Utilisation de scratch.

I. POUR COMMENCER

Le programme officiel propose de commencer par des labyrinthes…
Scratch n'est pas le seul programme à travailler de la même façon. Il peut être intéressant de commencer par là avec des élèves :
https://studio.code.org/s/starwarsblocks/stage/1/puzzle/1
(Issu du site : https://hourofcode.com/fr/learn)
Vous pourrez explorer ce site pour trouver des activités avec starwars, minecraft …

Les parties suivantes sont issues du document officiel eduscol disponible ici :
https://cache.media.eduscol.education.fr/file/Algorithmique_et_programmation/67/9/RA16_C4_MATH_algorithmique_et_programmation_N.D_551679.pdf

A. *Parcours d'un labyrinthe avec scratch*

Cet exemple de première séance a comme objectif principal la prise en main par les élèves du logiciel.
Il s'agit :

- de commencer à utiliser le vocabulaire de Scratch pour savoir nommer les objets de l'interface ;
- de comprendre le fonctionnement des blocs, leur « aimantation », leur duplication par un clic droit dans la fenêtre de script ;
- de savoir exécuter et mettre au point un script, de déclencher son exécution par un clic sur le drapeau vert ;
- d'enregistrer son programme.

On ne cherche surtout pas ici à découvrir l'intégralité des commandes disponibles !
On peut par exemple proposer d'ouvrir un fichier préparé par le professeur, avec un arrière plan déjà dessiné, un bloc utilisateur « départ », un script détectant le succès.
L'arrière-plan peut être celui-ci:



Le bloc « départ » permet de positionner le « chat » à sa bonne place et à la bonne taille.
Le défi proposé aux élèves est de combiner des déplacements (« avancer de » et « tourner ») pour amener le chat à la cible jaune.
On peut présenter dans la fenêtre de script le squelette du script visé : déclenchement par le drapeau vert, utilisation du bloc départ, et les trois blocs aimantés qu'il faut dupliquer et « attacher » pour obtenir le script final.



En ayant déjà placé un bloc « avancer de 100 », l'élève peut tester que le chat arrive au bout du premier trajet horizontal, ce qui lui donne une première idée de l'échelle en pixels. Il est d'ailleurs possible de proposer à l'élève, pour l'aider, d'afficher les coordonnées du chat en cochant les cases devant et de la catégorie « Mouvement ». Le professeur a préparé ces éléments de script, qu'il n'est pas nécessaire de présenter aux élèves lors d'une première séance : Aux élèves qui réussiraient très vite à terminer l'exercice, on peut expliquer le fonctionnement de ces deux blocs. Une autre idée consiste à simplifier leur script à l'aide d'un bloc de répétition : Cette façon de procéder, donner un même objectif à tous et proposer selon l'avancement de chacun des compléments de recherche, peut être utilisée régulièrement, tout au long du cycle.

II. QUELQUES EXEMPLES D'ACTIVITES

Ces exemples d'activités ne prétendent pas recouvrir exhaustivement le programme. Ils ne sont en rien prescriptifs, mais fournissent simplement des pistes de travail qui pourront inspirer les professeurs.

A. Un jeu à trois personnages

Lien vers l'activité : PoissonsChatChiens.sb2

Cette activité se prête facilement à un découpage, en ajoutant au fur et à mesure un personnage et des interactions, ce qui permet une pédagogie différenciée. Elle permet de revoir la gestion des déplacements et introduit l'utilisation conjointe de plusieurs personnages, ainsi que leur interaction. On rencontre également naturellement la notion de variable, et l'utilisation du chronomètre intégré à Scratch. On peut commencer avec le seul lutin habituel (le chat). On demande d'abord aux élèves de gérer son déplacement à l'aide des quatre flèches du clavier, ce qui relève de la programmation événementielle. On ajoute un nouveau lutin, un chien, qui doit effecteur indéfiniment des allers-retours du bord gauche au bord droit de l'écran, ce qui est facilement réalisé à l'aide d'une répétition infinie et des blocs et. Un défi supplémentaire, pour les élèves les plus à l'aise, consiste à demander de programmer un parcours aléatoire du chien. Pour l'intérêt du jeu, il convient de prévoir un déplacement pas trop rapide. On souhaite alors que chaque lutin réagisse au moment où il rencontre l'autre : le chien pourra simplement dire bonjour et le chat exprimer sa frayeur. La « rencontre » est détectée par le capteur et il faudra la tester au sein d'une répétition infinie. Il est probable que la nécessité de cette boucle n'apparaisse pas immédiatement aux élèves et que certains se contentent d'effectuer le test une seule fois. Il peut être intéressant de comparer la gestion directe d'un événement comme l'appui d'une touche avec celle du contact détecté par la valeur d'un capteur. La distinction entre les scripts de chaque lutin est également un élément qui mérite d'être souligné.

