Création avec Scratch d'un jeu de plateau utilisant la méthode des Tuiles

Objectif : Adapter au logiciel Scratch une méthode utilisée par les développeurs pour créer des jeux de plateau (Déplacement dans un labyrinthe, Pac Man, Bomber Man, Petits Chevaux, …). Cette méthode consiste à utiliser un tuilage pour permettre de dessiner le plateau et pour gérer les déplacements sur ce plateau.

Dans ce qui suit, on va expliquer la méthode en prenant un exemple. Le jeu, très simple, que nous allons construire consiste à déplacer un personnage dans un jardin pour lui faire trouver des œufs de Pâques. Pour cela, on procède en deux temps.

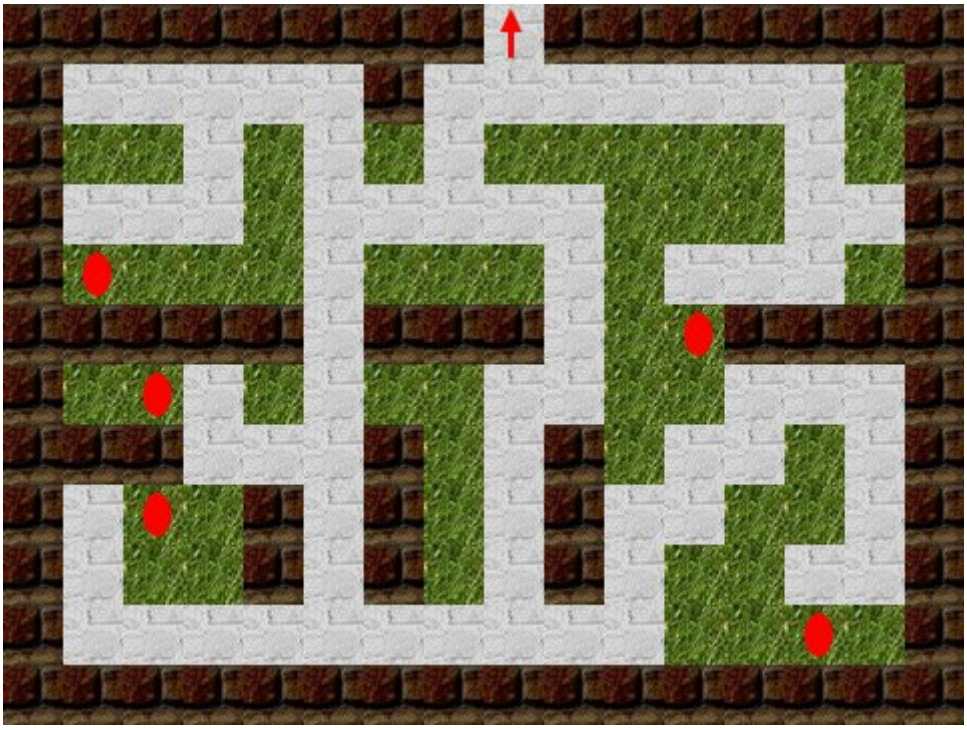
- Partie A : On va créer, avec un fichier tableur et un fichier Scratch, l'image du jardin (la map) qui est en fait un labyrinthe que l'on va construire à l'aide de la méthode des tuiles ;
- Partie B : On va utiliser cette image dans un autre fichier Scratch pour créer le jeu en lui-même qui consiste donc à récupérer les œufs de Pâques placés dans ce jardin.

Logiciels qui seront utilisées dans ce document : Scratch, Photofiltre, le tableur d'Open Office

Partie A : Création de l'image du jardin et du tableau de données le définissant

Le jardin dans lequel va évoluer le personnage sera constitué de murs, d'herbe, d'un chemin, d'œufs et d'une sortie. Il ne pourra se déplacer que sur le chemin et sur les emplacements des œufs. Ce jardin est donc, en fait, un labyrinthe. Dans cette partie, on va détailler une méthode qui permet de construire ce jardin et qui sera valable pour la construction de n'importe qu'elle labyrinthe. Cette méthode permettra de générer une image et une liste de données définissant cette image.

