Événements: communication entre lutins

Exercice 1

→ Ouvrez le fichier "Evenement_1.sb2"

Ce programme simule le dialogue entre deux lutins.

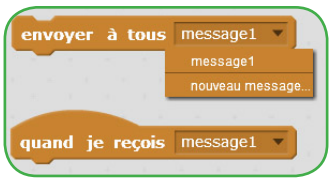
→ Après le lancement du programme, pour déclencher les différentes phrases, appuyez sur les touches 1,2,3 et 4 du pavé numérique (dans cet ordre, sans quoi le dialogue n'a pas de sens...)

Nous aimerions que ce ne soit plus l'utilisateur qui déclenche le dialogue, mais les lutins eux-même.
Pour cela, nous allons utiliser des **messages** et des **écouteurs**.

« Mais comment faire en sorte que les lutins se parlent ?»

Il existe dans le menu événement trois blocs de communication:

«Envoyer à tous...»:
Le lutin envoie un message à tout le monde. Ce message peut être un mot, ou une phrase que les autres lutins peuvent écouter.

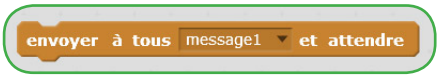


«Quand je reçois...»:
Ce bloc est un écouteur. Le lutin qui entend le message associé va déclencher la séquence d'instructions sous cet écouteur.

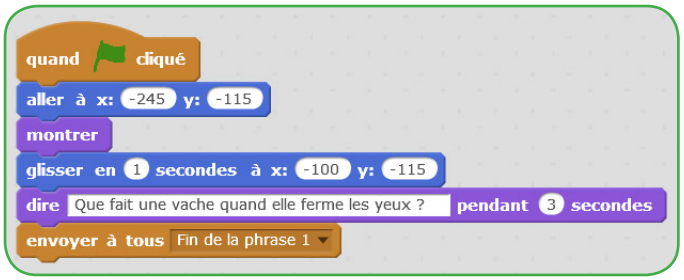
Par exemple: *Lutin 1* *Lutin 2:*

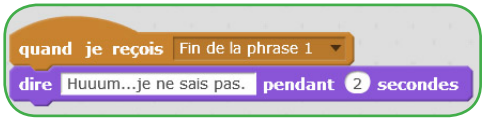
Lorsque le lutin 1 est cliqué, il envoie à tous le monde le message "Disparait !".
Le lutin 2 qui entend ce message lance alors l'instruction "cacher" et donc disparait.

«Envoyer à tous...et attendre»:
Le lutin envoie un message, et stoppe l'exécution de sa séquence tant que les lutins "écouteurs" n'ont pas terminé leur séquence déclenchée.



→ Modifiez ces scripts du lutin "Chat" et du lutin "Giga" comme ceci:





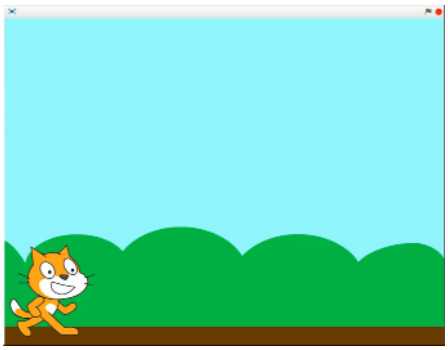
→ Modifiez d'une façon analogue les autres scripts pour que le dialogue s'enchaîne de lui même.

Exercice 2

→ Ouvrez le fichier “Evenement_2.sb2”

Ce programme comporte 4 lutins:
- le chat qui est visible
- le parallélogramme, le rectangle et le trapèze qui pour l’instant sont cachés.

Vous allez programmer une QCM (Question à Choix Multiple) comme sur la vidéo ci-dessous.



→ Programmez chaque lutin pour qu’au lancement du programme, il se montre et se place aux coordonnées suivantes:
Le chat en (0;-120) , le trapèze en (-160;60) , le parallélogramme en (0 ; 60) et le rectangle en (160 ; 60).

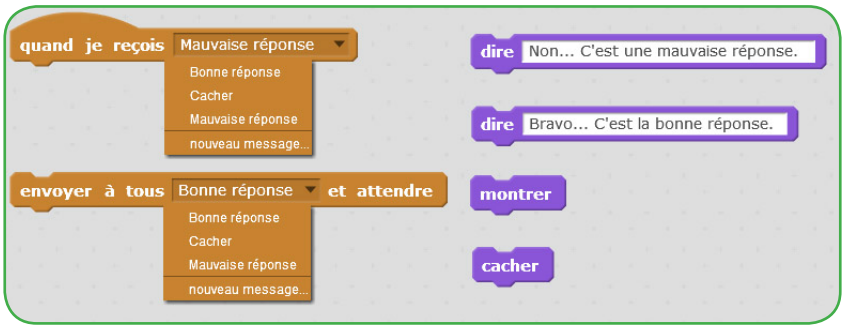
→ Faites dire au chat “Cliquez sur le quadrilatère qui n’est pas un parallélogramme”.