On peut d'ores et déjà ajouter une variable globale qui permet de gérer un score, chaque rencontre du chat et du chien faisant baisser le score. On propose alors d'ajouter un troisième personnage, un poisson. Il s'agit de programmer un déplacement aléatoire dans le plan de ce lutin. Une fois cette étape franchie, on gère l'interaction chat-poisson : à chaque contact, on incrémente le score, et on souhaite faire disparaître le poisson (le chat l'a mangé) et le faire réapparaître à un endroit aléatoire après un délai fixé. Il s'agit donc de simuler le fait que le chat mange les poissons qu'il réussit à attraper et qu'à chaque fois un nouveau poisson apparaît, même si c'est un lutin unique qui gère les poissons. La difficulté tient au délai fixé. Cela permet d'introduire le chronomètre de Scratch, mais également une variable qui conservera l'heure où le poisson a été attrapé : le script du poisson étant le seul qui ait besoin de cette variable, on la Définira comme variable privée du lutin, à la différence du score que plusieurs lutins ont besoin de modifier ou de consulter, qui doit[1] donc être une variable publique (ou globale). Cette distinction est une première introduction aux concepts de la programmation-objet qui n'a évidemment pas à être formalisée. La commande qui permet d'attendre le délai fié (ici à 2 secondes) ne va pas de soi et mérite sans doute un moment collectif de réflexion et d'explication : Une dernière étape consiste à ajouter un quatrième lutin, l'arbitre, qui apparaît quand le score devient négatif (après des rencontres du chat et du chien) ou dépasse une valeur fiée (le chat a réussi à manger suffisamment de poissons). Là encore, on met en évidence la nécessité d'une boucle infinie pour scruter la valeur d'une variable globale, le score, et il peut être

[1] Le fichier joint PremiersDessins.sb2 montre un exemple d'activité. L'utilisation d'un message a permis de regrouper les tracés de deux figures différentes dans le même script : ce n'est évidemment pas recommandé pour des premières séances. Le lecteur est invité à ouvrir et expérimenter à partir du document joint PoissonsChatChiens.sb2 en parallèle de sa lecture.

utile de réfléchir au nombre de scripts qui se déroulent en parallèle à un moment donné. On peut imaginer différents prolongements : par exemple, un étang où nage le poisson, le chat ne pouvant l'attraper, et le manger, qu'au bord de l'étang. Ou encore la présence de plusieurs poissons… Un exemple de production est proposé dans le fichier joint PoissonChatChien.sb2.

