Programmation par blocs

Groupe Algorithmique au Collège - I.R.E.M. de Clermont*

11 décembre 2019

Table des matières

*Joffrey Cottin (Lycée Saint-Pierre, Cusset); Pascal Lafourcade (UCA); Florian Liautaud (Collège Jean-Jacques Soulier, Montluçon); Guy Mas (Collège Joliot-Curie, Aubière); Mickaël Meyroneinc-Condy (Lycée Murat, Issoire); Malika More (UCA); Gaëtan Perrin (Collège Jean Vilar, Riom).

Résumé

Le document ci-après est le fruit du travail du groupe Algorithmique au Collège de l'I.R.E.M. de Clermont.

Il est écrit pour la version 2.0 du logiciel Scratch. Une version concernant Scratch 3.0 sera mise en ligne dans le courant de l'année 2020.

Il est à destination des enseignants du second degré à la recherche de traces écrites pour leurs élèves.

Nous avons essayé de produire un discours le plus scientifique possible, tout en restant accessible pour des élèves. Il ne s'agit ni d'un modèle, ni d'un document officiel.

Nous avons aussi choisi de donner peu d'exemples, afin de permettre aux enseignants et aux élèves de réaliser des documents personnels.

Ce document s'appuie aussi sur deux productions d'autres groupes I.R.E.M., notamment la Commission Inter-Irem Informatique [1] et la Commission Inter-Irem Lycée [2], dont on trouvera les références dans la bibliographie.

1 Initiation

1.1 Fenêtres

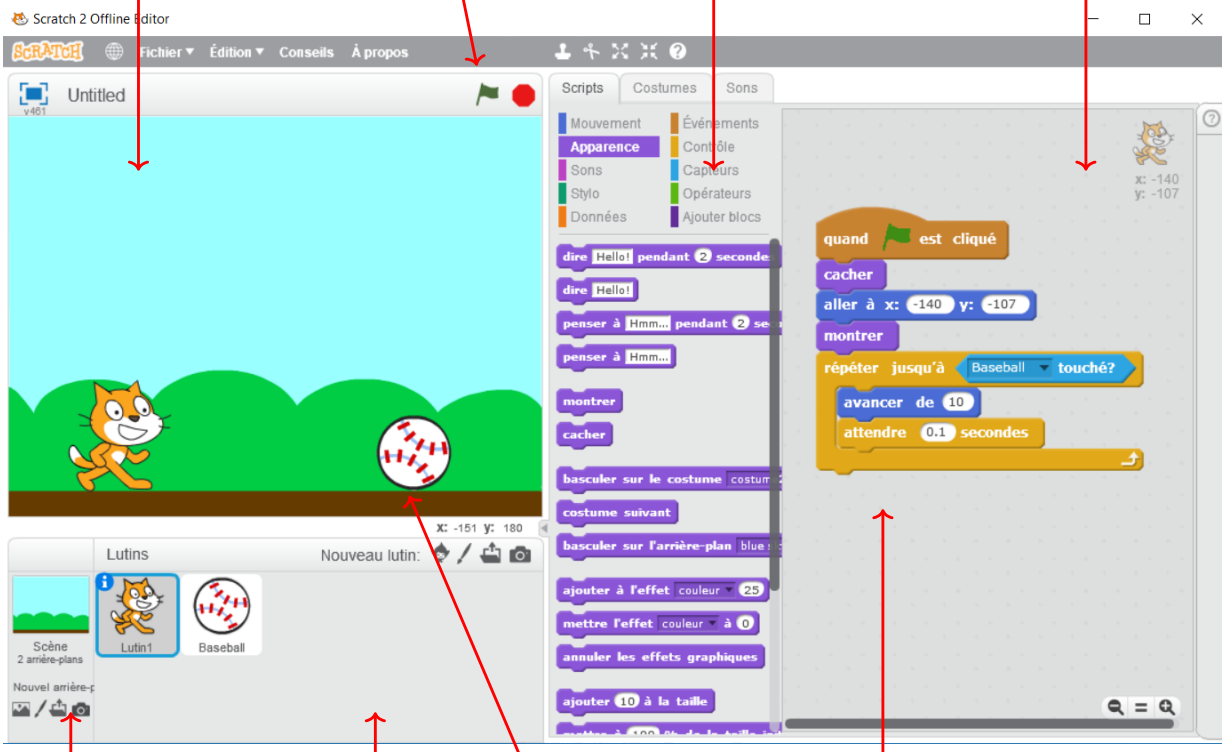
Lorsque l'on ouvre Scratch ou mBlock, on se retrouve devant plusieurs fenêtres dont chacune a sa spécificité.

