Activité (deconnectée)- Initiation à la programmation avec le jeu du robot

TOUTES LES FICHES

par Roseline Faliph

CC BY SA

PUBLIC	PARTICIPANTS	ANIMATEURS	NIVEAU	PRÉPARATION	ACTIVITÉ

Description

Cette fiche d'activité vous présente une activité débranchée d'initiation à la programmation, à travers un jeu où les participant-es occupent le rôle de robots exécutant un programme.

Objectifs

Algorithme
Bug
Boucle
Procédure
Efficacité en programmation

Matériel

Un paysage avec des obstacles représentés : il peut être imprimé sur une grand feuille, fabriqué avec un très grand drap, ou même dessiné au sol avec des craies (proposition de paysage à télécharger ci dessous) !
De petits objets à déposer sur le paysage : cailloux, fleurs, feuilles, objets (vrais ou imprimés et plastifiés)
Quelques exemples de programmes à imprimer et plastifier (voir fichiers ci-dessous).
Des tablettes (facultatif)

Contenus utilisés

Application LightBot (facultatif)

Pré-requis

Aucun

1 Introduction

L'objectif de ce parcours est de sensibiliser des publics à la **programmation** informatique dans un contexte d'activité en extérieur avec peu ou pas de connexion Internet. Il s'adresse aux enfants ou à des groupes parents-enfants qui n'ont pas de connaissance en programmation. L'idée est de combiner des phases d'activités déconnectées et des phases de découverte sur tablette.

Cette activité est très largement inspirée du projet de médiation imaginé par **Marie Duflot-Kremer**, enseignante-chercheuse à l'Université de Lorraine. Depuis 2011, elle s'est "spécialisée" dans l'initiation aux concepts informatiques sans ordinateur, pour tous les âges et tous les niveaux.

Vous pouvez retrouver sa proposition initiale d'atelier ici :
https://members.loria.fr/MDuflot/files/med/robot.html

2 Etape 1 : Atelier déconnecté - le robot idiot

Cette activité propose à tous de réaliser et d'exécuter des programmes informatiques, sans matériel informatique. Pour cela, le médiateur dispose un drap sur le sol (voir image dans les fichiers à télécharger ci-dessous).

Les participants sont regroupés en groupes de deux : un participant joue la mémoire de l'ordinateur (il lit le programme), l'autre participant joue le processeur de l'ordinateur (il exécute le programme).

On peut dire également que l'un des participants sera un robot-randonneur (il recense les fleurs sauvages par exemple ou il ramasse les déchets, à vous d'inventer l'histoire de votre choix !) et l'autre participant sera l'ordinateur qui lui donne des instructions pour se déplacer dans un paysage de montagne.

Ces instructions ont déjà été rédigées, mais attention ! Certaines contiennent des erreurs. Aux participants de trouver les bonnes instructions, et de corriger celles qui risquent de causer la perte de notre petit robot !

1. Présentation du décor

Nous voici à la montagne, il y a une rivière, un volcan, une forêt très dense, un lac, des ponts… Le robot doit se déplacer dans ce paysage mais il doit rester sur les points. La forêt, le volcan, le lac et la rivière sont donc interdits.

2. Découverte du langage

Pour commencer, tous les participants sont des **robots**. Tous se mettent sur le tapis, chacun sur un point, et suivent les indications données par l'animateur, à choisir entre un pas à droite, à gauche, en avant ou en arrière.
Dès qu'un participant se trouve dans le décor (rivière, lac, forêt ou volcan) ou bien sort du tapis, il se met sur le côté. Quand tout le monde est sur le côté on peut recommencer.

Précisions à apporter :

- *Un pas c'est se déplacer d'un point, pas deux*
- *Pas de déplacement en diagonale*
- *On reste toujours dans la même orientation. On fait des pas de côté mais on ne tourne pas sur soi-même*
- *On est un robot, donc on exécute les instructions sans les changer, même si elles nous envoient dans le décor*
- *Rappeler gauche et droite pour ceux qui pourraient se tromper. Pour les enfants qui ne connaissent pas leur gauche et leur droite, on peut utiliser des rubans de couleur au poignet : un pas du côté rose, un pas du côté vert… comme le propose Marie Duflot-Kremer.*

Ils viennent d'utiliser des **algorithmes**, comme un ordinateur.

Un algorithme est une **succession d'instructions** qui permet de résoudre un problème.

3. Exécution des programmes en binôme

On met le point de départ comme montré sur le plan (tout en bas, 7ème point à partir de la gauche). Tous les programmes vont démarrer à partir de là. Pour chaque programme on va répartir la tâche entre un participant qui joue le robot et un participant qui joue le rôle de l'ordinateur. Faire **exécuter les programmes de 1 à 4**.

Précisions : le robot et l'ordinateur ne savent qu'exécuter les tâches qui leur sont attribuées. Il n'est pas question de faire un pas de trop ou d'ajouter une flèche en cours de route. L'ordinateur ne fait pas cela ! On peut même bander les yeux du "robot" si besoin.