Voici deux copies d'écran de ce que nous allons construire.



La map (le jardin)

Un extrait de la liste Scratch

Dans la liste Scratch contenant les données définissant le jardin, on remarque que les 8 premières valeurs sont des 1 et que si on compte les tuiles dans le sens de lecture habituel (de gauche à droite et de haut en bas), les 8 premières tuiles sont des murs. On constate ensuite que la 9 ième valeur est un 4 et que la neuvième tuile est la Sortie. En fait le chiffre 1 code une tuile mur et le chiffre 4 code la tuile sortie.

1. Organisation du dossier contenant le jeu :

Dans une premier temps, on crée un dossier qui porte le nom du jeu (Jeu_De_Paques) dans lequel, on regroupe tous les fichiers utiles. A l'intérieur, on crée les sous-dossiers suivants :

1_Data : dossier qui contient le fichier tableur qui permet de créer la structure du labyrinthe
2_Tuiles : dossier qui contient les images de toutes les tuiles utilisées dans le jeu
3_AssemblageTuiles : dossier qui contient le fichier Scratch qui va créé l'image du jardin et le tableau de données le définissant.
4_Map : dossier qui contient l'image jpeg du labyrinthe
5_Jeu_De_Paques : dossier qui contient le jeu sous format Scratch

2. Définir les dimensions des tuiles et les dimensions du labyrinthe :

Avant toutes choses, il faut définir, en pixels, les dimensions du labyrinthe et les dimensions des tuiles. On a donc besoin de définir 4 valeurs qui seront attribuées, dans le fichier Scratch, aux 4 variables suivantes :

Larg_Tuile : largeur en pixels d'une tuile
Haut_Tuile : hauteur en pixels d'une tuile
nb_Tuile_Larg : nombre de tuiles en largeur ;
nb_Tuile_Haut : nombre de tuiles en hauteur ;

Les dimensions de la fenêtre Scratch sont 480 pixels en largeur et 360 pixels en hauteur. Ainsi, si on décide que les dimensions en pixels du labyrinthe sont 480×360 , il faut faire en sorte que :

$$Larg\_Tuile \times nb\_Tuile\_Larg = 480 \text{ et } Haut\_Tuile \times nb\_Tuile\_Haut = 360$$

Dans l'exemple qui est présenté dans ce document, les valeurs attribuées aux 4 variables, en pixels, sont les suivantes : *Larg_Tuile*=30 pixels ; *Haut_Tuile*=30 pixels ; *nb_Tuile_Larg*=16 ; *nb_Tuile_Haut*=12

3. Définir les différentes tuiles selon le scénario du jeu :

Dans notre exemple, 5 tuiles ont été utilisées.

Tuile 0 (en vert dans le tableur) : tuile correspondant à de l'herbe sur laquelle il n'est pas autorisé de marcher
Tuile 1 (en marron dans le tableur) : tuile correspondant au mur sur laquelle il n'est pas autorisé de marcher
Tuile 2 (en bleu dans le tableur) : tuile correspondant au chemin sur laquelle il est autorisé de marcher
Tuile 3 (en vert dans le tableur) : tuile correspondant à l'emplacement d'un œuf. On pourra se déplacer sur cette tuile sans y rester.
Tuile 4 (en bleu dans le tableur) : tuile correspondant à l'emplacement de la sortie du labyrinthe. On pourra se déplacer sur cette tuile et, lorsqu'on sera dessus, on sera autorisé à y rester uniquement si tous les œufs ont été récupérés.

On a aussi besoin de créer une sixième tuile qui sera la tuile qui contiendra le personnage. Mais cette construction sera décrite dans la partie B.