Lorsqu'un programme est exécuté, ses effets sont visibles dans la **fenêtre d'exécution** ou **scène**.

Drapeau vert

Les blocs de commande sont accessibles via différents menus colorés depuis la **fenêtre des blocs**.

On glisse les blocs depuis la fenêtre des blocs dans la **fenêtre de programmation**.



Dans la **fenêtre des arrières-plans**, on peut sélectionner l'arrière-plan dont on modifie les propriétés (script, etc).

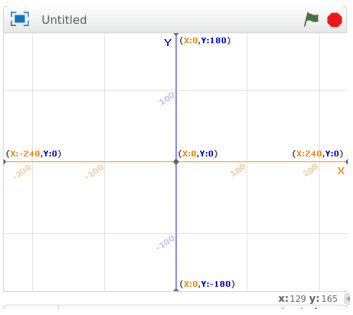
Dans la **fenêtre des lutins** on peut sélectionner le lutin dont on modifie les propriétés (script, etc).

Lutin : Personnage à qui on fait exécuter les instructions indiquées par **le script** qui lui est associé.

1.2 Repère

La fenêtre d'exécution est un plan repéré, dont l'origine est au centre de la fenêtre. Ses dimensions ne peuvent pas être changées :

— les abscisses vont de -240 à 240 ;
— les ordonnées vont de -180 à 180.



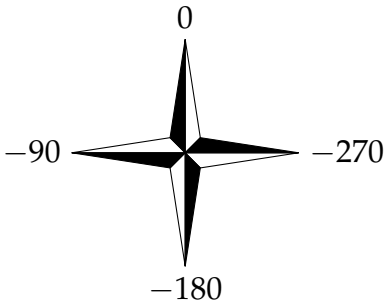
1.3 Stylo

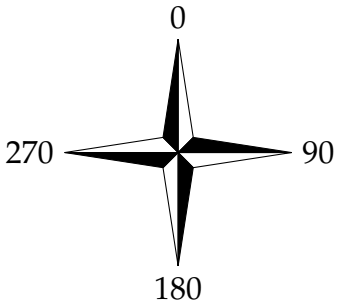
Chaque lutin est muni d'un stylo qui lui permet d'écrire sur la scène. La position de la pointe du stylo correspond aux coordonnées du lutin. Plusieurs paramètres sont associées au stylo (position d'écriture, couleur, taille, intensité). Par exemple : 

1.4 Angles

Il existe dans Scratch trois instructions relatives aux angles :

Le plan est orienté comme ci-dessous. L'instruction permet de mettre le lutin dans une position fixée par rapport à ce repère.



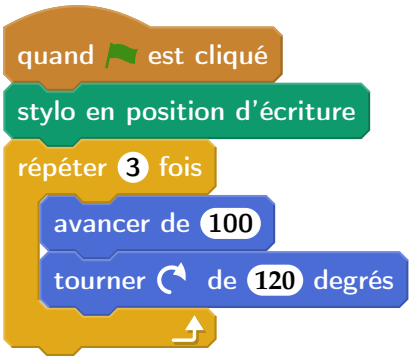


Le lutin tourne dans le sens indiqué par le bloc, d'un certain nombre de degrés, par rapport à sa position.

Règle

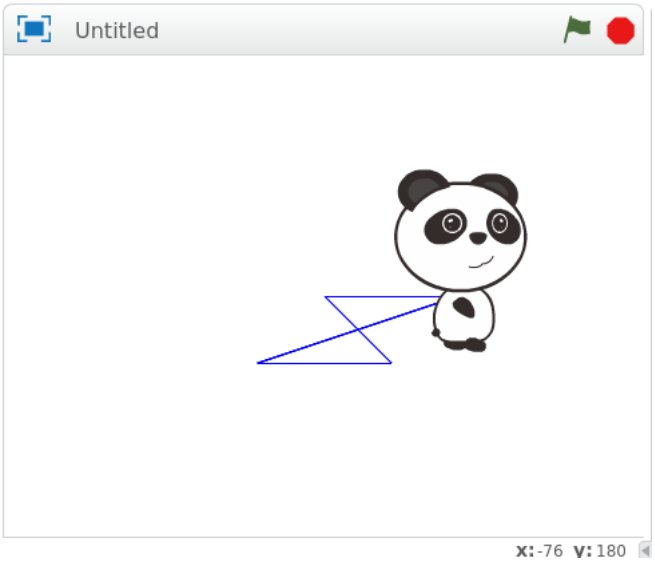
Pour tracer avec Scratch un angle mathématique donné, il faut demander au programme de tracer le supplémentaire de cet angle.

