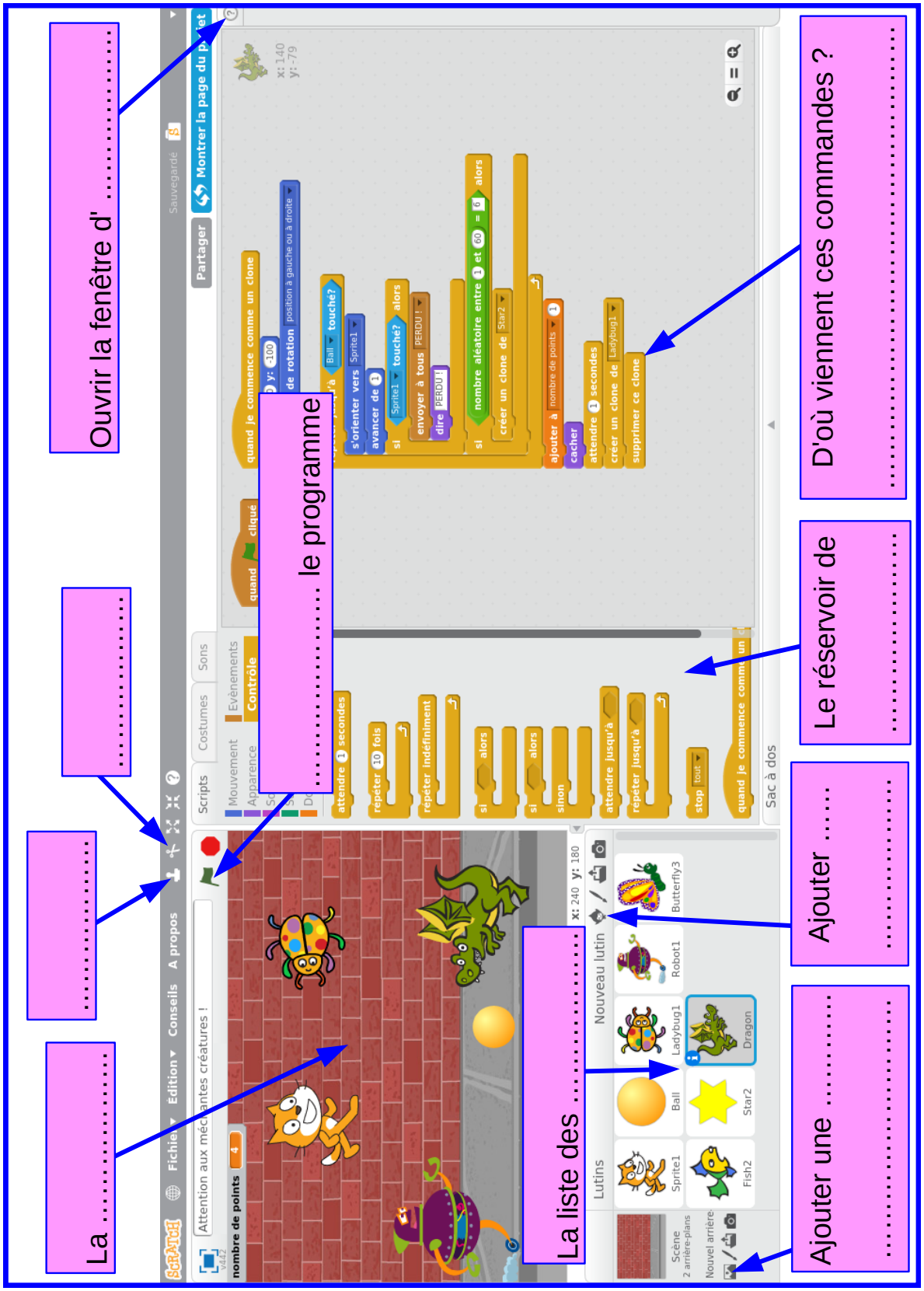
Ces exercices sont prévus pour être réalisés avec l'aide de l'Aide-mémoire disponible en seconde partie de ce document.

Michel Di Salvo, étudiant à la HEP Lausanne

Exercice 1 – A la découverte de Scratch

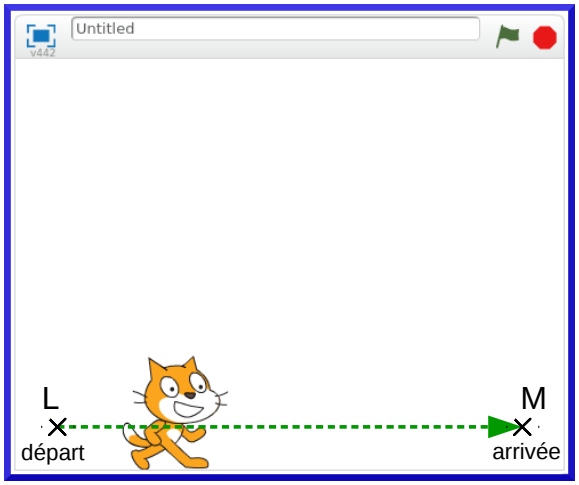
Voici une copie d'écran de l'espace de travail de Scratch. **Complète les lacunes :**

⟹ **Aide-mémoire**, paragraphes « **Scratch : comment ça fonctionne ?** » et « **L'espace de travail de Scratch** », pages 2 et 3.



Exercice 2 – Mon premier programme

Dans cet exercice, tu vas créer un programme qui met le chat en mouvement : le chat va traverser la scène du point **L** au point **M** :



Coordonnées des points :

L (−190 ; −130)
M (190 ; −130)

a) Va sur la **page principale de Scratch**.

b) **Crée un nouveau programme**.

⟹ **Aide-mémoire**, paragraphe « **Comment créer un nouveau programme ?** », page 5.

c) **Insère les commandes nécessaires** pour faire avancer le chat du point **L** au point **M**. La durée du déplacement est de **4 secondes**.

⟹ **Aide-mémoire**, paragraphe « **Comment mettre en mouvement une créature ?** », pages 7 à 10.

d) **Exécute le programme**. Observe ce qu'il se passe sur la scène, et corrige les erreurs éventuelles.

⟹ **Aide-mémoire**, paragraphe « **Comment exécuter un programme ?**, page 7.

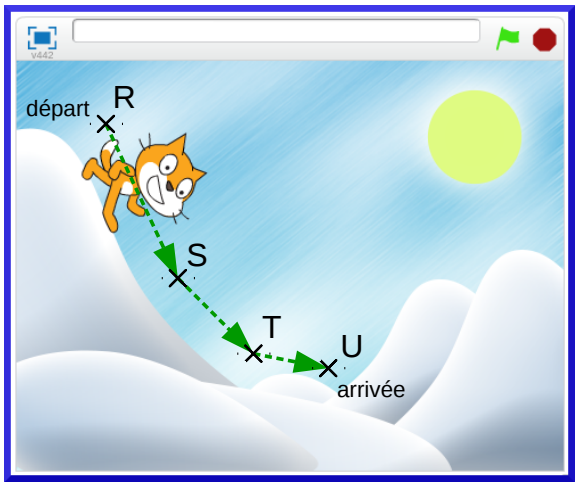
e) **Sauvegarde ton travail** sous le nom « **exercice 2** ».

⟹ **Aide-mémoire**, paragraphe « **Comment sauvegarder son travail ?** », page 6.

f) Quitte l'espace de travail et retourne sur la **page principale de Scratch**.

Exercice 3 – En montagne

Dans cet exercice, tu vas créer un programme dans lequel le chat glisse le long de la pente enneigée d'une montagne :



Coordonnées des points :

R (−160 ; 150)
S (−90 ; 0)
T (−20 ; −70)
U (30 ; −80)

Orientation du chat :

→ entre **R** et **S** : 155
→ entre **S** et **T** : 135
→ entre **T** et **U** : 101

a) Crée un nouveau programme.

b) **Insère les commandes nécessaires** pour faire avancer le chat le long du parcours présenté ci-dessus. Chaque déplacement dure **1 seconde**.

⟹ **Aide-mémoire**, paragraphe « **Comment faire avancer une créature le long d'un parcours ?** », page 11.

c) Exécute le programme, et vérifie que le chat se mette en mouvement.

d) Sauvegarde ton travail sous le nom « **exercice 3** ».

e) Ajoute l'**image d'arrière-plan** représentant les montagnes : il s'agit de l'image « **slopes** » qui se trouve dans le thème « **Vacances** ».

⟹ **Aide-mémoire**, paragraphe « **Comment insérer une image en arrière-plan ?** », pages 12 et 13.

f) Exécute le programme, et vérifie que le chat glisse le long de la montagne de gauche. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

g) Sauvegarde ton travail.

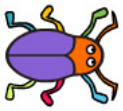
Exercice 4 – Le plus de tours possible

Dans cet exercice, tu vas créer un programme dans lequel une créature doit rester en mouvement le plus longtemps possible.

a) Crée un nouveau programme.

b) **Ajoute** l'insecte « **Beetle** » : cet insecte se trouve dans la catégorie « **Animaux** ».

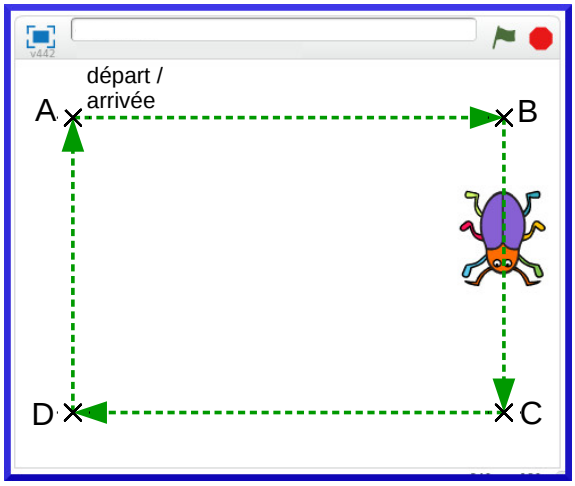
⟹ **Aide-mémoire**, paragraphe « **Comment insérer une nouvelle créature ou un nouvel objet ?** », pages 14 et 15.



c) **Supprime le chat**.

⟹ **Aide-mémoire**, paragraphe « **Comment supprimer une créature ?**, page 16.

d) **Insère les commandes nécessaires** pour que l'insecte effectue le parcours suivant :



- L'insecte part du point **A** (−190 ; 130) .
- Il avance ensuite jusqu'au point **B** (190 ; 130) .
- Puis il avance jusqu'au point **C** (190 ; −130) .
- Il avance ensuite jusqu'au point **D** (−190 ; −130) .
- Finalement il revient au point **A**.

Chaque déplacement dure **2 secondes**.

Informations sur l'orientation d'une créature :
⟹ **Aide-mémoire**, paragraphe « **Les commandes de mouvement** », tableau « **Les principales orientations** », page 17.

e) Exécute ton programme, et corrige les éventuelles erreurs jusqu'à ce que tout fonctionne correctement.

f) Sauvegarde ton travail sous le nom « **exercice 4** ».

Suite de l'exercice à la page suivante

g) **Modifie ton programme** pour que l'insecte continue à tourner **indéfiniment**, en effectuant autant de tours que possible, jusqu'à ce qu'on quitte le programme.

⟹ **Aide-mémoire**, paragraphe « **Les boucles** », pages 18 à 20.

h) Exécute le programme, et corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

i) Ajoute un **compteur de tours** qui **affiche** le **nombre de tours** effectués par l'insecte.

⟹ **Aide-mémoire**, paragraphe « **Les variables** », pages 21 et 22.

j) Exécute le programme, et vérifie que le nombre de tours affiché dans la scène soit correct. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

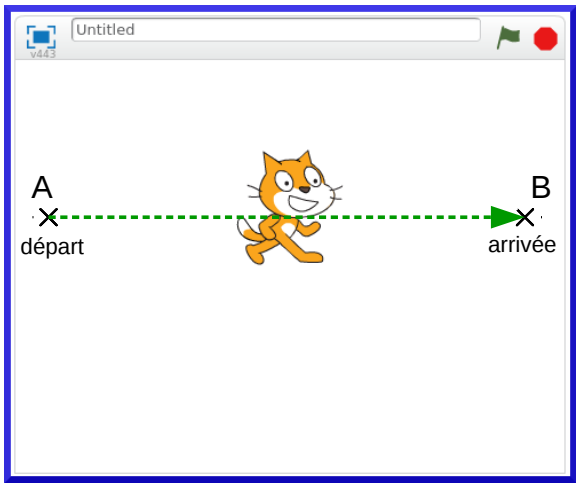
k) Sauvegarde ton travail.

Exercice 5 – Obstacle à éviter

Dans cet exercice, tu vas créer un programme dans lequel le chat traverse la scène de gauche à droite en contournant un obstacle.

a) Crée un nouveau programme.

b) Insère les commandes nécessaires pour faire avancer le chat du point **A** (−200 ; 0) au point **B** (200 ; 0) :



Tu as le choix de la durée du déplacement.

c) Exécute le programme, et vérifie que tout fonctionne correctement.

d) Sauvegarde ton travail sous le nom « **exercice 5** ».

e) Ajoute le bâtiment « **Marble Building** » : ce bâtiment se trouve dans la catégorie « **Choses** ».

⟹ **Aide-mémoire**, paragraphe « **Comment insérer une nouvelle créature ou un nouvel objet ?** », remarque 3) , page 14.

f) Place ce bâtiment **au centre** de la scène.

⟹ **Aide-mémoire**, remarque 1) , page 14.

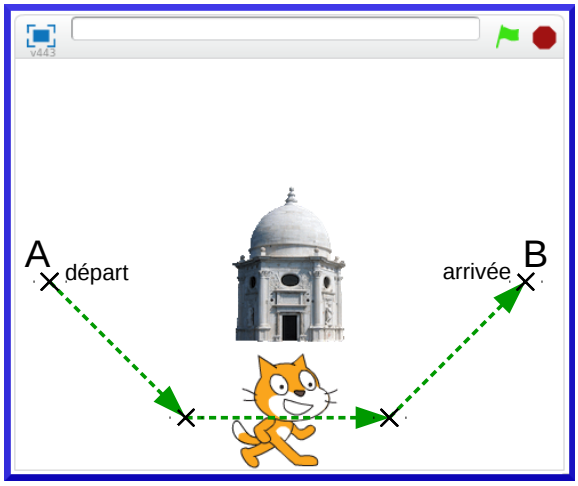
g) Où sont passées les commandes que tu avais insérées précédemment pour faire avancer le chat ? Fais le nécessaire pour **afficher** ces commandes à nouveau (il suffit d'**un seul** clique).

⟹ **Aide-mémoire**, remarque 2) , page 14.

Suite de l'exercice à la page suivante

h) Le bâtiment que tu viens d'insérer constitue un obstacle au passage du chat : **modifie ton programme** de manière à ce que le chat aille du point **A** au point **B** en contournant l'obstacle, sans le toucher.

Tu as le choix du parcours que le chat va emprunter : tu peux par exemple utiliser un parcours de ce type :



Tu peux modifier les commandes déjà insérées, et ajouter de nouvelles commandes.

Informations sur le système de coordonnées de Scratch :
⟹ **Aide-mémoire**, paragraphe « **Le système de coordonnées de Scratch** », page 4.

i) Exécute ton programme. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

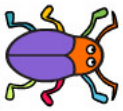
j) Sauvegarde ton travail.

Exercice 6 – Garde le contrôle !

Dans cet exercice, tu vas créer un programme dans lequel l'utilisateur pourra déplacer une créature à n'importe quel endroit de la scène en utilisant les **flèches du clavier**.

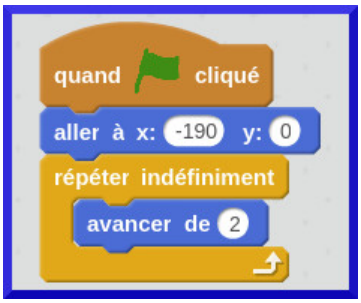
a) Crée un nouveau programme.

b) Ajoute l'insecte « **Beetle** » (catégorie « **Animaux** ») :



c) Supprime le chat.

d) Insère les commandes suivantes :



e) Exécute le programme, et observe ce qu'il se passe. Explique ce qu'il se passe :

.....

.....

.....

f) **Modifie ton programme** pour que l'insecte avance **uniquement** lorsqu'on appuie sur la **flèche droite** du clavier.

⟹ **Aide-mémoire**, paragraphe « **Les structures de contrôle** », pages 23 à 25.

g) Exécute le programme, et observe ce qu'il se passe. S'il ne se passe rien, appuie sur la **flèche droite** du clavier, et vérifie que l'insecte se déplace vers la droite.

h) **Modifie ton programme** de manière à pouvoir déplacer l'insecte à n'importe quel endroit de la scène en utilisant les **flèches du clavier** :

⟹ La flèche droite fait avancer l'insecte vers la droite.

⟹ La flèche gauche fait avancer l'insecte vers la gauche.

⟹ La flèche haut fait avancer l'insecte vers le haut.

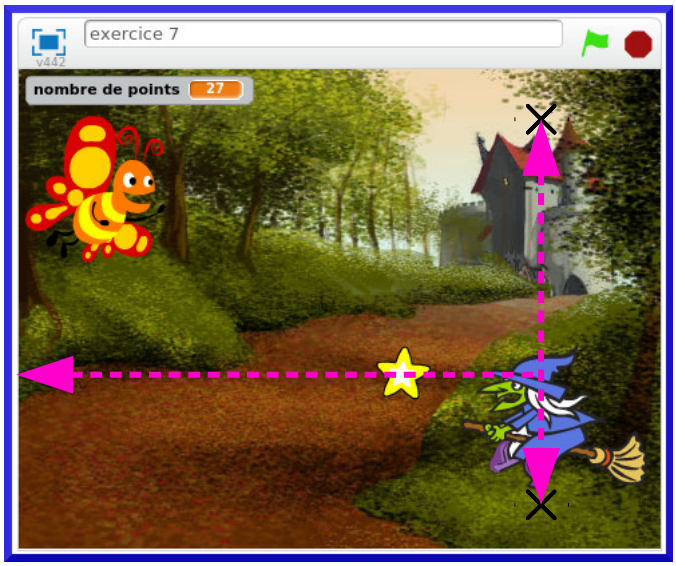
⟹ La flèche bas fait avancer l'insecte vers le bas.

i) Exécute le programme, et observe ce qu'il se passe lorsque tu appuies sur les flèches du clavier. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

j) Sauvegarde ton programme sous le nom « **exercice 6** ».

Exercice 7 – Mon premier jeu

Dans cet exercice, tu vas créer un jeu dans lequel une sorcière tire des étoiles sur un papillon. Le joueur doit déplacer le papillon en utilisant les flèches du clavier et en évitant de se faire toucher par les étoiles tirées par la sorcière. La sorcière est continuellement en mouvement : elle monte et elle descend. Le joueur marque 1 point à chaque fois qu'il évite une étoile. Le but du jeu est d'obtenir le plus de points possible. La partie se termine lorsque le joueur se fait toucher par une étoile :



Ce jeu contient :

- l'**image d'arrière-plan** « **castle3** » qui se trouve dans le thème « **Château** »,
- le papillon « **Butterfly2** » qui se trouve dans la catégorie « **Animaux** »,
- la sorcière « **Witch** » qui se trouve dans la catégorie « **Fantaisie** »,
- l'étoile « **Star1** » qui se trouve dans la catégorie « **Choses** ».

a) Crée un nouveau programme.

b) **Construis le jeu présenté ci-dessus.**

⟹ **Aide-mémoire**, paragraphe « **Un exemple de conception de jeu** », pages 27 à 34.

c) Exécute le programme, et évite les étoiles tirées par la sorcière. Vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

d) Sauvegarde ton travail sous le nom « **exercice 7** ».

Exercice 8 – De plus en plus réaliste

Dans cet exercice, tu vas créer un programme dans lequel un personnage marche sur un terrain de sport. Les graphismes seront plus réalistes que dans les exercices précédents : les jambes, les pieds et les bras du personnage seront en mouvement pendant qu'il marche. Pour trouver comment faire, tu devras utiliser l'**aide de Scratch**.

a) Crée un nouveau programme.

b) Ajoute le personnage « **Avery Walking** » qui se trouve dans la catégorie « **Gens** ». La taille d'Avery doit correspondre à **80%** de sa taille initiale. Place Avery **au centre** de la scène, c'est-à-dire au point de coordonnées (0 ; 0) .

Modification de la taille d'une créature :
⟹ **Aide-mémoire**, paragraphe « **Les commandes d'apparence** », page 26.

c) Supprime le chat.

d) Insère **l'image d'arrière-plan** « **playing-field** » qui se trouve dans le thème « **Sport** » :

e) Ouvre **la fenêtre d'aide de Scratch** et clique sur « **Comment faire** ».
Sélectionne « **Animations** » et clique sur « **Faire marcher le lutin** ».

⟹ **Aide-mémoire**, paragraphe « **Comment obtenir de l'aide ?** », page 39.

f) **Insère les commandes nécessaires** pour faire marcher Avery de manière à ce que **ses jambes bougent** : aide-toi de la **fenêtre d'aide de Scratch**.

A chaque étape :

- l'avancement est de **25**
- l'attente est de **0,2 seconde** (on utilise **un point** à la place de la virgule).

g) Exécute le programme et vérifie qu'Avery marche correctement. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

h) Sauvegarde ton travail sous le nom « **exercice 8** ».