B. *Un jeu de pong*

Lien vers l'activité : Pong.sb2
Le professeur peut choisir ce type d'activité avec différents objectifs : travailler la notion de direction angulaire relative ou absolue, découvrir et utiliser les constructions conditionnelles, découvrir et utiliser des variables, programmer une gestion d'événements. Selon le(s) choix qu'il effectue, il développera plus tel ou tel aspect de l'activité, proposera des défi différents aux élèves, prolongera l'activité suivant diverses directions. On peut commencer par une activité concernant le rebond sur les bords d'une balle. Il vaut mieux choisir pour commencer un lutin dont la direction est immédiatement visible (le chat, une flche), et reporter à plus tard l'utilisation d'un lutin en forme de balle. De la même façon, il est sans doute utile d'afficher la direction du lutin (en cochant la case appropriée tout en bas de la catégorie « Mouvement »). Une répétition infinie de , , permet alors l'observation de ce qui se passe aux bords : un objectif est que les élèves conjecturent les règles de changement de direction. On peut alors proposer un défi qui est de réaliser le même processus, mais sans utiliser le bloc. Il faut toutefois prendre garde à repérer l'approche des bords, en ajoutant des répétitions infinies de test sur l'abscisse (inférieure à -215 ou supérieure à 215) et sur l'ordonnée.
Ce défi n'est pas si simple pour des élèves inexpérimentés. On ajoute deux raquettes (que les élèves peuvent dessiner, un simple rectangle suffi). On pensera à les nommer de façon explicite, raquette gauche et raquette droite, par exemple. Il n'est pas très difficile d'associer à chaque raquette un script qui permet à l'utilisateur de la faire monter ou descendre, à l'appui d'une touche ou d'une autre. Il faudra donc choisir quatre touches différentes du clavier, pour le joueur gauche et le joueur droit. Il s'agit là d'une programmation événementielle très simple à mettre en œuvre. Pour l'instant, la balle et les raquettes agissent sans aucune interaction. Il convient maintenant de gérer le rebond de la balle sur chaque raquette. Pour cela on crée une boucle infinie qui teste le contact à l'aide du premier bloc de la catégorie « Capteurs ». Si la raquette gauche est touchée, il suffi de changer la direction de la balle en l'opposé de la direction actuelle. Remarquons que puisqu'il s'agit de modifier la direction de la balle, c'est bien sur le script de la balle qu'il faut travailler, et non sur celui de la raquette ! C'est un objectif important que les élèves comprennent bien quelles commandes sont à écrire dans le script de la balle et quelles autres dans celui de chaque raquette : on introduit ici des concepts liés à la programmation-objet, sans avoir pour autant besoin du moindre formalisme. Une autre remarque s'impose ici : on peut bien sûr créer un unique script long et compliqué pour la balle, mais il est sans doute plus simple de créer plusieurs scripts associés chacun à un comportement particulier : le déplacement perpétuel sur le terrain de jeu, la gestion du rebond sur les bords (si on ne veut pas utiliser la commande), le contact avec la raquette gauche, le contact avec la raquette droite. Cela introduit à la fois la notion de programmation d'actions en parallèle et permet une mise au point beaucoup plus facile des programmes.
Plusieurs prolongements sont envisageables.
En premier lieu, on peut ajouter un score pour chacun des deux joueurs, en créant deux variables score gauche et score droit (toujours utiliser des identifiants explicites et non abrégés), qui sont incrémentées à chaque rebond réussi sur la raquette correspondante. Les élèves peuvent décider si la balle rebondit sur les côtés gauche et droit au cas où les raquettes l'auraient ratée, ou si dans ce cas la partie continue avec la balle au centre et une direction tirée au hasard. On peut également ajouter des pénalités au score adéquat dans ce cas de balle ratée.
Un prolongement plus ardu consisterait à gérer des niveaux de difficulté, à chaque fois qu'un score dépasse une dizaine (par exemple) : ce pourra correspondre à une accélération de la balle, à des raquettes moins longues, etc.
Un exemple de production est proposé dans le fichier joint Pong.sb2.

C. *Un jeu avec des pièces*

Lien vers l'activité : Francs.sb2
L'activité est reprise d'un exercice du rallye mathématique de l'académie de Lyon.
On dispose de quatre pièces, disposées initialement respectivement du côté face, pile, face, pile.
Chaque clic sur une pièce la retourne, ainsi que son éventuelle voisine de droite.
Le joueur gagne quand les quatre pièces sont du même côté.
Bien entendu, on peut varier le nombre de pièces, ainsi que la disposition initiale —mais attention, certaines dispositions ne sont pas résolubles !
L'objectif de cette activité est de travailler les notions suivantes :

- instructions conditionnelles ;
- utilisation d'une variable ;

- utilisation simple d'une liste ;
- gestion d'événements déclenchés par la souris ;
- déclenchement de plusieurs actions en parallèle ;
- échange de messages entre objets.

