Algorithmique avec Scratch

On présente ici quelques activités d'initiation à l'algorithmique pouvant être réalisées avec des élèves de 5e.

Mots-clefs. Algorithmique, pseudo–code, Scratch, TICE.

Niveau. Cinquième, Quatrième, Troisième.

Le programme de mathématiques en cycle 4 (5e–4e–3e) applicable à la rentrée 2016 préconise d'initier les élèves à la programmation événementielle à partir de la classe de 5e, avant d'introduire progressivement au long du cycle les notions d'actions exécutées en parralèle, de variable informatique, de boucle et d'instruction conditionnelle. L'objet de cet article est de présenter un compte–rendu d'activités effectuées par l'auteur avec des élèves de 5e dans le but de les initier aux concepts de base de l'algorithmique.

A. Découverte de l'éditeur

La version en ligne de l'éditeur de pseudo–code `Scratch`, que l'on peut trouver à l'adresse suivante :

`https://scratch.mit.edu/projects/editor/`

se présente sous la forme illustrée en figure 1.

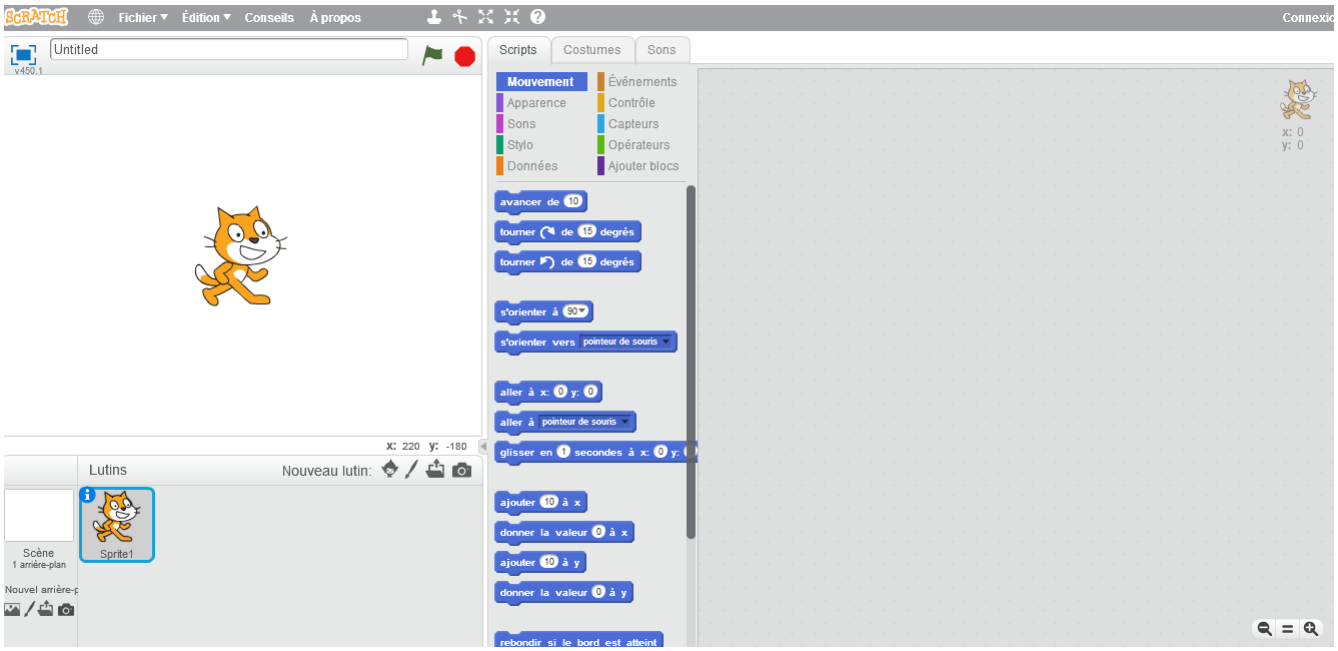

Figure 1 – Éditeur en ligne pour `Scratch`.