Exercice 9 – Le logiciel de dessin

Dans cet exercice, tu vas créer un programme de dessin : l'utilisateur du programme pourra dessiner des traits dans la scène en appuyant sur la barre d'espace du clavier et en déplaçant le pointeur de souris. L'utilisateur pourra choisir la couleur des traits qu'il dessine en appuyant sur les touches r (rouge), o (orange), j (jaune), v (vert), t (turquoise), b (bleu) et m (magenta). Différents traits de différentes couleurs pourront être dessinés. La touche e permettra d'effacer le dessin (avant la suppression du dessin, l'utilisateur devra confirmer son intention).

a) Crée un nouveau programme.

b) Ajoute le stylo « **Pencil** » qui se trouve dans la catégorie « **Choses** ».

c) Supprime le chat.

d) Au lancement du programme, les exigences suivantes doivent être satisfaites :

- La taille du stylo doit correspondre à **20%** de sa taille initiale.
  ⟹ **Aide-mémoire**, paragraphe « **Les commandes d'apparence** », page 26.
- L'épaisseur des traits est mise à **2**.
  ⟹ **Aide-mémoire**, paragraphe « **Les commandes de dessin** », page 41.

Insère les commandes nécessaires pour satisfaire ces exigences.

e) Pendant l'exécution du programme, les exigences suivantes doivent être satisfaites **en permanence** :

- Le stylo suit en permanence le **pointeur de souris**.
  ⟹ **Aide-mémoire**, paragraphe « **Les commandes de mouvement** », pages 17 et 18.
- **Si** la barre d'espace du clavier est appuyée, **alors** le stylo est placé en position d'écriture. **Sinon** le stylo est relevé.
  ⟹ **Aide-mémoire**, paragraphe « **Les commandes de dessin** », page 41.

Insère les commandes nécessaires pour satisfaire ces exigences.

f) Exécute le programme et vérifie que tu puisses dessiner des traits en appuyant sur la barre d'espace du clavier et en déplaçant le pointeur de souri sur la scène (la barre d'espace doit rester appuyée pendant le traçage d'un trait). Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

g) Sauvegarde ton travail sous le nom « **exercice 9** ».

Suite de l'exercice à la page suivante

h) **Ajoute les structures de contrôle** permettant de modifier la couleur du stylo en appuyant sur les touches r (rouge), o (orange), j (jaune), v (vert), t (turquoise), b (bleu) et m (magenta).

⟹ **Aide-mémoire**, paragraphe « **Les commandes de dessin** », page 41.

i) Exécute le programme et vérifie que tu puisses changer la couleur du stylo. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

j) **Ajoute une structure de contrôle** permettant de tout effacer si on appuie sur la touche **e**. Avant la suppression du dessin, l'utilisateur devra confirmer son intention :

> **Si** la touche e est pressée, **alors** :
>
> - Le programme demande à l'utilisateur « Es-tu sûr de vouloir effacer ton dessin ? oui/non » et attend sa réponse.
> - **Si** la réponse est « oui », **alors** tout est effacé.

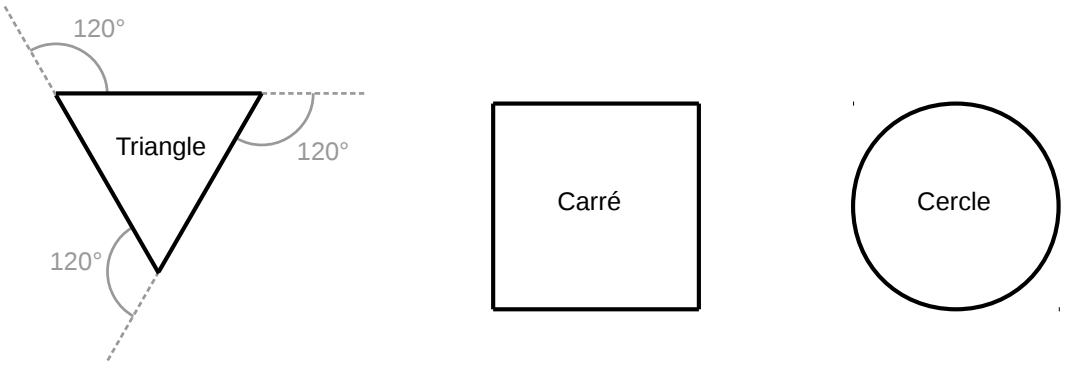
⟹ **Aide-mémoire**, paragraphe « **Comment interagir avec l'utilisateur du programme ?** », page 40.

k) Exécute le programme. Appuie sur la touche **e**, et vérifie que le dessin est effacé si tu réponds « **oui** ». Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

l) Sauvegarde ton travail.

Facultatif :

Ajoute les commandes nécessaires pour pouvoir dessiner des formes simples, comme par exemple, des triangles, des carrés et des cercles : un triangle se dessine lorsqu'on appuie sur la touche x, un carré se dessine lorsqu'on appuie sur la touche y, et un cercle se dessine lorsqu'on appuie sur la touche z.

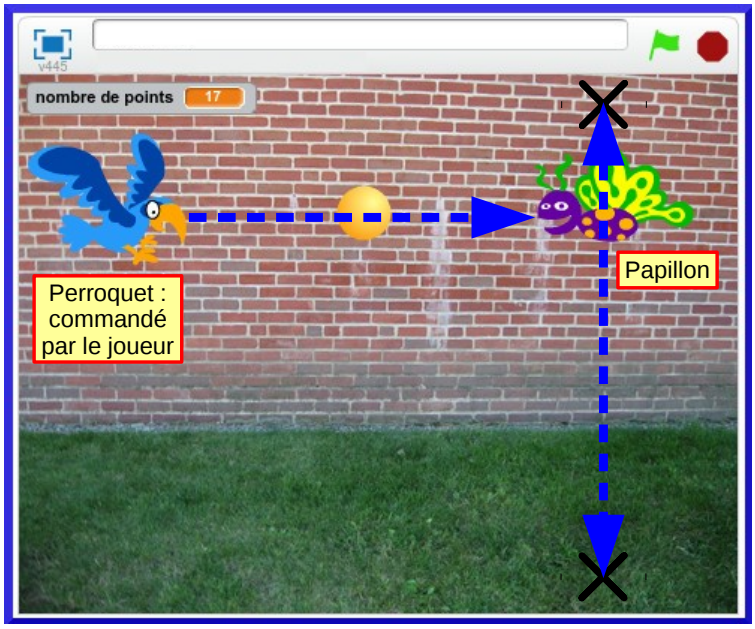


Exécute le programme et vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles jusqu'à que tout fonctionne correctement. Sauvegarde ton travail.

Exercice 10 – Perroquet contre papillon

Dans cet exercice, tu vas créer un jeu dans lequel un perroquet tire des balles sur un papillon. Le joueur déplace le perroquet et tire les balles sur le papillon. Voici les règles du jeu :

- **Le perroquet** se trouve sur la gauche de la scène : **le joueur** peut faire **monter ou descendre le perroquet** en appuyant sur **les flèches haut et bas du clavier**.
- **Le perroquet tire une balle** à chaque fois que le joueur appuie sur **la barre d'espace du clavier** : les balles sont tirées vers la droite.
- Lorsque le joueur tire une balle, il doit attendre **0,5 seconde** avant de pouvoir tirer la balle suivante.
- **Le papillon** se trouve sur la droite de la scène : **il monte et il descend indéfiniment**.
- Le joueur obtient des points à chaque fois que le papillon se fait toucher par une balle.
- Le but du jeu est d'obtenir le plus de points possible.



Ce jeu contient :

- l'image d'arrière-plan « **brick wall2** » qui se trouve dans le thème « **Ville** »,
- le perroquet « **Parrot2** » qui se trouve dans la catégorie « **Animaux** »,
- le papillon « **Butterfly1** » qui se trouve dans la catégorie « **Animaux** »,
- l'objet « **Ball** » qui se trouve dans la catégorie « **Choses** ».

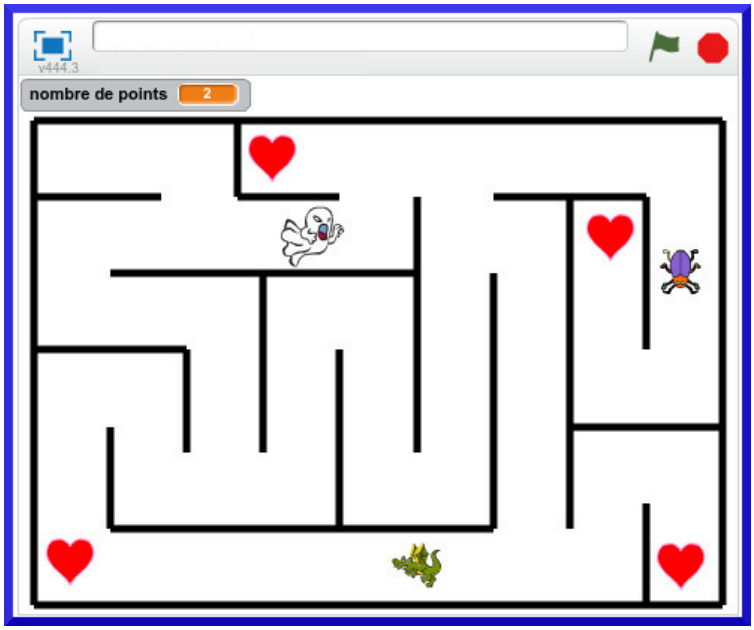
Crée un nouveau programme, et **construis le jeu présenté ci-dessus**. Exécute le programme, et vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement. Sauvegarde ton travail sous le nom « **exercice 10** ».

Facultatif :

Ajoute les commandes nécessaires pour que **le papillon tire des pommes** sur le perroquet : les pommes sont tirées vers la gauche. Le joueur perd des points à chaque fois qu'il se fait toucher par une pomme. La partie se termine si le nombre de points devient négatif.

Exercice 11 – Le labyrinthe hanté

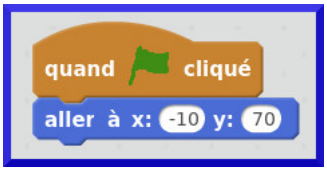
Dans cet exercice, tu vas créer un jeu dans le lequel un insecte, un fantôme et un dragon se déplacent à l'intérieur d'un labyrinthe (le fantôme peut franchir les parois du labyrinthe). Le joueur doit déplacer l'insecte en utilisant les flèches du clavier et en évitant de se faire toucher par le fantôme ou par le dragon. Le but du jeu est de récupérer tous les cœurs qui se trouvent à l'intérieur du labyrinthe. Le joueur marque 1 point à chaque fois qu'il récupère un cœur. La partie se termine lorsque le joueur se fait toucher par le fantôme ou par le dragon :



Partie 1 – Les préparatifs :

a) Crée un nouveau programme.

b) Télécharge le fichier « **labyrinthe.svg** » depuis l'ordinateur vers Scratch : utilise l'icône « Importer un lutin depuis un fichier » à droite de l'étiquette « Nouveau lutin ». Ajoute ensuite les commandes suivantes pour positionner le labyrinthe précisément :



Exécute le programme pour valider le positionnement du labyrinthe.

c) Supprime le chat.

d) Ajoute un **compteur de points** : crée une nouvelle variable appelée « **nombre de points** ».

e) Sauvegarde ton travail sous le nom « **exercice 11** ».

Suite de l'exercice à la page suivante

Partie 2 – L'insecte :

f) Ajoute l'insecte « **Beetle** » qui se trouve dans la catégorie « **Animaux** ».

g) Au lancement du programme, les exigences suivantes doivent être satisfaites :

- L'insecte doit se positionner au point de coordonnées (−200 ; 125).
- La taille de l'insecte doit correspondre à **35%** de sa taille initiale.

⟹ **Aide-mémoire**, paragraphe « **Les commandes d'apparence** », page 26.

- L'insecte doit être **visible**.

⟹ **Aide-mémoire**, paragraphe « **Les commandes d'apparence** », page 26.

- Le **compteur de points** doit être remis **à zéro**.

Insère les commandes nécessaires pour satisfaire chacune de ces exigences.

h) Pendant le jeu, le joueur déplace l'insecte dans le labyrinthe en utilisant les flèches du clavier. **Ajoute les commandes nécessaires** pour que le joueur puisse déplacer l'insecte dans le labyrinthe.

⟹ **Aide-mémoire**, paragraphe « **Comment faire avancer une créature à l'intérieur d'un labyrinthe ?** », cas 1) , page 35.

i) Exécute le programme et vérifies que tu puisse déplacer l'insecte à l'intérieur du labyrinthe en utilisant les flèches du clavier. Vérifie qu'il soit impossible de traverser les parois du labyrinthe. Corrige les erreurs éventuelles jusqu'à ce que tout fonctionne correctement.

Partie 3 – Le fantôme :

j) Ajoute le fantôme « **Ghost2** » qui se trouve dans la catégorie « **Fantaisie** ».

k) Lors du lancement du programme, les exigences suivantes doivent être satisfaites :

- Le fantôme doit se positionner au point de coordonnées (0 ; 0).
- Le fantôme doit se comporter selon le style de rotation « **position à gauche ou à droite** ».

⟹ **Aide-mémoire**, paragraphe « **Les commandes de mouvement** », pages 17 et 18.

- La taille du fantôme doit correspondre à **30%** de sa taille initiale.

Insère les commandes nécessaires pour satisfaire chacune de ces exigences.

l) Pendant le jeu, le fantôme se déplace de manière aléatoire en traversant les parois du labyrinthe. **Ajoute les commandes nécessaires** pour que le fantôme se déplace de manière aléatoire dans la scène.

⟹ **Aide-mémoire**, paragraphe « **Comment faire avancer une créature de manière aléatoire ?** », page 38.

m) Exécute le programme, et vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles.

Suite de l'exercice à la page suivante

Partie 4 – Le dragon :

n) Ajoute le dragon « **Dragon** » qui se trouve dans la catégorie « **Fantaisie** ».

o) Lors du lancement du programme, les exigences suivantes doivent être satisfaites :

- Le dragon doit se positionner au point de coordonnées (-200 ; 25).
- Le dragon doit s'orienter **vers la droite**.
- Le dragon doit se comporter selon le style de rotation « **position à gauche ou à droite** ».
- La taille du dragon doit correspondre à **25%** de sa taille initiale.

Insère les commandes nécessaires pour satisfaire chacune de ces exigences.

p) Pendant le jeu, le dragon se déplace de manière autonome à l'intérieur du labyrinthe. **Ajoute les commandes nécessaires** pour que le dragon se déplace à l'intérieur du labyrinthe de manière autonome.

⟹ **Aide-mémoire**, paragraphe « **Comment faire avancer une créature à l'intérieur d'un labyrinthe ?** », cas 2) , page 36.

q) Exécute le programme, et vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles.

Partie 5 – Les cœurs à récupérer :

r) Ajoute le cœur « **Heart** » qui se trouve dans la catégorie « **Choses** »). La taille du cœur doit correspondre à **25%** de sa taille initiale.

s) Pendant le jeu, **le joueur** doit récupérer les cœurs en les touchant. Lorsqu'il récupère un cœur, le nombre de points augmente. Ajoute les commandes nécessaires à la récupération des cœurs. Duplique le cœur autant de fois que tu le souhaites, et place les cœurs obtenus à différents endroits du labyrinthe.

⟹ **Aide-mémoire**, paragraphe « **Comment insérer des objets que le joueur doit récupérer ?** », pages 37 et 38.

t) Exécute le programme, et fais avancer l'insecte pour récupérer tous les cœurs que tu as insérés. Vérifie que tout fonctionne correctement. Corrige les erreurs éventuelles.

Suite de l'exercice à la page suivante

Partie 6 – La fin de la partie :

u) La partie se termine si l'insecte se fait toucher par le fantôme : sélectionne le **fantôme**, et ajoute **la structure de contrôle** nécessaire à l'arrêt du jeu :

> **Si** l'**insecte** est touché, **alors** :
> - Le fantôme envoie le message « **PERDU !** ».
> - Le fantôme dit « **PERDU !** ».

v) De même, la partie se termine si l'insecte se fait toucher par le dragon : sélectionne le **dragon**, et ajoute **la structure de contrôle** nécessaire à l'arrêt du jeu :

> **Si** l'**insecte** est touché, **alors** :
> - Le dragon envoie le message « **PERDU !** ».
> - Le dragon dit « **PERDU !** ».

w) Lorsque le joueur a perdu la partie, l'insecte disparaît, et le joueur ne peut plus récupérer de cœurs : sélectionne l'**insecte**, et **ajoute les commandes nécessaires** pour que l'insecte disparaisse lorsqu'il reçoit le message « **PERDU !** ».

x) Exécute le programme, et corrige les erreurs éventuelles.

y) Sauvegarde ton travail.

Exercice 12 – La somme infinie

Marina, qui est très douée en math, a proposé un défi à ses camarades de classe :

> « Je suis sûr que personne n'arrivera à deviner à quel nombre correspond la somme suivante :
>
> $$\frac{8}{1\cdot(1+2)}+\frac{8}{5\cdot(5+2)}+\frac{8}{9\cdot(9+2)}+\frac{8}{13\cdot(13+2)}+\frac{8}{17\cdot(17+2)}+\frac{8}{21\cdot(21+2)}+\ldots$$
>
> et il faut continuer à additionner comme ça indéfiniment ! »

Julie, la meilleure amie de Marina, a une idée pour calculer cette somme : elle va créer un programme informatique qui va calculer cette somme à l'aide d'une boucle. Julie découvre qu'en utilisant une variable **N**, cette somme peut s'écrire de la manière suivante :

$$\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 1}}+\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 5}}+\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 9}}+\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 13}}+\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 17}}+\underbrace{\frac{8}{N\cdot(N+2)}}_{\textbf{N vaut 21}}+\ldots$$

- Au début **N vaut 1**,
- Ensuite **N augmente de 4** à chaque étape.

Avant de commencer à réaliser son programme, Julie fait un schéma :

Au lancement du programme :

- Mettre la somme à 0.
- Mettre N à 1.

Après :

- Ajouter $\frac{8}{N\cdot(N+2)}$ à la somme.
- Ajouter 4 à N.

} A répéter indéfiniment !

A toi de créer le programme permettant de calculer cette somme :

a) Crée un nouveau programme.

b) Crée une nouvelle variable appelée « **somme** ».

c) Crée une nouvelle variable appelée « **N** ».

d) **Insère les commandes nécessaires** pour réaliser le programme proposé par Julie.
⟹ **Aide-mémoire**, paragraphe « **Les opérateurs mathématiques** », page 42.

e) Exécute le programme et vérifie que la valeur indiquée par la variable « **somme** » se stabilise après quelques secondes.

f) A quel nombre la somme proposée par Marina correspond-t-elle ? .....

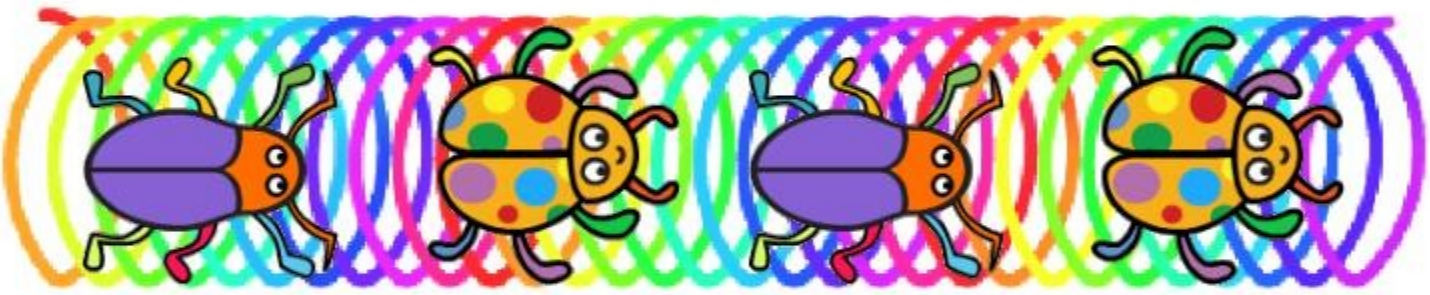
g) Sauvegarde ton travail sous le nom « **exercice 12** ».

h) Combien d'additions faut-il approximativement réaliser avant d'obtenir le bon résultat ? Ce calcul serait-il faisable sans ordinateur ?

