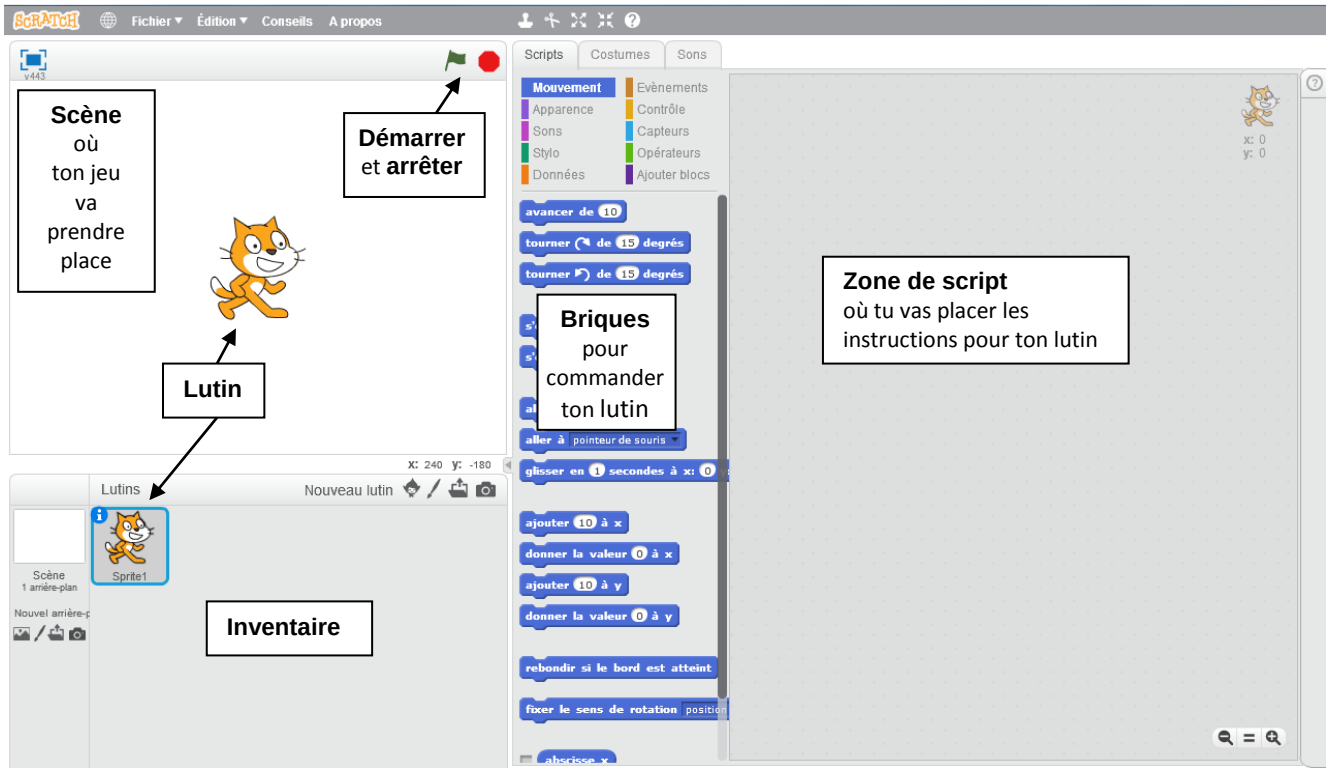
Sommaire :

A) Les bases de Scratch :

Scratch est un environnement de développement graphique qui permet de réaliser des animations interactives très facilement, grâce notamment à un langage de programmation (ou script) visuel.

1) Interface :

A l'ouverture de Scratch, tu vois :



Un « lutin » est un personnage ou un objet de ton jeu. Les lutins peuvent se déplacer et être actif ou être des accessoires qui restent là.

Les lutins ne peuvent rien faire par eux-mêmes. L'action d'un lutin vient des scripts de la fenêtre de script. Les zones de scripts sont propres à chaque lutin.

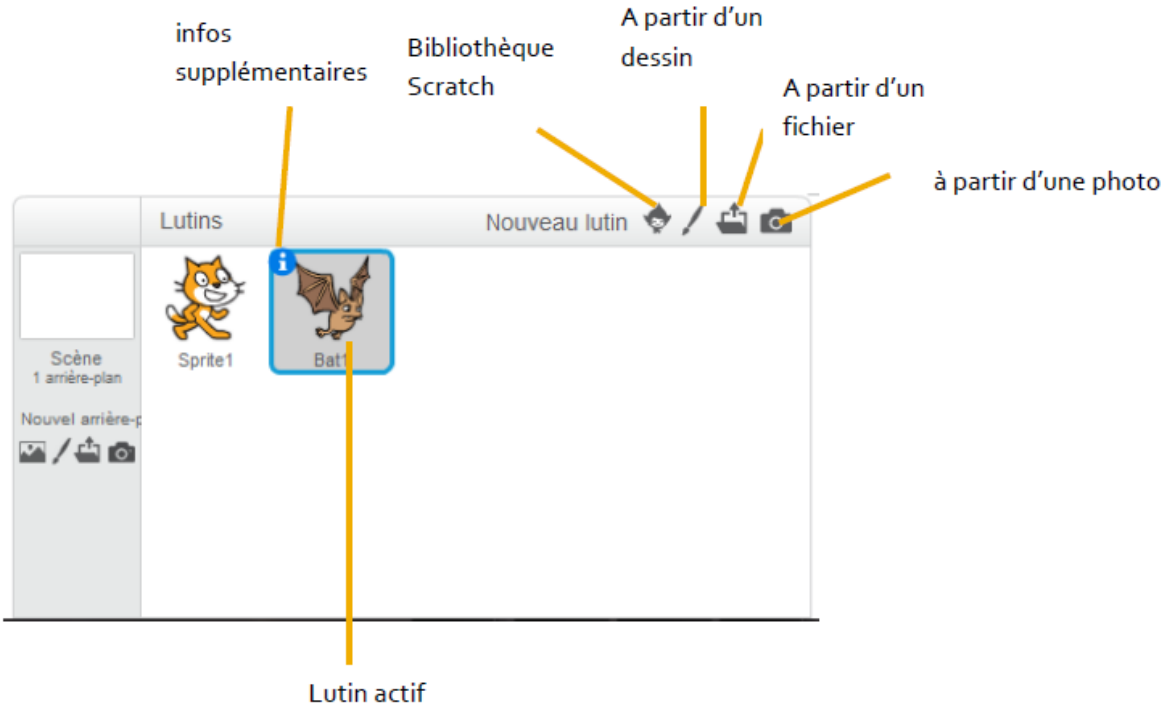
Ces scripts sont des instructions pour dire au lutin ce qu'il doit faire. Les instructions sont des ensembles de briques. Tu fais glisser ces briques depuis la zone des briques vers la zone "Scripts". Ces briques s'emboîtent comme un puzzle pour créer les instructions.