Les programmes 2 et 4 ont chacun un "bug". Cela permet d'expliquer que si le programme est faux, l'ordinateur va l'exécuter bien sagement et faire consciencieusement un truc faux. Il ne s'est pas trompé, c'est le programmeur qui a fait une erreur. Dans chacun de ces deux programmes, ajouter une seule flèche (vers

la droite dans le 2 ou vers le haut dans le 4) permet de corriger le programme. Cela montre bien qu'une toute petite erreur peut entraîner un gros problème (robot noyé), mais que parfois il n'y a pas grand chose à changer pour avoir un programme correct. Cela permet également de mentionner qu'il arrive à tous les informaticiens de faire des erreurs dans leurs programmes (ils sont très longs et complexes en général) mais qu'un bon informaticien va tester/vérifier son programme jusqu'à trouver et corriger les bugs.

4. Changement de jeu d'instructions

Les programmes 1 à 4 permettent de se promener sur le tapis, mais notre robot peut faire plus, comme collecter des échantillons : pierre, feuilles, déchets… Ces variantes seront à réaliser selon le temps disponibles et l'âge de participants :

- disposer sur le tapis les objets comme montré sur le plan et ajouter une instruction **"ramasser"** qui permet de prendre tous les objets situés sur le point où l'on se trouve,
- regrouper les flèches identiques consécutives, par exemple écrire ? x 4 au lieu de ? ? ? ?. Cela évite des erreurs en comptant les flèches et rend les programmes plus compacts (cf programme 5),
- remplacer les instructions "pas de côté" par des instructions de quart de tour à droite ou à gauche (cf programme 6).

5. Écriture de programmes

Une fois que les participants ont apprivoisé le langage de programmation c'est à eux d'**écrire leurs propres programmes** (utiliser une ardoise par exemple, plus facile pour corriger qu'une feuille de papier).

Une liste d'exemples de programmes à proposer :

- ramasser trois objets, peu importe lesquels,
- revenir au point de départ avec deux objets dans les mains,
- ramasser trois cailloux et aller au bord de la forêt,
- ramasser un objet de chaque sorte,
- terminer sur un pont avec dans ses mains une fleur, un caillou et une feuille,
- ramasser toutes les feuilles et faire le tour de la forêt,
- ramasser (dans n'importe quel ordre) deux fleurs et deux feuilles,
- ramasser, dans cet ordre, une feuille puis une fleur puis une feuille puis une fleur (et rien d'autre).

En faisant cela on peut remarquer qu'il existe plusieurs programmes possibles pour faire la même chose. On peut si on a le temps **comparer des programmes** qui effectuent la même tâche, en termes d'efficacité (en informatique on parle de complexité).

Comparer les programmes inventés :

Un programme peut faire le même travail qu'un autre et être **plus court** à écrire, ou nécessiter moins de pas, ou de tours. Parfois il est assez évident qu'un programme qui va droit au but est **plus efficace** qu'un autre qui ferait un tour du lac, un tour de la forêt et un tour du volcan avant d'effectuer ce qu'on lui demande.

D'autre fois c'est plus difficile de comparer. Entre deux robots, peut-être qu'un avance super vite mais se baisse très lentement. On voudra alors plutôt l'envoyer dans la presqu'île du lac pour ramasser trois objets d'un coup, plutôt que de faire un chemin moins long mais de se baisser trois fois.

De manière générale entre deux programmes corrects on va naturellement préférer le plus efficace. Suivant les cas l'efficacité peut se mesurer en termes de temps/nombre d'étapes à exécuter, ou d'énergie consommée, ou encore d'espace mémoire utilisé.

3 Etape 2 : Atelier connecté - LightBot

Proposer aux participants de réinvestir leurs connaissances sur une application : LightBot.

Elle ne nécessite **pas de connexion Internet**. Le parcours se fait cette fois sur l'application. Il y a souvent plusieurs solutions pour résoudre chaque niveau. Il peut être intéressant de comparer les solutions trouvées par chacun.

Il est possible de mettre les participants par groupe de deux. Les notions de programmation que l'on retrouve dans LightBot sont :

- **Le contrôle séquentiel**: les commandes (ou instructions) sont exécutées les unes après les autres par le programme.
- **Une procédure** : un bloc de code programmé à l'avance, qui permet de tirer parti de modèles réutilisables et donc de gagner en temps et en efficacité.
- **Une boucle** : une structure de code qui permet de répéter une instruction indéfiniment (souvent tant qu'une condition n'est pas respectée).
- **Déboguer**: exécuter et ré-exécuter un programme, tester des solutions pour corriger des erreurs dans un programme informatique.

4 Variante

On peut utiliser l’activité “Programmation orale” en introduction mais elle se prête moins à une activité extérieure en groupe.

5 Document à imprimer ou reproduire

Le paysage : paysage à reproduire

Les programmes : les programmes à imprimer

6 Pour aller plus loin

Conseil médiation

Pour aller plus plus loin sur le sujet, nous vous conseillons de vous référer à la fiche outil “Initiation à la programmation“