Aide-mémoire

La programmation informatique avec Scratch

M. Di Salvo



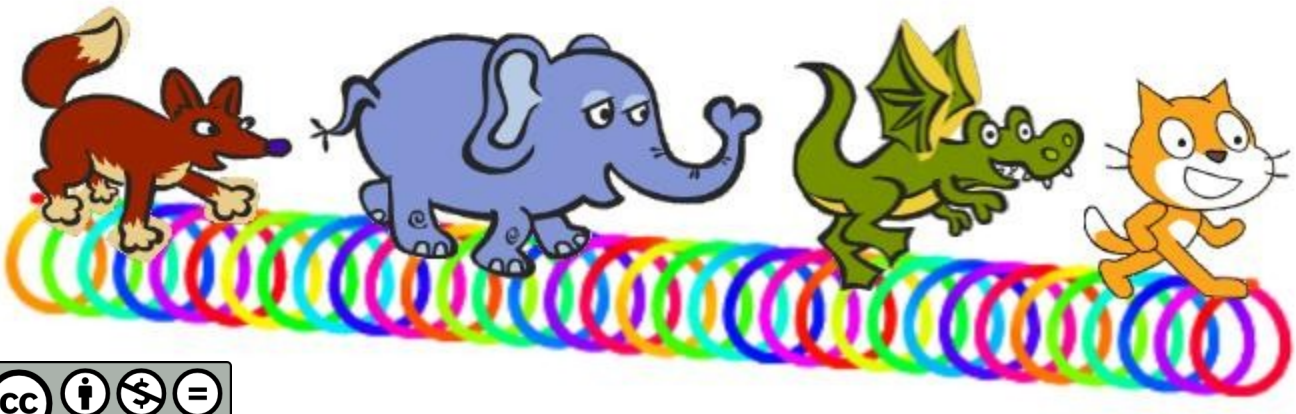
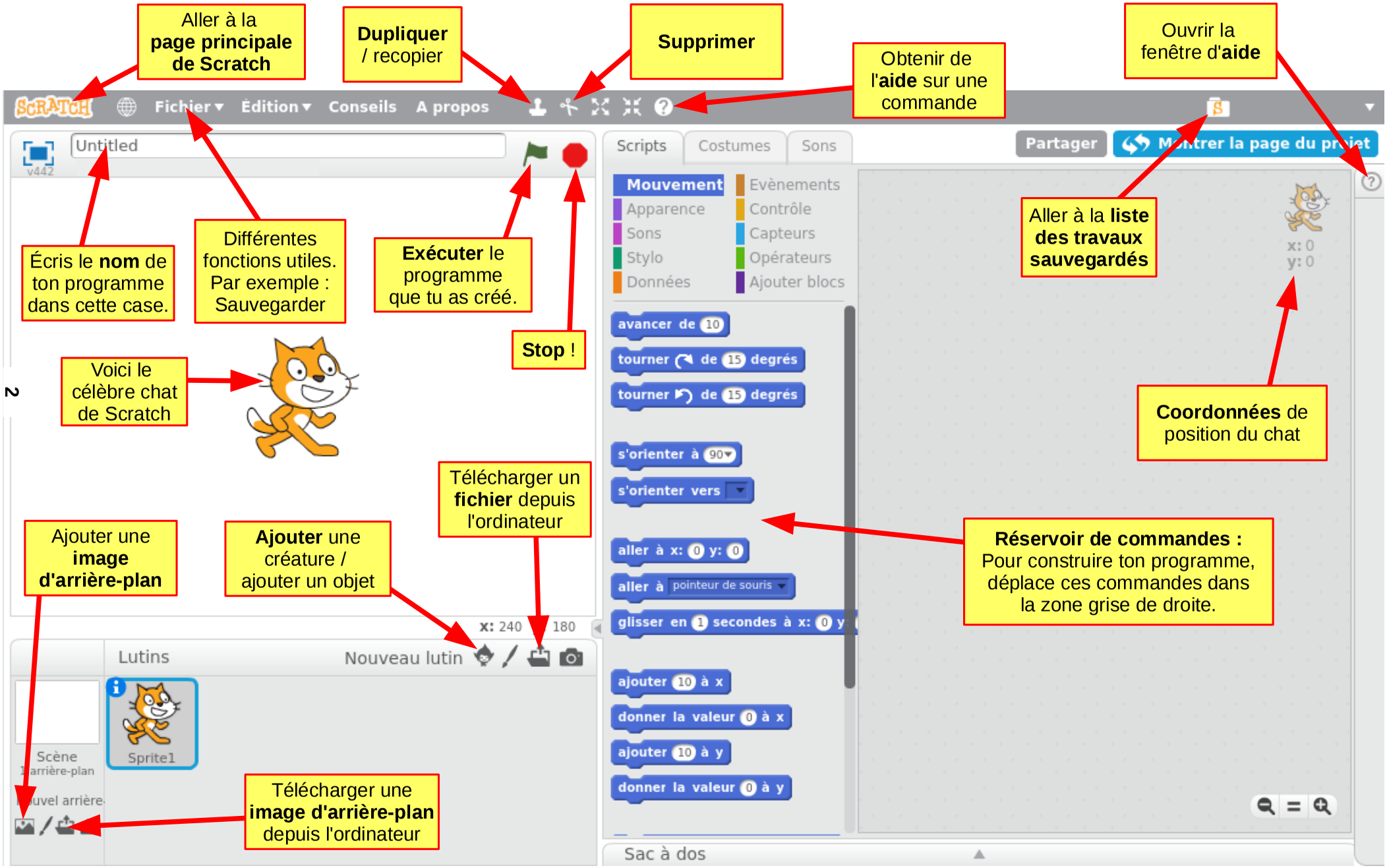


Table des matières

Les images contenues dans ce document sont tirées du site internet de Scratch. Scratch est un projet du Lifelong Kindergarten Group au sein du MIT Media Lab. Voir `scratch.mit.edu` .

Scratch : comment ça fonctionne ?



L'espace de travail de Scratch :

La barre du haut :

Elle contient des icônes utiles.

La liste des Rubriques :

Les différentes rubriques de commandes sont listées ici. Clique sur la rubriques qui contient les commandes dont tu as besoin.

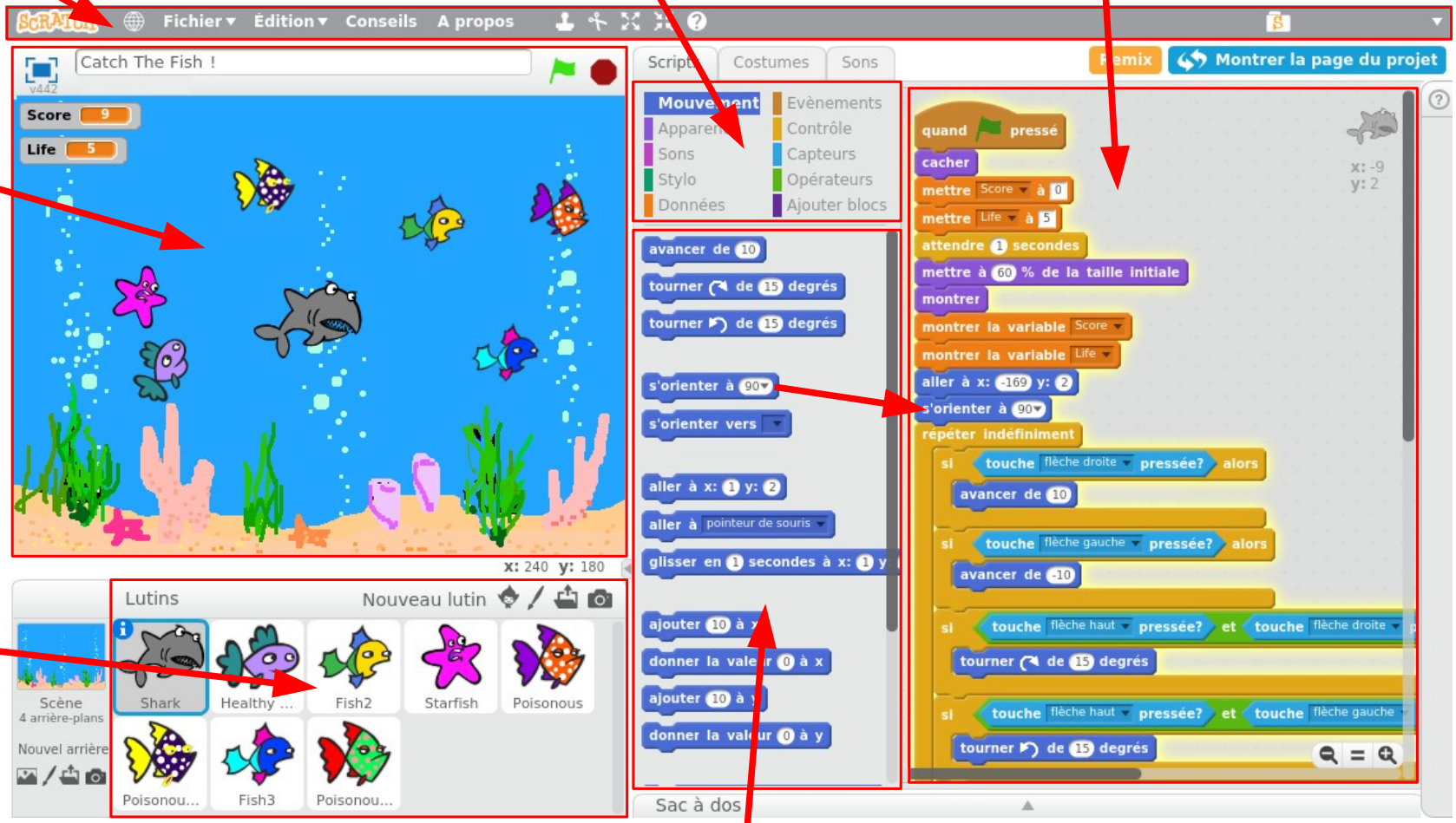
La zone grise de droite :

C'est ici que tu construis ton programme : place les commandes dans cette zone.

La Scène :

C'est ici que le programme que tu construis s'affichera.

C'est ici que tu pourras jouer au jeu que tu construis.



La liste des Créatures :

Les créatures que tu insères dans ton programme s'afficheront ici.

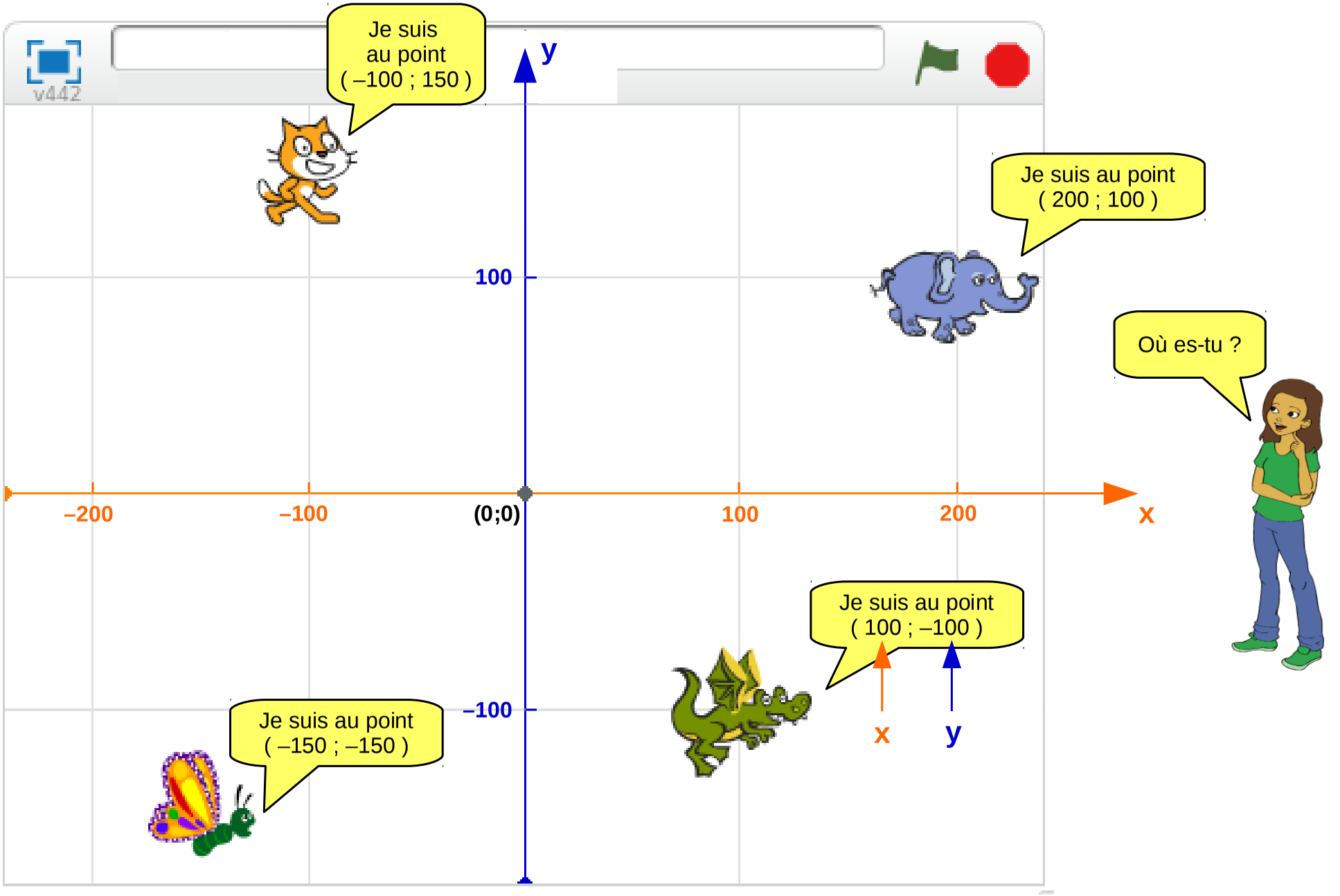
Lorsque tu construis ton programme, clique sur la créature sur laquelle tu aimerais travailler.

Le réservoir de Commandes :

Les commandes à utiliser pour construire ton programme se trouvent ici.

Pour construire ton programme, prends les commandes dans ce réservoir, et déplace-les dans la zone grise de droite.

Le système de coordonnées de Scratch :



Comment accéder à Scratch ?

Pour accéder à Scratch, on ouvre une navigateur internet, et on va à l'adresse internet suivante :

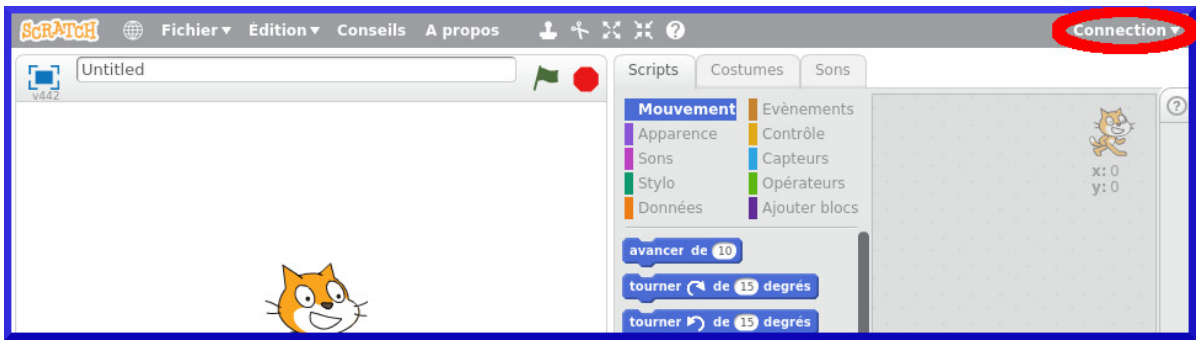
scratch.mit.edu

On arrive ainsi sur la **page principale de Scratch**.

Comment se connecter à son compte Scratch ?

Pour se connecter depuis la page principale de Scratch, on clique sur « **Se connecter** » en haut à droite de la fenêtre :

Pour se connecter depuis l'espace de travail de Scratch, on clique sur « **Connexion** » en haut à droite de la fenêtre :



On écrit ensuite son **Nom d'utilisateur** et son **Mot de passe** dans les cases prévues à cet effet, puis on clique sur « **Se connecter** ».

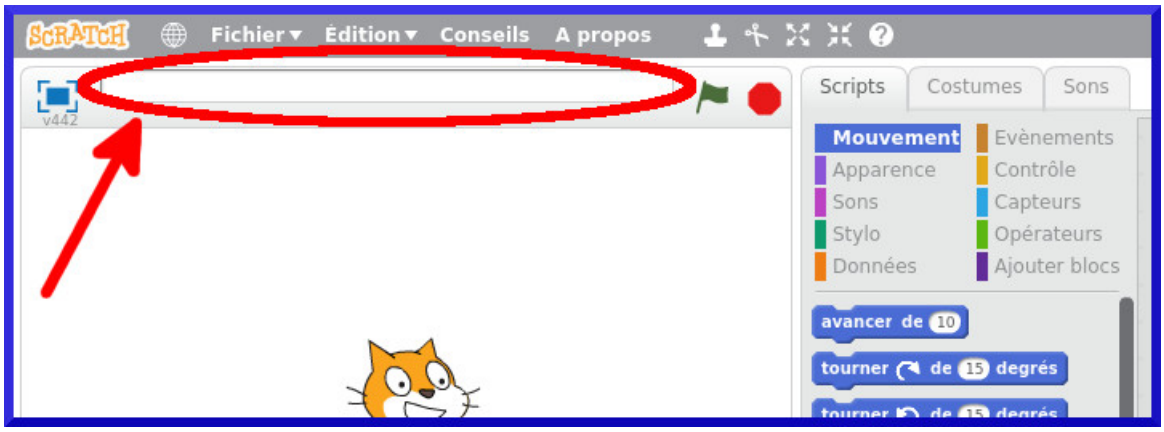
Comment créer un nouveau programme ?

Pour créer un nouveau programme, on va sur la page principale de Scratch, et on clique sur « **Créer** » :

L'**espace de travail** de Scratch se charge, puis s'affiche.

Comment sauvegarder son travail ?

Pour sauvegarder son travail, on écrit le nom sous lequel on souhaite sauvegarder son travail dans la case prévue à cet effet :

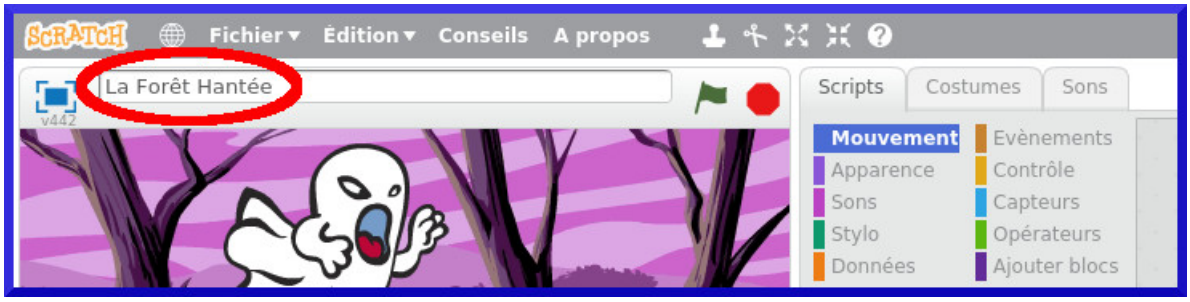


Ensuite, on clique sur « **Fichier** », puis sur « **Sauvegarder maintenant** ».

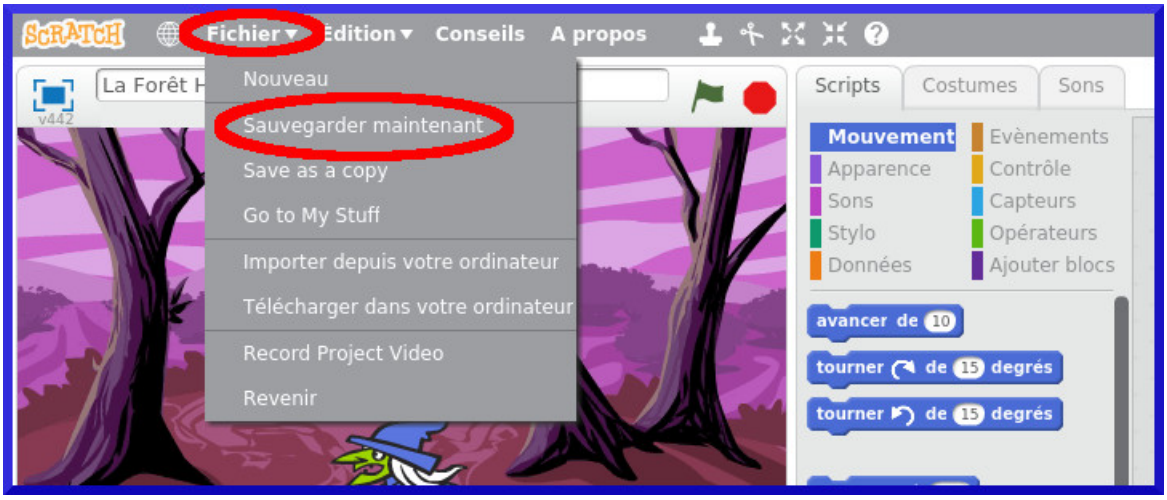
Exemple :

On a créé un jeu, et on souhaite sauvegarder ce jeu sous le nom « La Forêt Hantée » :

1) On écrit « La Forêt Hantée » dans la case prévue à cet effet :



2) On clique ensuite sur « **Fichier** », puis sur « **Sauvegarder maintenant** » :

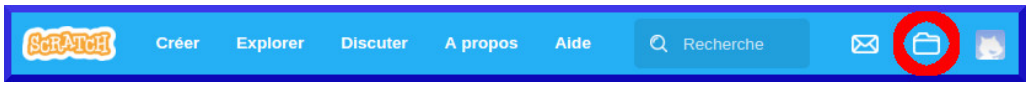


Remarque importante :

Pour pouvoir sauvegarder son travail, il faut obligatoirement **être connecté** à son compte Scratch.

Comment accéder aux travaux qui ont été sauvegardés ?

