Scratch

Sommaire

Introduction

Avantages et inconvénients de scratch

- La programmation de base est intuitive. Scratch est un véritable langage de programmation.
- Sur le plan didactique, il pemet aussi de réaliser des applications (parfois complexes) très attractives pour les enfants.
- Scratch 3 dispose d'extensions : 'Mindstorm', 'Détection vidéo', 'Synthèse vocale', etc.
- Contrairement à d'autres langages de programmation, il est très 'permissif' et ne génère pas d'erreurs lors de l'exécution … d'où parfois des difficultés pour 'débugger' les scripts.
- L'éditeur de scratch 3 et l'espace de travail sont moins conviviaux et faciles d'utilisation que ceux de scratch 2.

Installation et description

Utilisation en ligne : Scratch 3.0

Configuration requise

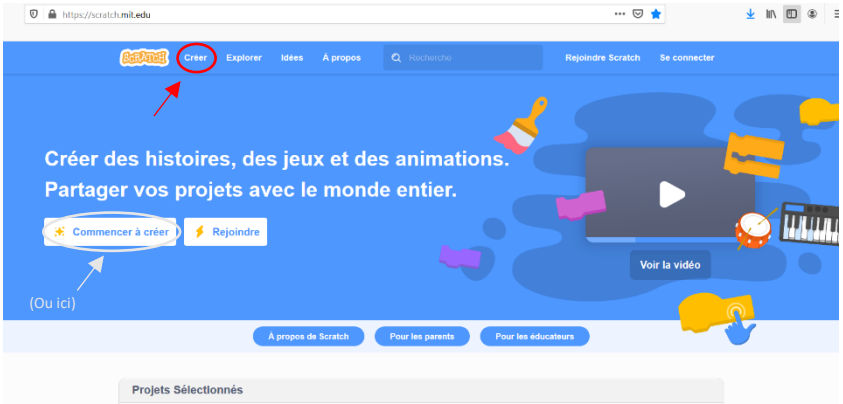
Un navigateur relativement récent est nécessaire pour exécuter Scratch 3.0 :

- Google Chrome : version 63 ou ultérieure
- Mozilla Firefox : version 57 ou ultérieure
- Edge : version 15 ou ultérieure
- Safari : version 11 ou ultérieure

Pour utiliser Scratch directement en ligne : https://scratch.mit.edu/

Sans créer de compte

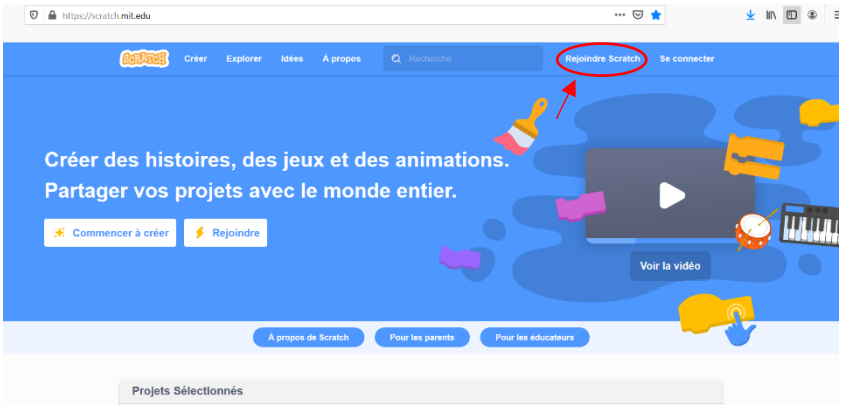
Cliquer sur « Créer » en haut à gauche pour accéder à l'interface.



Vous pouvez sauvegarder vos projets sur votre ordinateur (« Sauvegarder sur votre ordinateur » dans « Fichier » en haut à gauche).

Créer un compte

Cliquer sur « Rejoindre Scratch » en haut à droite et suivre les instructions.



Pour accéder à votre compte par la suite, il suffit de vous connecter depuis la page d'accueil (« Se connecter » en haut à droite).

Une fois connecté, vos créations sont automatiquement sauvegardées (vous les retrouvez dans la rubrique « Mes projets » de votre compte). Vous pouvez aussi sauver vos projets sur votre ordinateur, comme expliqué précédemment.

Utilisation hors ligne : Scratch Desktop

Configuration requise

L'installation de Scratch Desktop nécessite :

- Windows 7+ / MacOS 10.13+ (High Sierra ou ultérieur) / ChromeOS / Android 6.0+
- Environ 500 Mo d'espace disque

Pour utiliser Scratch sans connexion internet, il faut installer l'application Scratch Desktop.