Chaque pièce est représentée par un lutin, avec deux costumes, pour le côté pile et le côté face. On propose, dans le fichier joint Francs.sb2 , un exemple de production autour de cette activité. Dans cet exemple, on a également créé deux costumes pour l'arrière-plan, avec un costume pour signaler la victoire et un autre pour préciser qu'on aurait pu gagner en moins de clics. En effet, avec la configuration initiale choisie, il existe des solutions en seulement 2 clics. Une difficulté de l'activité tient à ce qu'un clic sur une pièce doit entraîner une action sur sa voisine : c'est une occasion naturelle d'introduire la notion de message entre lutins. Les actions déclenchées par un clic sur la pièce et la réception d'un message sont très similaires, mais différentes : c'est là aussi une difficulté.

Dans un objectif de différenciation pédagogique, on peut apporter de l'aide en fournissant le script de la pièce 3, par exemple, et demander d'écrire les scripts des pièces 1, 2 et 4, ce qui demande une bonne analyse des actions, et correspond bien aux compétences « mettre au point un programme simple » et « reconnaître des schémas ». L'initialisation de la variable qui compte le nombre de clics et de la liste qui retient les faces courantes des pièces est réalisée, à l'appui du drapeau vert, par un script lié à l'arrière-plan. Cette méthode permet de distinguer ce qui relève du comportement global de l'application et ce qui est spécifique de chaque pièce. On notera qu'on a choisi également de tester la victoire dans le même script, lié à l'arrière-plan.
L'utilisation d'une liste paraît indispensable, car chaque lutin est le seul à avoir accès au numéro de son costume courant : aucun lutin ne peut donc tester l'égalité des côtés des quatre pièces en utilisant seulement le bloc. Cette activité ne demande qu'une utilisation très élémentaire d'une liste, et peut donc servir d'introduction à cette notion.
Le test de victoire peut être réalisé de plusieurs façons : par une imbrication de plusieurs conditionnelles, comme dans la production jointe, mais aussi en utilisant les connecteurs logiques. Ce peut être intéressant de ne pas guider les élèves sur telle ou telle voie, mais de les laisser choisir. On peut s'attendre à ce que les élèves cherchent un bloc pour tester des égalités simultanées, et expliquer qu'on ne peut tester que l'égalité de deux quantités. On a choisi de représenter les valeurs pile et face par les nombres 1 et 0, ce qui permet d'utiliser qui devra naturellement faire l'objet d'un travail spécifique : une fois qu'on aura trouvé comment modifier un élément d'une liste, il faudra encore trouver comment une seule expression arithmétique simple permet de passer de pile à face et vice versa. L'utilisation d'images trouvées sur Internet pour les lutins figurant les pièces sera l'occasion d'évoquer la question de licence liée à l'utilisation de ces images : sont-elles du domaine public ? Est-ce le cas pour les pièces de 1 €, par exemple ?