Pour accéder aux travaux qui ont été sauvegardés, on va sur la page principale de Scratch, et on clique sur l'icône  :



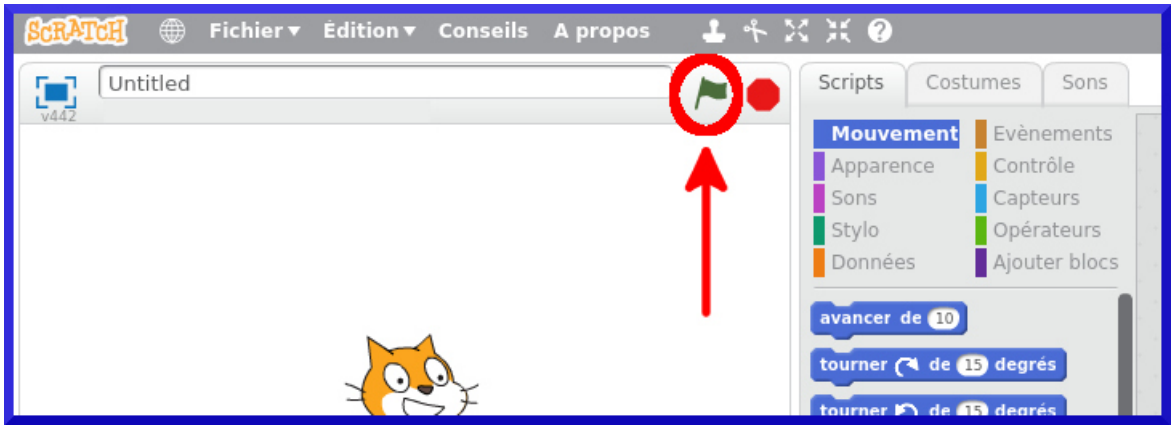
Une nouvelle fenêtre contenant la liste des travaux sauvegardés apparaît. On sélectionne le travail auquel on souhaite accéder, puis on clique sur « **Voir à l'intérieur** ».

Remarque :

Pour pouvoir accéder aux travaux sauvegardés, il faut obligatoirement **être connecté** à son compte Scratch.

Comment exécuter un programme ?

Pour exécuter un programme, on clique sur l'icône  :



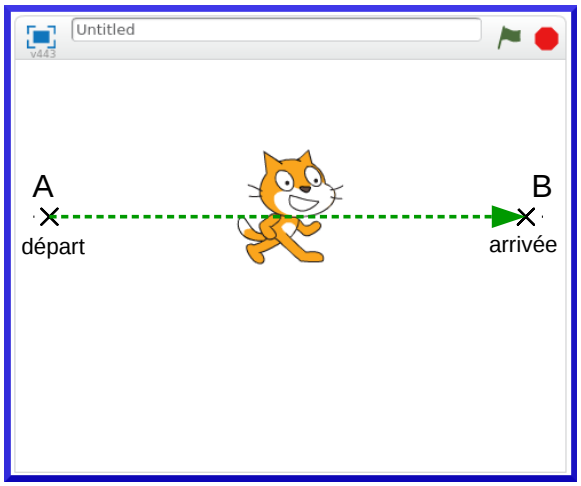
Comment mettre en mouvement une créature ?

Pour mettre une créature en mouvement, on insére les commandes nécessaires dans la zone grise de droite, puis on clique sur l'icône  .

Voir l'exemple à la page suivante

Exemple :

Dans cet exemple, un chat va traverser la scène de gauche à droite :



Coordonnées des points :

A (-190 ; 52)
B (190 ; 52)

1) Tout d'abord, on insère la commande permettant le lancement du programme. Il s'agit de la commande 

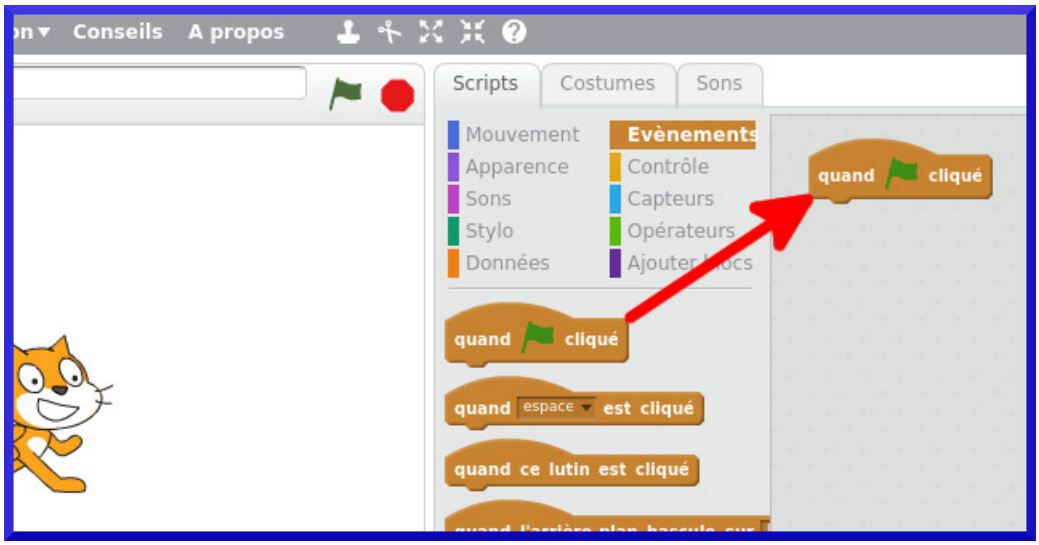
. Cette commande se trouve dans la catégorie « **Événements** » :

$\Longrightarrow$ On clique sur « **Événements** » pour afficher les commandes d'événements :

Les commandes d'événements apparaissent dans le réservoir de commandes.

$\Longrightarrow$ On place la commande 

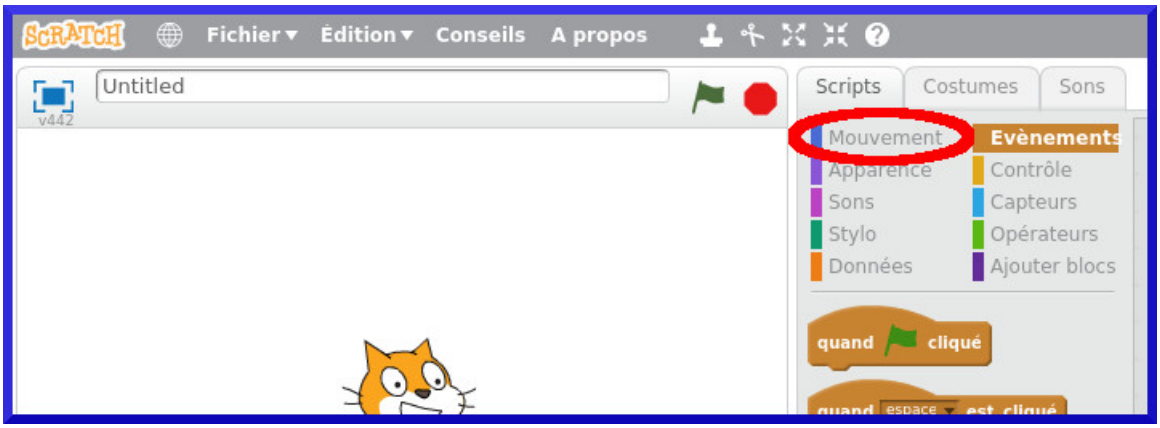
dans la zone grise de droite :



2) Ensuite on insère la commande qui va placer le chat au point de départ. Il s'agit de la commande 

. Cette commande se trouve dans la rubrique « **Mouvement** » :

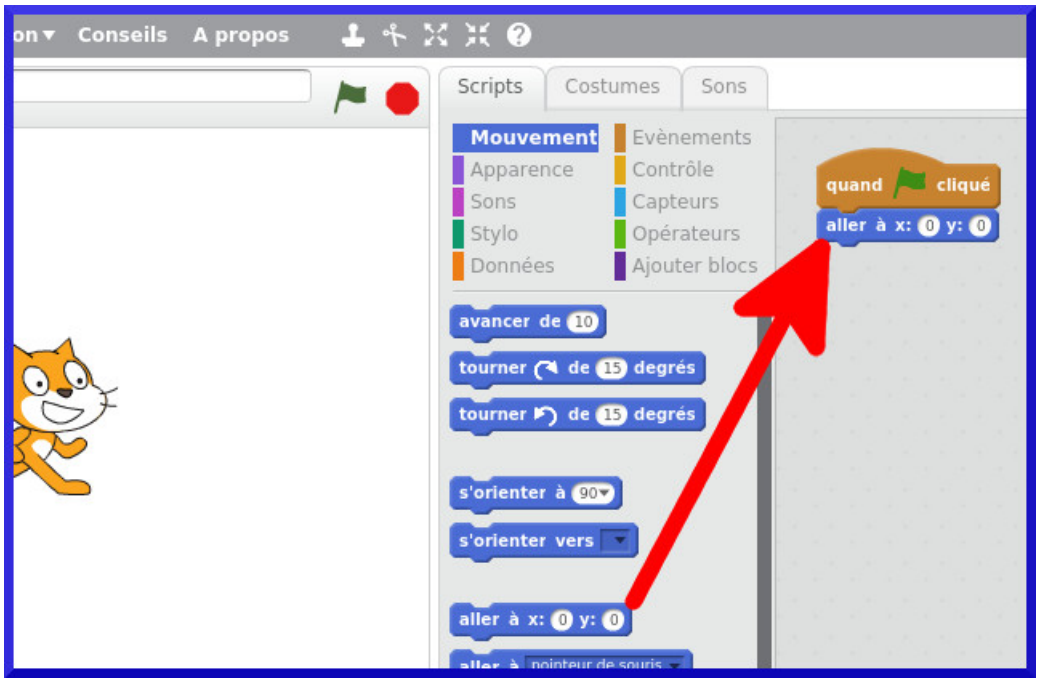
$\Longrightarrow$ On clique sur « **Mouvement** » pour afficher les commandes de mouvement :



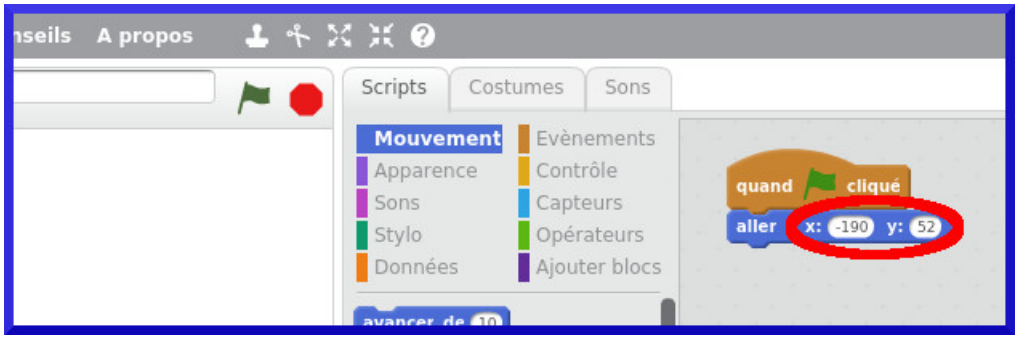
Les commandes de mouvement apparaissent dans le réservoir de commandes.

$\Longrightarrow$ On place la commande 

dans la zone grise de droite, **juste en dessous** de la première commande :



$\Longrightarrow$ Le point de départ du chat correspond au point **A** $(\ -190\ ;\ 52\ )$. On écrit les coordonnées du point de départ dans la commande :



3) Finalement on insère la commande qui va faire avancer le chat jusqu'à son point d'arrivée. Il s'agit de la commande 

. Cette commande se trouve dans la rubrique « **Mouvement** » :

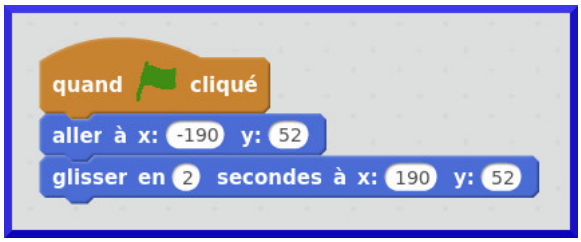
$\Longrightarrow$ On place la commande 

dans la zone grise de droite, **juste en dessous** des commandes précédentes.

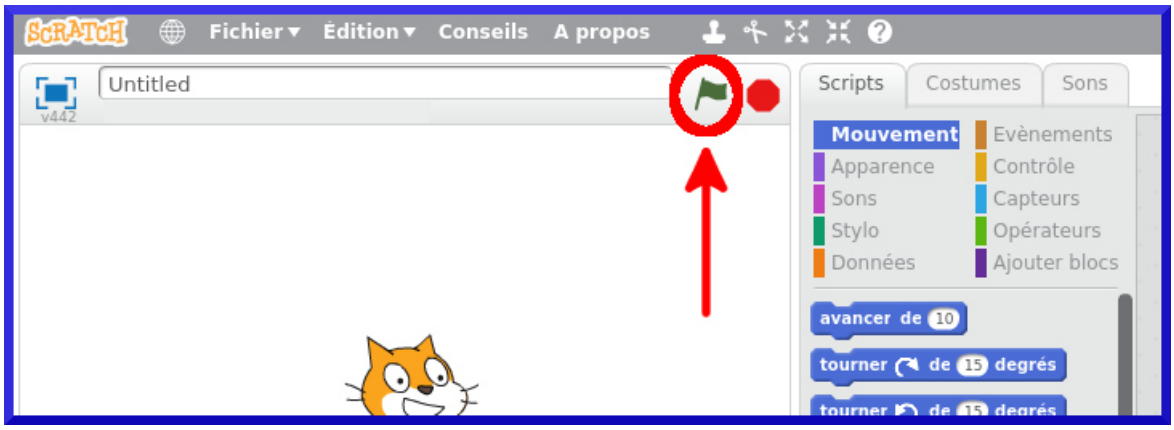
$\Longrightarrow$ Le point d'arrivée correspond au point **B** (190 ; 52) . On écrit les coordonnées du point d'arrivée dans la commande, dans les cases x et y.

$\Longrightarrow$ On souhaite par exemple que le déplacement s'effectue en 2 secondes : on écrit le nombre **2** dans la case prévue à cet effet.

La construction du programme est maintenant terminée : la zone grise de droite contient les trois commandes suivantes :



On clique sur l'icône  pour exécuter le programme :



Lorsqu'on clique sur l'icône  , le chat traverse la scène de gauche à droite.

Comment faire avancer une créature le long d'un parcours ?

Pour faire avancer une créature sur un parcours donné, il est nécessaire d'utiliser les commandes de mouvement suivantes :

$\Longrightarrow$ La commande 

pour **placer** la créature à son **point de départ**.

$\Longrightarrow$ La commande 

pour **orienter** la créature dans la bonne direction.

$\Longrightarrow$ La commande 

pour **faire avancer** la créature.

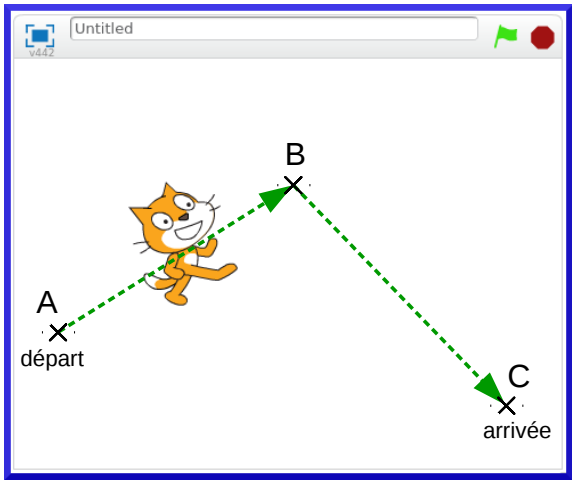
Les commandes 

et 

peuvent être utilisées **plusieurs fois**.

Exemple

Dans cet exemple, un chat va avancer le long du parcours suivant :



Coordonnées des points :

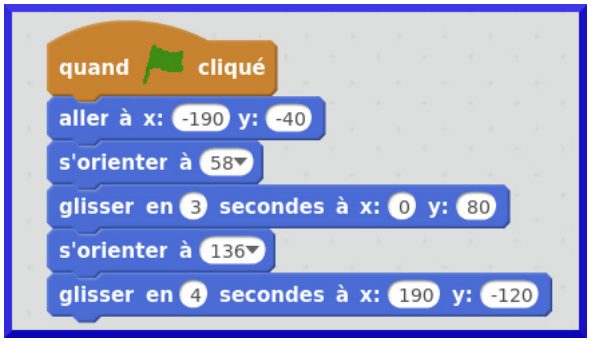
A (-190 ; -40)
B (0 ; 80)
C (190 ; -120)

Orientation du chat :

$\rightarrow$ entre **A** et **B** : 58
$\rightarrow$ entre **B** et **C** : 136

Le chat part du point **A** et arrive au point **C**. Le trajet entre **A** et **B** dure 3 secondes, et le trajet entre **B** et **C** dure 4 secondes.

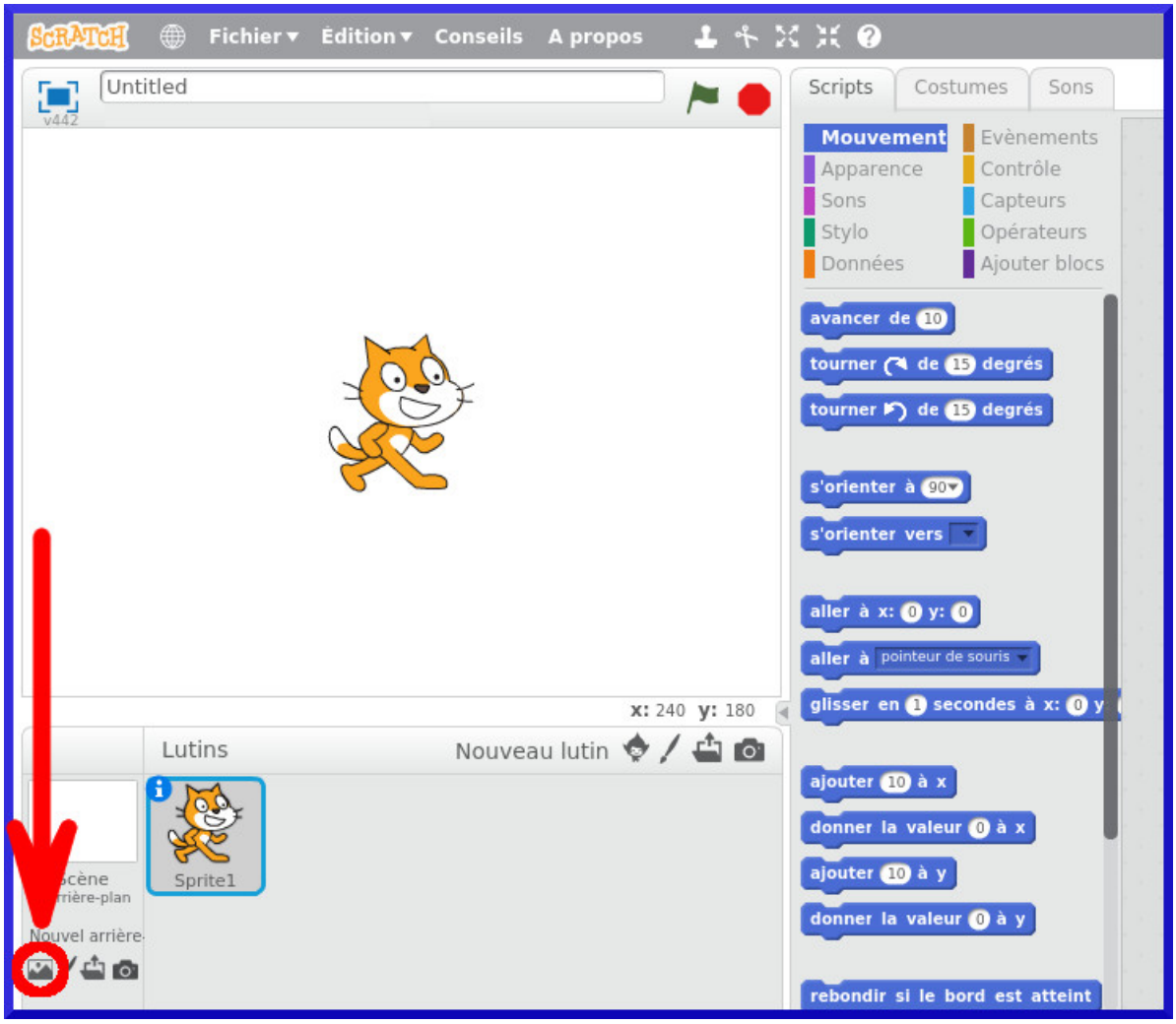
Pour que le chat effectue ce parcours, on insère les commandes suivantes dans la zone grise de droite :



On clique ensuite sur l'icône  pour exécuter le programme : le chat avance le long du parcours.

Comment insérer une image en arrière-plan ?