2) Lutins :

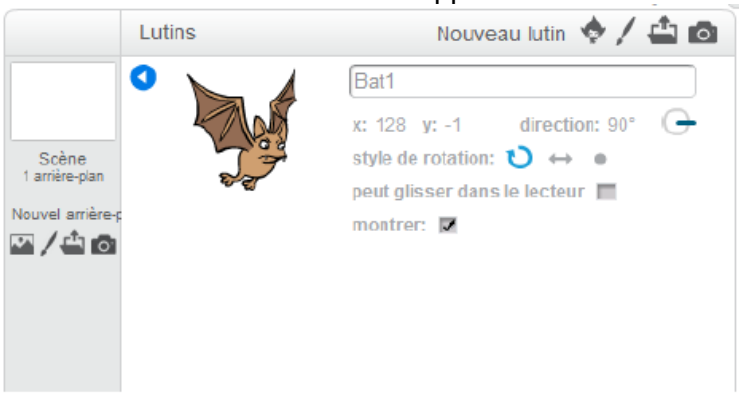
Les lutins sont les personnages et les objets que vous pouvez programmer. Un lutin ne voit et n'exécute que les programmes qui ont été écrit sur SA page de script (quand l'image en haut à droite de la zone de script représente le lutin en question).

En revanche quand un évènement arrive (appui sur le drapeau vert, appui sur espace..) tous les lutins le voient et tous les lutins peuvent potentiellement réagir à cet évènement.

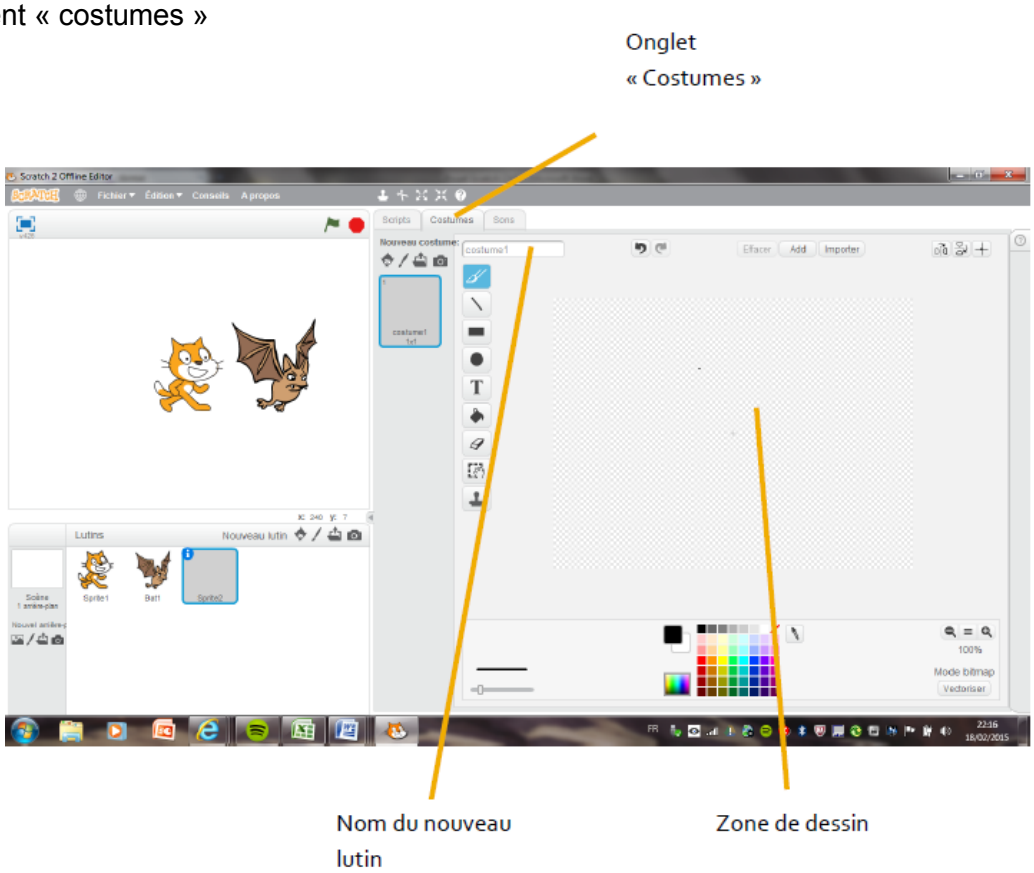
Les lutins sont gérés dans la fenêtre « inventaire » en bas à droite. C'est dans cette fenêtre que l'on peut ajouter/supprimer des lutins.



Lorsque l'on clique sur le bouton « i » des informations supplémentaires sur le lutin apparaissent.



Lorsqu'on clique sur « dessiner un lutin », la page de script est remplacée par une page de dessin, et l'onglet devient « costumes »

