Conseils aux parents ou enseignants

Les projets présentés dans cette page ont été réalisés dans l'ordre de présentation.

Les enfants au moment où nous avons commencé, avaient déjà des notions de séquence d'instructions, boucle, contrôle conditionnel.

Les exercices à faire sur code.org sont plaisants et les enfants aiment en général les faire. Il n'est pas obligatoire de tout faire.

https://studio.code.org/s/course2 https://studio.code.org/s/course3

Avant de commencer avec Scratch, il est bon de s'assurer que les enfants ont une certaine maitrise de l'ordinateur :

- Savoir naviguer dans l'arborescence des dossiers.
- Savoir créer un dossier.
- Savoir copier/coller un fichier
- Savoir dézipper un fichier
- Connaitre quelques extensions de fichier utilisé par Scratch : fichier .svg ou .png = image d'arrière-plan ou de costume de lutin, fichier .sb2 = programme scratch, fichier .sprite2 = un lutin avec ses costumes et ses scripts, fichier .wav = musique

Scratch est un merveilleux outil pour apprendre les grands principes de la programmation et ne se contente pas du b.a ba. La programmation de « clones » est déjà un pied dans la programmation objet.

C'est un outil très souple. La contre-partie de cette souplesse est qu'un projet très mal programmé, très mal conçu peut fonctionner. Son auteur aura la satisfaction de voir quelque chose qui marche, mais n'aura pas appris à programmer correctement.

Mon objectif, dans les initiations à la programmation avec Scratch, est de construire des bases un peu solides pour pouvoir passer sans trop de difficultées à un autre language de programmation.

Chaque programmeur a son style de programmation. Je pars très souvent de projets que je trouve sur la plate-forme Scratch lorsque les lutins ou les arrière-plans me plaisent, que le jeu est interessant. Dans 99% des cas, je remanie complètement le code, soit parce que j'estime que le projet est programmé n'importe comment, soit parce que la façon de programmer ne correspond pas à mon style.

Mes principes :

- Dans le script d'un lutin on ne peut avoir qu'un seul

- On ne multiplie jamais le nombre de lutin pour simplement se faciliter la vie.
- Lorsqu'on a besoin de synchroniser l'action de deux lutins, on utilise des messages explicites.
- Dans un jeu, il y a, la plupart du temps, un lutin guidé par le joueur. On commence par programmer les mouvements de ce lutin. On teste et on sauvegarde.
- On ajoute l'un après l'autre les autres éléments du jeu. On teste à chaque ajout, on vérifie que tout fonctionne bien et on sauvegarde.
- Lorsqu'on a besoin de variables, de préférence on utilise des variables locales, sauf si d'autres lutins ont besoin eux-aussi de connaitre ces variables. Je suis plus souple sur ce principe, s'il n'y a pas de danger de confusion lors de l'utilisation des variables, si ces variables doivent être affichées sur la scène, au début de l'utilisation des variables.

Vous le savez sans doute, mais j'insiste sur le fait qu'une séance de programmation se prépare. Même si on dispose du corrigé, il faut s'approprier le projet, il faut savoir comment tout fonctionne, il faut prévoir dans quel ordre on va faire faire les choses, sachant que cet

ordre peut être modifié en fonction des réactions, difficultés, envies des enfants.

Pendant la séance, si les enfants ont des idées qui ne correspondent pas toujours à ce qui est prévu, il faut les laisser faire. S'ils arrivent à les mettre en œuvre, c'est très bien, on garde et on essaie de rendre le reste cohérent avec ce qu'ils ont fait.

Mais si l'on n'est pas soi même très à l'aise avec Scratch ou avec la programmation, il vaut mieux être un peu plus dirigiste et s'en tenir au corrigé, afin d'arriver au bout. Rien n'est plus décevant qu'un projet que l'on n'arrive pas à faire fonctionner correctement.