Exemple : Pour tracer un triangle équilatéral de côté 100, dont on rappelle que tous les angles mesurent 60°, on peut utiliser le programme suivant :

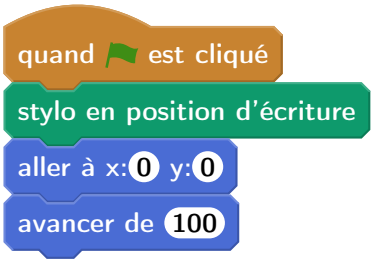


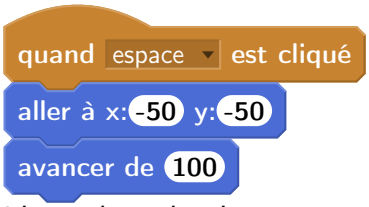
1.5 État du système

À la fin de l'exécution d'un programme, certaines valeurs internes de Scratch ne sont pas remises à zéro, comme par exemple la position du stylo, la position du lutin, l'orientation du lutin, etc ...



L'image ci-dessus a été obtenue avec l'enchaînement des deux programmes suivants, chacun traçant un segment à un emplacement spécifique.





L'état du stylo n'ayant pas changé, les diagonales, non désirées, sont tracés.

2 Variables

Généralement, les résultats des opérations effectuées par un ordinateur ne sont pas conservés en mémoire. Si on veut utiliser ou réutiliser une valeur plus tard, il faut la stocker en mémoire à un endroit précis. En informatique cet emplacement est appelé variable.

2.1 Qu'est ce qu'une variable informatique ?

Définition

Une variable est un emplacement dans la mémoire d'un ordinateur qui permet de stocker des informations.

Une variable porte un nom qui permet de retrouver l'information stockée à cet emplacement. Pour que vos programmes soient plus facile à lire il est important de nommer vos variables avec des noms précis (cote, longueur, angle,...).

2.2 Dans Scratch

On trouvera les blocs dont il est question dans le menu « Données » .
Le menu variable est au départ vide. Il faut en créer une pour avoir accès aux différentes commandes.

Bloc	Utilisation
mettre longueur à 0	**Affectation** : « 0 » est la valeur à mettre en mémoire à l'emplacement nommé « Longueur » « 0 » peut être aussi le résultat d'un calcul (2 + 3), une chaîne de caractère (« cerise »), une autre variable (cote).
ajouter à longueur 1	**Incrémentation** : La variable longueur est ici numérique, et on lui ajoute 1.
montrer la variable longueur	La variable est affichée dans la fenêtre principale.
cacher la variable longueur	La variable est cachée dans la fenêtre principale.

2.3 Propriétés de la variable

Affectation d'une valeur

Lors d'une affection du type mettre longueur à 0, la nouvelle valeur remplace l'ancienne valeur, qui est, elle, mise à jour dans la mémoire.

Affectation d'une autre variable

Lors d'une affectation du type mettre longueur à cote, le contenu de la variable longueur est remplacé par le contenu de la variable cote.
Pour sa part, le contenu de la variable cote n'a pas changé, il a simplement été recopié.

3 Structures conditionnelles

En informatique, on utilise une structure conditionnelle lorsqu'il est nécessaire de faire un choix.