4. Dessiner le labyrinthe sur un tableur

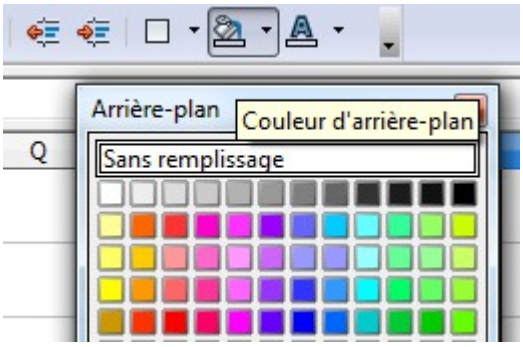
Dans un tableur, on dessine la structure générale du jardin en coloriant les différentes cellules aux couleurs définies plus haut et en attribuant le chiffre correspondant à la tuile (Herbe, Mur, Chemin, Œuf, Sortie...)

Voici le labyrinthe construit avec un tableur (Open Office) :

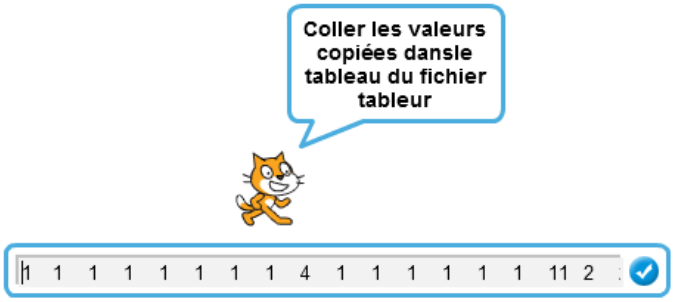
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	1	1	1	1	1	1	1	1	4	1	1	1	1	1	1	1		Description des tuiles :
2	1	2	2	2	2	2	1	2	2	2	2	2	2	2	0	1		0 : Herbe
3	1	0	0	2	0	2	0	2	0	0	0	0	0	2	0	1		1 : Mur
4	1	2	2	2	0	2	2	2	2	2	0	0	0	2	2	1		2 : Chemin
5	1	3	0	0	0	2	0	0	0	2	0	2	2	2	0	1		3 : Objet à récupérer
6	1	1	1	1	1	2	1	1	1	2	0	3	1	1	1	1		4 : Sortie du labyrinthe
7	1	0	3	2	0	2	0	0	2	2	0	0	2	2	2	1		
8	1	1	1	2	2	2	1	0	2	1	0	2	2	0	2	1		
9	1	2	3	0	1	2	1	0	2	1	2	2	0	0	2	1		
10	1	2	0	0	1	2	1	0	2	1	2	0	0	2	2	1		
11	1	2	2	2	2	2	2	2	2	2	2	0	0	3	0	1		
12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		

Aide avec le tableur d'Open Office :

- On modifiera les dimensions des cellules pour qu'elles soient carrées. Pour cela, dans un premier temps, on sélectionne toutes les colonnes, on clique-droit sur l'une des lettres sélectionnées, on choisit « largeur de colonne » et on règle la largeur à 1 cm, par exemple. Dans un deuxième temps, on sélectionne toutes les lignes, on clique-droit sur l'un des nombres sélectionnées, on choisit « hauteur de ligne » et on règle la hauteur à 1 cm.
- Voici comment colorier une cellule