Cette page web détecte de façon dynamique la langue de l'utilisateur et requière un plugin `flash` fonctionnel. Si la langue est mal détectée, il est possible de changer celle–ci à l'aide de l'icône en forme de globe située en haut à gauche de la fenêtre.

1. Interface

La zone de gauche de l'éditeur contient un personnage dessiné (ici un chat) à qui l'on va pouvoir donner des instructions. Pour ce faire, il suffit dans un premier temps de cliquer sur les commandes situées dans la zone centrale, qui divise ces dernières en plusieurs catégories :

- ◇ la catégorie "Mouvement", contenant les instructions permettant de déplacer le personnage dessiné dans son environnement ;
- ◇ la catégorie "Apparence", permettant d'agir sur l'aspect du personnage (*e.g.* en modifiant sa taille, son costume, ...) ;
- ◇ la catégorie "Sons", dont le nom résume bien les fonctions ;
- ◇ la catégorie "Stylo", permettant de programmer des dessins à l'écran ;
- ◇ la catégorie "Données", permettant la gestion de variables ;
- ◇ la catégorie "Événements", contenant les blocs permettant de débuter une série d'instruction lorsque une condition est remplie (*e.g.* si une touche particulière est pressée) ;
- ◇ la catégorie "Contrôle", contenant les instructions conditionnelles et les boucles ;
- ◇ la catégorie "Capteurs", permettant de recueillir de l'information (*e.g.* le personnage a–t–il été cliqué ?) ;
- ◇ la catégorie "Opérateurs", contenant les blocs permettant de réaliser différentes opérations mathématiques ou logiques (*e.g.* addition, soustraction, ou logique, ...) ;
- ◇ enfin, un sous menu "Ajouter blocs" permettant de créer ses propres instructions.

Notons également la présence de deux onglets supplémentaires au–dessus de ce menu, permettant à l'utilisateur de personnaliser les costumes et les sons émis par le personnage.

2. Syntaxe

Pour écrire un algorithme avec `Scatch`, il suffit de faire glisser les "blocs" correspondant aux instructions voulues dans la zone de droite de l'éditeur et de les emboîter entre eux. Ce système est très intuitif, puisque ces blocs ont des formes distinctes façon "pièces de puzzle" qui permettent à l'élève d'évaluer en un coup d'œil si elles pourront se combiner. Un suite d'instructions écrites avec `Scratch` ressemble donc à une "pile" de blocs compatibles, comme illustré en figure 2.

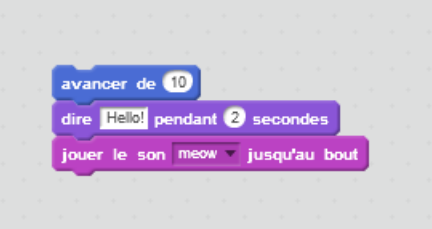

FIGURE 2 – Une suite d'instructions composée avec `Scratch`.

Lorsque l'on souhaite programmer des boucles ou des instructions conditionnelles, il peut être nécessaire d'emboîter des blocs les un à l'intérieur des autres. Par exemple, une boucle `Si` devrait faire appel à l'un des blocs de comparaison du sous–menu "Opérateurs". La encore, la forme des pièces est suffisamment distinctive pour éviter toute confusion.

B. Programmer avec Scratch

Dans un premier temps, il peut être profitable de laisser les élèves "jouer" avec l'éditeur, pour acquérir une première familiarité avec son interface. Notons que cliquer sur une instruction, sans nécessairement la déplacer dans la zone de droite de l'éditeur, causera son exécution ; cela peut permettre de se rendre compte *a priori* de l'effet de la plupart d'entre elles.