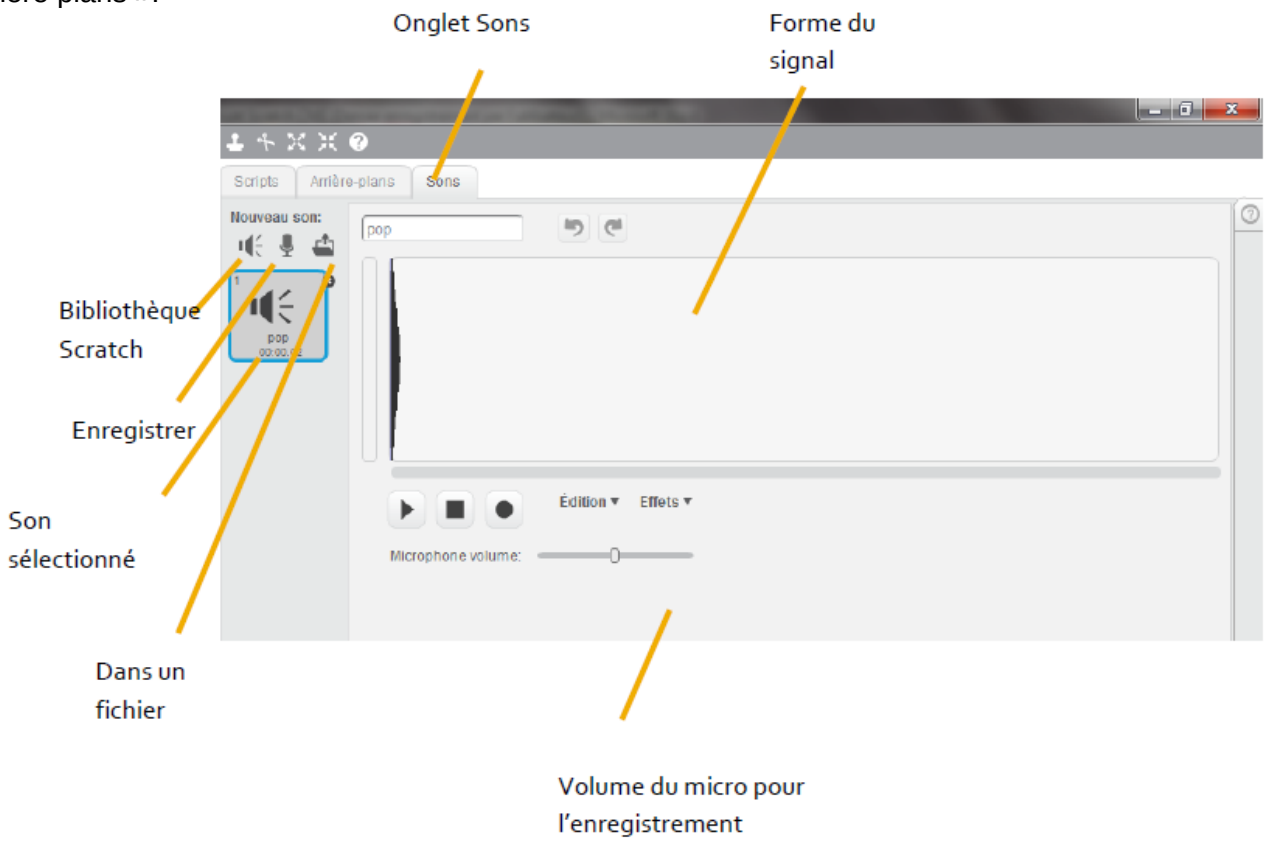


3) Arrière plans :

Les arrière-plans sont les décors des scènes. Comme les lutins ils sont programmables, et reçoivent également les évènements déclenchant les scripts.
Des icônes identiques à celles des lutins permettent d'ajouter des nouveaux arrières plans.

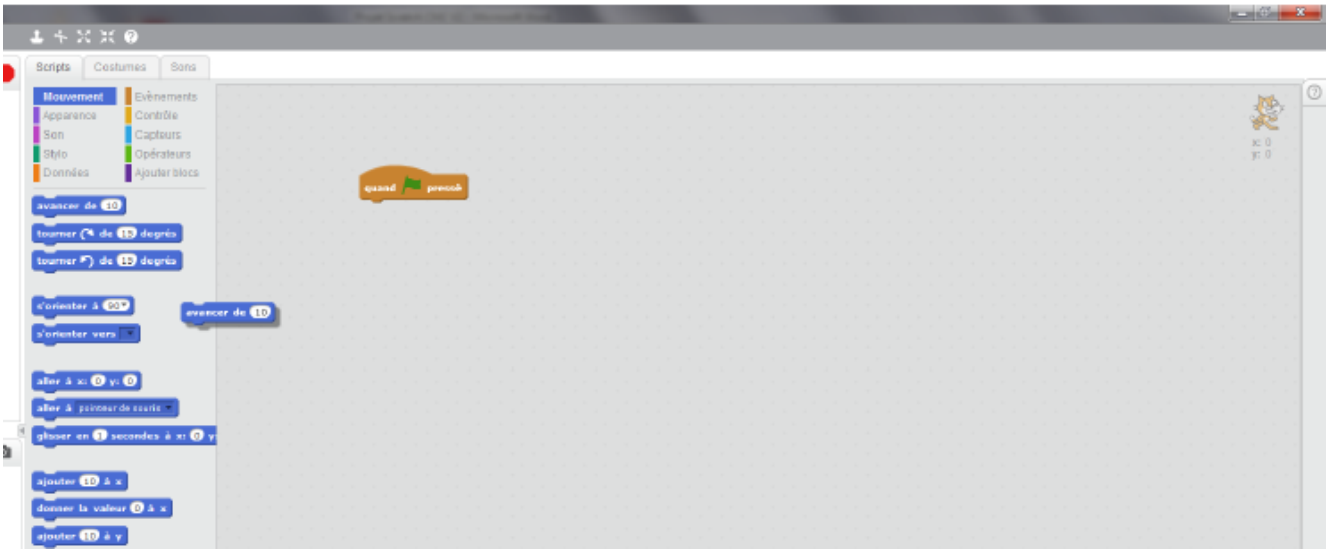
4) Sons et musiques :

Il est possible d'associer aux lutins et aux arrière-plans des sons et de musiques. Ces sons, comme les programmes, sont associés à un lutin ou à un arrière-plan. Les autres lutins ou arrière-plans ne les voient pas.
Cette fonctionnalité se trouve dans l'onglet « sons », le troisième onglet situé à droite de « costumes » ou « arrière plans ».

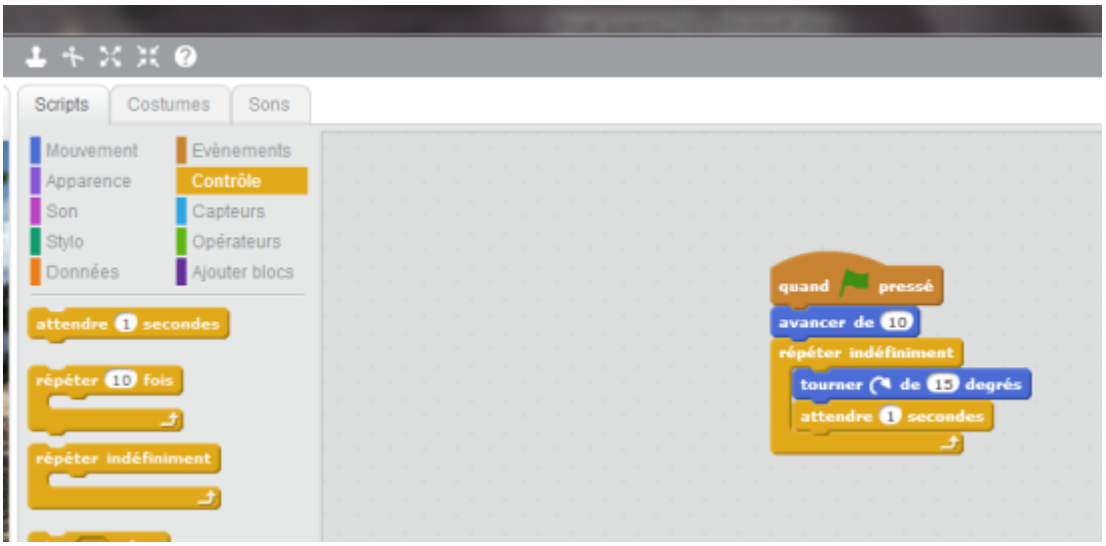


5) Programmation et fonctionnement des blocs :

Les blocs de programmation sont disponibles dans l'onglet « scripts » pour chaque lutin. On peut identifier le lutin que l'on programme en regardant le dessin en haut à droite de la zone de programmation.
Les blocs se positionnent sur la zone de script par simple glisser-déposer.



Les blocs s'enclenchent les uns à la suite des autres comme un puzzle.
Vous pouvez mettre autant de scripts que vous le souhaitez sur la page de script d'un lutin.



Un script doit toujours commencer par un évènement. C'est ce qui permettra de lancer le script. Les instructions seront ensuite exécutées dans l'ordre dans lesquelles vous les avez mises.