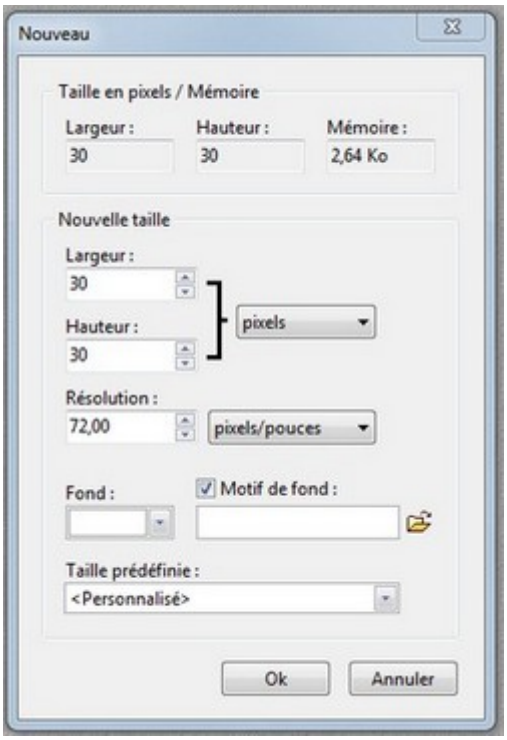
Problème : Lorsqu'on sélectionne à l'aide de la souris tout le tableau contenant le labyrinthe et qu'on le copie-colle dans une variable Scratch, à l'aide de l'instruction « demander » (voir ci-dessous), on obtient une chaîne de caractères contenant une alternance de chiffres d'espaces voir même de double espace. La suppression de ces espaces sera présentée dans l'étape 2 de la partie A.6.

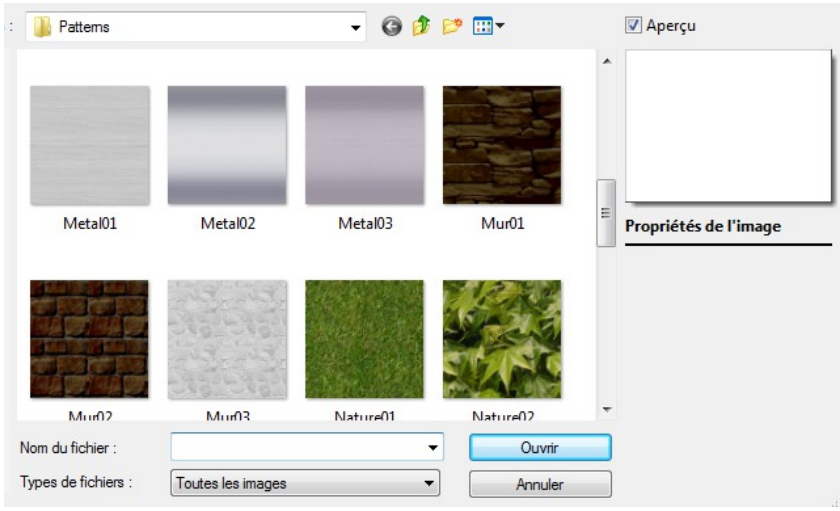


5. Dessin des tuiles avec Photofiltre et enregistrement dans le fichier 2_Tuile

On souhaite maintenant dessiner les 5 tuiles. Pour cela on peut utiliser Photofiltre (ou Paint). Dans le menu Fichier de Photofiltre, choisissez *Nouveau.* Dans la fenêtre nommée *Nouveau* qui s'ouvre, réglez la largeur et la hauteur à 30 pixels, puis cochez la case à cocher *Motif de fond* puis cliquez sur l'icône :

Une deuxième fenêtre s'ouvre alors dans laquelle se trouve un dossier nommé *Patterns* dans lequel vous trouverez différents motifs qui vous permettrons de créer de jolies tuiles.





Il faut sélectionner l'une de ces images et l'enregistrer dans le dossier *2_Tuile* en la nommant de manière explicite.

Dans le labyrinthe que nous allons créer, voici les 5 tuiles qui seront utilisées :

Design de la tuile					
Nom de la tuile et du lutin Scratch	Herbe	Mur	Chemin	Œuf	Sortie
Valeur correspondante dans la liste Scratch	0	1	2	3	4

Nous avons maintenant, enfin, tout ce qu'il nous faut pour commencer à programmer !
Ouvrons donc un fichier Scratch que l'on enregistre dans le dossier 3_AssemblageTuiles.