Pour insérer une image d'arrière-plan, on clique sur l'icône . Cet icône se trouve en-dessous de l'étiquette « **Nouvel arrière** » :

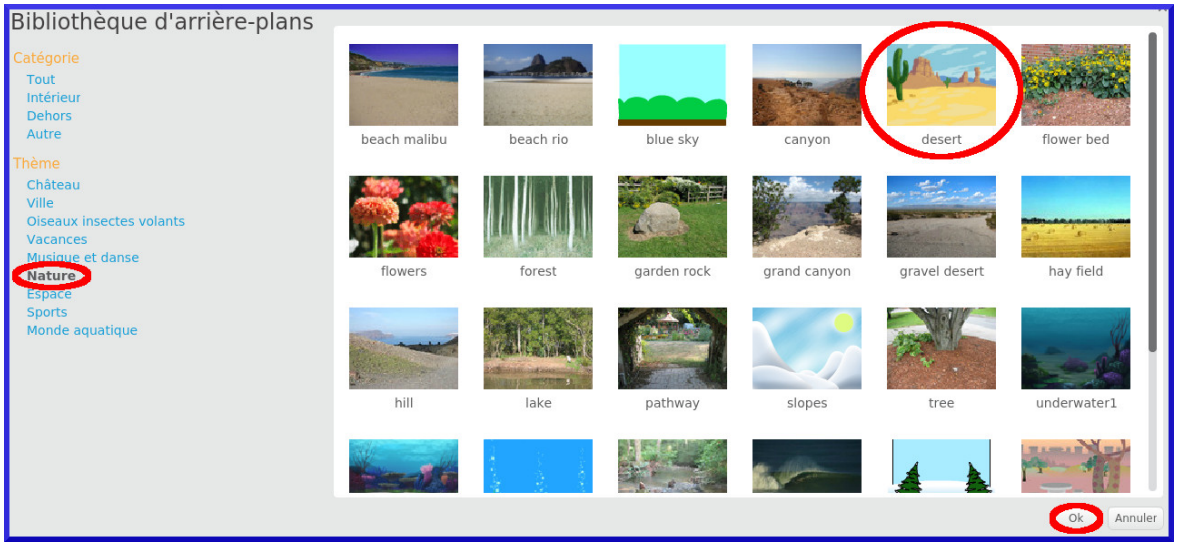


Lorsqu'on clique sur cet icône, une nouvelle fenêtre s'ouvre : il s'agit de la « **Bibliothèque d'arrière-plans** ». Cette fenêtre permet de sélectionner l'arrière-plan qu'on souhaite insérer.

Voir Exemple et Remarque à la page suivante

Exemple :

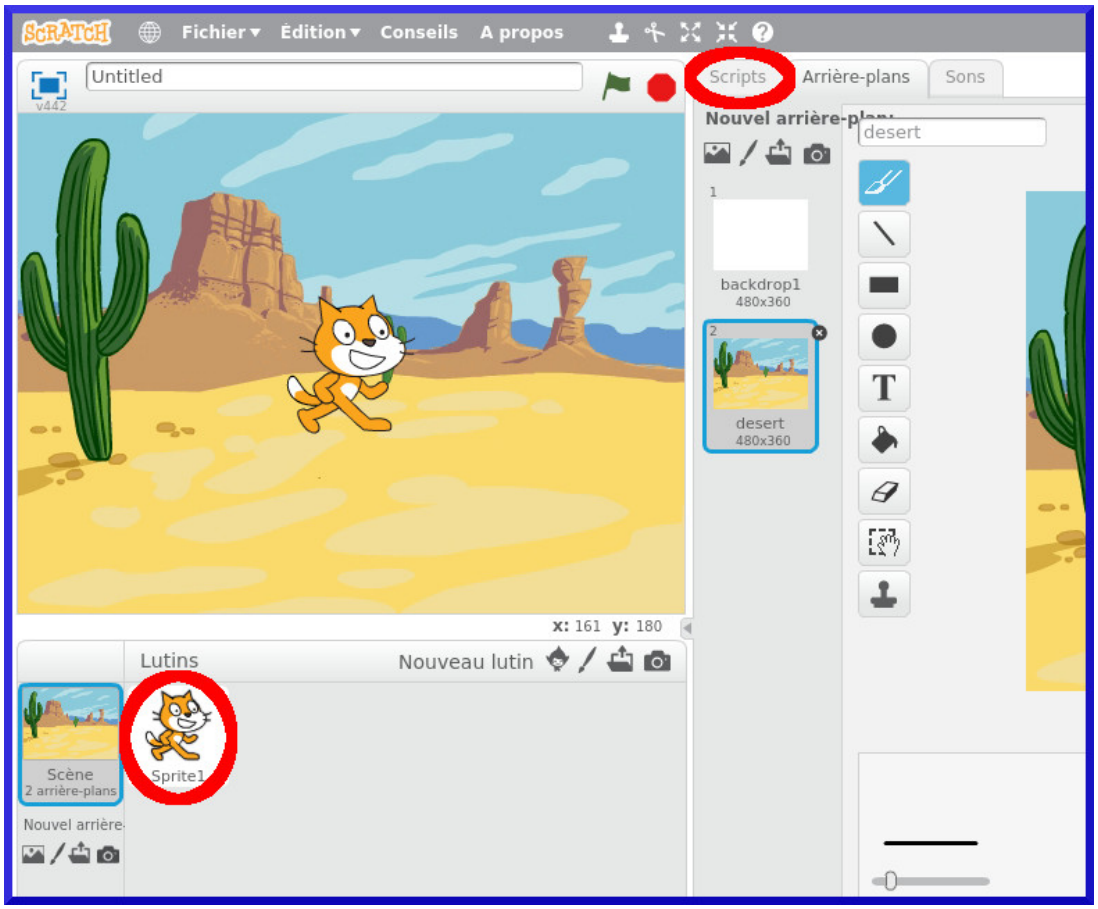
Pour insérer une image de désert en arrière plan, on clique sur l'icône : la bibliothèque d'arrière-plans s'ouvre. On clique ensuite sur le thème « Nature », puis sur l'image « desert », et on clique sur « Ok » :



L'arrière-plan « desert » apparaît dans la scène.

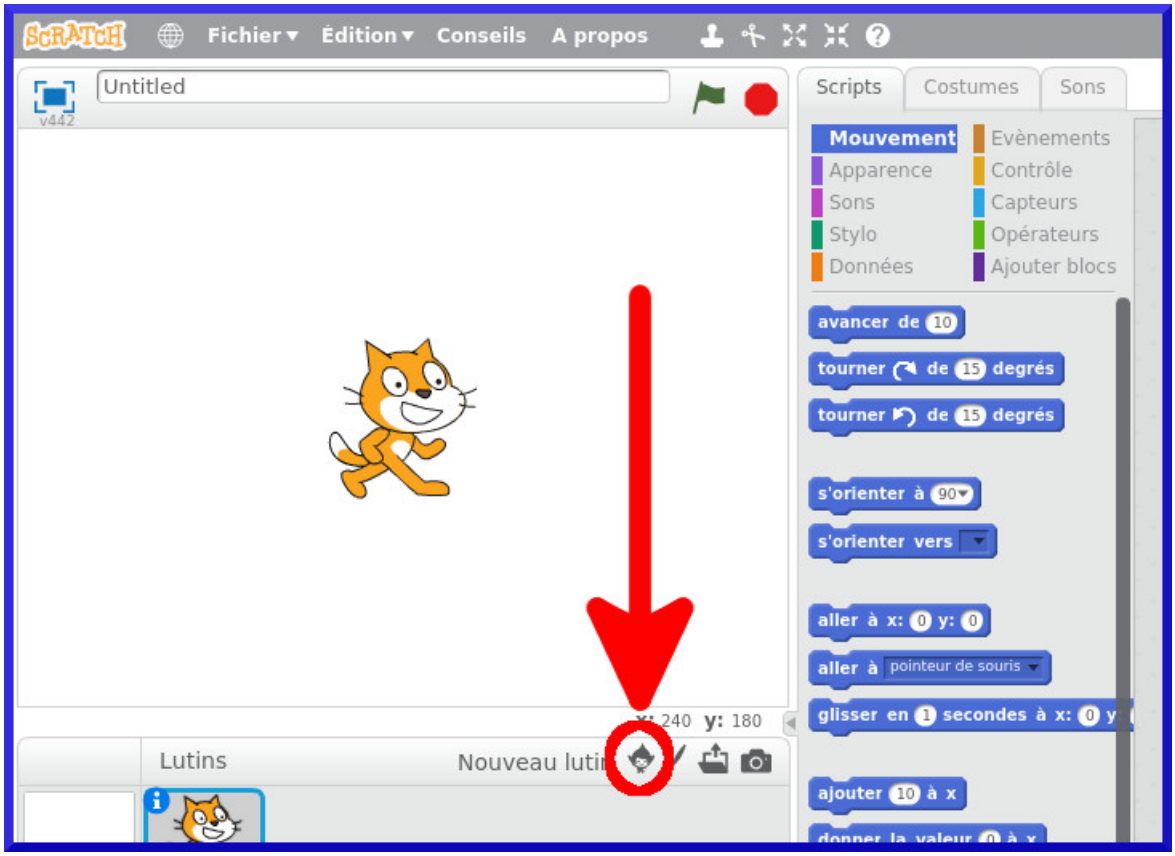
Remarque :

Une fois l'arrière-plan ajouté, il faut revenir à l'affichage standard pour pouvoir continuer à travailler : on clique sur « **Scripts** », puis on clique sur la créature sur laquelle on travaille :



Comment insérer une nouvelle créature ou un nouvel objet ?

Il est possible d'insérer plusieurs créatures dans un même programme. Pour insérer une nouvelle créature, on clique sur l'icône  . Cet icône se trouve à droite de l'étiquette « **Nouveau lutin** » :



Lorsqu'on clique sur cet icône, une nouvelle fenêtre s'ouvre : il s'agit de la « **Bibliothèque des lutins** ». Cette fenêtre permet de sélectionner la créature qu'on souhaite insérer. Une fois la créature insérée, elle apparaît dans la scène, ainsi que dans la liste des créatures.

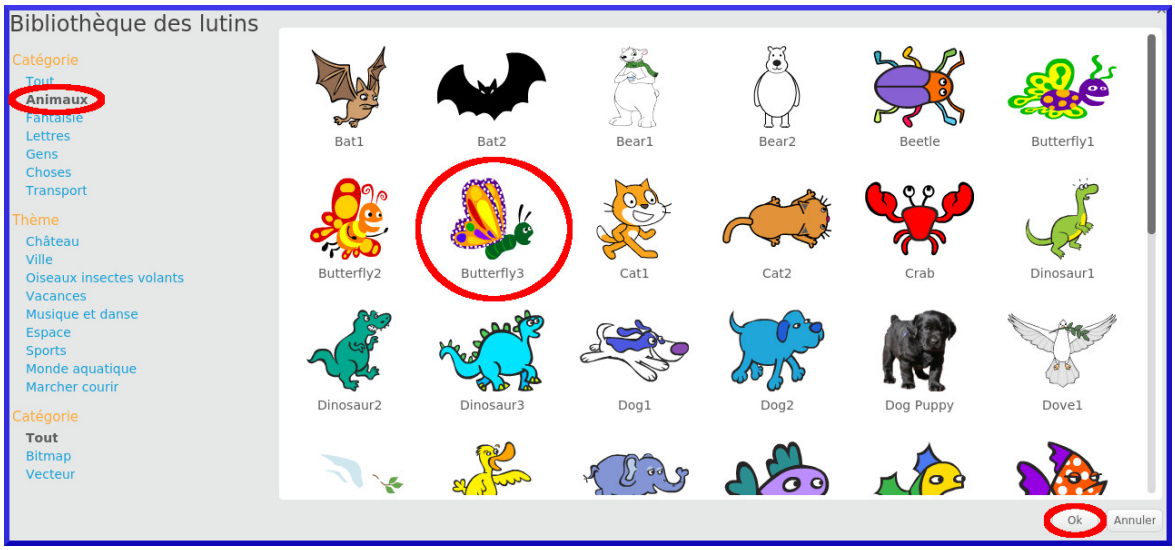
Remarques :

1) Il est possible de déplacer une créature à n'importe quel endroit de la scène en utilisant le pointeur de souri.
2) Lorsqu'il y a plusieurs créatures, on utilise la **liste des créatures** pour sélectionner la créature sur laquelle on travaille. La liste des créatures indique quelle est la créature sélectionnée.
3) Pour ajouter un **nouvel objet**, on procède de la même manière que pour ajouter une nouvelle créature :
   — Les créatures se trouvent dans les catégories « Animaux » et « Fantaisie ».
   — Les objets se trouvent dans la catégorie « Choses ».
4) Il est aussi possible d'ajouter :
   — des lettres ⇒ catégorie « Lettres »,
   — des personnages ⇒ catégorie « Gens »,
   — des moyens de transport ⇒ catégorie « Transport ».

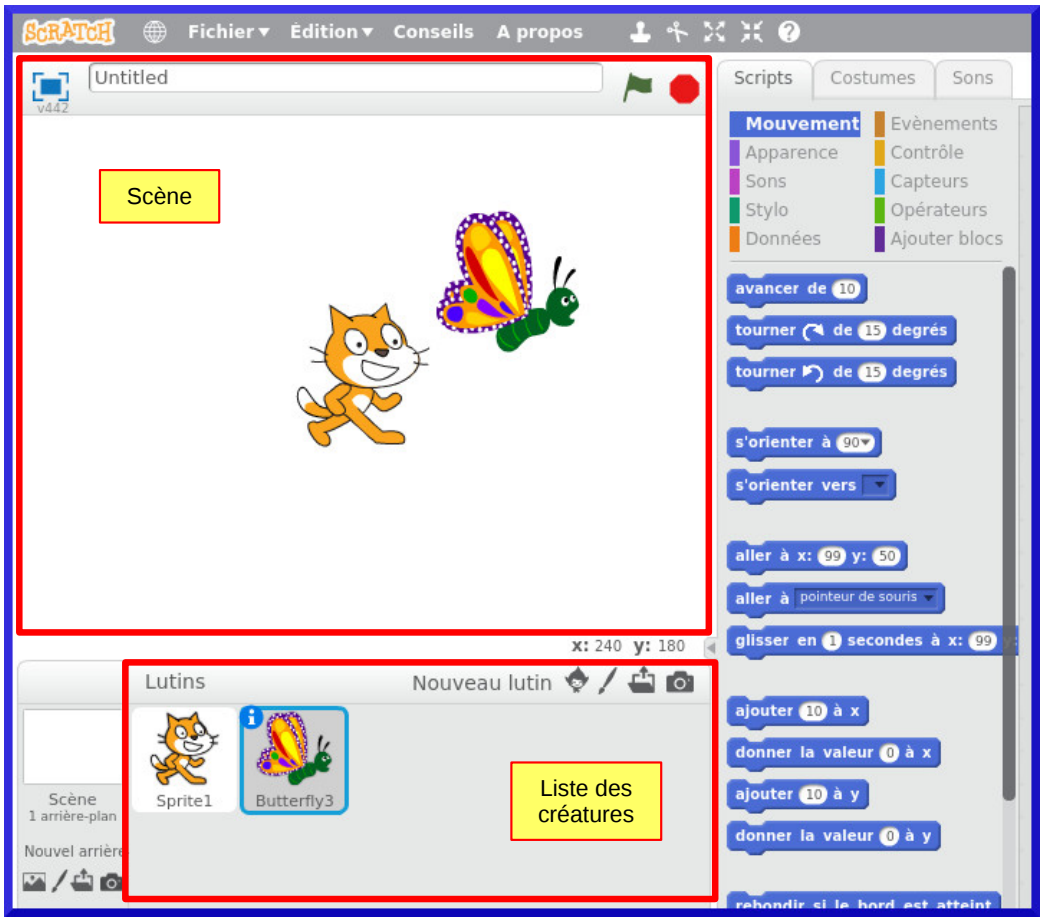
Voir l'exemple à la page suivante

Exemple

Pour insérer un papillon (par exemple le papillon « Butterfly3 »), on clique sur l'icône  : la bibliothèque des lutins s'ouvre. On clique ensuite sur la catégorie « Animaux », puis sur le papillon « Butterfly3 », et on clique sur « Ok » :



Le papillon apparaît dans la scène, ainsi que dans la liste des créatures :



Comment supprimer une créature ?

Pour supprimer une créature, on sélectionne l'icône  . Cet icône se trouve dans la barre du haut :

On clique ensuite sur la créature à supprimer. La créature disparaît.

Exemple :

Pour supprimer le chat, on sélectionne l'icône  , puis on clique sur le chat :

Le chat disparaît.

Remarque :

L'icône  permet aussi de supprimer des commandes dans la zone grise de droite.

Les commandes de mouvement

Cette commande place la créature à l'endroit indiqué par les coordonnées (x ; y). Le déplacement s'effectue de manière instantanée.

Cette commande place la créature à l'endroit où se trouve la créature sélectionnée dans le menu déroulant. Le déplacement s'effectue de manière instantanée.

Cette commande fait avancer la créature vers l'endroit indiqué par les coordonnées (x ; y). La durée du déplacement est indiquée en secondes.

Cette commande oriente la créature dans la direction indiquée. Les quatre orientations principales sont les suivantes :

Orientation	Valeur
Vers la droite	90
Vers la gauche	−90
Vers le haut	0
Vers le bas	180

Les principales orientations

Cette commande oriente la créature en direction de la créature sélectionnée dans le menu déroulant. Cette commande est très utile pour faire avancer une créature en direction d'une autre créature.

Cette commande fait avancer la créature de la distance indiquée. Le déplacement de fait de manière instantanée. Avant d'utiliser cette commande, il faut fixer la direction du déplacement en utilisant la commande 

ou 

.

Cette commande fait tourner la créature dans le sens des aiguilles d'une montre. L'angle de rotation est indiqué en degrés.

Cette commande fait tourner la créature dans le sens inverse des aiguilles d'une montre. L'angle de rotation est indiqué en degrés.

Cette commande permet de régler le style de rotation. Cette commande est très utile lorsqu'une créature se retrouve la tête à l'envers.

Cette commande inverse le sens du mouvement lorsque la créature touche le bord de la scène.

Les boucles

En programmation informatique, on a souvent besoin de **répéter** une même commande un grand nombre de fois, et il est beaucoup trop long de recopier la commande autant de fois que nécessaire.

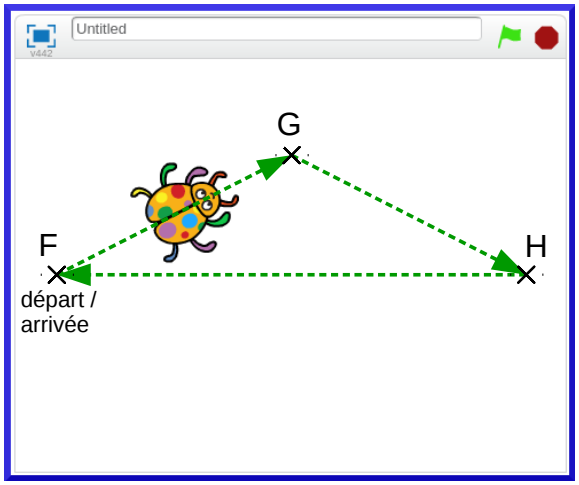
C'est pourquoi il existe un outil, appelé ***boucle***, qui permet de n'insérer qu'une seule fois une commande qui se répète beaucoup de fois. Cet outil se trouve dans la rubrique « **Contrôle** » :

On insère la commande une seule fois, et on indique combien de fois il faut répéter cette commande.

Voir l'exemple à la page suivante

Exemple :

Dans cet exemple, on va faire avancer une créature le long du parcours suivant :



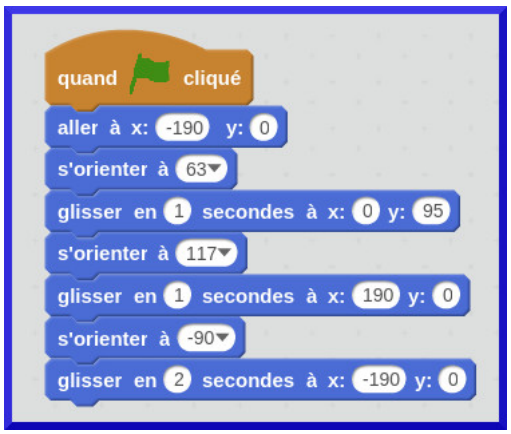
Coordonnées des points :

F (-190 ; 0)
G (0 ; 95)
H (190 ; 0)

Orientation de la créature :

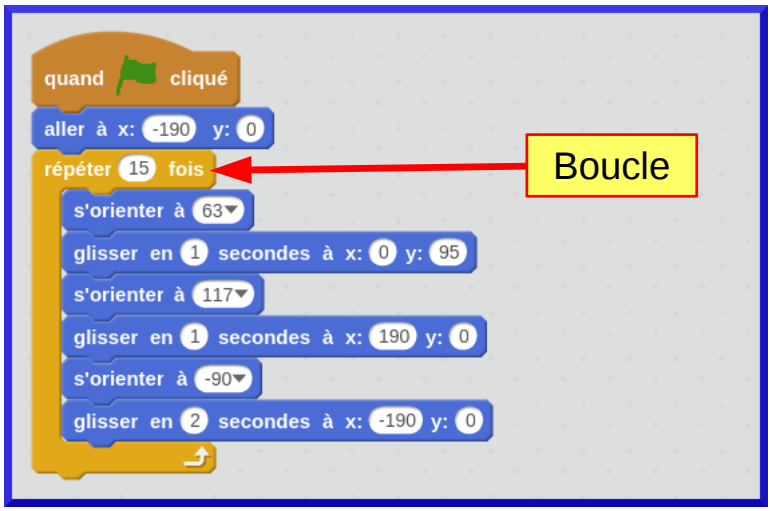
→ entre **F** et **G** : 63
→ entre **G** et **H** : 117
→ entre **H** et **F** : -90

La créature part du point **F** et avance jusqu'au point **G**. Elle avance ensuite jusqu'au point **H**, puis elle retourne au point **F** : on insère les commandes suivantes dans la zone grise de droite :



Lorsqu'on clique sur l'icône , la créature n'effectue qu'**un seul** tour, et s'arrête au point **F**. Comment faire pour que la créature effectue **plusieurs tours** ?