Formes

Il existe deux formes principales :

Si « condition » est vraie **Alors**
    « séquence d'instructions 1 »
Sinon
    « séquence d'instructions 2 »
Exemple
Si le nombre se termine par 0 **Alors**
    dire le nombre est divisible par 10
Sinon
    dire le nombre n'est pas divisible par 10

Si « condition » est vraie **Alors**
    « séquence d'instructions »
Exemple
Si le nombre se termine par 0 **Alors**
    dire le nombre est divisible par 10

3.1 Conditions

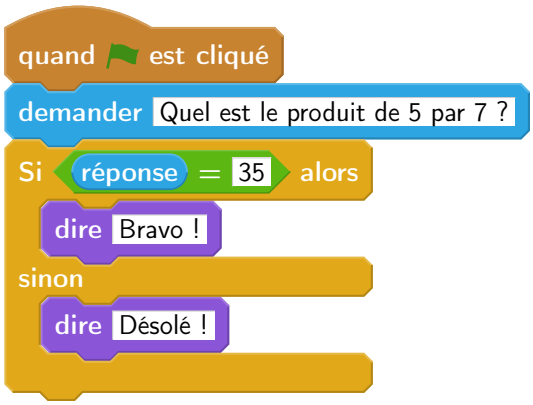
Pour faire fonctionner une structure conditionnelle, il faut évaluer si une condition est vraie ou fausse. Dans Scratch, les opérateurs de comparaison ont une forme hexagonale. On les nomme « Prédicats » .

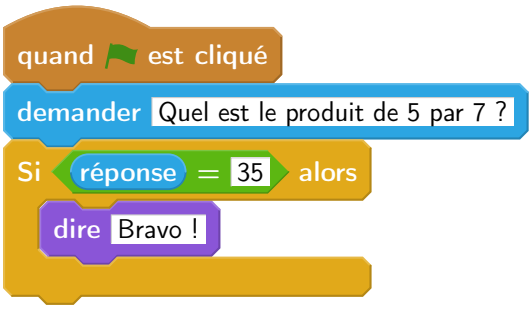
Menu Opérateurs	
tomate = patate	Teste l'égalité des deux valeurs,qui peuvent être des variables, et renvoie ici « faux ».
7 < 12	Teste si la première valeur est plus petite que la seconde valeur et renvoie ici « vrai ».
cote > longueur	Teste si la valeur de la variable « cote » est plus grande que la valeur de la variable « longueur ».
patate contient r ?	Teste la présence du caractère « r » dans la chaine de caractère « patate » et renvoie `faux`.
annee > 1969 et annee < 1980	Renvoie vrai si les deux conditions sont vraies en même temps, (pour 1972 dans annee, on a vrai)
annee > 1969 ou annee < 1980	envoie vrai si l'une des deux conditions est vraie, (pour 1900 dans annee, on a vrai)
non annee > 1969	Répond vrai si la condition est fausse, (pour 1972 dans annee, on a faux)
Menu Données	
menu contient poissons ?	Répond vrai si la liste « menu » contient la chaine de caractère « poissons » .

Menu Capteurs	
touche le pointeur de souris ?	Teste si le lutin touche le pointeur de la souris et répond vrai si c'est le cas.
couleur touchée ?	Teste si le lutin touche un élément de la couleur choisie et répond vrai si c'est le cas.
couleur touche ?	Teste si la partie colorée d'un lutin touche une autre partie colorée d'un autre objet et répond vrai si c'est le cas.
touche espace pressée ?	Teste si la touche du clavier sélectionnée est pressée et répond vrai si c'est le cas.
souris pressée ?	Teste si un bouton de la souris est pressé et répond vrai si c'est le cas.