Nous allons programmer les quatre lutins pour que lorsque l’utilisateur clique sur l’un des quadrilatères, ils communiquent entre eux selon l’algorithme suivant:
- Les trois quadrilatères écoutent le message “Cache-toi”. Lorsqu’ils entendent cet événement, ils se cachent.
- Le chat écoute le message “Bonne réponse”. Lorsqu’il entend cet événement, il affiche un message de félicitations.
- Le chat écoute aussi le message “Mauvaise réponse”, pour lequel il affichera à l’inverse un message négatif.
- Quand l’utilisateur clique sur l’un des quadrilatères, il effectue la séquence suivante:
  - Il envoie à tous les lutins le message “Cache-toi” et attend que tous les lutins quadrilatères soient cachés.
  - Il se montre ensuite: c’est donc à présent le seul quadrilatère visible.
  - Il envoie à tous les lutins le message “Bonne réponse” si c’est le trapèze, ou le message “Mauvaise réponse” dans le cas inverse.

« Mais pourquoi envoyer les messages “Bonne/Mauvaise réponse” à tous les lutins alors que seul le chat les écoute ?»

Et bien... c’est comme cela !
Il est impossible avec Scratch de cibler à qui on envoie un message.
Le lutin qui ne l’écoute pas va tout simplement l’ignorer.
Ici, les lutins quadrilatères entendent aussi le message “Bonne réponse”, mais eux ne sont pas programmés pour y réagir...
De la même manière, le chat entend aussi le message “Cache-toi”, mais lui n’est pas programmé pour se cacher en conséquence.

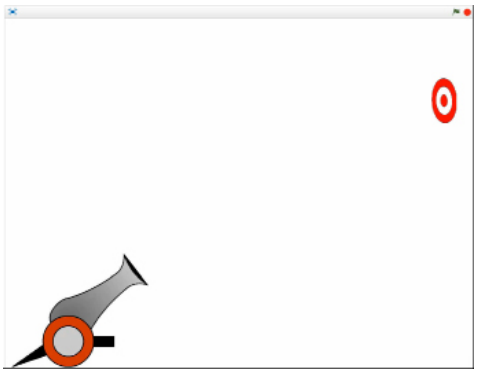
→ Utilisez les blocs ci-dessous pour compléter les scripts des lutins comme indiqué par l’algorithme proposé.



Exercice 3

→ Ouvrez le fichier “Evenement_3.sb2”

Vous allez programmer le jeu décrit dans la vidéo ci-dessous.

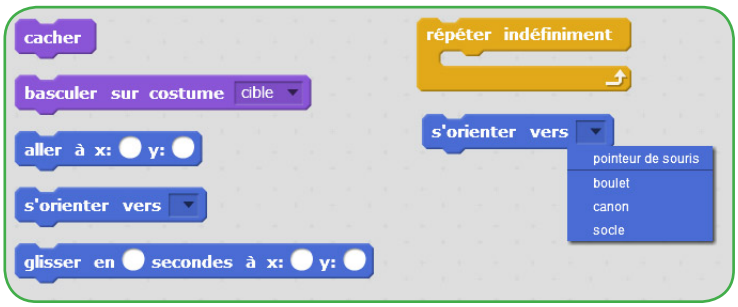


→ Programmez chaque lutin pour qu’au lancement du programme, il effectue les instructions suivantes:

-Le boulet se cache.

- La cible se place en (210;180), met son costume “cible”, puis répète indéfiniment un glissement en une seconde en (210;-180), suivi d’un glissement retour en une seconde en (210;180).
- Le canon se place en (-175; -130), puis il répète indéfiniment une orientation vers le pointeur de souris.
- Le socle se place en (-180;-150) et se place au premier plan pour cacher le canon.

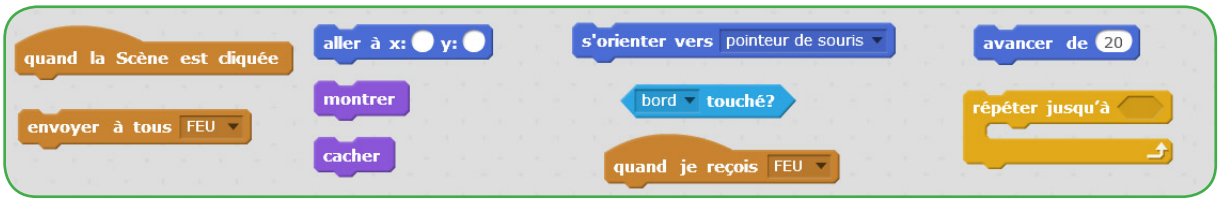
Pour cela, vous pouvez utiliser les blocs suivants:



→ Pour gérer le tir, nous allons utiliser l’algorithme suivant:

- Lorsqu’on clique sur l’arrière plan, celui-ci envoie le message “Feu”.
- Le boulet écoute ce message et lance alors cette séquence:
  - Il se place en (-175;-130) et se montre.
  - Il s’oriente vers le pointeur de la souris.
  - Il répète une avancée de 20 jusqu’à ce que les bords soient touchés.
  - Une fois le bord atteint, le boulet se cache.

En utilisant les blocs suivants, programmez l’arrière-plan et le boulet.



→ Il reste enfin à gérer la colision du boulet avec la cible. Pour cela, nous allons suivre l’algorithme suivant:

- Dans la boucle qui fait avancer le boulet, on teste si celui-ci touche la cible. Si c’est le cas, le boulet envoie le message “Touché”.
- La cible écoute ce message et lance alors la séquence suivante:
  - Elle change son costume en “Boom!”.
  - Elle stoppe tous les scripts, ce qui arrête le jeu.

En utilisant les blocs suivants, complétez les scripts du boulet et de la cible.