Pour que la créature effectue **plusieurs tours**, on clique sur la rubrique « **Contrôle** », et on ajoute **une boucle** :



Lorsqu'on clique sur l'icône , la créature effectue **15 tours**.

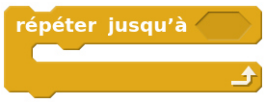
Les différents types de boucle

Il existe trois types de boucles :

1) Le nombre fois que le contenu de la boucle se répète est indiqué :

2) Le contenu de la boucle se répète **indéfiniment**, jusqu'à ce qu'on quitte le programme :

3) Le contenu de la boucle se répète jusqu'à ce qu'à ce qu'une certaine condition soit satisfaite :

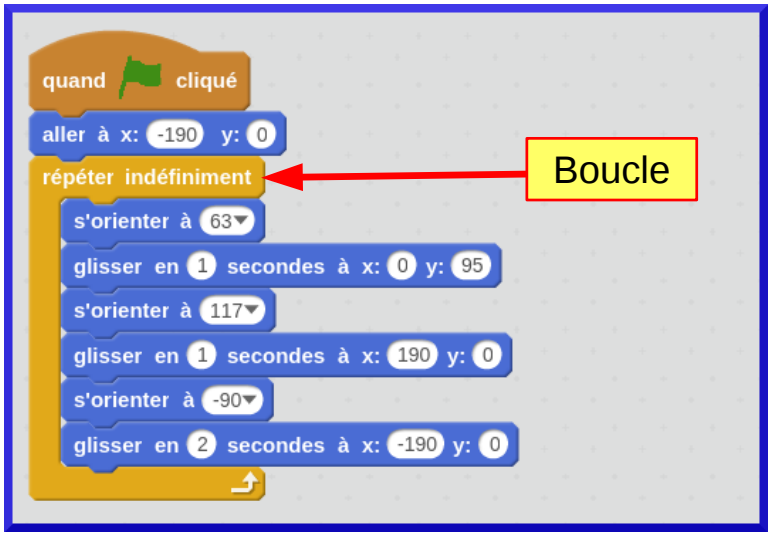


Si la condition est déjà remplie au début, le contenu de la boucle ne s'exécute jamais.

Ces boucles se trouves dans la rubrique « **Contrôle** ».

Exemple :

Dans cet exemple, la créature reste **indéfiniment** en mouvement : la créature avance, et ne s'arrête jamais d'avancer : on utilise la boucle suivante :



Les variables

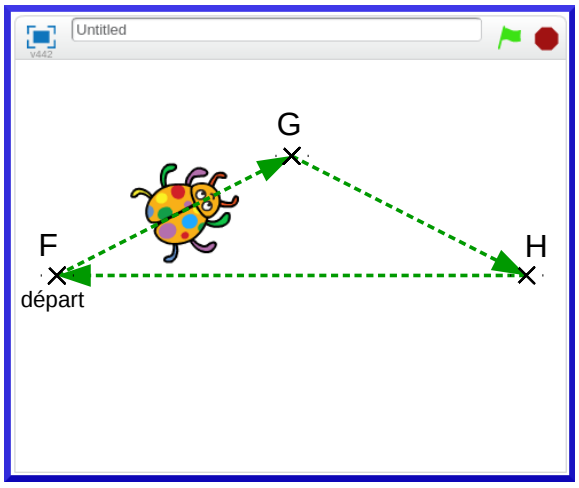
En programmation informatique, on a souvent besoin de compter quelque chose. Par exemple, dans un jeu, il faut **compter le nombre de points** que le joueur obtient. Dans un jeu de course de voitures, il faut **compter le nombre de tours de circuit** effectués par chaque voiture.

C'est pourquoi il existe un outil, appelé ***variable***, qui permet de compter quelque chose. Cet outil se trouve dans la rubrique « **Données** ».

Exemple :

Dans cet exemple, on va faire avancer une créature sur un circuit, et on va **compter le nombre de tours** effectués par la créature. Ce nombre sera affiché dans la scène.

Voici le circuit le long duquel la créature se déplace :



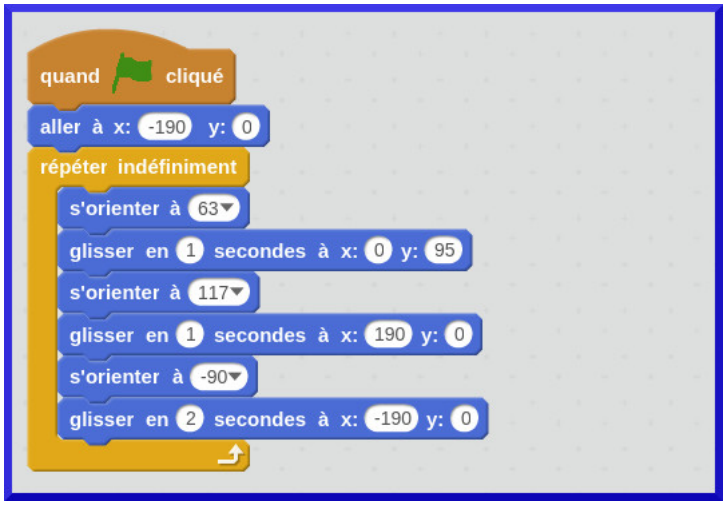
Coordonnées des points :

F (−190 ; 0)
G (0 ; 95)
H (190 ; 0)

Orientation de la créature :

→ entre **F** et **G** : 63
→ entre **G** et **H** : 117
→ entre **H** et **F** : −90

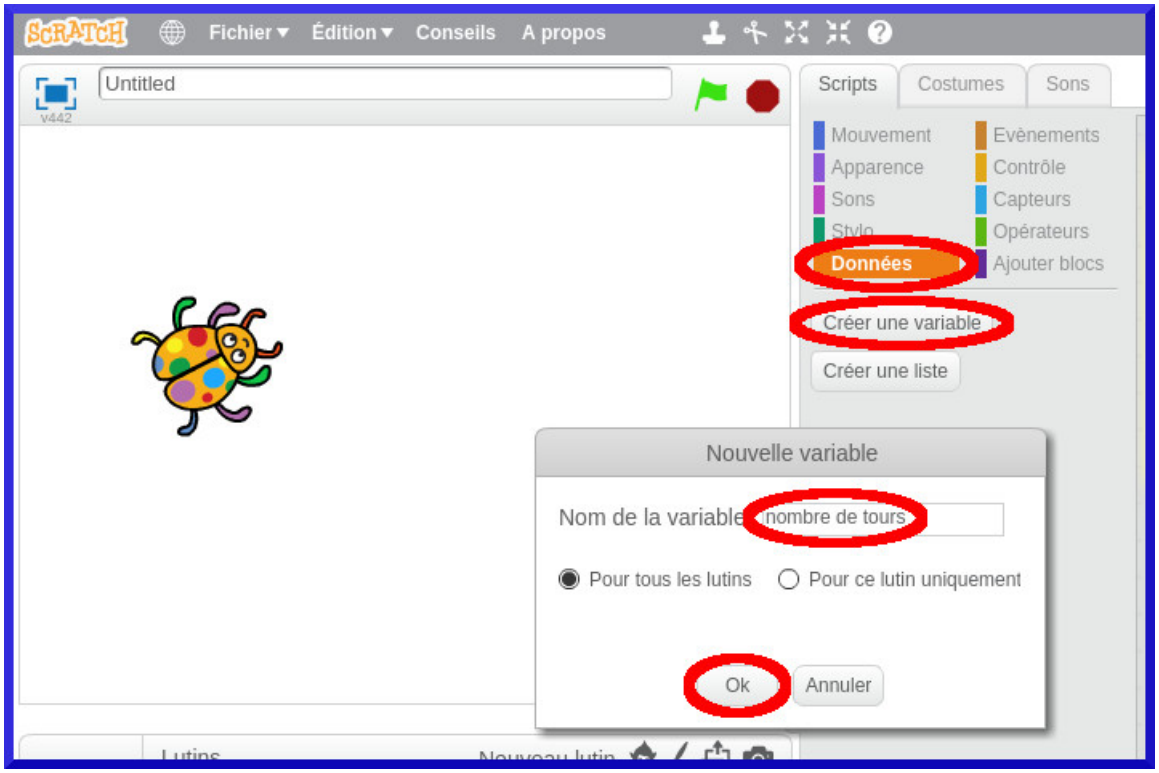
Les commandes suivantes font avancer la créature le long de ce circuit :



Lorsqu'on clique sur l'icône , la créature avance indéfiniment le long du circuit, sans jamais s'arrêter. Comment faire pour **compter le nombre de tours** qu'elle effectue ?

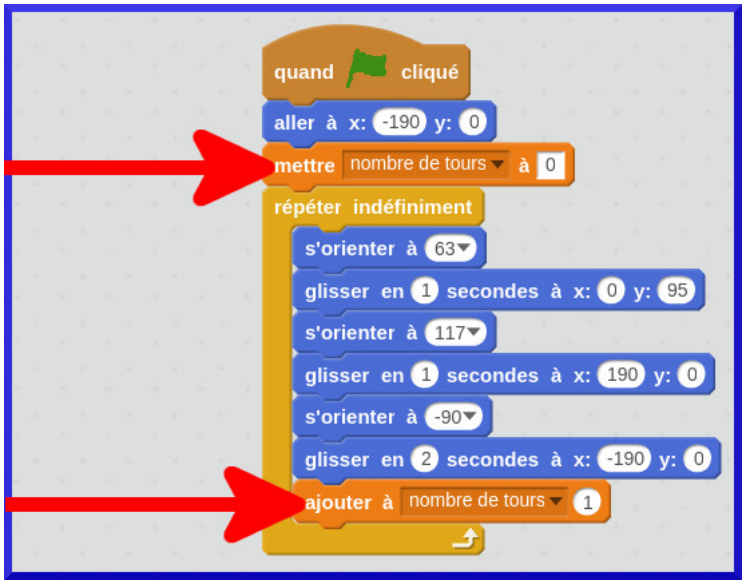
Voir la suite de l'exemple à la page suivante

Pour compter le **nombre de tours** effectués par la créature, on clique sur la rubrique « **Données** », puis sur le bouton « **Créer une variable** ». On écrit ensuite « **nombre de tours** » dans la case « Nom de la variable », et on clique sur « **Ok** » :



La variable « **nombre de tours** » s'affiche dans la scène.

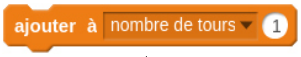
Pour que le nombre de tours affiché soit correct, il faut ajouter les deux commandes suivantes :



Ces deux commandes se trouvent dans la rubrique « **Données** » :

⟹ La commande 

remet le compteur à zéro avant l'exécution de la boucle : lorsqu'on lance le programme, la créature n'a encore effectué aucun tour. Le comptage démarre donc depuis 0.

⟹ La commande 

augmente le nombre de tours à chaque fois que la créature termine un tour (arrivée au point **F**).

Les structures de contrôle

En programmation informatique, On a souvent besoin qu'une commande s'exécute **uniquement** dans une certaine situation, et qu'elle ne s'exécute pas si la situation est différente. Par exemple, dans un jeu de course de voiture, on aimerait que la commande qui fait freiner la voiture soit exécutée **uniquement** lorsque le joueur appuie sur la touche de freinage du joystick, et qu'elle ne soit pas exécutée si le joueur n'appuie pas sur cette touche.

C'est pourquoi il existe un outil, appelé ***structure de contrôle***, qui permet d'indiquer dans quelle situation la commande doit être exécutée. Cet outil se trouve dans la rubrique « **Contrôle** » :

La condition d'exécution doit être placée après le si . La commande dont l'exécution dépend de la condition doit être insérée à l'intérieur de la structure de contrôle.

Exemple :

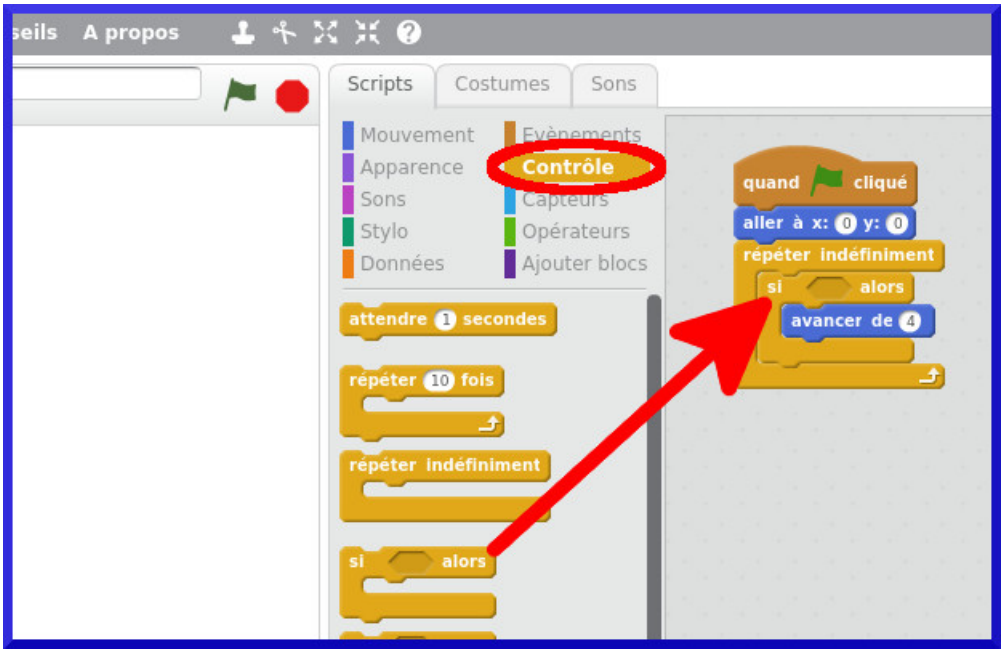
Dans cet exemple, l'utilisateur du programme va utiliser les flèches gauche et droite du clavier pour déplacer une créature à gauche et à droite :

1) On commence par insérer les commandes suivantes :

La commande avancer de ( ) se trouve dans la rubrique « **Mouvement** ».

2) On ajoute ensuite une **structure de contrôle** qui va faire avancer la créature **uniquement** lorsqu'on appuie sur la **flèche droite** du clavier :

   $\Longrightarrow$ On clique sur la rubrique « **Contrôle** », et on ajoute la **structure de contrôle** :



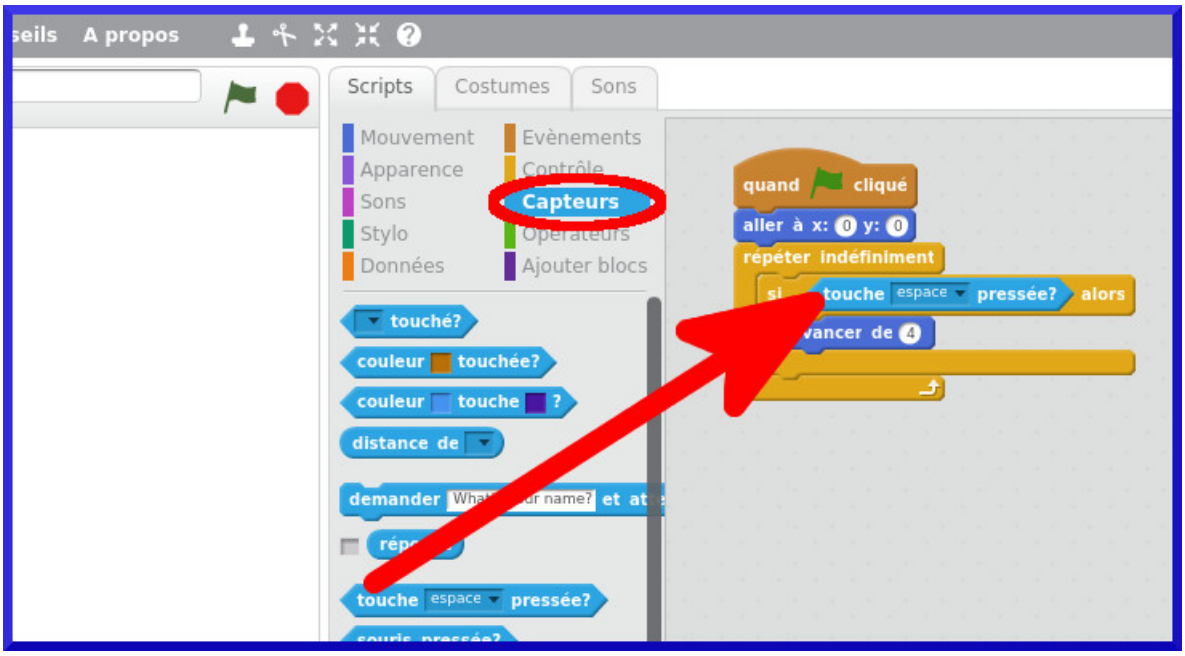
3) On insère la commande qui va détecter si la flèche droite du clavier est appuyée. Il s'agit de la commande 

qui se trouve dans la rubrique « **Capteurs** » :

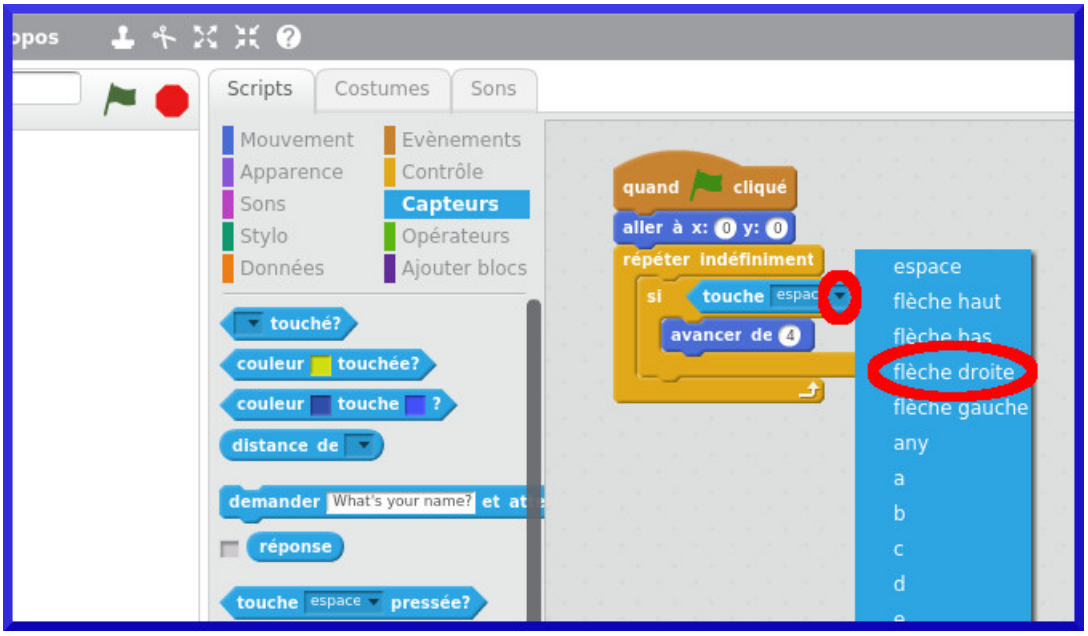
$\Longrightarrow$ On clique sur la rubrique « **Capteurs** », et on insère la commande 

. Cette commande doit être placée dans la case située après le 

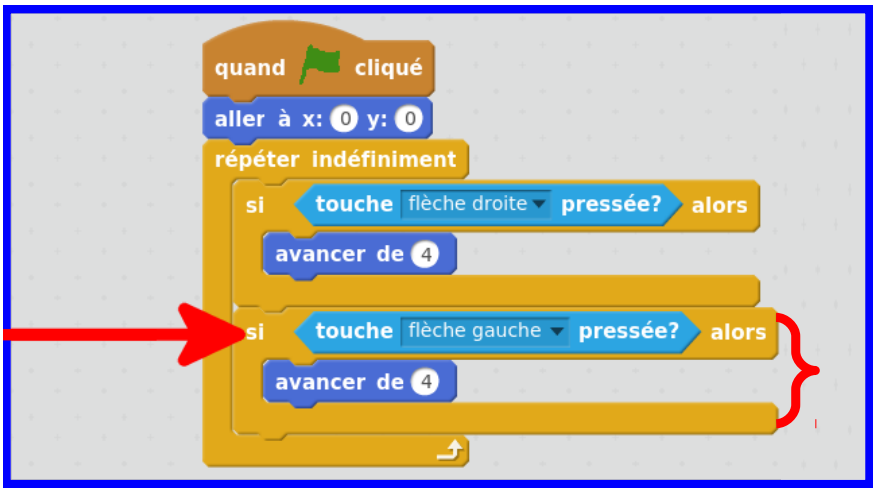
de la structure de contrôle :



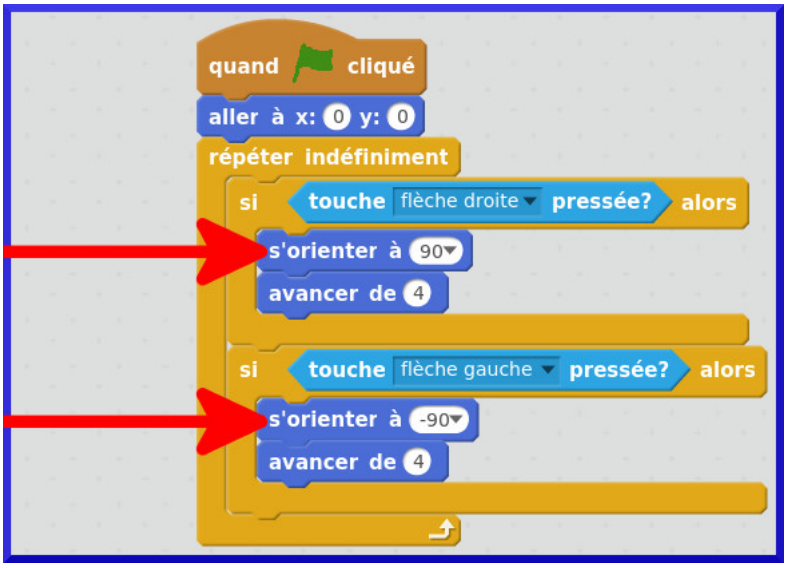
$\Longrightarrow$ On clique sur l'icône  de la commande qu'on vient d'insérer, et on sélectionne « **flèche droite** » :



4) En procédant de la même manière, on ajoute la structure de contrôle qui va faire avancer la créature lorsqu'on appuie sur la **flèche gauche** du clavier :

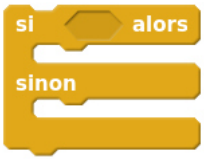


5) Finalement, on ajoute les commandes nécessaires pour **orienter** la créature dans la bonne direction :



Lorsqu'on clique sur l'icône  , on peut faire avancer la créature à droite ou à gauche en appuyant sur les flèches droite et gauche du clavier.