1. Premiers algorithmes

Une fois les élèves familiarisés avec l'interface, une première étape naturelle serait la programmation d'une suite d'instructions simple, du type de celle illustrée en figure 2, ne faisant appel qu'à des instructions aisément compréhensibles (en se limitant par exemple aux trois premiers sous–menus : Mouvement, Apparence et Sons).

Un tel bloc de code ne sera toutefois exécuté qu'en cliquant dessus ; le moment peut donc être opportun d'introduire les pièces "initiales" du sous–menu "Événements", qui se placent en tête de bloc et conditionnent son exécution à un certain stimulus utilisateur. Par exemple, nous pouvons reprendre le bloc de la figure 2 en lui intimant de s'exécuter dés que la touche A du clavier est pressée (*cf.* figure 3).

FIGURE 3 – Une suite d'instructions composée avec Scratch, cette fois–ci avec un bloc initial.

On peut aussi complexifier les instructions données, en utilisant les outils simples de repérage dans l'espace fournis par le menu "Mouvements" ; par exemple en forçant le personnage à suivre le pointeur de la souris du "regard" ou influer sur sa taille.

2. Vers les actions en parallèle

Scratch permet nativement de gérer plusieurs blocs de code simultanément. Il suffit pour ce faire de les positionner séparément dans la zone de droite de l'écran, comme dans l'exemple illustré figure 4.

FIGURE 4 – Deux blocs avec des déclencheurs différents.

Dans cet exemple simple, le personnage réagit indépendamment à deux instructions différentes : si il est cliqué, il se mettra à miauler et si la touche "espace" est pressée, il affichera le message "Hello" pendant deux secondes. On peut de cette façon faire programmer de petits jeux simples aux élèves, par exemple en leur permettant de déplacer le personnage grâce aux flèches de direction (*cf.* figure 5).

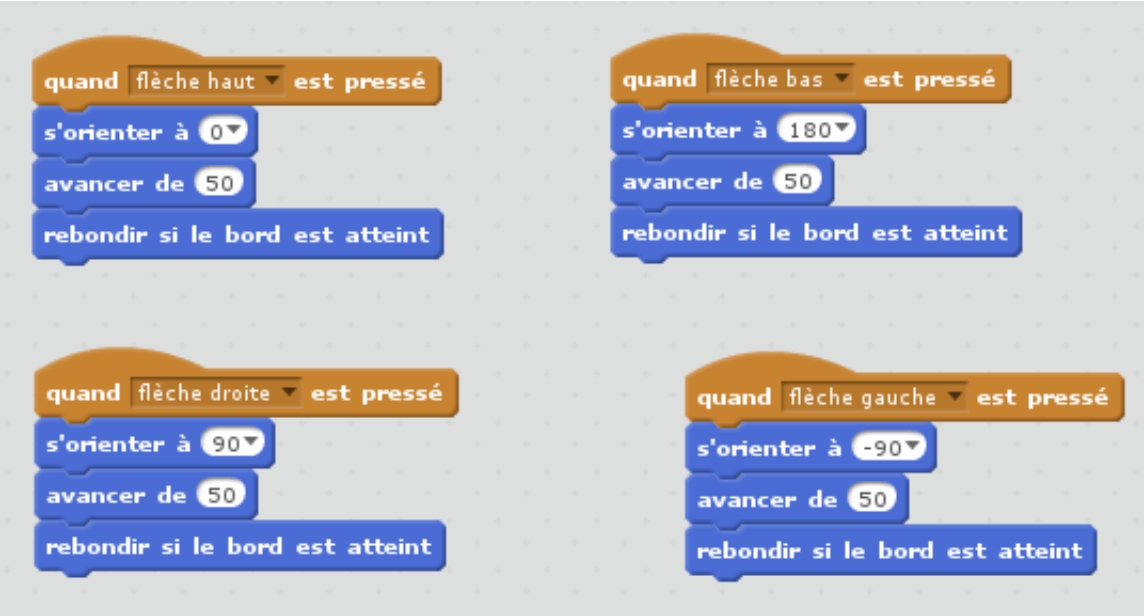

FIGURE 5 – Déplacement à l'aide des flèches directionnelles.