D. *Un jeu de tic-tac-toe*

Lien vers l'activité : TicTacToe.sb2
Le lecteur est invité à expérimenter à partir du fichier joint TicTacToe.sb2 , et à regarder en détail les scripts proposés en parallèle à sa lecture. Cette activité est plus ambitieuse, et le fichier joint n'est qu'un exemple de production qui demande un niveau de maîtrise important. Mais on pourra s'en inspirer, car elle se découpe aisément en sous-problèmes dont la résolution permet d'approcher de nombreuses notions différentes. On se propose de faire jouer tour à tour le joueur (l'utilisateur du programme) et la machine. C'est le joueur qui joue le premier. Les marques du joueur sont des croix vertes, celles de la machine des disques rouges. Ainsi chacune des 9 cases du jeu peut-elle se retrouver dans 3 états différents. Chaque case est simulée par un lutin, qui porte 3 costumes : un carré blanc, une croix ou un disque. Jouer revient donc à changer le costume d'une case blanche. On introduit ainsi le concept d'attribut d'un objet : il s'agit ici du costume. Ces attributs sont privés : seul un lutin peut savoir quel est son costume actuel. Or l'arbitre, pour décider de la victoire du joueur ou de la machine, doit connaître l'état de chaque case du jeu. C'est pourquoi on introduit une liste des valeurs des différentes cases : cette liste étant une variable globale, les lutins pourront mettre à jour la valeur qui leur correspond, et l'arbitre examiner les valeurs de toutes les cases qui l'intéressent. On peut commencer en choisissant les valeurs 0, 1, 2 pour coder les trois états. La victoire est déclarée lors de l'alignement de 3 croix ou de 3 disques, et se teste donc en faisant par exemple la somme des trois valeurs correspondantes 1+1+1 = 3 ou 2+2+2 = 6. En jouant on s'aperçoit cependant que l'arbitre rend des décisions inexactes. Il est sans doute intéressant de laisser les élèves trouver sans aide du professeur l'erreur commise : en effet, 0+1+2 = 3 donc l'alignement blanc / croix / disque provoque, à tort, la victoire. Une façon simple de corriger le programme consiste à choisir les valeurs 0, 1 et 4 pour coder les trois états : il suffi de vérifier que la seule décomposition de 3, respectivement de 12, comme somme de trois de ces valeurs est 3 = 1+1+1, respectivement 12 = 4+4+4.
Le joueur joue en cliquant une case blanche, chaque lutin a donc un petit script qui gère l'événement clic de souris. Il faut tester, par une conditionnelle, que la case cliquée était blanche, ce que le lutin peut faire en examinant son costume actuel (lui en a bien le droit) ou en consultant la liste globale des valeurs des cases. Après avoir joué, on passe la main à l'arbitre, pour qu'il puisse éventuellement constater une victoire, en lui envoyant un message, puis on cède la main à la machine, par un nouveau message « À la machine ! » que reçoit en fait

l'arbitre. L'arbitre choisit alors au hasard une case jusqu'à trouver une blanche (avec une boucle répéter tant que) qu'il conserve dans une variable globale case avant d'envoyer à son tour un message « coup machine ». Chaque lutin gère la réception de ce message, en testant s'il concerne bien sa case à l'aide de la variable de même nom. Si tel est le cas, il poursuit de façon analogue à ce qu'il sait déjà faire pour répondre à un clic.
Dans le cadre d'un projet courant sur plusieurs séances, on peut distribuer le travail de conception des scripts à différents élèves, en distinguant par exemple, pour un lutin simulant une case, le script à utiliser en réponse à un clic ou en réponse au message « coup machine », et, pour l'arbitre, le test de victoire et la commande du déroulé du jeu, avec la gestion des messages.

E. A vous de jouer.

Réalisez un document que vous pourriez fournir aux élèves, présentant une activité sur scratch, et présentant si possible de la différentiation. Vous préciserez les compétences pré requises et les objectifs de l'activité.

Pour vous aider un peu vous pouvez observer l'énoncé en annexe qui est loin d'être parfait (je l'ai réalisé pendant mon année de stage) mais qui permet de voir comment mettre en place de la différentiation. Cette activité était la seconde activité en salle informatique nous avions fait une heure sur le site une heure de code (lien au début de ce document). Nous avions déjà parlé d'algorithme dans le cadre de la construction d'une figure par translation plus tôt dans l'année en parlant d'étapes à réaliser dans un ordre précis comme une recette de cuisine, mais aucun formalisme ou vocabulaire spécifique n'avait été donné. Les élèves connaissaient donc déjà la logique des blocs et l'objectif de la séance était de découvrir le logiciel afin de pouvoir l'utiliser facilement par la suite. Les classes qui ont travaillé sur cette séquence étaient des classes avec un niveau de base très élevé avec des élèves très autonomes.

La présentation de l'énoncé est aussi importante que son contenu. Vous remarquerez ici l'usage d'un code couleur (utilisé dans tous les documents de l'année) avec les objectifs à atteindre par tous en vert et les objectifs « bonus » en mauve. Soignez donc l'apparence de votre énoncé. Une fois terminé, envoyez vos travaux à : yannick.feron@ac-lille.fr

Idées de thèmes :
Ceci est une petite liste d'idées cependant ne vous sentez pas limité à ces idées soyez créatifs !
Un calculateur d'aires, travailler l'addition des nombres relatifs, un programme de calcul, la réalisation d'une figure (du même type que celle que vous avez faites en début d'année)…

Annexe :

A la découverte de Scratch

Activité 1 : Réaliser une animation simple d'un objet.