La structure de contrôle double :



Cette structure de contrôle permet d'indiquer à la fois :

⟹ les commandes qui seront exécutées si la condition est remplie : ces commandes doivent être insérées en-dessous du si alors ,

⟹ les commandes qui seront exécutées si la condition **n**'est **pas** remplie : ces commandes doivent être insérées en-dessous du sinon .

Les commandes d'apparence

Cette commande affiche une bulle contenant un texte, comme dans une bande dessinée.

Cette commande modifie la **taille** d'une créature.

$\Longrightarrow$ Cette commande est très utile pour rendre une créature plus petite : on écrit un nombre entre 1 et 100. Plus le nombre est petit, plus la créature est petite.

$\Longrightarrow$ Il est aussi possible d'agrandir une créature : on écrit un nombre plus grand que 100.

Cette commande fait disparaît une créature : la créature n'est plus affichée dans la scène. Cependant les commandes concernant la créature continuent à s'exécuter.

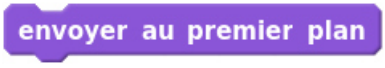
Cette commande s'utilise à chaque fois qu'on utilise la commande 

: elle permet de faire réapparaître la créature qui a été cachée. Attention : lorsqu'on utilise la commande 

, il se peut que la créature soit invisible lors du lancement du programme ! Il est donc important de placer la commande 

au début de chaque programme pour garantir que la créature s'affiche lors du lancement du programme.

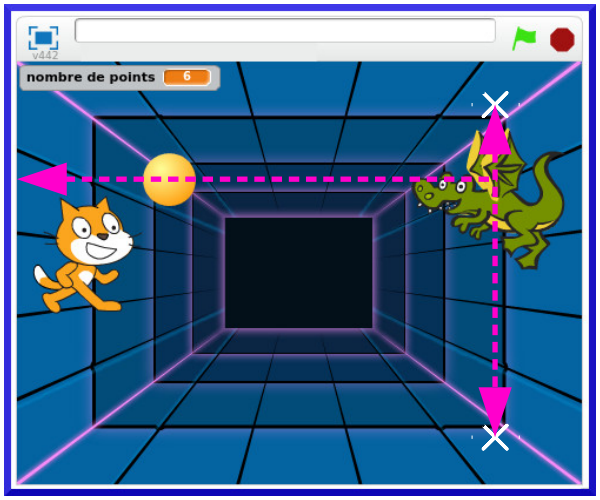
Cette commande modifie l'**image d'arrière-plan** affichée dans la scène. Avant d'utiliser cette commande, il faut avoir inséré plusieurs images différentes d'arrière-plans. On sélectionne l'image d'arrière-plan désirée dans le menu déroulant.



Cette commande affiche la créature au premier plan, c'est-à-dire devant les autres créatures.

Un exemple de conception de jeu

Dans cet exemple, on va créer un jeu dans lequel un dragon tire des balles sur un chat. Le joueur doit déplacer le chat en utilisant les flèches du clavier et en évitant de se faire toucher par les balles tirées par le dragon. Le dragon est continuellement en mouvement : il monte et il descend. Le joueur marque 1 point à chaque fois qu'il évite une balle. Le but de jeu est d'obtenir le plus de points possible. La partie se termine lorsque le joueur se fait toucher par une balle :



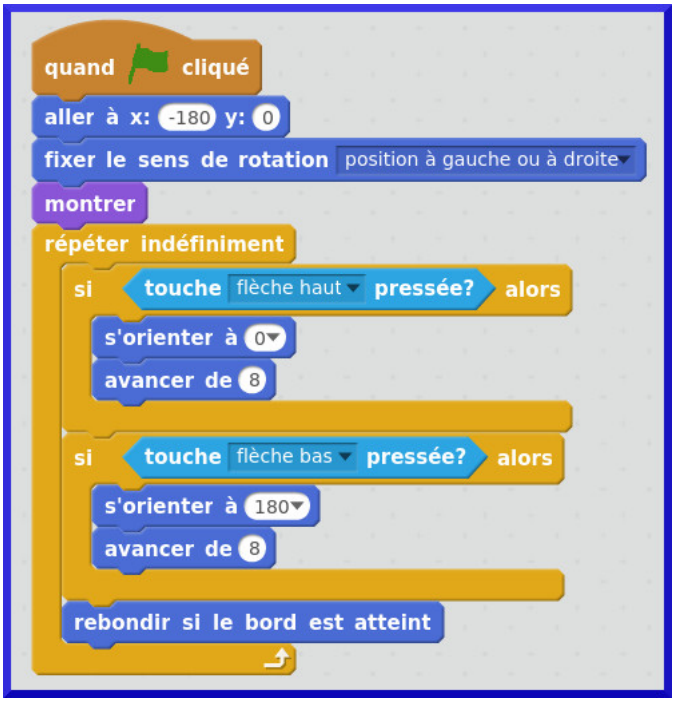
1) On ajoute les deux objets suivants :

   $\Longrightarrow$ le dragon « Dragon » qui se trouve dans la catégorie « Fantaisie »,

   $\Longrightarrow$ la balle « Ball » qui se trouve dans la catégorie « Choses ».

Voir la suite de l'exemple à la page suivante

2) On sélectionne le chat, et on insère les commandes qui permettent au joueur de déplacer le chat en appuyant sur les touches « **flèche haut** » et « **flèche bas** » du clavier :

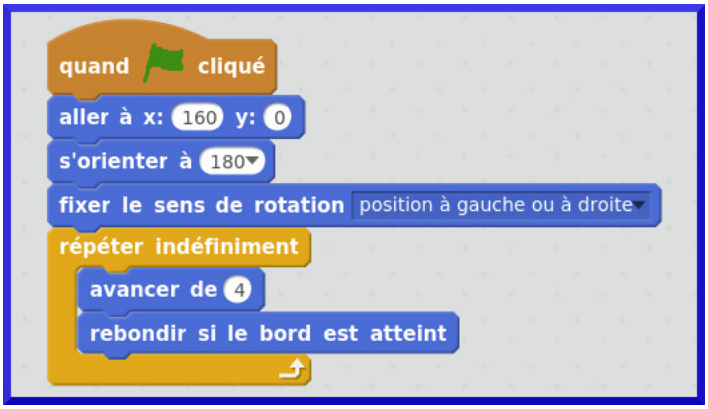


$\Longrightarrow$ Ces commandes concernent le chat.

$\Longrightarrow$ La commande  se trouve dans la rubrique « **Apparence** ».

$\Longrightarrow$ Les commandes  et  se trouvent dans la rubrique « **Mouvement** ».

3) On sélectionne le dragon, et on insère les commandes qui font monter et descendre le dragon :



$\Longrightarrow$ Ces commandes concernent le dragon.

Voir la suite de l'exemple à la page suivante

4) Le dragon est dirigé dans la **mauvaise direction** : on retourne le dragon pour qu'il regarde en direction du chat :

$\Longrightarrow$ On clique sur « **Costumes** » :

$\Longrightarrow$ On clique sur l'icône  :

Le dragon se retourne, et se retrouve dans la bonne direction.

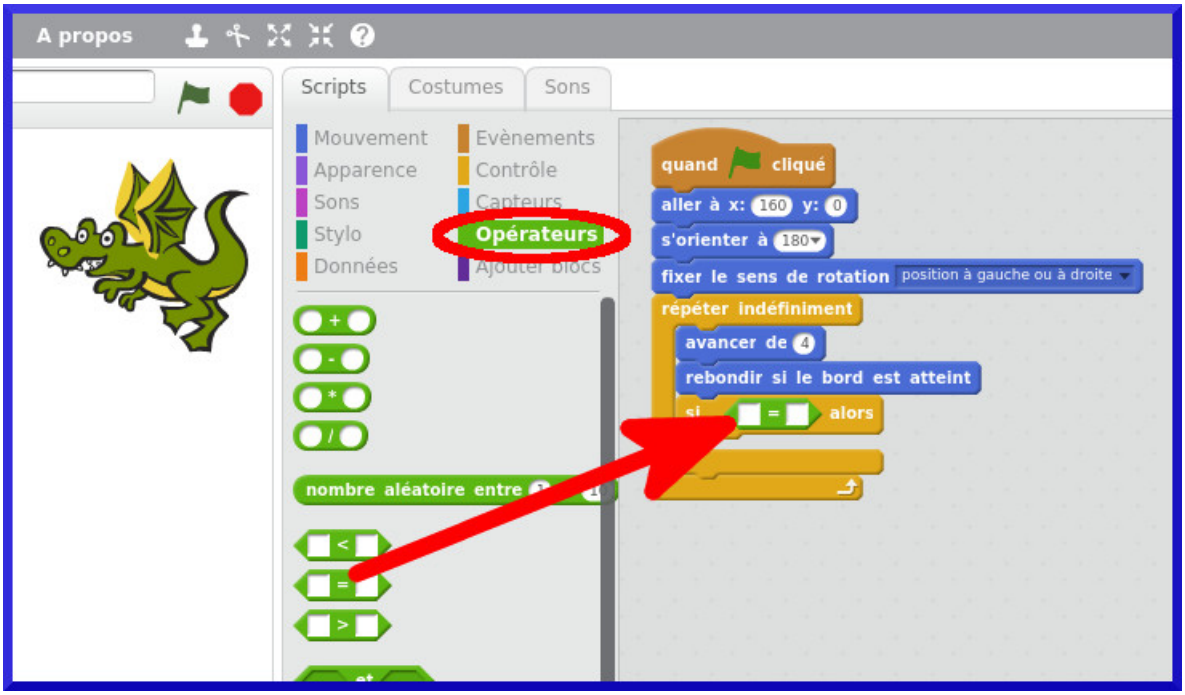
$\Longrightarrow$ On clique sur « **Scripts** » pour revenir à l'affichage habituel :

Voir la suite de l'exemple à la page suivante

5) On ajoute une **structure de contrôle** qui va permettre au dragon de tirer des balles :

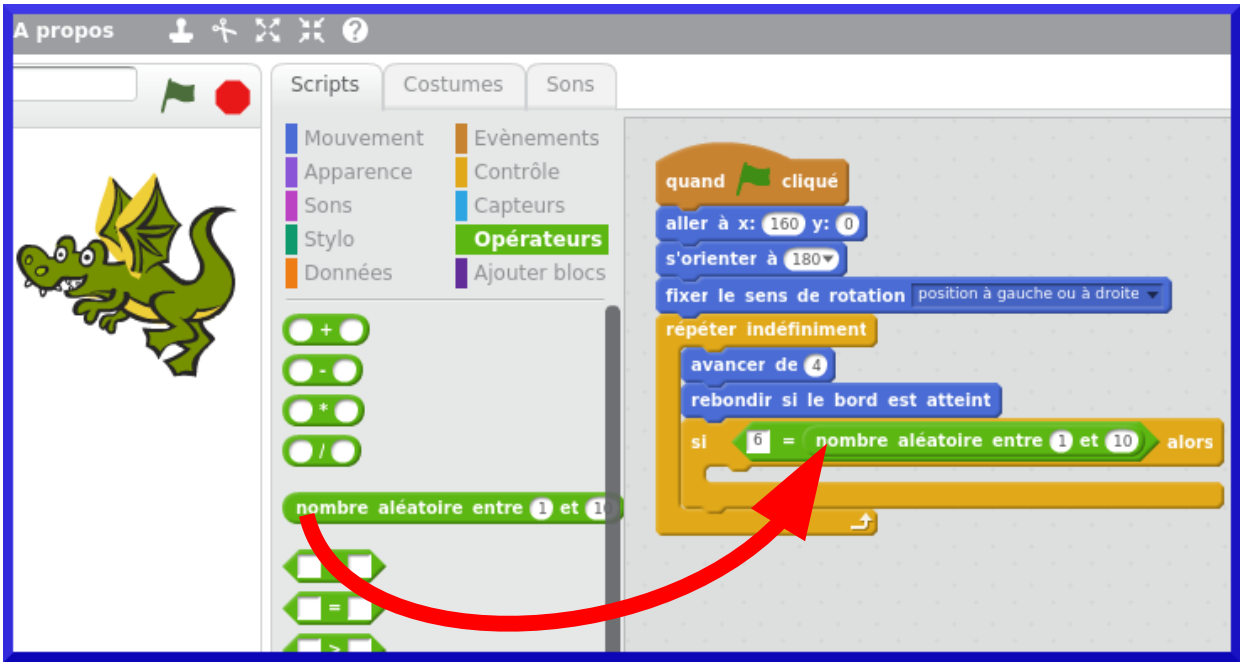
$\Longrightarrow$ On clique sur la rubrique « **Contrôle** », et on ajoute la **structure de contrôle** : on la place **à l'intérieur** de la boucle, **après** les commandes de mouvement.

$\Longrightarrow$ On clique sur la rubrique « **Opérateurs** », et on place la commande  dans la case située après le  de la structure de contrôle :



$\Longrightarrow$ On écrit le nombre **6** dans la case **de gauche** de la commande  .

$\Longrightarrow$ On place la commande  dans la case **de droite** de la commande  :



Voir la suite de l'exemple à la page suivante

$\Longrightarrow$ On écrit les nombres **1** et **60** dans les cases de la commande 

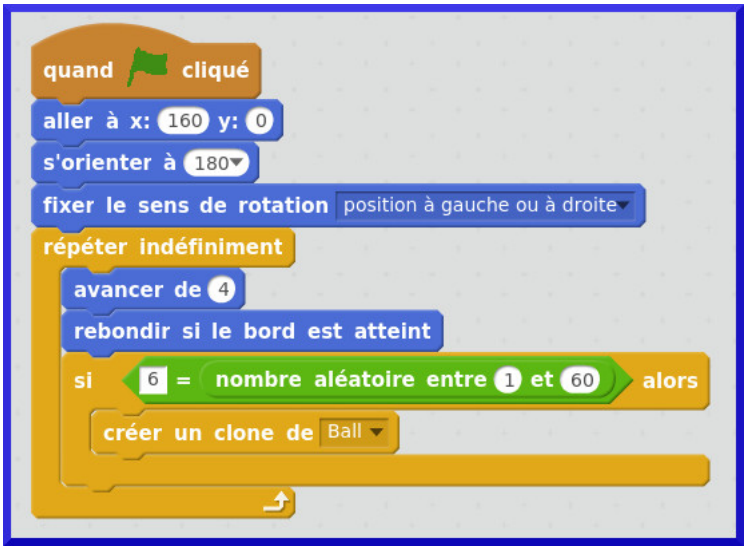
:

$\Longrightarrow$ On clique sur la rubrique « **Contrôle** », et on insère la commande 

à l'intérieur de la structure de contrôle.

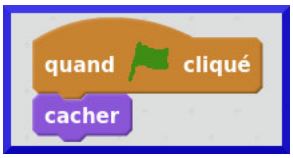
$\Longrightarrow$ On sélectionne « Ball » dans le menu déroulant de cette commande.

La structure de contrôle permettant au dragon de tirer des balles est maintenant terminée : la zone grise de droite contient les commandes suivantes :



Voir la suite de l'exemple à la page suivante

6) Lorsque que la partie commence, aucune balle n'a encore été tirée pas le dragon. Il faut donc que la balle reste cachée : on sélectionne la balle, et on insère les commandes suivantes :



$\Longrightarrow$ Ces commandes concernent la balle.

$\Longrightarrow$ La commande  se trouve dans la rubrique « **Apparence** ».

7) On insère les commandes qui font avancer les balles après qu'elles aient été tirées par le dragon :

$\Longrightarrow$ Ces commandes concernent la balle. On place ces nouvelles commandes **à côté** des commandes qu'on a déjà insérées au point 6).

$\Longrightarrow$ Les commandes  et  se trouvent dans la rubrique « **Contrôle** ».

$\Longrightarrow$ La commande  se trouve dans la rubrique « **Mouvement** ». Comme les balles sont tirées par le dragon, on sélectionne « Dragon » dans le menu déroulant de cette commande.

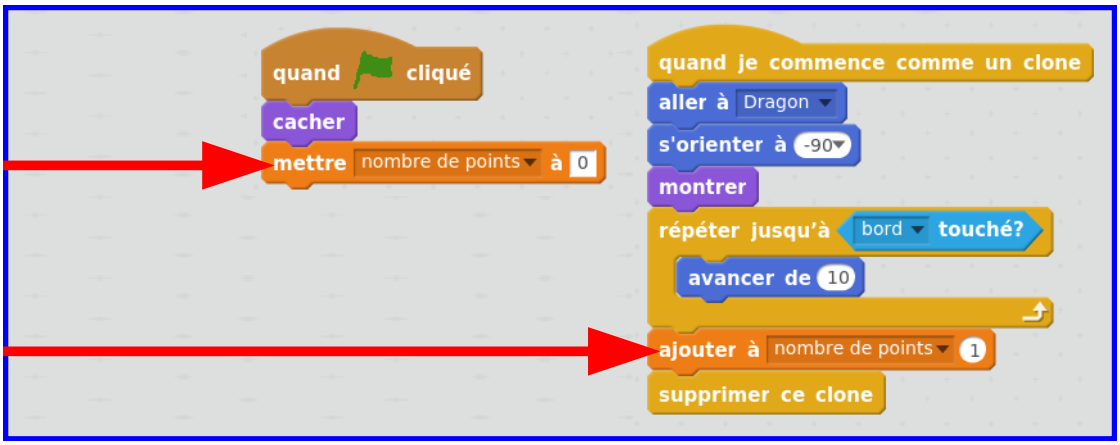
$\Longrightarrow$ La commande  se trouve dans la rubrique « **Capteurs** ». On sélectionne « **bord** » dans le menu déroulant de cette commande.

Voir la suite de l'exemple à la page suivante

8) On ajoute un **compteur de points** : à chaque fois que le chat esquive une balle, le joueur marque 1 point :

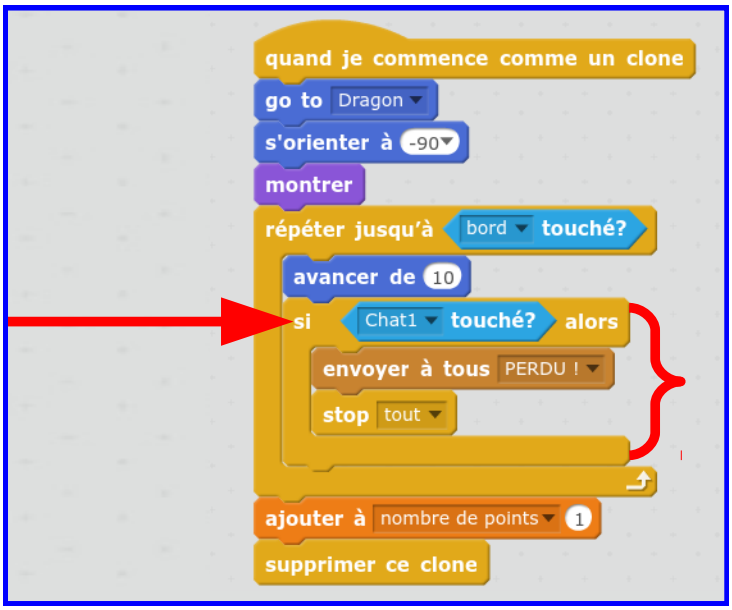
⟹ On clique sur « **Données** », et on ajoute une **nouvelle variable** appelée « **nombre de points** ».

⟹ On ajoute les deux commandes nécessaires au comptage des points :



Ces commandes concernent la balle.