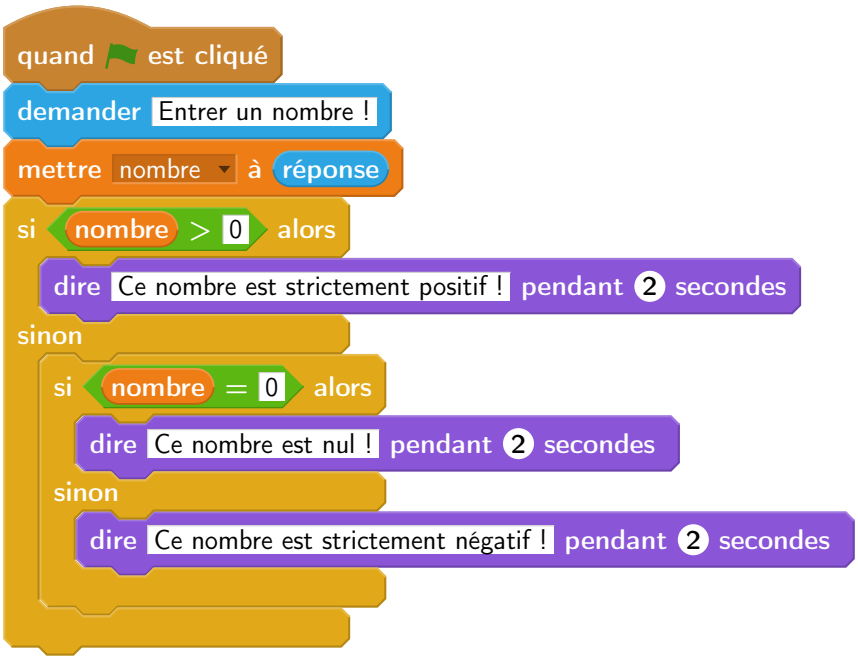
3.2 Structures conditionnelles

On trouvera les blocs dont il est question dans le menu « COntrôles » .
Exemple : Écrire un programme qui demande à l'utilisateur le produit de 5 par 7.





Exemple : Écrire un programme Scratch qui demande un nombre entier à l'utilisateur et qui fait dire au lutin si le nombre est strictement positif ou non.



4 Boucles

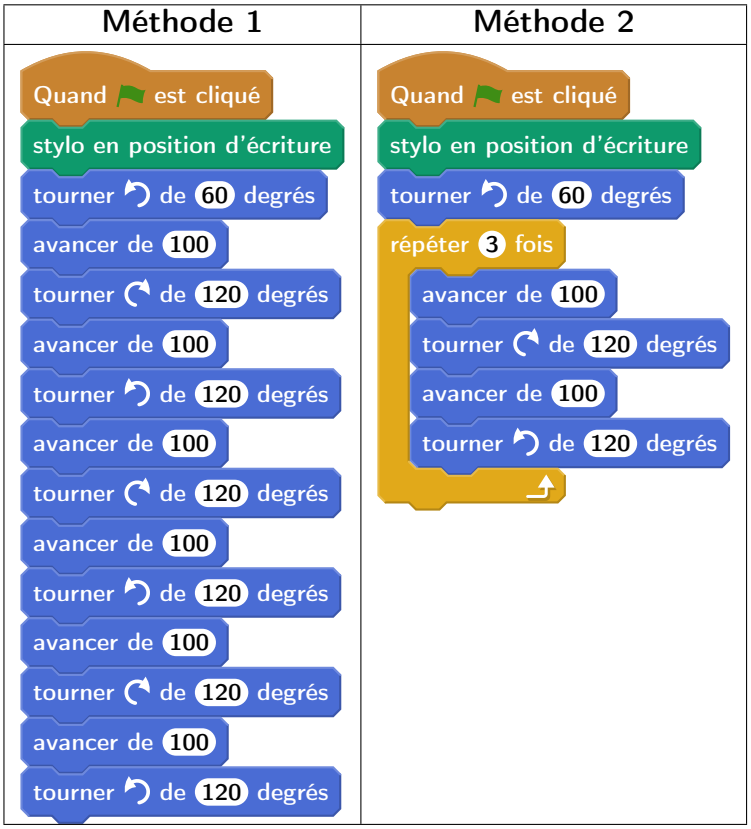
Une boucle est une structure de contrôle qui permet de répéter une instruction (ou une séquence d'instructions) plusieurs fois.
Dans Scratch, les blocs « Boucles » se trouvent dans l'onglet **Contrôle**.

4.1 Les boucles pour répéter des instructions plusieurs fois

Le bloc utilisé dans ce cas est : 

Toutes les instructions placées à l'intérieur de ce bloc seront répétées n fois où n peut être une variable.

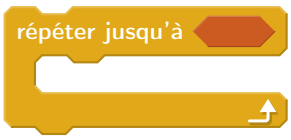
Exemple :
Pour réaliser la frise ci-dessous, on peut envisager la construction de deux façons :



Les blocs **4** à **7** se répètent trois fois et peuvent donc être insérés à l'intérieur du bloc 

4.2 Les boucles pour répéter une instruction jusqu'à ce qu'une condition d'arrêt soit vérifiée

Le bloc utilisé dans ce cas est :



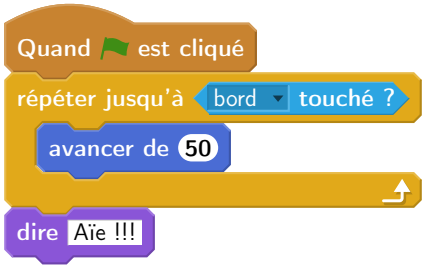
Toutes les instructions placées à l'intérieur de ce bloc seront répétées jusqu'à ce que la condition soit vérifiée.