1) Utiliser les costumes et déplacer automatiquement le personnage :

Objectif : Faire marcher le chat

Le but est de faire marcher le chat. Pour cela, nous utiliserons les blocs suivants :

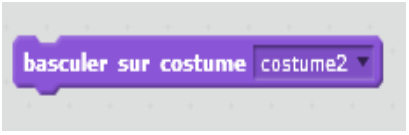
Bloc 1 :



Bloc2 :



Bloc 3 :



Bloc 4 :



Bloc 5 :



Explications sur la fonction des blocs :

- Bloc 1 : C'est la condition de départ. Ici lorsque l'on clique le drapeau vert, les instructions qui suivent seront exécutées.
- Bloc 2 : C'est une boucle, c'est-à-dire une répétition, on demande au logiciel de répéter un certain nombre de fois une série d'actions.
- Bloc 3 : Permet de changer l'apparence du personnage.
- Bloc 4 : Permet de faire avancer le personnage d'un nombre de pas.
- Bloc 5 : Permet d'attendre avant de réaliser l'instruction suivante.

Premier programme :

-Réaliser une série d'instructions permettant de changer l'apparence du chat toutes les 2 secondes.

-Modifier votre programme pour que le lutin avance de 10 entre chaque changement de costume.

Appelez le professeur et passez à la suite en attendant son arrivée.

Pour aller plus loin (ces questions utilisent des blocs qui ne sont pas expliqués ci-dessous, à vous de les trouver) :

-Modifier votre programme pour que le lutin fasse demi-tour lorsqu'il arrive au bord de l'écran.

-Modifier votre programme pour que le lutin se trouve toujours au milieu de l'écran et orienté vers la droite après que l'on ait cliqué sur le drapeau vert.

Sauvegardez votre programme et créez en un nouveau.

2) Animer le personnage et déplacement à partir du clavier :

Objectif 1 : Créer un personnage animé.

Ici nous allons suivre une démarche pas à pas pour changer de personnage, lui créer trois costumes et changer ses costumes à intervalles réguliers. Nous utiliserons des blocs similaires à ceux précédents.

Utilisation de scratch.

Changer de lutin :

- Cliquez sur choisir un nouveau lutin dans la bibliothèque (première icone à côté de nouveau lutin).
- Dans la catégorie fantaisie, cliquez sur Ghost 1.
- Faites un clic droit sur le chat et supprimez le.
- Cliquez sur costumes

Vous arrivez dans la partie qui permet de modifier manuellement les couleurs des lutins.

- Faites un clic droit sur le fantôme (celui situé dans le bandeau au centre de l'écran) afin de le dupliquer deux fois.
- Changez la couleur du second fantôme et donnez un air méchant au troisième à l'aide du crayon (les outils de dessin sont dans la partie droite de l'écran)

Vous avez maintenant trois costumes pour votre fantôme.

-Créez un programme qui lorsque le drapeau vert sera pressé permettra de répéter indéfiniment un changement de costume toutes les deux secondes.

Si cela fonctionne passez à l'objectif 2.

A retenir pour le cours : Les instructions du type répétez 10 fois ou répétez indéfiniment sont appelées des boucles. Elles permettent de répéter une action un nombre donné de fois.

Objectif 2 : Le déplacer avec les touches du clavier

Ici nous allons suivre une démarche pas à pas pour que le programme fasse avancer et pivoter le fantôme lorsque l'on enfonce la barre espace du clavier.
Sans effacer votre programme précédent, réalisez le programme ci-dessous.
Le premier bloc correspond à la condition de déclanchement des actions.
Le second fait avancer le lutin et le troisième le fait tourner.



Complétez le programme pour :
- Que le lutin recule lorsque l'on appuie sur la flèche de gauche.
- Que le lutin s'agrandisse lorsque l'on appuie sur la flèche du haut.
- Que le lutin rétrécisse lorsque l'on appuie sur la flèche du bas.
Appelez le professeur et passez à la suite en attendant son arrivée.

Pour aller plus loin :
Créez un programme permettant que lorsque le drapeau vert est enfoncé le fantôme poursuive le pointeur de la souris.

Thierry LOOF
Maj Yannick FERON