9) La partie se termine lorsque le chat est touché par une balle : on ajoute la **structure de contrôle** nécessaire à l'arrêt du jeu lorsque le chat est touché par une balle :



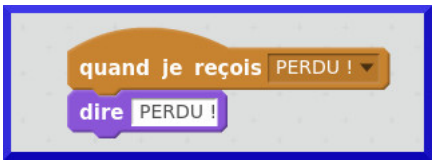
⟹ Ces commandes concernent la balle.

⟹ La commande  se trouve dans la rubrique « **Événements** ». On sélectionne « **nouveau message...** » dans le menu déroulant de cette commande, puis on écrit « **PERDU !** » dans la case « Nom du message », et on clique sur « **Ok** ».

⟹ La commande  se trouve dans la rubrique « **Contrôle** ».

Voir la suite de l'exemple à la page suivante

10) Quand le joueur perd, le dragon affiche le message « **PERDU !** » : on sélectionne le dragon, et on ajoute les commandes suivantes :

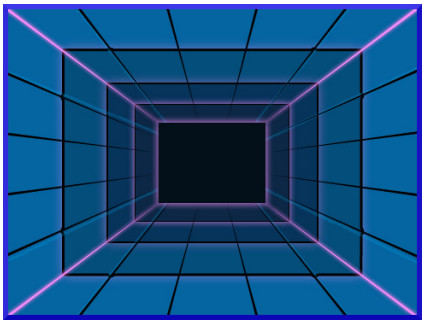


$\Longrightarrow$ Ces commandes concernent le dragon. Elles sont placées **à côté** des commandes déjà insérées aux points 3) et 5).

$\Longrightarrow$ La commande  se trouve dans la rubrique « **Événements** ».

$\Longrightarrow$ La commande  se trouve dans la rubrique « **Apparence** ». On écrit « **PERDU !** » dans la case de cette commande.

11) Finalement, on ajoute l'**image d'arrière-plan** « neon tunnel » qui se trouve dans le thème « Espace » :



La construction du jeu est maintenant terminée. On peut cliquer sur l'icône  pour y jouer !

Comment faire avancer une créature à l'intérieur d'un labyrinthe ?

Pour faire avancer une créature à l'intérieur d'un labyrinthe, on utilise une **structure de contrôle** qui empêche la créature de traverser les parois du labyrinthe :

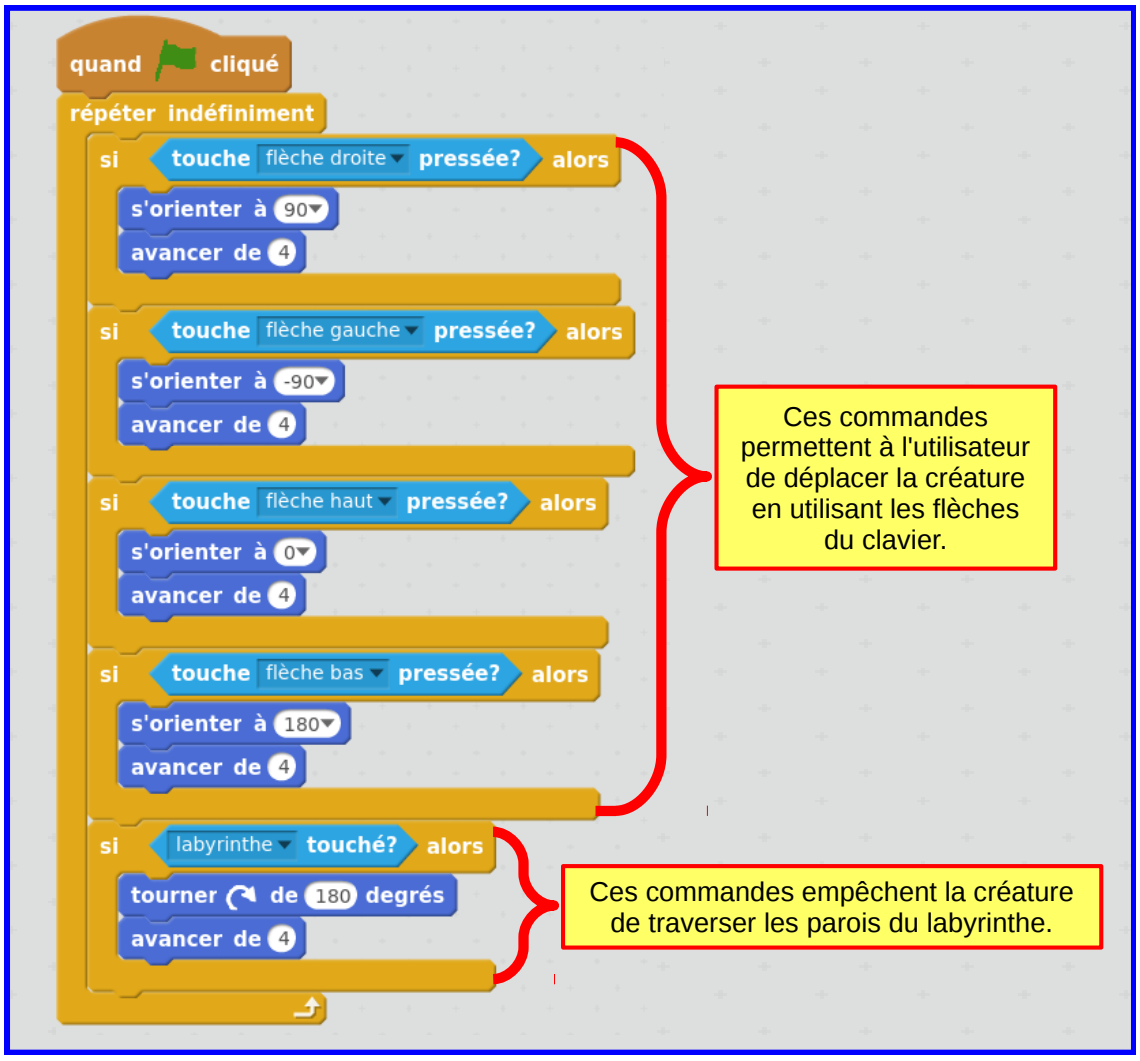
Il y a deux cas de figure possibles :

1) C'est l'utilisateur qui déplace la créature en utilisant les flèches du clavier.

2) La créature se promène toute seule dans le labyrinthe, de manière autonome.

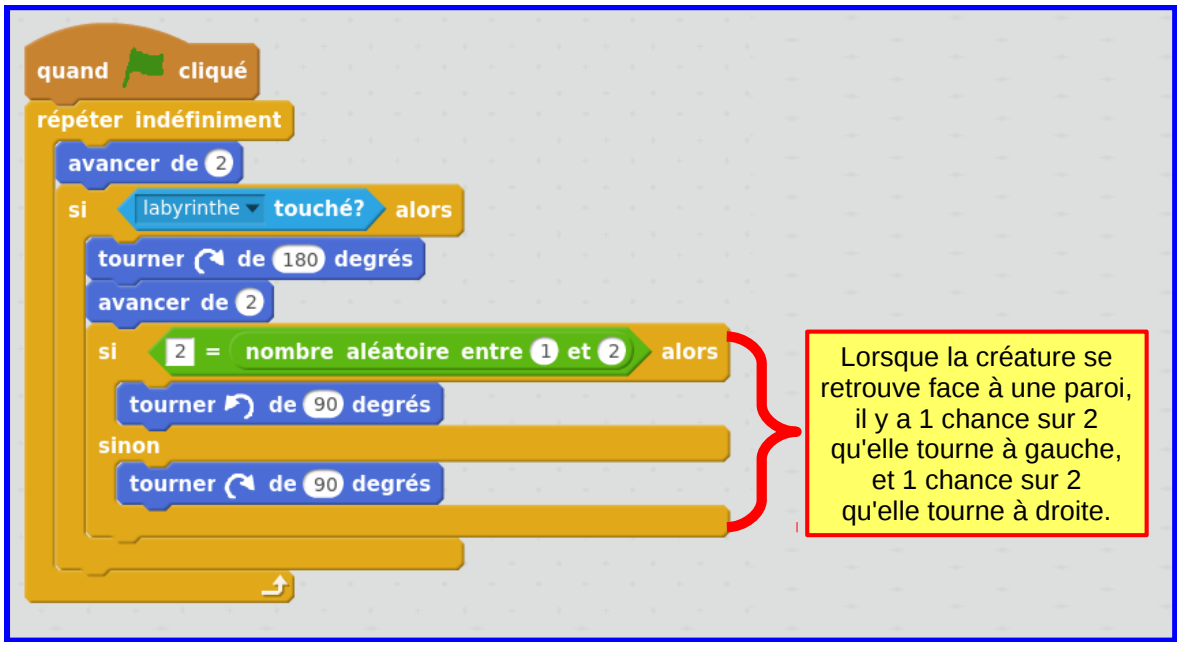
Cas 1 : l'utilisateur déplace la créature en utilisant les flèches du clavier :

Pour que l'utilisateur puisse déplacer une créature à l'intérieur d'un labyrinthe en utilisant les flèches du clavier, on insère les commandes suivantes :



Cas 2 : la créature se promène dans le labyrinthe de manière autonome :

Pour que la créature se promène seule dans le labyrinthe, de manière autonome, on utilise une **structure de contrôle double** fonctionnant grâce à un **nombre aléatoire** :



Remarques :

$\Longrightarrow$ Avant d'insérer les commandes présentées ci-dessus, il faut charger le fichier contenant le labyrinthe. Le contenu de la commande [touché?] dépend du nom de ce fichier.

$\Longrightarrow$ La **taille** de la créature doit être adaptée à taille des allées du labyrinthe.
Souvent il est nécessaire d'ajuster la taille de la créature en plaçant la commande [mettre à ( ) % de la taille initiale] avant la boucle.

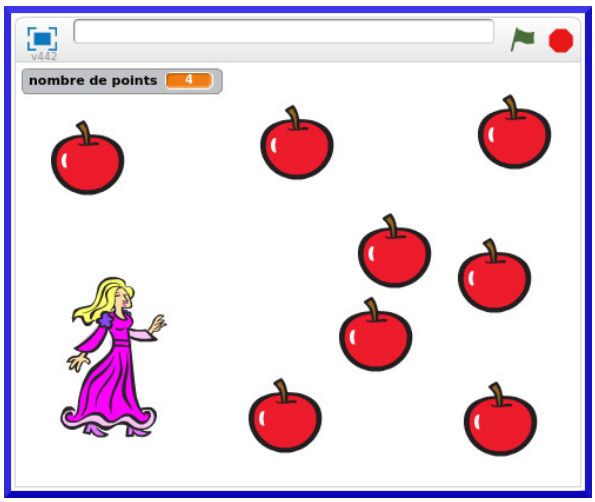
$\Longrightarrow$ Les valeurs insérées dans les commandes [avancer de ( )] doivent être adaptées à la taille des allées du labyrinthe, ainsi qu'à la taille de la créature.

Comment insérer des objets que le joueur doit récupérer ?

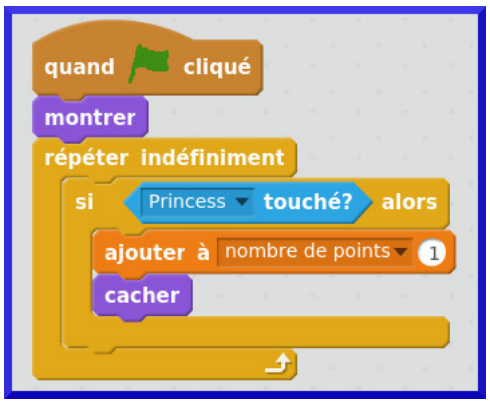
Dans un jeu, il est fréquent que le joueur doive récupérer des objets : par exemple des pièces d'or, des clés, des points bonus, des potions magiques, ou des cœurs...

Exemple :

Dans cet exemple, le joueur fait avancer une princesse en utilisant les flèches du clavier. Le but du jeu est de récupérer le plus de pommes possible : pour récupérer une pomme, le joueur doit la toucher avec la princesse. A chaque fois que le joueur récupère une pomme, le nombre de points augmente :



Pour permettre au joueur de récupérer des pommes, on commence par ajouter un exemplaire de la pomme à récupérer, puis on insère les commandes suivantes :



Ces commandes concernent la pomme.

Voir la suite de l'exemple à la page suivante

On duplique ensuite la pomme en plusieurs exemplaires : on sélectionne l'icône  , puis on clique sur la pomme, dans la liste des créatures :

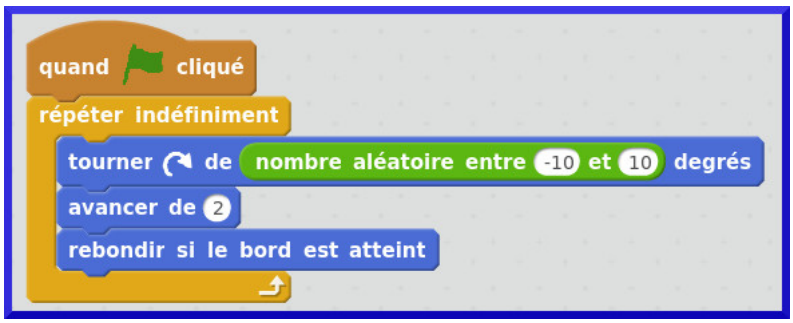
⟹ Une fois dupliquée, la nouvelle pomme peut être déplacée à n'importe quel endroit de la scène en utilisant le pointeur de souri.

⟹ Les commandes qui concernent la pomme sont automatiquement dupliquées. Il n'y a donc pas besoin de s'en occuper.

⟹ Il est possible de dupliquer la pomme autant de fois qu'on souhaite afin d'obtenir le nombre de pommes souhaité.

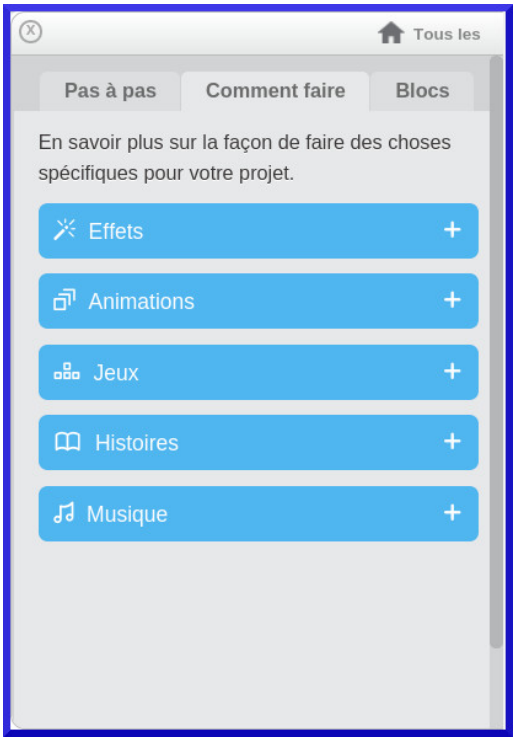
Comment faire avancer une créature de manière aléatoire ?

Pour faire avancer une créature de manière aléatoire, on utilise une **boucle** contenant une commande de mouvement munie d'un nombre aléatoire. Par exemple :



Comment obtenir de l'aide ?

Pour obtenir de l'aide, on clique sur l'icône . Cet icône se trouve tout à droite de l'espace de travail de Scratch. Lorsqu'on clique sur l'icône , une fenêtre d'aide s'ouvre sur la droite :



⟹ **Pas à pas** : ici se trouvent des marches à suivre qui te permettent de créer des programmes étape par étape. Chaque étape est illustrée par une petite vidéo explicative. Parmi les programmes proposés, on trouve des animations, des jeux, et beaucoup d'autres choses...

⟹ **Comment faire** : ici se trouve des notices qui expliquent comment procéder pour faire quelque chose en particulier (par exemple : comment faire marcher une créature ?). Ces notices sont classées en cinq catégories : Effets, Animations, Jeux, Histoires, Musique. Chaque catégorie contient plusieurs notices.

⟹ **Blocs** : ici se trouve le mode d'emploi de chacune des commandes de Scratch. Ces modes d'emploi sont classés selon les rubriques de commandes de Scratch. Chaque mode d'emploi explique à quoi sert la commande et comment l'utiliser. Chaque mode d'emploi présente un exemple d'utilisation de la commande.

Comment interagir avec l'utilisateur du programme ?

En programmation informatique, on a souvent besoin de **poser des questions** à l'utilisateur du programme, et de **récupérer les réponses** de l'utilisateur pour les exploiter. Par exemple, lorsqu'on sauvegarde son travail dans un traitement de texte, il faut que le programme puisse **demander à l'utilisateur** sous quel nom il souhaite sauvegarder son travail. Dans un jeu de course de voitures, il faut que le programme puisse **demander au joueur** sur quel circuit il souhaite conduire et quel voiture il souhaite utiliser.

Pour **poser une question** à l'utilisateur du programme, on utilise la commande 

.

Pour **récupérer la réponse** de l'utilisateur, on utilise la commande 

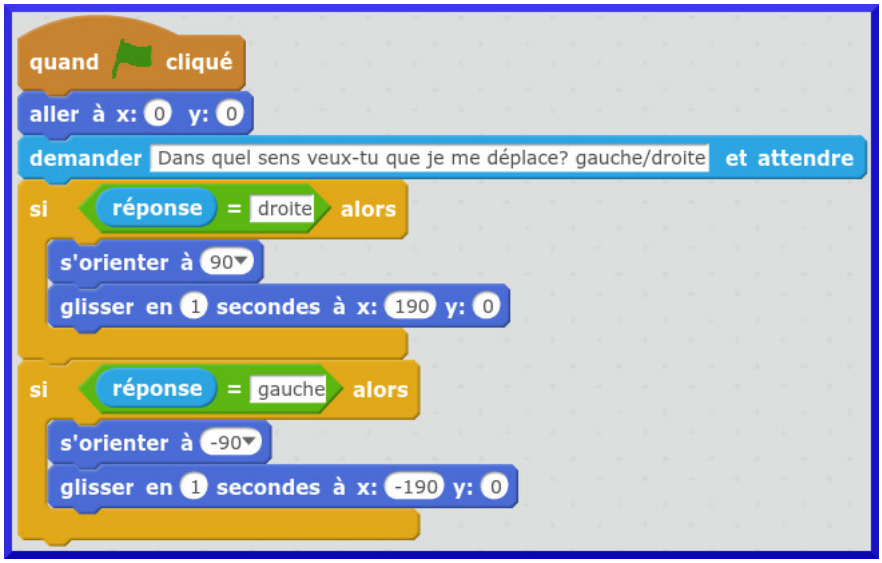
.

Ces commandes se trouvent dans la rubrique « **Capteurs** ».

Exemple :

Dans cet exemple, la créature va demander à l'utilisateur dans quel sens elle doit se déplacer : l'utilisateur devra décider si la créature se déplace à droite ou si elle se déplace à gauche. Si l'utilisateur répond « droite », la créature va se déplacer à droite. Si l'utilisateur répond « gauche », la créature va se déplacer à gauche :

Pour réaliser ce programme, on insère les commandes suivantes :



Lorsqu'on clique sur l'icône  , la créature demande dans quel sens elle doit se déplacer. L'utilisateur écrit « droite » ou « gauche » dans la barre au bas de la scène, puis clique sur l'icône  : la créature se déplace à droite ou à gauche de la scène.

Les commandes de dessin

Les commandes de dessin se trouvent dans la rubrique « **Stylo** ».

Cette commande change la **couleur** du stylo : elle fixe la couleur du stylo à la couleur correspondant à la valeur indiquée dans la case blanche. Le tableau suivant indique les valeurs correspondant aux principales couleurs :

Couleur	Valeur
rouge	0
orange	24
jaune	36
vert	60
turquoise	96
bleu	132
magenta	176

Les principales couleurs

Cette commande fait varier la couleur du stylo : plus le nombre indiqué est élevé, plus la variation de couleur est importante.

Cette commande modifie l'**épaisseur des traits** dessinés par le stylo. Plus le nombre indiqué est élevé, plus les traits sont épais.

Cette commande permet au stylo de tracer des traits en se déplaçant.

Cette commande interrompt le traçage des traits : le stylo ne dessine plus.

Cette commande efface tous les traits dessinés par le stylo. Attention : une fois le dessin effacé, il n'est plus possible de le récupérer !

Les opérateurs mathématiques

Les opérateurs mathématiques se trouvent dans la rubrique « **Opérateurs** ».

Il s'agit de l'opérateur d'**addition**. Dans les cases blanches, on peut **écrire des nombres** ou **insérer des variables**. Par exemple :

Il s'agit de l'opérateur de **soustraction**.

Il s'agit de l'opérateur de **multiplication**.

Il s'agit de l'opérateur de **division**.

Cet opérateur indique le **reste** d'une division entière.

Cet opérateur permet de calculer la valeur d'une **fonction** : on sélectionne la fonction à calculer dans le menu déroulant de l'opérateur. Par exemple :

Les fonctions trigonométriques fonctionnent **en degrés**.

Il s'agit de l'opérateur de **nombre aléatoire** : lorsque cet opérateur est exécuté, un nombre aléatoire est tiré au sort. Ce nombre est compris entre les deux nombres indiqués dans les cases blanches de l'opérateur. Lorsque le programme est exécuté plusieurs fois, le nombre tiré est différent à chaque exécution. Lorsque cet opérateur est placé dans une boucle, le nombre tiré est différent à chaque répétition.