Téléchargement

Télécharger gratuitement Scratch Desktop : https://scratch.mit.edu/download

Installation

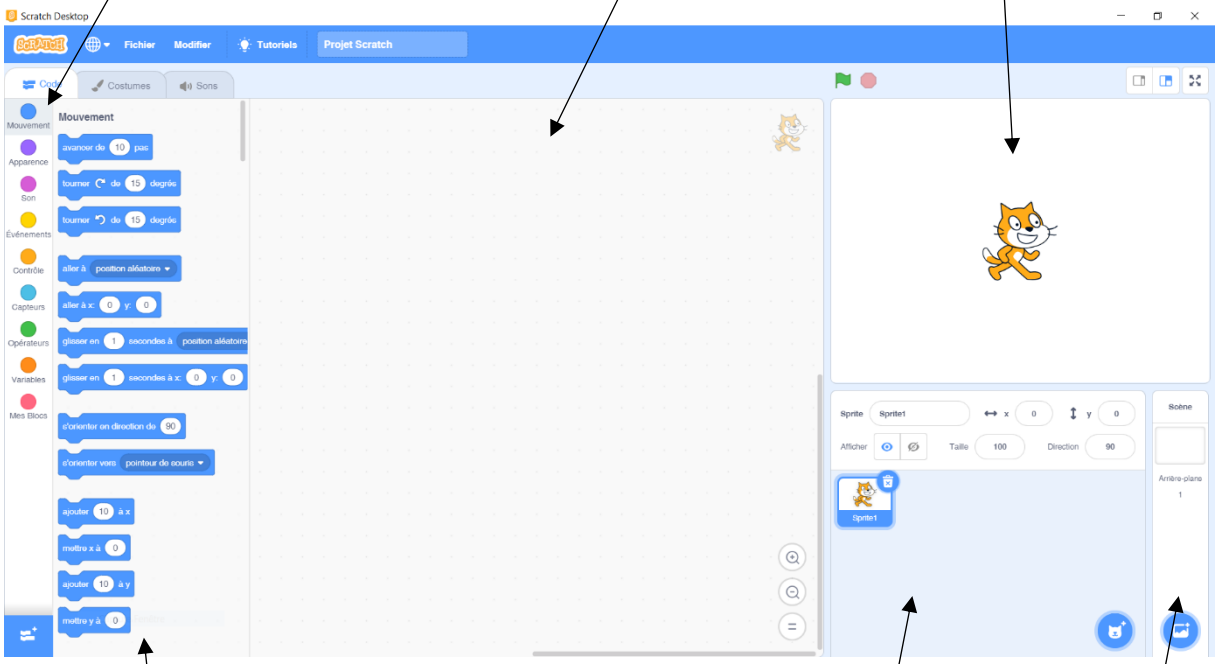
Une fois le téléchargement terminé, cliquez sur le fichier téléchargé pour l'exécuter.
Lors de la première ouverture, l'interface est par défaut en Anglais mais on peut facilement changer de langue en cliquant sur le « Globe » en haut à gauche. Par la suite, Scratch s'ouvrira dans la langue sélectionnée.

Description de l'interface

Dans ces MENUS se trouvent les instructions à glisser dans la zone de script.

C'est dans la ZONE DE SCRIPT que l'on assemble les instructions du programme.

La SCÈNE permet de visualiser l'exécution du programme.

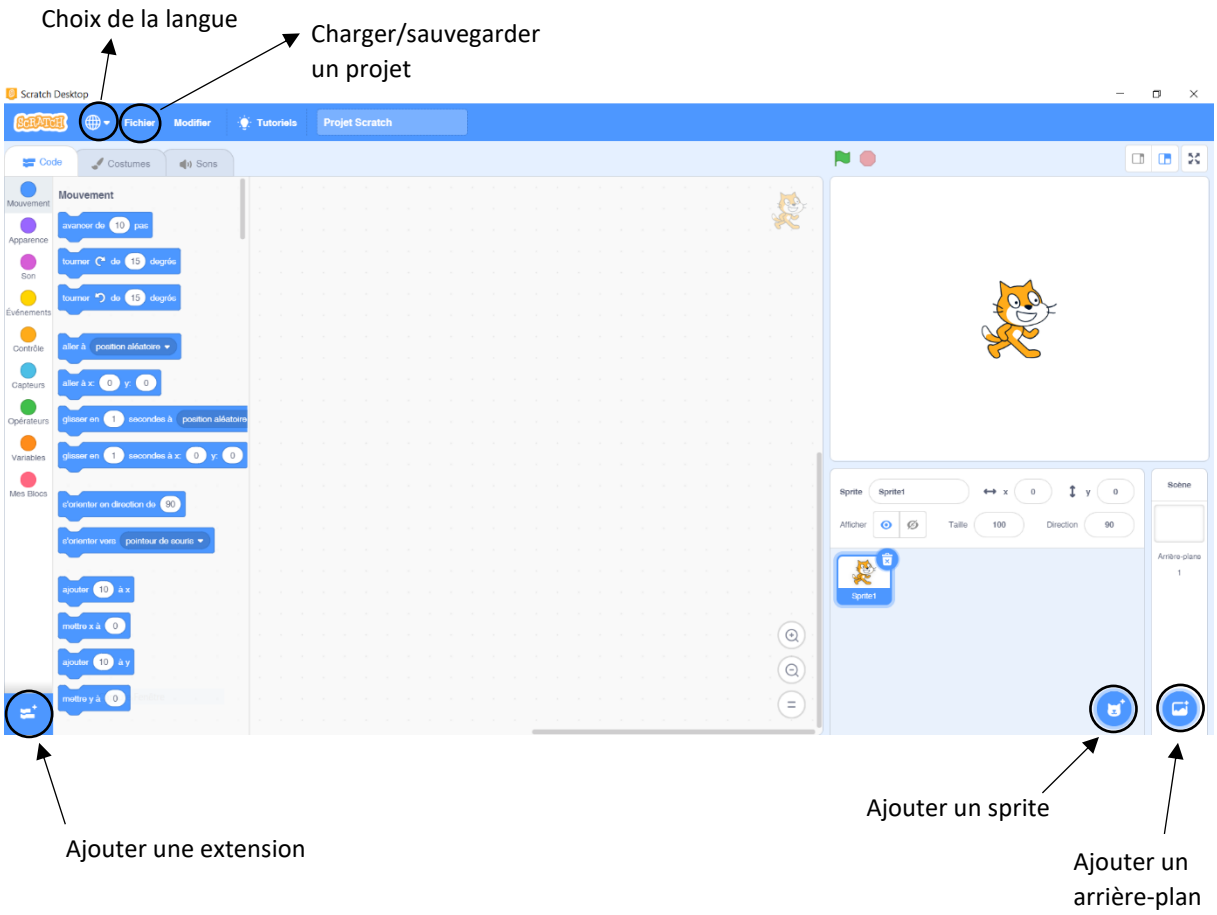


Chaque menu propose des INSTRUCTIONS à faire glisser dans la zone de script.

Zone de gestion des SPRITES.

Zone de gestion des ARRIÈRE-PLANS.

Fonctionnalités principales



Liens utiles

Description des blocs, aide concernant Scratch, etc. : https://fr.scratch-wiki.info/

Pour des exemples de script et tutoriels, voir : https://scratch.mit.edu/

Dans la suite, tous les exercices dirigés sont réalisés en scratch 3, sauf ceux de ClassCode qui sont en Scratch 2. Il n'est pas possible d'importer des fichiers scratch 3 dans scratch 2. L'inverse est possible.