Pour aborder des choses plus complexes, on peut tenter d'approcher la notion d'actions en parallèle à l'aide de clones. `Scratch` permet en effet de dupliquer le personnage actif et d'agir sur chacun de ses clones de façon indépendante.

FIGURE 6 – Déplacement à l'aide des flèches directionnelles.

Observons par exemple le pseudo–code présenté en figure 6. Ce dernier est composé de deux blocs : celui de gauche s'éxécute lorsque la touche "espace" est pressée et fait usage de l'instruction "créer un clone" du sous–menu "Contrôle" ; cette dernière va faire apparaître un deuxième personnage auquel nous allons également pouvoir donner des instructions.

Dans le second bloc, nous commençons par la pièce "quand je commence comme un clone" du sous–menu "Contrôle" : cela signifie que ce qui suit sera exécutée à chaque apparition d'un nouveau clone. De fait, lorsque nous pressons la touche "espace", les deux blocs de la figure 6 s'exécuteront en parallèle. Notons qu'il est préférable de faire le ménage après chaque clonage, pour éviter l'encombrement du plan de travail ...

3. Instruction conditionnelle

Le pseudo–code `Scratch` permet également de manipuler des instructions conditionnelles de type `Si–Sinon`, ceci pouvant se faire de façon très intuitive grâce à la structure "pièces de puzzle" évoquée précédemment.

FIGURE 7 – Différentes pièces pour une instruction conditionnelle.

La figure 7 illustre cet état de fait : la structure globale de l'instruction (trouvée dans le sous–menu "Contrôle") possède une encoche à coté du mot clé "si" pouvant recevoir n'importe quelle instruction ayant la forme appropriée (un hexagone). Celles–ci peuvent aussi bien être des résultats d'opérations mathématiques ou logiques (sous–menu "Opérateurs") que des capteurs liés à des actions de l'utilisateur (sous–menu "Capteurs").

FIGURE 8 – Un exemple d'instruction conditionnelle.

Notons pour finir que les encoches carrées peuvent recevoir une donnée (nombre, variable) entrée manuellement ou n'importe quel bloc de forme elliptique (sous–menu "Opérateurs").

4. Boucles

De la même façon que pour les instructions conditionnelles, `Scratch` permet une approche intuitive de la notion de boucle en aidant les élèves à identifier visuellement la "bonne place" des instructions et conditions. À l'heure actuelle, ce pseudo–code permet nativement d'effectuer des boucles conditionnelles (`Tant que`) ou suivant un compteur (`Pour`) ; notons toutefois que le compteur ne peut pas être utilisé comme variable si on utilise le second choix (on peut toutefois retomber sur ses pieds en incrémentant manuelle le compteur d'une boucle conditionnelle).

FIGURE 9 – Boucles.

C. Pour aller plus loin

Ce court article ne se veut bien entendu pas exhaustif, et il est possible d'aller beaucoup plus loin dans le "pseudo–codage" avec `Scratch`. Il est par exemple possible, en faisant un large usage du sous–menu "Capteurs" de programmer de petits jeux interactifs mettant en scène un ou plusieurs personnages. De même, cet article s'intéressant principalement à la mise en pratique des notions enseignées par l'auteur en classe de cinquième, l'usage de variables n'a pas été abordé en détails.

Références

[En] Aide (officielle) de `Scratch` en anglais : `http://scratch.mit.edu/help/`.

[Fr] Aide (officieuse) de `Scratch` en français : `http://scratchfr.free.fr/`.

Arnaud GIRAND
`arnaud.girand@ens-cachan.org`
Collège Jean Mermoz
24, rue du 2e régiment de dragons
02000 Laon