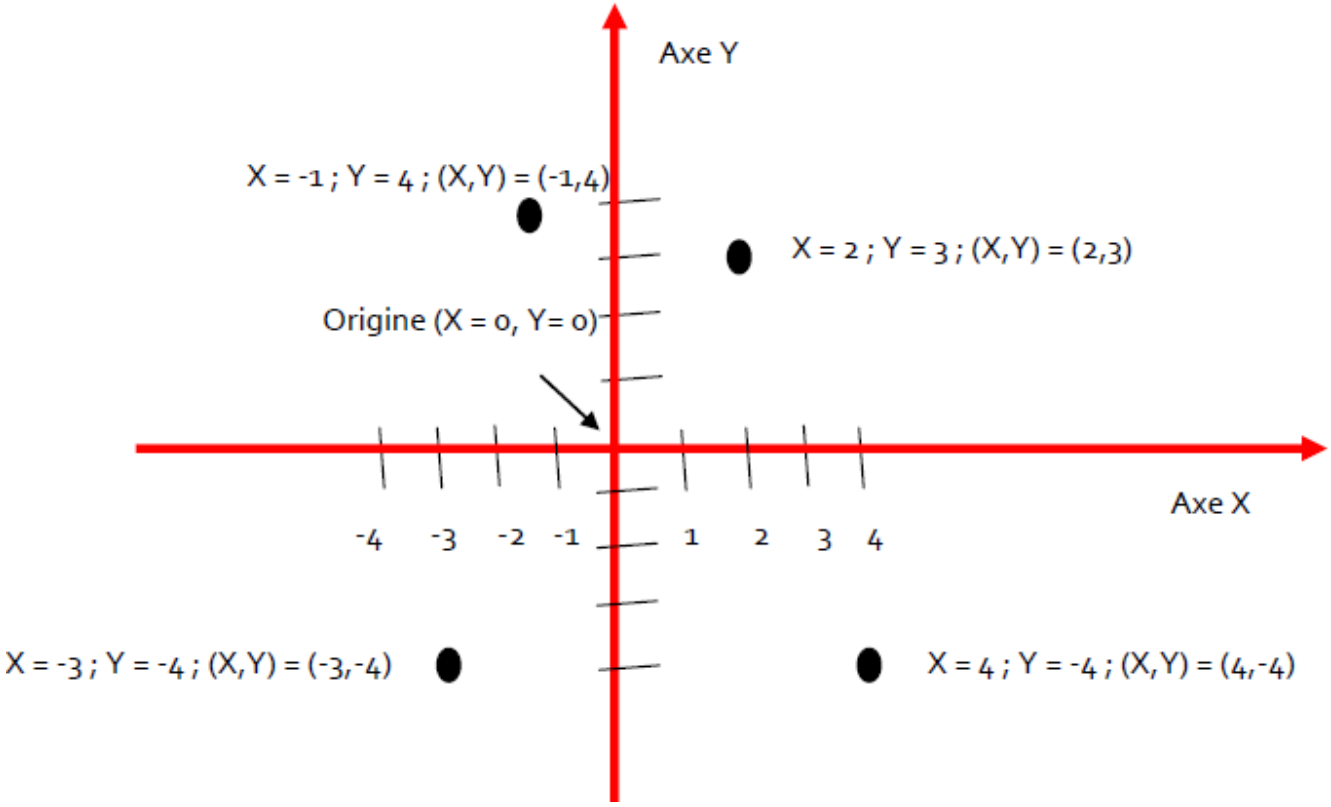
Point important, tous les scripts, de tous les lutins s'exécutent en même temps (ou plutôt parallèlement), donc si tous les lutins ont un script qui démarre avec le même évènement, alors tous ces scripts s'exécuteront en même temps.

On peut copier un script d'un lutin à l'autre en le glissant sur l'image du lutin destinataire.

B) Notions importantes à savoir pour écrire des programmes :

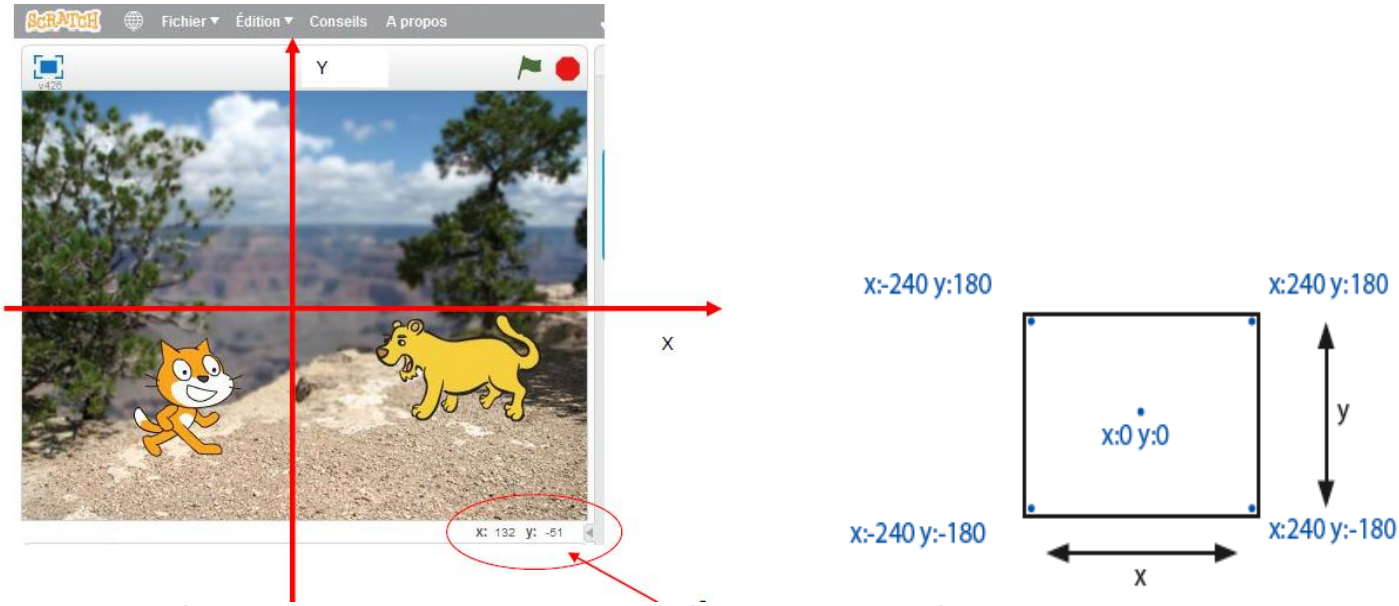
1) Coordonnées dans le plan :

Les lutins et objets dans la scène sont repérés par leurs coordonnées cartésiennes. Les coordonnées cartésiennes permettent de repérer un point dans un plan selon un axe horizontal X et un axe vertical Y.