Premiers pas

ClassCode, initiation en autonomie

https://project.inria.fr/classcode/initiation-a-scratch-en-autonomie/

Toutes les vidéos sont en scratch 2 !

Les indispensables :

- Action 2 : découverte de l'interface
- Action 3 – Prenez le contrôle du petit chat
- Action 6 – Changez de décor
  Astuce : avant de commencer, chargez les lutins 'Cat' et 'Hippo1' et orientez le lutin 'Hippo1' dans le bon sens
- Action 7 – Posez vos conditions
- Action 8 – Synchronisation
- Action 10 – Contrôler votre héros avec les flèches du clavier

ClassCode, les ressources à la carte & les parcours de formation

https://pixees.fr/classcode-v2/

Petits conseils de programmation

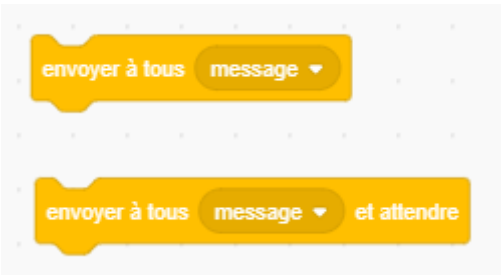
Initialisation des variables et démarrage des applications

Toujours démarrer l'application avec le drapeau vert en initialisant les variables que l'on utilise. Les valeurs des variables sont sauvegardées même lorsque l'on termine l'application avec le bouton rouge ! Il en est de même des listes. C'est potentiellement « piégeux ». Pour les listes, une initialisation simple permise par scratch 3 : « Supprimer tous les éléments de la liste ».

Lutin contrôle

Regrouper les scripts qui sont indépendants d'un lutin spécifique dans un lutin « contrôle » (commandes générales, etc.) qui peut rester invisible.

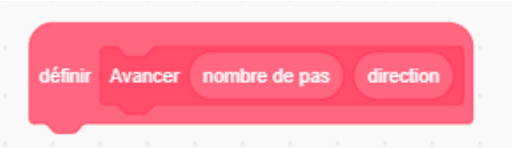
Envoi de messages



L'instruction « envoyer à tous …. et attendre » garantit que le script s'exécutera complètement. A privilégier, notamment, lorsqu'il s'agit de gros scripts (graphiques par exemple) afin de ne pas

s'arracher les cheveux en constatant qu'un script qui apparemment est 'nickel' ne s'exécute pas bien (s'interrompt parfois, parfois pas, de manière aléatoire en cours d'exécution).

Utilisation des blocs



Il est recommandé de fractionner les longs scripts en blocs d'instructions pour éviter les codes 'spaghetti'. Cela améliore la lisibilité et permet de raccourcir le code lors de la répétition des mêmes séquences d'instruction.

Ne pas oublier que les variables propres au bloc (en rouge) sont utilisables uniquement à l'intérieur du bloc et ne peuvent être modifiées (lecture seule).

Attention, les blocs sont uniquement valables pour le lutin dans lequel ils ont été créés. Il n'est pas possible de créer des blocs « globaux » pour l'ensemble des lutins. Un appel à un bloc créé dans un lutin à partir d'un autre lutin (par exemple après avoir copié une série d'instructions d'un lutin à un autre comportant un appel à un bloc) peut malheureusement parfois donner la fausse illusion que cela fonctionne !

Si l'on a besoin d'utiliser des blocs d'instructions identiques dans plusieurs lutins, on peut penser à faire un script dans le lutin contrôle et à l'activer par un message à partir des autres lutins. On peut aussi recopier la définition du bloc de lutin à lutin.

Commentaires dans les scripts

Il est recommandé généralement d'utiliser des commentaires pour s'y retrouver dans le code. Malheureusement les commentaires se comportent parfois un peu bizarrement en scratch 3 surtout lorsque les scripts sont longs et que l'on doit déplacer des portions importantes de ceux-ci. Une alternative : donner des noms explicites aux blocs et variables (scratch permet beaucoup de fantaisie dans les appellations contrairement à d'autres langages de programmation).

Exercices guidés

- Quelques effets graphiques
- Les clones (génération et gestion de clones)
- Les fractions
- Tuez le virus ! (jeu)
- Le labyrinthe (circulation autonome dans un labyrinthe comportant des obstacles)

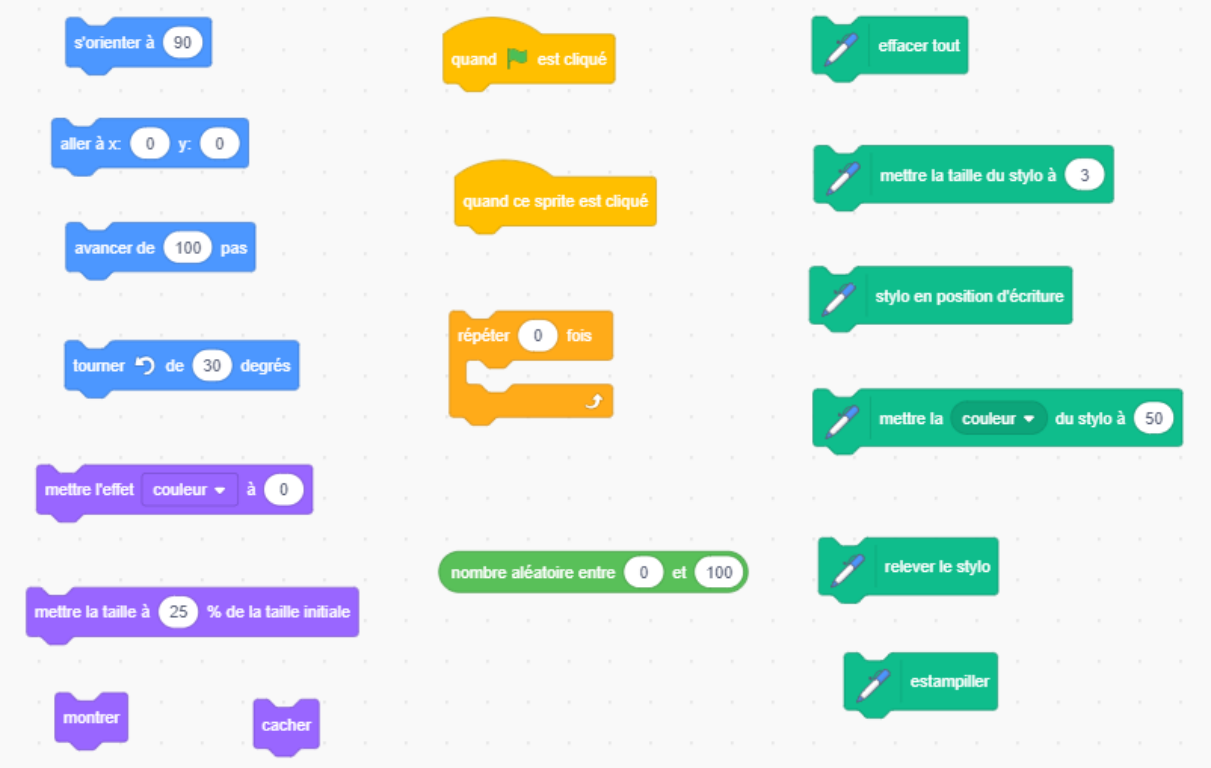
 Annexes :