6. Création du fichier Scratch qui va créer l'image du jardin :

Objectif : Ce fichier doit pouvoir être utilisé pour créer n'importe qu'elle labyrinthe.

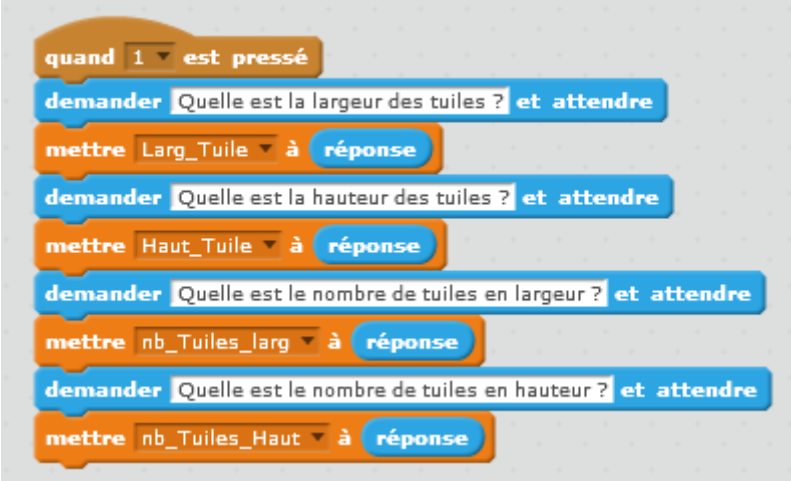
On commence par créer un fichier nommé Jeu_De_Paques_Assemblage_Tuile que l'on enregistre dans le dossier 3_Assemblage_Tuile , puis on procède en 4 étapes.

Etape 1 : *Permettre la saisie des dimensions du labyrinthe*

Dans un premier temps, on crée les variables Scratch suivantes :
Larg_Tuile ; *Haut_Tuile* ; *nbTuile_Larg* ; *nb_Tuile_Haut*.

Dans un deuxième temps, on crée la procédure qui permet à l'utilisateur du fichier d'affecter des valeurs à ces 4 variables en les saisissant au clavier.

Dans l'exemple qui suit, après avoir appuyé sur la touche 1 du clavier, une série de questions sont posées à l'utilisateur qui lui permettent de saisir les dimensions en pixels qu'il souhaite donner aux tuiles et au jardin.



Etape 2 : *Récupération des données contenues dans le fichier tableur et affectation de ces données à une liste Scratch*

On sélectionne, dans le fichier Tableur, le tableau qui contient la structure du jardin (du labyrinthe). On le copie (Ctrl + C) et on le colle (Ctrl + V) dans une variable de Scratch, on obtient une chaîne de caractères. Celle-ci contient les chiffres contenus dans le tableau dans l'ordre de lecture classique mais elle contient aussi des espaces (voir même des doubles espaces) intercalés entre ces différents chiffres. On souhaite supprimer ces espaces et mettre ces chiffres dans une liste. Pour cela, on procède de la manière suivante.

On crée, tout d'abord, une variable nommée *Data* et une liste nommée *Tableau*.

Avant toute chose, on veut permettre à l'utilisateur de pouvoir effacer le contenu de la variable *Tableau* à tout moment.

La procédure qui permet de le faire est présentée ci-contre.

Lorsque l'utilisateur tape sur la touche 2 du clavier, la liste *Tableau* est effacée.

A cette occasion, on crée une variable *Taille_Tableau* à laquelle on attribue la valeur de la longueur de la liste *Tableau*.

Pour mettre dans la liste *Tableau* les chiffres contenus dans le tableau du fichier tableur, on peut procéder comme ci-contre.

L'utilisateur, avant toute chose, tape sur la touche 2 pour vider la liste *Tableau* de tout caractère.