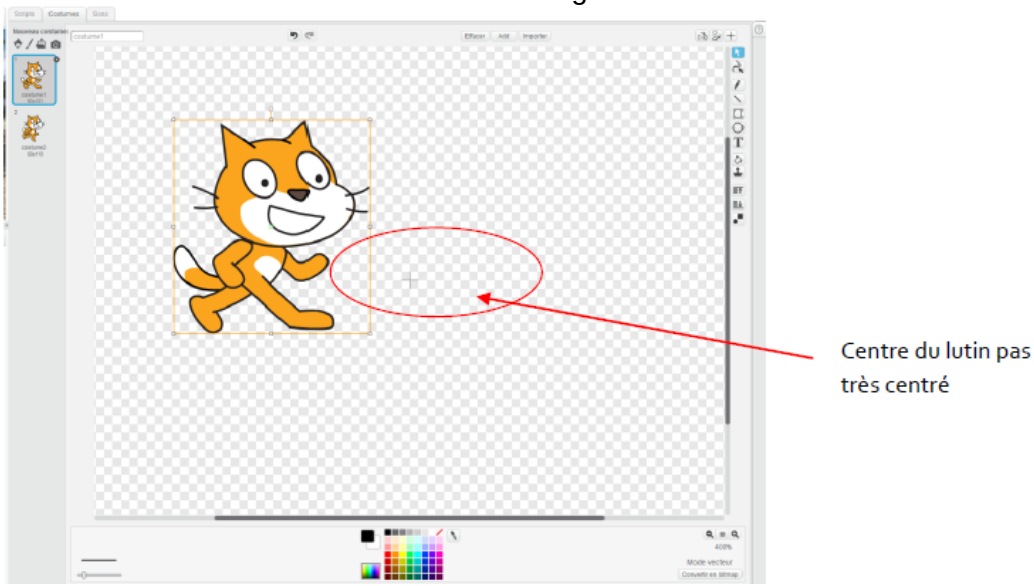
La scène est un rectangle de 480 unités de large et 360 unités de hauteur.

Dans la scène, l'origine (X=0, Y=0) est au centre de l'image. Tout ce qui est au dessus de l'axe horizontal correspond à un Y positif, en dessous à un Y négatif. Tout ce qui est à droite de l'axe vertical correspond à un X positif, à gauche à un X négatif.

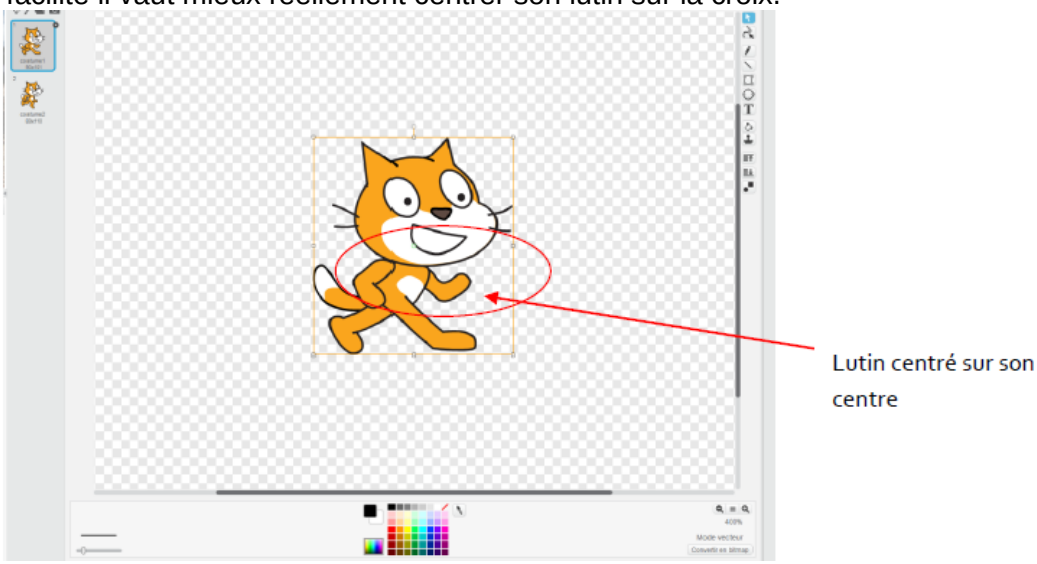


Les coordonnées de la pointe de la souris sont affichées en bas de la scène.

Le lutin, lui, est repéré par rapport à son « centre ». Pour voir ou celui-ci est situé, allez dans l'onglet « Costume » de votre lutin. Le centre du lutin est la croix grise.