- Les fichiers nécessaires pour les exercices dans le sous-répertoire 'lutins à importer'
- Chaque sous-épertoire contient lui-même 2 sous-répertoires : 'corrigé' et 'lutins à importer'. Le sous-répertoire corrigé contient l'application complète ainsi que tous les lutins individuels utiles avec leur code. Il est recommandé de ne pas les utiliser avant d'avoir essayé de réaliser l'exercice. Pour les exercices, seuls les fichiers dans le sous-répertoire 'lutins à importer' sont nécessaires.

 Pour les exercices, il faut importer des lutins « standards » (ceux disponibles à partir de scratch desktop) mais aussi des lutins « non standards » qui se trouvent dans le répertoire 'lutins à importer' et se finissent par l'extension .sprite3

EXERCICE 1 : QUELQUES EFFETS GRAPHIQUES

Blocs et instructions utiles



1. Importer les lutins :

   - pencil (standard) : position -191, -127, taille 40
   - apple (standard) : position 179, -127, taille 85
   - gomme (non standard) : position -1, -152, taille 40
   - Abby (standard) : caché

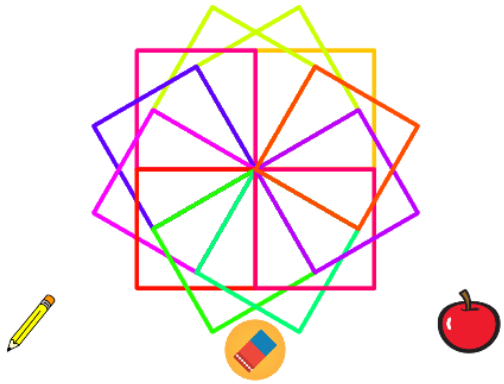
Prendre l'habitude de positionner les lutins sur la scène au moyen des scripts.

2.  Lutin ‘Abby’

C’est le lutin ‘contrôle’.

 Il efface tous les éléments graphique de la scène et se cache. Il est donc invisible.

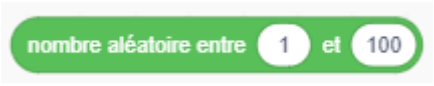
3.  Lutin ‘pencil’



Lorsque l’on clique sur ce lutin :

- Le crayon se dirige au centre de la scène avec une taille de 25
- Il dessine à partir du centre et vers la gauche 12 carrés décalés d’un angle de 30°. Caractéristiques des carrés : côté de 100 pas, couleur aléatoire (entre 0 et 100 – palette de couleur scratch), taille du stylo 3
- Le premier carré est orienté dans la direction 90°
- Une fois la figure terminée, le lutin revient à sa position et à sa taille initiale

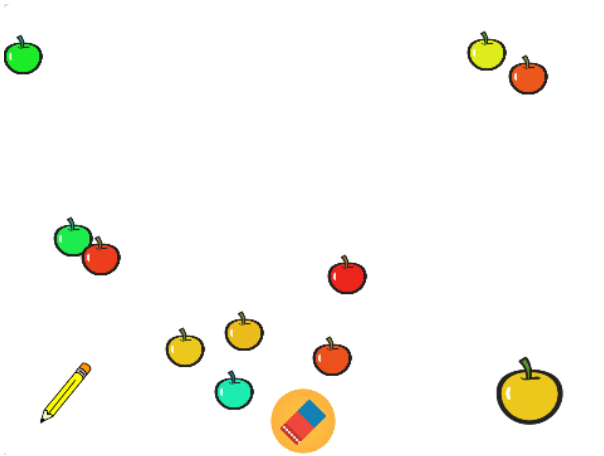
 Pour dessiner les carrés, pensez à imbriquer deux boucles : une qui dessine un carré et une qui effectue 12 fois une rotation de 30°

Pour générer un nombre aléatoire (entier) 