Exemple :

Quelles que soient la position et l'orientation du lutin, si l'on répète l'instruction 

celui-ci touchera l'un des bords après un certain nombre de répétition. On ne sait pas, *a priori*, combien de fois il faut répéter l'instruction 

avant de toucher un bord.



4.3 En programmation évènementielle

Le bloc utilisé dans ce cas est :

Toutes les instructions placées à l'intérieur de ce bloc seront répétées lors de l'exécution du programme jusqu'à son arrêt par l'utilisateur via le bouton `Arrêt`.

Exemple :

Déplacer le lutin à l'aide des flèches directionnelles :

5 Listes

En informatique, une liste contient des informations rangées dans un certain ordre, ce qui permet de les retrouver plus facilement.

On trouvera les blocs dont il est question dans le menu « Données » .

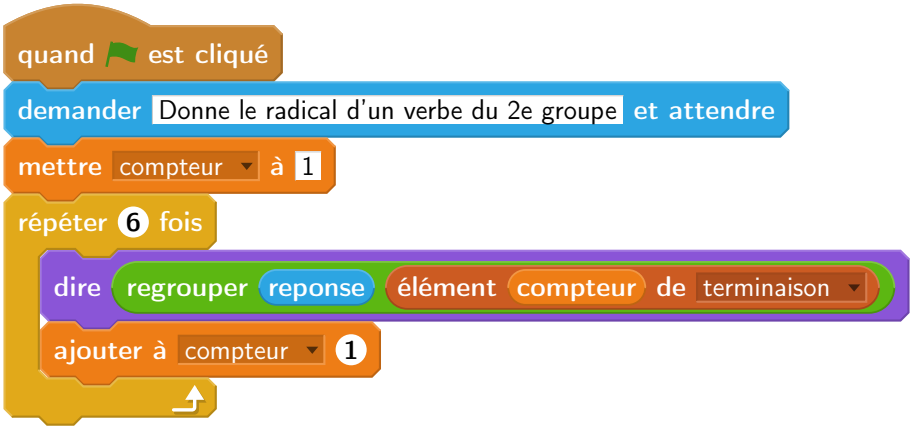
Dans Scratch, il faut créer une liste dans le menu associé avant de voir apparaître les instructions que l'on peut utiliser.

Bloc	Explications
	Ajoute la chaine de caractère « exemple » en dernière position de la liste.
	Insère en 4^e position de la liste « Somme » le résultat de « 2+5 ».
	Remplace la valeur en position 7 de la liste par la valeur « 04 73 01 02 03 ».
	Supprime l'élément en position « 2 » et renumérote la suite liste.
	Teste si le nombre 18 est un élémentde de la liste « nombres ».
	Renvoie la valeur en 4^e position de la liste « Telephone ».
	Renvoie la longueur de la liste « Telephone », c'est à dire le nombre d'élément dans cette liste.

Exemple :

On souhaite créer un petit programme qui conjugue les verbes du deuxième groupe au présent de l'indicatif.

On crée donc une liste des terminaisons, et une variable compteur pour la boucle. On utilise alors le programme suivant.



6 Évènements

6.1 Qu'est-ce qu'un évènement en programmation ?

Dans le cas de Scratch, on parle de programmation évènementielle, c'est à dire que le programme attend qu'il se passe quelque chose avant d'exécuter une séquence d'instructions programmée.

Les évènements les plus simples sont 

ou 

.

Dans Scratch, les évènements sont toujours placé en début de script, leur forme imposant ce fonctionnement.

6.2 Les différents type d'évènements dans Scratch.

6.2.1 Évènements déclenchés par l'utilisateur.

Intervention de l'utilisateur

Cette catégorie regroupe les déclencheurs qui imposent une action de l'utilisateur, qui peut être un clic sur le drapeau vert, ou sur un lutin précis, ou alors un appui sur une touche du clavier

Attention ! Lorsque l'on créé deux séquences différentes dans le même programme avec le même évènement déclenchant, on peut avoir des problèmes d'exécution, l'ordre dans lequel Scratch effectue les instructions n'étant pas toujours le même.