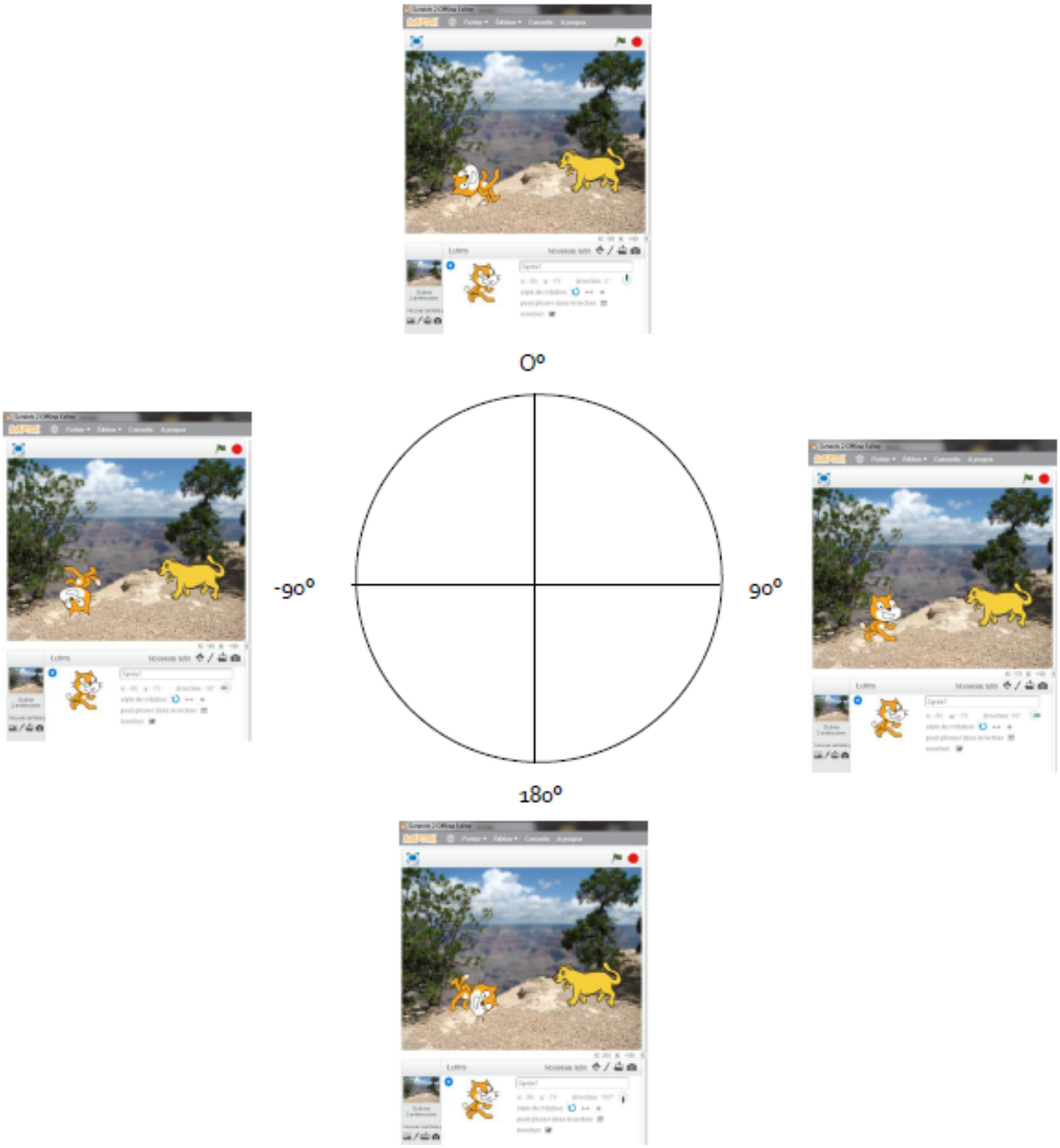


Pour plus de facilité il vaut mieux réellement centrer son lutin sur la croix.



2) Orientation et rotations :

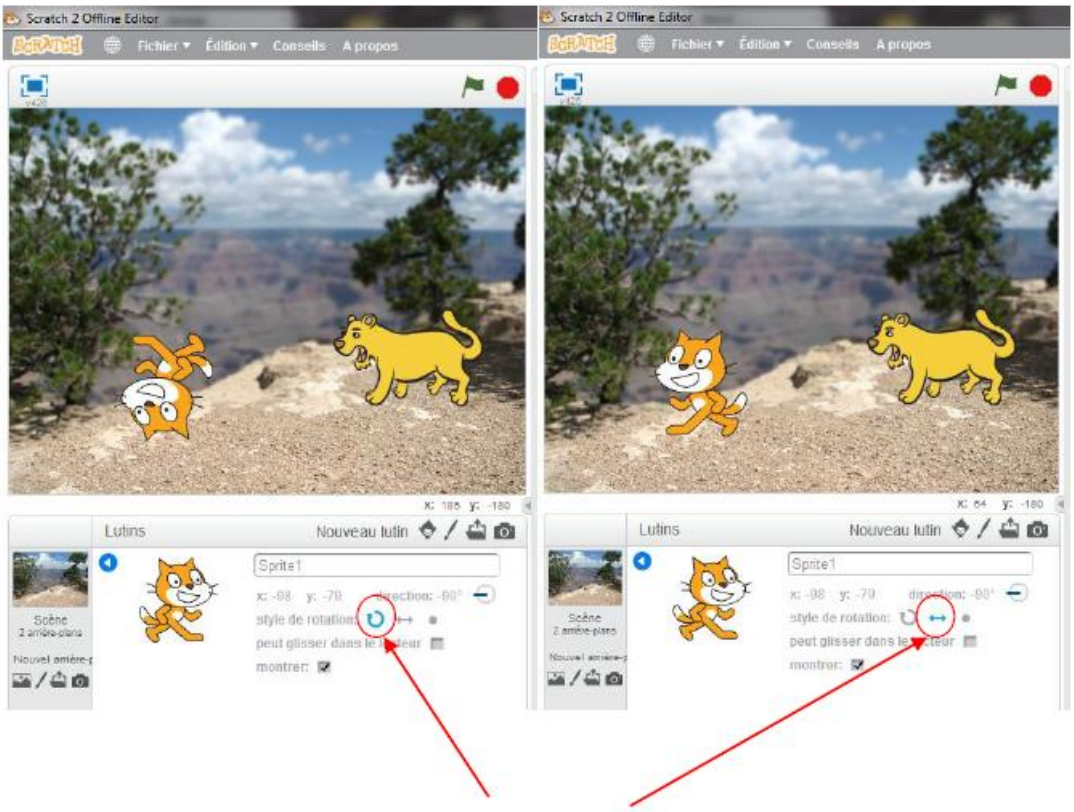
Vous pouvez orienter vos lutins. Les orientations sont repérées par un angle selon le schéma ci-dessous (regardez bien l'orientation du chat dans l'image) :



Quand vous demandez à votre chat de tourner de 15°, il va donc tourner dans le sens des aiguilles d'une montre de 15°. Si vous lui dites de tourner de -15° il tournera dans le sens inverse des aiguilles d'une montre (pour les mathématiciens, ça s'appelle le sens trigonométrique).

Vous remarquez que quand on demande au chat de s'orienter vers la gauche (-90°), il se retrouve la tête en bas. C'est ennuyeux.
Pour éviter cela, il faut contraindre les mouvements du chat. Cliquez sur le petit i au niveau de votre lutin et dans « style de rotation » choisissez ↔

Miracle : le chat a de nouveau la tête en haut.

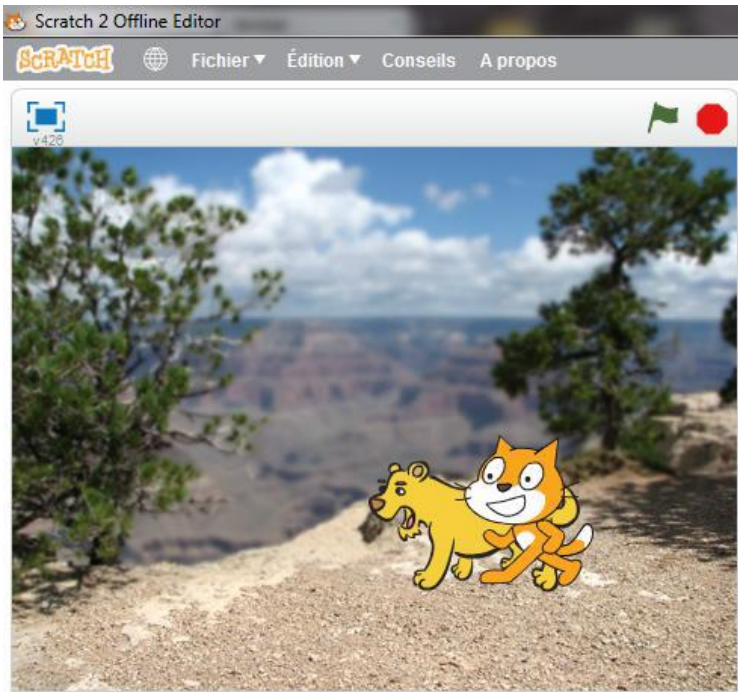


La différence est la !

3) Plans :

Les lutins et les objets dans Scratch sont positionnés sur plusieurs plans ou couches qui se superposent.