4.  Lutin ‘gomme’

   Efface tout dessin sur la scène

5.  Lutin ‘apple’



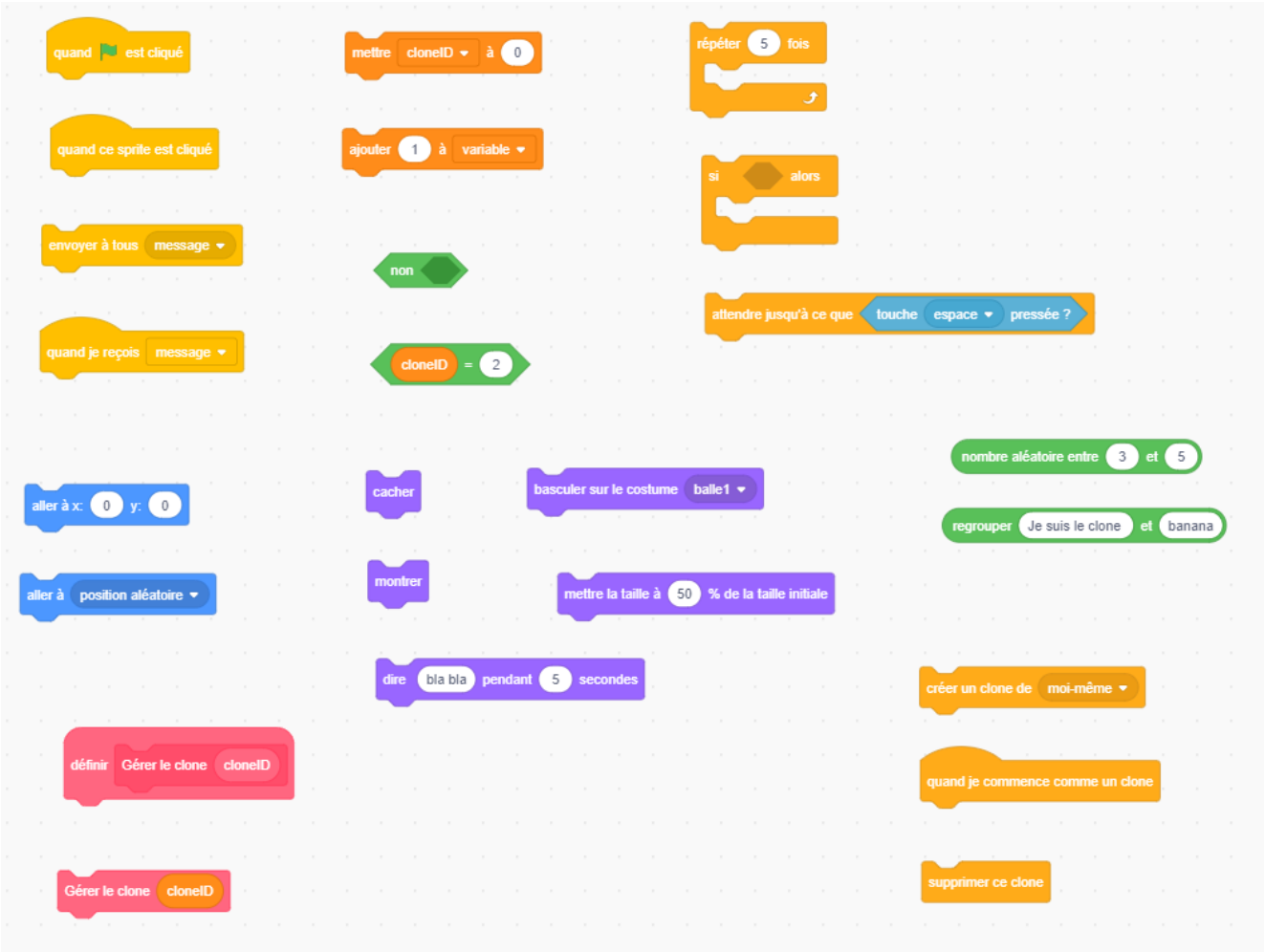
Lorsque l’on clique sur ce lutin :

- Il génère 10 répliques de lui-même par estampillage (taille 50), position et couleur aléatoire (couleur entre 0 et 100 – palette de couleur scratch)
- Revient à sa postion initiale

Pour l’estampllage, il faut utiliser si possible des images vectorielles natives (.svg) sous peine d’une pixellisation importante. Le fait d’importer une image .jpg ou .png et de la convertir en image vectrorielle dans l’éditeur scratch n’améliore rien à la qualité !

EXERCICE 2 : LES CLONES

Blocs et instructions utiles



1. Importer le lutin ‘Ball’ (standard)
2.  Positionner le lutin en 195,-142 taille 100 couleur bleue

3. Lorsque le lutin est cliqué :

   - Générer 5 clones qui prennent une couleur aléatoire (rouge, vert ou violet) et se positionnent aléatoirement sur la scène (taille 50).
   - Les clones se présentent pendant 5 secondes : « Je suis le clone x ».
   - Le message « aller au centre et changer de couleur » est généré à destination du clone 2. Celui-ci réagit à ce message en disant « Je suis le clone 2 et je vais me déplacer au centre en changeant de couleur ». Il se déplace ensuite au centre de la scène en changeant sa couleur en jaune.

4. Lorsque les clones sont générés, rendre le lutin inactif

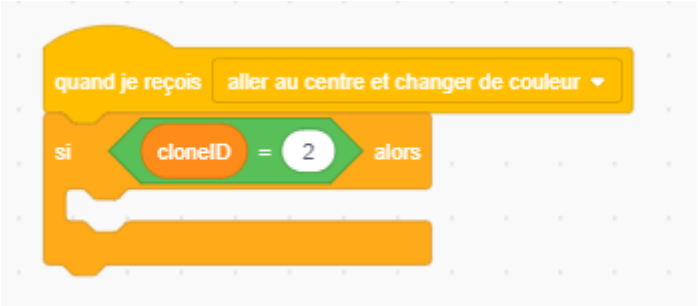
 Astuce : prévoir une variable d’état pour rendre le lutin inactif lorsqu’il est cliqué après la génération des clones.

5. Lorsque la barre d’espace est pressée, tous les clones disparaissent sauf le lutin parent.

 Astuce : renommer les costumes des lutins pour pouvoir accéder plus facilement à leur nom (balle1, balle2, etc.)