Par un copier-coller, il affecte à la variable *Data* la chaîne de caractères copiée dans le tableau du tableur Open Office.

Toute la chaîne de caractères contenues dans *Data* est lue, dans l'ordre habituel de lecture, caractère par caractère. On rappelle qu'elle est constituée d'une alternance de chiffres, d'espaces et de doubles espaces. On rappelle également qu'un espace est un caractère.

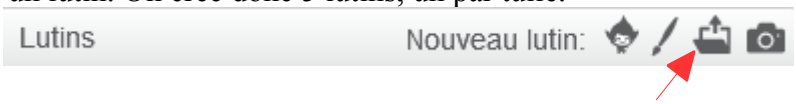
A chaque fois que l'on rencontre un chiffre, on le met, à la suite, dans une nouvelle ligne de la liste *Tableau*.

A la fin de la structure *Répéter*, la liste *Tableau* contient tous les chiffres définissant le labyrinthe.

On a « nettoyé » Data des espaces qu'elle contenait.

Etape 3 : *Création des lutins*

Chacune de 5 tuiles est un lutin. On crée donc 5 lutins, un par tuile.



Pour cela, dans le menu "Nouveau Lutin", on clique sur l'icône "importer lutin depuis un fichier" et on ouvre chacune des images des tuiles contenues dans le dossier 2_Tuiles.

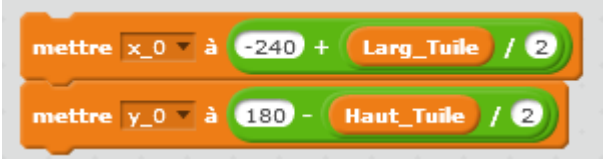
Etape 4 : *Assemblage des tuiles*

Remarques:

- On va placer les tuiles en commençant en haut à gauche et en continuant dans le sens de lecture habituel (de gauche à droite et de bas en haut).
- Les coordonnées d'une tuile sont les coordonnées de son centre. Comme dans notre cas les tuiles sont des carrés de 30 pixels, on va placer la première tuile au point de coordonnées :

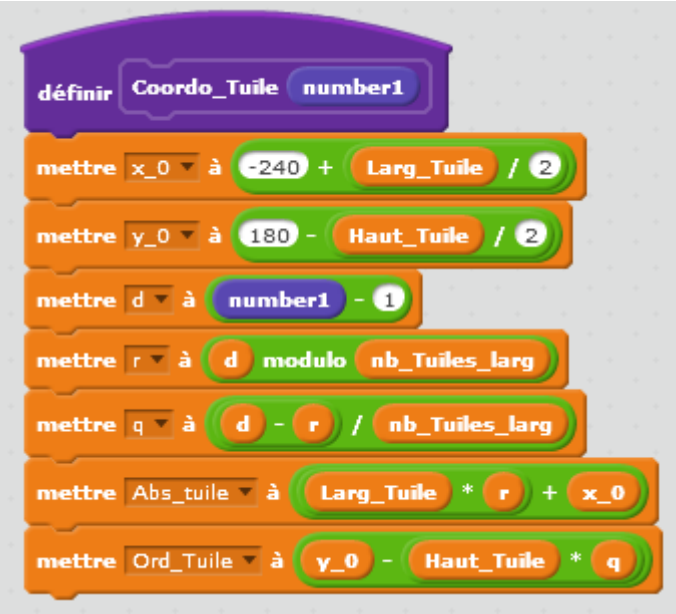
$$(-240+15;180-15)=(-225;165)$$

Dans le cas général, on utilisera les deux variables x_0 et y_0 que l'on affectera des valeurs suivantes :



L'assemblage des tuiles se fait à l'aide de deux procédures et d'une fonction (un bloc en langage Scratch). Elles sont détaillées ci-dessous.