Typiquement, l'arrière plan est derrière, comme son nom l'indique. Sur l'image ci-dessous le chat est au premier plan, la lionne au second, et l'arrière plan…toujours en dernier.



Evidemment ces histoires de plans n'interviennent que quand deux lutins se croisent. Il faut choisir lequel cache l'autre.
Pour modifier le positionnement des lutins il suffit d'utiliser, dans la zone de script, les deux instructions suivantes, que vous trouvez dans les blocs « Apparence » :

4) Conditions :

Les conditions permettent de tester des variables. Les conditions commencent par un « Si », sont suivies d'un « alors » et quelquefois d'un « sinon ». On utilise tout le temps des conditions dans le langage courant :

« **Si** il fait beau **alors** je vais skier»
« **Si** il fait beau **alors** je vais skier sinon je reste chez moi »
« **Si** il fait plus de 25°C **alors** je mets un short **sinon** je mets un pantalon »

Dans le premier cas on ne précise pas ce qu'il se passe s'il ne fait pas beau.
On teste ici la variable « météo » ou « température », En langage de programmation on pourrait l'écrire :

Si météo = beau **alors** va au ski sinon reste à la maison.
Si température > 25°C **alors** mets un short **sinon** mets un pantalon.

On peut évidemment imbriquer plusieurs niveaux de conditions :

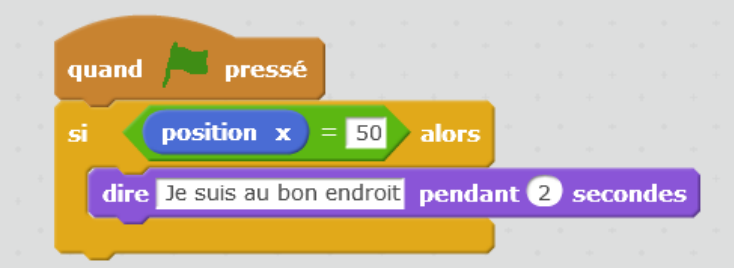
Si température > 30°C **alors** mets un maillot de bain
sinon si température > 20°C **alors** mets un short
sinon mets un pantalon.

On trouve les blocs de conditions dans l'onglet « **Contrôles** » :

Les cases hexagonales vides sont censées être remplies par un test (sinon on ne teste rien du tout.)

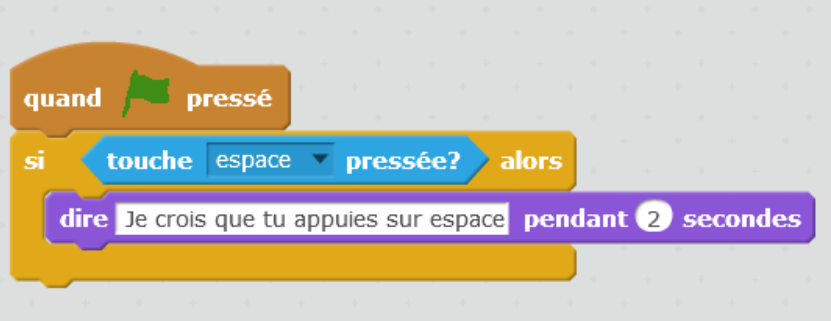
Vous trouverez deux types de tests possibles dans Scratch :

- **Des opérateurs** (voir le chapitre qui y est consacré plus bas), qui permettent notamment de tester des valeurs :



Ici on teste que la position sur l'axe X du lutin est 50, si c'est le cas, le lutin dit « Je suis au bon endroit »

- **Des capteurs** qui détectent les interactions du joueur avec le jeu ou des lutins entre eux.

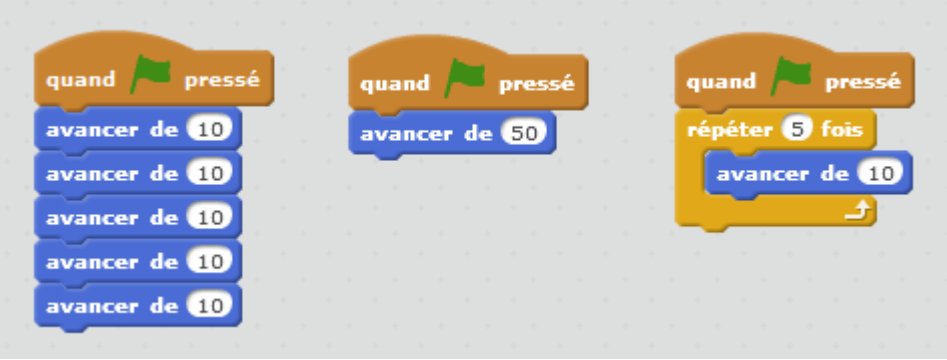


Quand le jeu démarre, si la touche « espace » est pressée, le lutin nous en fera part.

5) Boucles :

Les boucles sont extrêmement utiles en programmation, en effet elles permettent de faire une série d'instructions plusieurs fois sans avoir à écrire plusieurs fois cette série d'instruction. C'est pratique car en bon informaticien fainéant, ça fait moins de travail à faire et puis, ce qui ne gâche rien, le programme est plus lisible.
Vous trouverez les blocs « **boucles** » dans l'onglet « **contrôles** ».

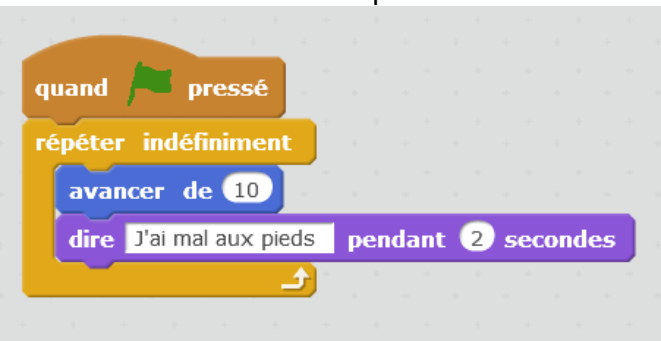
Prenons quelques exemples. Les trois scripts ci-dessous donnent le même résultat :



Le lutin va avancer de 50 pas. (En pratique, au moment de l'exécution, vous noterez une légère différence entre les deux premiers scripts et le troisième).
Dans ce cas, évidemment, le second script est le plus « intelligent » car il fait avancer de 50 pas en une seule instruction.

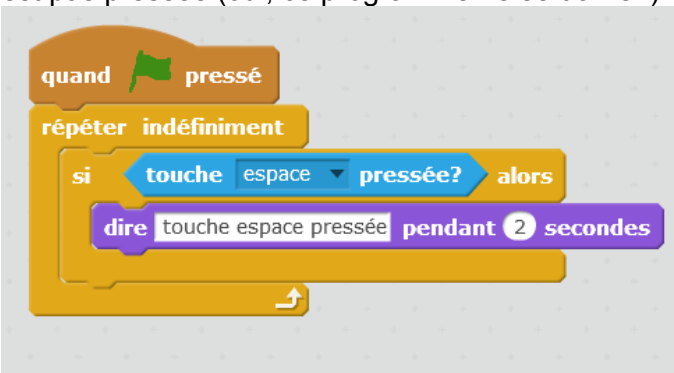
Il existe deux autres types de boucle très utiles : la **boucle dite « infinie »** et la **boucle avec condition de fin**.