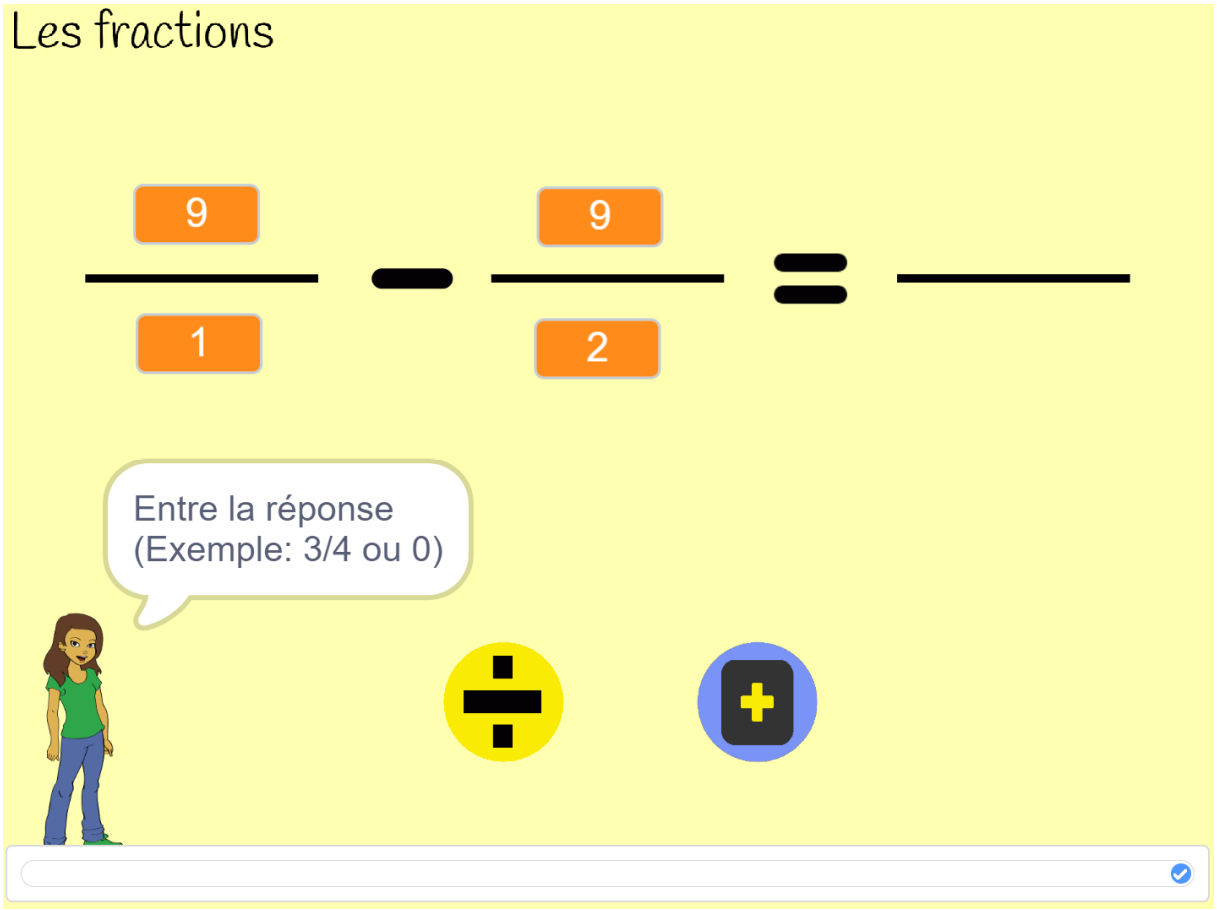
 Principe de la gestion des clones : Lorsqu’un lutin est cloné, le clone hérite de copies des attributs de son lutin parent, y compris ses variables. Une propriété héritée démarre identique à la propriété du parent au moment où le clone est créé. Mais après cela, si les attributs et les variables du clone changent, cela n’affectera pas ceux du parent. De même, les changements effectués sur le lutin parent n’affectent pas les clones.

Il faut penser à créer une variable locale cloneID et à affecter un numéro d’identification à chaque clone avant sa création. Une fois les clones créés, ils peuvent réagir différemment, notamment aux messages reçus.

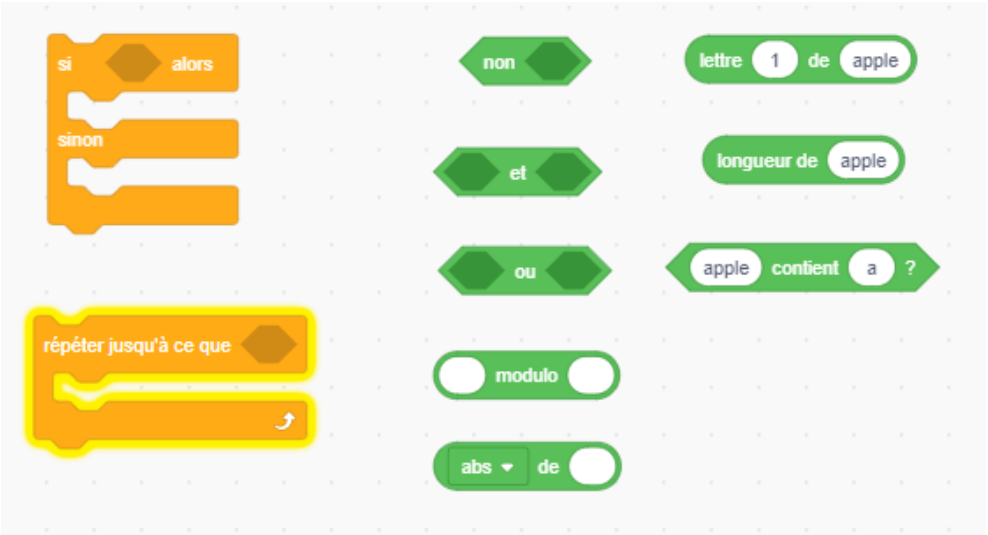


EXERCICE 3 : PROFESSEUR DE FRACTIONS

Apprendre les fractions avec Abby



Blocs et instructions utiles non encore vus jusqu'à présent



1. Importer les lutins :

   - 'Abby' (standard)
   - 'opérations' (non standard)
   - 'ligne horizontale' (non standard)
   - 'égalité' (non standard)
   - 'bouton plus et moins' (non standard)
   - 'bouton fractions' (non standard)