6.2.2 Évènements déclenchés par un programme. Messages.

Déclenchement interne

Scratch permet l'envoi de messages entre lutins. Il suffit d'insérer la commande 

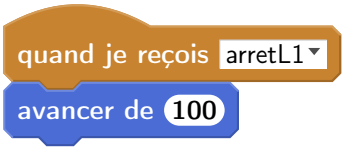
dans un script.

Tous les lutins que l'on veut faire réagir à l'envoi du message doivent avoir un script comportant l'évènement 

en première instruction.

Exemple :

Dans un jeu, on souhaite que le lutin 2 avance lorsque le lutin 1 a touché le bord de la scène. Voici un exemple de séquences d'instructions possibles pour cette situation.

Lutin 1	Lutin 2
	

Remarque : On peut créer de nouveaux messages et il faut penser à donner des noms explicites, comme pour l'exemple ci-dessus « ArrêtL1 » au lieu de « message 1 » pour faciliter la lecture.

7 Blocs

En programmation, il est utile de simplifier son code le plus possible, pour qu'il soit lisible et compréhensible par quelqu'un d'autre.
Plutôt que de réécrire des mêmes séquences d'instructions à plusieurs endroit d'un script, dans Scratch, on va utiliser les « blocs » pour faire ce travail.

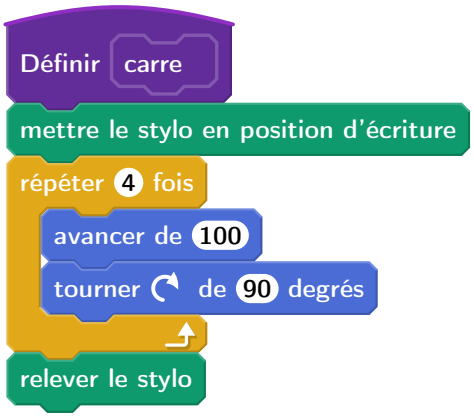
7.1 Qu'est-ce qu'un bloc ?

Définition

Un bloc est une partie de programme que l'on a nommée et mise à part du programme principal pour la mettre en évidence.

Dans Scratch, les blocs sont à définir dans le menu *Blocs*.
Ensuite, pour construire un nouveau bloc on utilise les blocs déjà existant comme dans un programme normal.

Le bloc ci-dessous permet de faire tracer un carré au lutin.

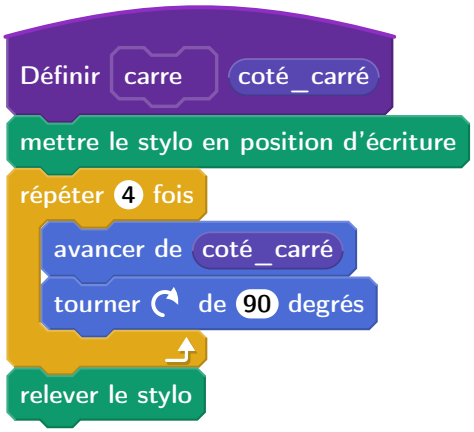


On peut l'utiliser dans le programme principal de la manière suivante.

7.2 Bloc simple, bloc à paramètre

Dans l'exemple précédent, le bloc est figé, et il est simplement exécuté, sans être modifié, 10 fois par le programme principal. On peut introduire un paramètre afin de faire un bloc qui ne fait pas la même chose selon la valeur donnée en entrée.

Le bloc ci-dessous permet de faire tracer au lutin un carré de côté `côté_carré`, qui représente la longueur donnée par l'utilisateur.



On peut l'utiliser dans le programme principal de la manière suivante.

Références

[1] Sylvie ALAYRANGUES et al. « Une analyse des exercices d'algorithmique et de programmation du brevet 2017. » In : *Repères IREM* Numéro 116 (2019). URL : https://hal.archives-ouvertes.fr/hal-02077738.

[2] Emmanuel BEFFARA, Malika MORE et Cécile PROUTEAU. *Algorithmique et programmation au cycle 4 : Commentaires et recommandations du groupe Informatique de la CII Lycée.* 2017. URL : http://www.univ-irem.fr/IMG/pdf/algoetprogaucycle4ciilycee-2.pdf.