Bloc Scratch :



Commençons par la fonction Coordo_Tuile

On rappelle que la première tuile est celle qui est située en haut à gauche de la fenêtre.
On rappelle aussi que le sens de lecture classique (de gauche à droite et de bas en haut) définit l'ordre des tuiles.

Grâce à la fonction Coordo_Tuile ou plutôt grâce au bloc Coordo_Tuile, on souhaite obtenir les coordonnées dans la fenêtre Scratch d'une tuile connaissant son indice dans la liste *Tableau*.

Ce bloc se trouve ci-contre.

Explications :

a. **x_0** et **y_0** sont les coordonnées de la première tuile.
b. **number1** est le paramètre qui est transmis et qui va être transformé par la fonction (par le bloc). Dans notre cas ce sera le numéro de la tuile dont on cherche les coordonnées.
c. **d** prend la valeur **number1-1**
d. **r** est le reste de la division euclidienne de **d** par le nombre de tuile en largeur. Ainsi **r+1** est le numéro de la colonne dans laquelle se trouve la tuile.
e. **q** est le quotient du reste de la division euclidienne de **d** par le nombre de tuile en largeur. Ainsi **q+1** est le numéro de la ligne dans laquelle se trouve la tuile.
f. **abs_tuile** prend la valeur **larg_Tuile*r+x_0** . C'est l'abscisse du centre de la tuile
g. **ord_tuile** prend la valeur **y_0-Haut_Tuile*q** . C'est l'ordonnée du centre de la tuile

Première procédure : elle permet de dessiner une tuile lorsque le lutin correspondant reçoit le message portant son nom.

On clique sur une tuile, par exemple, la tuile "Chemin".

On code les instructions ci-contre de manière à ce que, lorsque ce lutin "Chemin" reçoit le message "chemin", une tuile "Chemin" soit dessinée au point de coordonnées (*x;y*).
En fait, le lutin va "tamponner" un exemplaire de lui même au point de coordonnées (*x;y*).

On procède de la même manière pour les 4 autres tuiles.

Deuxième procédure : On tamponne les tuiles en utilisant les données de la liste *Tableau*.

On crée 4 variables Scratch *x, y, Abs_Tuile, Ord_Tuile* qui vont permettre de se déplacer dans la fenêtre.

Même si cela n'est pas obligatoire, on initialise ses 4 variables en prenant :
x=x_0 ; y=y_0 ; Abs_Tuile=x_0 et *Ord_Tuile=y_0*

On crée une variable i que l'on initialise à 1. Cette variable sera l'indice de la liste *Tableau*.

On parcourt la liste *Tableau* dans l'ordre et pour la $i^{ème}$ valeur de *Tableau* :

- on affecte à *x* et à *y* les coordonnées de la $i^{ème}$ tuile à placer grâce au bloc nommé Coordo_Tuile qui est présenté ci-dessus.

- on envoie un message au lutin correspondant au $i^{ème}$ élément de la liste *Tableau*.

Exemple : si le $18^{i ème}$ élément de la liste *Tableau* est un 2, on envoie à tous les lutins le message "chemin" ce qui a pour conséquence, compte tenu de ce que l'on a décrit pour la première procédure, que le lutin "chemin" est tamponné (dessiné) au point de coordonnées (*x;y*).

Comme on répète ceci pour toutes les valeurs contenues dans la liste *Tableau*, on obtient l'image du jardin dans la fenêtre.

7. Récupération de l'image du labyrinthe et de la liste *Tableau* :

- Récupération de l'image du jardin

A l'aide d'un clique droite sur la fenêtre dans laquelle se trouve le labyrinthe, on copie l'image (« save of stage ») dans le dossier 4_Map après l'avoir nommée Jardin.jpeg.

- Récupération de la liste Scratch nommée *Tableau :*

Pour pouvoir utiliser les valeurs contenues dans la variable Scratch dans un autre fichier, il suffit de faire une copie de ce fichier, de l'ouvrir et de supprimer tout ce qui n'est pas utile pour la construction du jeu.