2. Importer l'arrière-plan 'Blue sky', changer sa couleur en jaune pâle et renommer le costume en 'jaune' - Mettre le titre « Les fractions » dans le coin supérieur gauche
   Dupliquer 3 fois l'arrière-plan
   Nommer les costumes 'vert', 'orange' et 'rouge' et changer la couleur du fond en la couleur correspondante

3.  Effacer tout dessin et positionner :
   - 'Abby' en -207, -112, taille 50
   - 'bouton fractions' en -41,-97, taille 25
   - 'bouton plus et moins' en 59,-97, taille 25

4.  Lutin 'bascule plus et moins'

   - Bouton bascule + ou +/- (uniquement des nombres positifs / nombres positifs et négatifs pour les opérandes des fractions)
   - Par défaut, uniquement des nombres positifs – Initialisation d'une variable 'nombres positifs et négatifs' à 0

5.  Lutin 'nouvelle fraction'

   - Génère un événement 'Nouvelles fractions' quand il est pressé. Celui-ci est exécuté dans le lutin 'Abby'.
   - Le script :

     ✓ Affiche les barres de séparation des numérateurs et dénominateurs des fractions à résoudre et du résultat en clonant 3 fois le lutin 'ligne horizontale' (en -160,70 / 0,70 / 160,70 Taille 15)
     ✓ Génère aléatoirement une opération à appliquer aux fractions et l'afficher en -77,70 taille 200
     ✓ Affiche le signe d'égalité avant le résultat en 80,70 taille 180
     ✓ Initialise aléatoirement les opérandes des deux fractions à résoudre (soit uniquement des nombres positifs soit des nombres positifs et négatifs). Chaque opérande est limitée à un chiffre. Les nombres des dénominateurs ne peuvent être des 0.
     ✓ Affiche les opérandes des deux fractions à résoudre (affichage des variables)


Lutin 'Abby'

- Se positionne en -207,- 112 taille 50
- Il s'agit du lutin 'contrôle' qui :

✓  précise qu'il faut appuyer sur le bouton jaune pour obtenir un nouvel exercice

✓ demande à l'élève d'introduire une réponse sous une forme fractionnaire (Ex : 1/4 ou 0)

✓ sépare la réponse en 1 numérateur et 1 dénominateur.

 Dans Scratch, une variable contenant un nombre peut être traitée comme une chaîne de caractère.

 Pour séparer les 2 opérandes dans des variables NumRep (numérateur de la réponse) et DenRep (dénominateur de la réponse) :

- Si la réponse est 0 ce n'est pas la peine de séparer les opérandes car le résultat est nul. Mettre 0 dans NumRep et DenRep
- Le script examine chaque caractère de la réponse, détecte la barre de fraction qui sépare numérateur et dénominateur et recopie chaque caractère dans une variable NumRep (numérateur de la réponse) ou DenRep (dénominateur de la réponse)

 Si une erreur de forme est commise dans la réponse (exemple : 10/2a), le résultat est considéré comme fautif.

✓ résout le problème pour obtenir 'la bonne réponse'

Le calcul s'exécute dans le lutin 'opérations '

✓ vérifie que la réponse de l'élève est exacte et lui donne un feed-back sous une des formes suivantes :

- « Super ! La réponse est correcte ! » - costume abby-c – arrière-plan 'vert'

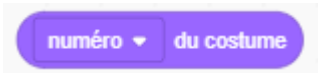
- « La réponse 'X' est correcte mais peut être simplifiée. Essaye à nouveau ! » - costume abby-a – arrière-plan 'orange'
- La réponse 'X' n'est pas correcte. Essaye à nouveau ! » - costume abby-b - arrière-plan 'rouge'

✓ Si la réponse est correcte, il affiche le résultat dans la fraction 'résultat' (montrer les variables résultat). Si la réponse est 0, il ne montre que le numérateur.

La réponse de l'élève est correcte si les opérandes de la réponse de l'élève et du résultat correct sont égales OU si le numérateur de la réponse de l'élève et celui du résultat correct sont 0. La réponse de l'élève est correcte mais peut être simplifiée si « Numérateur de la réponse x Dénominateur exact = Dénominateur de la réponse x Numérateur exact »

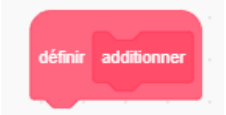
6.  Lutin 'opérations'

- Se positionne en -77,70 taille 200 - costume en fonction du type d'opération (est généré aléatoirement pour chaque exercice).

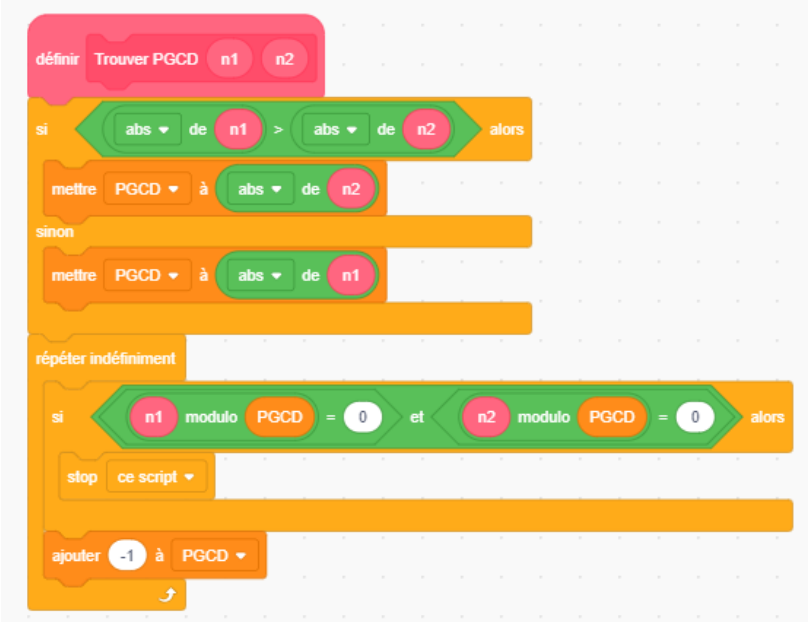
peut être utile

- Calcule le résultat de l'opération sur les 2 fractions dans les variables 'numérateur' et'dénominateur'.