La **boucle infinie** sert à faire les mêmes instructions en permanence sans indication de fin :

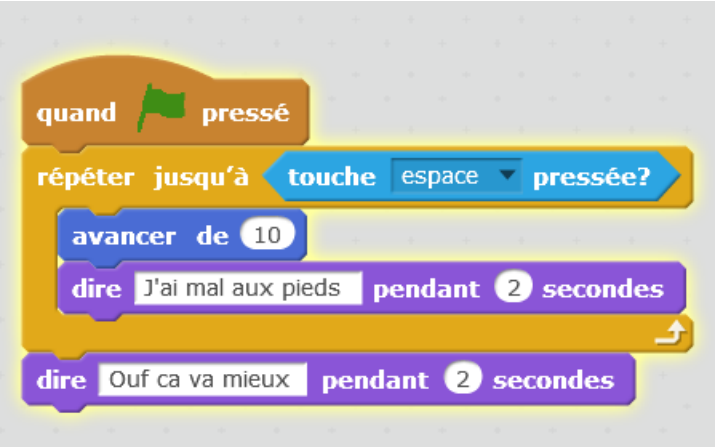


Le lutin va avancer de dix pas et dire « j'ai mal aux pieds » indéfiniment, enfin, jusqu'à temps qu'on arrête l'animation. Ce type de boucle sert par exemple à faire défiler des personnages, ou encore à tester de manière répétée si une action a eu lieu.

Dans l'exemple ci-dessous, le lutin dira « touche espace pressée » quand j'appuie sur la touche espace, et ne dira rien quand elle n'est pas pressée (oui, ce programme ne sert à rien) :



Enfin **la boucle avec condition de fin** permet d'exécuter de manière répétée plusieurs instructions jusqu'à ce qu'un évènement arrive :



Dans cet exemple, le lutin continue sa marche forcée en se plaignant de ses pieds, jusqu'à ce qu'un humain bienveillant le libère (on sort alors de la boucle) en pressant la touche « espace ». Il nous dit alors que ça va mieux.

Notez bien la différence entre les deux derniers exemples : même si la condition (touche espace pressée) est la même, dans la boucle infinie le fait de presser « espace » n'arrête pas la boucle, dans le cas de la boucle avec condition, c'est justement l'appui sur « espace » qui arrête la boucle.

6) Opérateurs :

Vous trouvez les blocs « opérateurs » dans l'onglet opérateurs (vert clair).
Vous remarquerez qu'il y a deux formes de blocs, des hexagonaux et d'autres avec les coins arrondis.

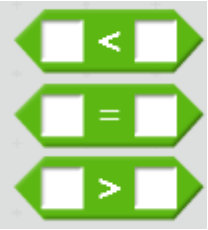
Commençons par les plus évidents, les blocs arrondis sont ceux qui permettent de réaliser des opérations :

- celles qui s'appliquent à des nombres et que vous connaissez : addition, soustraction, multiplication, division

- celles qui s'appliquent à des mots ou suites de lettres :
  - regrouper ou concaténer deux mots : par exemple regroupe « bonjour » « monsieur » donne « bonjourmonsieur »
  - trouver la lettre numéro n d'un mot
  - compter le nombre de lettres dans un mot

Les blocs hexagonaux sont utilisés dans les conditions (voir chapitre conditions), ils permettent de réaliser des tests :

- comparer des valeurs : égalité, supérieur, inférieur.



- les opérateurs logiques : et, ou, non

Arrêtons-nous quelques instants pour bien comprendre ces opérateurs en reprenant les exemples déjà vu avec les conditions (voir plus haut) :

◦ L'opérateur ET permet de tester que deux choses sont exactes. Par exemple : » Si il fait beau ET que c'est le weekend alors je vais skier ».
On ira skier que si les deux conditions sont réunies, il fait beau et c'est le weekend.

◦ L'opérateur OU permet de tester qu'au moins une des conditions est vérifiée. Par exemple : « Si c'est le weekend OU les vacances alors je peux faire la grasse matinée ». Bien sûr si les deux sont vraies alors on peut dormir aussi.

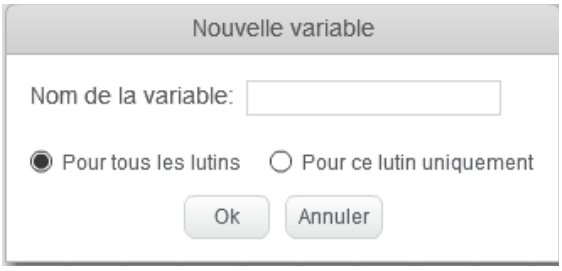
◦ L'opérateur NON permet de tester le contraire. Par exemple on veut écrire :
« Si il ne fait pas beau alors je ne vais pas skier ».

C'est possible avec l'opérateur NON :
« Si NON il fait beau alors je ne vais pas skier » ou encore
« Si NON météo= beau alors je ne vais pas skier »

7) Variables :

Autre élément extrêmement important en programmation : les variables.
Les variables permettent de stocker une valeur qui pourra être utilisée par le programme. Cette valeur peut changer, d'où le nom « variable ».

Dans Scratch, vous pouvez créer des variables en allant dans les blocs « **données** » et en cliquant sur « **créer une variable** ».



Vous devez lui donnez un nom. Essayez de donner un nom clair, afin de pouvoir vous repérer plus facilement quand vous souhaitez l'utiliser. Evitez par exemple « toto » ou « abcd ».

On vous demande également si c'est une variable pour tous les lutins ou pour ce lutin seulement.
Pour savoir quelle option choisir, il faut savoir si plusieurs lutins du jeu doivent voir cette valeur ou si seulement le lutin concerné en a besoin.

Une fois la variable créée vous voyez que de nouveaux blocs sont apparus.

8) Messages et synchronisation :

Comme nous l'avons vu, les lutins ont leur monde bien à eux. Ils ne savent pas ce que font les autres lutins et n'ont pas d'interaction sauf éventuellement lorsqu'ils se touchent.

Afin de pouvoir communiquer entre eux (et également avec l'arrière plan, qui est un lutin bien particulier), on utilise des messages. Ces messages nous permettent de synchroniser et déclencher des actions.

Vous **trouverez les messages** dans la rubrique « **Evènements** ».

Vous pouvez envoyer des messages et recevoir des messages. Quand un lutin envoie un message, tout le monde peut le voir, mais tout le monde n'est pas obligé de réagir. Vous pouvez créer autant de message que vous voulez, encore une fois, essayez d'être explicite quand vous créez le message, pour autant n'écrivez pas un roman.

Pour **créer un nouveau message**, dans le bloc « **envoyer un message** » cliquez sur la liste déroulante puis sur « **nouveau message** ».