- Calcule le PGCD

  Pour ne pas trop perdre de temps voici le script



- Simplifie le résultat en divisant le numérateur et le dénominateur par le PGCD

EXERCICE 4 : TUEZ LE VIRUS !

Un infirmier en scaphandre de haute protection doit tuer des virus en leur envoyant un vaccin. Des arbres se promenant sur la scène viennent lui compliquer la tâche. Si un virus ou un arbre touche l'infirmier, il perd une vie (10 vies au départ). S'il ne lui reste aucune vie ou s'il n'a pas tué au moins 5 virus avant que 25 virus n'aient défilé, il a perdu.

1. Mise en place du cadre général du jeu

   Importer les lutins et l'arrière-plan

   - 'pico' (lutin standard 'pico')
   - 'infirmier'
   - 'arbre'
   - 'vaccin'
   - 'virus'
   - 'terminé'
   - arrière-plan standard 'Blue Sky'

2.  Tous les lutins doivent être invisibles après lancement de l'application avec le drapeau vert sauf le lutin 'contrôle' (costume pico-a) qui demande de presser la touche 'x' et attend avant la poursuite du programme. Le jeu proprement dit et les principaux scripts des lutins démarrent après que la touche 'x' ait été pressée (voir infra). Le lutin 'contrôle' disparaît alors et laisse la place aux lutins du jeu.

   NB : La taille des lutins est à ajuster

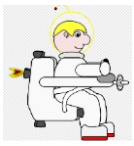
   Des variables globales sont initialisées lors du lancement de l'application :

   - score (0)
   - virus (25)
   - vies (10)
   - partie terminée (0)

   Les variables 'score' et 'vies' doivent être visibles sur la scène

   Après lancement de l'application avec le drapeau vert, la scène se présente comme ceci.

Après avoir pressé ‘x’

3.  Lutin ‘infirmier’ (Taille 16, direction 0)

À partir du moment où la touche ‘x’ a été enfoncée, le lutin apparaît et peut se déplacer dans les 4 directions (haut, bas, gauche, droite) au moyen des flèches de direction.

Le lutin :

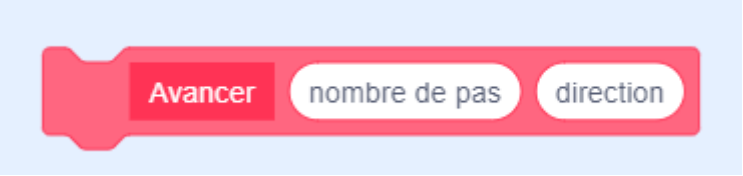
- Démarre à une position de départ fixe
- Rebondit lorsqu’un des bords de la scène est atteint
- Avance par déplacement de 10 pas

- Lorsqu'il est tourné vers la gauche, reste dans cette direction après être descendu ou monté
- Dit 'Aïe !' pendant 0.5 secondes lorsqu'il est touché par un arbre ou un virus
- Revient à la position de départ s'il touche un virus ou un arbre

Le script :

- s'arrête si la partie est terminée
- met à jour la variable 'vies'

Pour la lisibilité, il est recommandé de créer un bloc qui a pour but de faire avancer le lutin d'un certain nombre de pas dans une des 4 directions (haut, bas, gauche, droite) au moyen des flèches de direction



4.  Lutin 'arbre' (Taille 40, direction -90)

A partir du moment où la touche 'x' a été enfoncée, le lutin continuellement :

- Démarre du bord droit de la scène
- Avance 6 à 12 pas (aléatoire)
- Se déplace jusqu'au bord gauche et disparaît s'il n'a pas touché l'infirmier
- Disparaît s'il a touché l'infirmier
- Réapparaît à la coordonnée de départ

Le script s'arrête si la partie est terminée (Cf. variable 'partie terminée')

5.  Lutin 'virus' (Taille 35)

Ce script est semblable à celui du lutin 'arbre'

A partir du moment où la touche 'x' a été enfoncée, le lutin, continuellement :

- Démarre du bord droit de la scène mais à une hauteur aléatoire
- Avance de 6 à 12 pas (aléatoire)
- Se déplace jusqu'au bord gauche de la scène
- Disparaît s'il a touché l'infirmier, s'il a atteint le bord gauche ou s'il est atteint par le vaccin.
  Le lutin 'virus' « apprend » qu'il a été touché par le vaccin par l'intermédiaire d'un message « Touché par le vaccin »

Le script :

- Met à jour la variable virus (nombre de virus restant)
- S'arrête si la partie est terminée

6.  Lutin 'vaccin' :

Le vaccin est symbolisé par un petit rectangle rouge.

Le vaccin :

- Est disponible à partir du moment où la touche 'x' a été enfoncée
- Est lancé à partir de l'extrémité pointue de ce qui ressemble au bâton de ski du lutin 'infirmier' lorsque la barre d'espacement est pressée. 1 nouveau vaccin ne peut être lancé que 0.5 secondes après le précédent.
- Se dirige vers la droite ou vers la gauche en fonction de la position de l'infirmier
- Disparaît lorsqu'il touche un virus ou atteint le bord de la scène

Un message « Touché par le vaccin » est envoyé à tous si le virus est touché

Suggestions :

- Penser à déterminer la direction actuelle suivie par l'infirmier au moment du tir (revenir éventuellement au lutin infirmier)
- Pour contrôler le vaccin : on commencer par créer un clone puis on le gère par l'instruction « Quand je commence comme un clone ». On finit par le supprimer quand il n'est plus nécessaire.

7. 

Lutin 'terminé' :

Le script :

- Est actif à partir du moment où la touche 'x' a été enfoncée
- Quand le jouer n'a plus de vies ou que le stock de virus est à 0, le jeu se termine par la **proclamation du résultat par le lutin « contrôle ».** Si le joueur a gagné, celui-ci prend le costume « pico-b » et dit « Gagné – Score : x ». S'il a perdu, il prend le costume « pico-d » et dit simplement « Perdu ! ». Le lutin 'terminé' s'affiche finalement au milieu de la scène