Remarque : on peut aussi prévoir de mettre dans le fichier Tableur les valeurs des 4 variables : *Larg_Tuiles* ; *Haut_Tuile* ; *nb_Tuile_Larg ; nb_Tuile_Haut.*

Partie B :

Dans cette partie, on souhaite créer, avec le logiciel Scratch, un jeu de chasse aux œufs (Jeu de Pâques). Le personnage se déplace, à l'aide des flèches du clavier dans le jardin créé partie A et récupère des œufs. Lorsqu'il les a tous trouvés, il peut sortir du jardin.

On réutilise l'image du jardin et le tableau de données qui ont été créés dans la Partie A.

1. Réutilisation du fichier Jeu_De_Paque_AssemblageTuile

Pour gagner du temps et pour ne pas multiplier les blocs inutiles, on peut faire une copie du fichier Jeu_De_Paque_Assemblage_Tuile et supprimer ce qui ne nous servira pas. On garde donc uniquement :

les 3 tuiles Oeuf, Chemin, Herbe et leurs blocs

le bloc Coordo_Tuile que l'on attribuera à l'arrière plan

la plupart des variables....mais surtout la liste *Tableau* contenant toutes les données définissant le tuilage.

2. Placer la map en arrière plan

Cliquez sur l'icone qui permet "d'importer un nouvel arrière plan dans un fichier" puis ouvrez l'image du jardin créée dans la partie A que vous avez enregistrée dans le dossier 4_Map et nommée Jardin.

3. Création de la tuile du personnage

On laisse quelques instants le fichier Scratch de côté et on ouvre le logiciel Photofiltre pour créer le chasseur d'œufs. Pour cela, on prend l'image d'un personnage sur Internet. On active la transparence et on l'enregistre au format PNG.

Dans notre cas, les dimensions de l'image contenant ce personnage devront être 30×30 pixels.

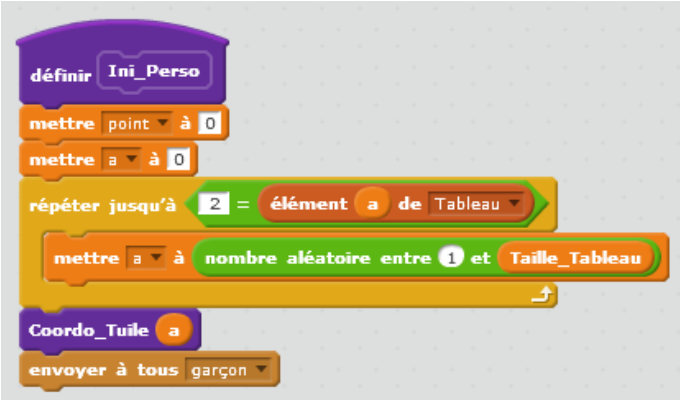
Nommons, par exemple, cette image Garçon.png

4. Initialisation de la position du personnage

Au lancement du jeu, on souhaite placer de manière aléatoire le personnage sur le chemin. On rappelle que les tuiles Chemin sont codées par le chiffre 2.

Ainsi, parmi toutes les lignes de la liste *Tableau* qui contiennent un 2, on veut en choisir une de manière aléatoire.
Pour cela, on choisir au hasard des lignes de la liste *Tableau* jusqu'à ce qu'on obtienne une ligne contenant un 2. On calcule les coordonnées, dans la fenêtre scratch, de la tuile correspondante puis on dessine à cet emplacement, par dessus, la tuile Garçon.

Plus précisément, lorsque l'on clique sur le drapeau vert, on appelle une procédure, le bloc Ini_Perso, dont une image figure ci-contre et qui est décrite ci-dessous.



La variable *Point*, qui compte le nombre d'œufs récupérés, est initialisée à 0.
La variable *a* qui désigne les numéros des lignes de la liste *Tableau*, est aussi initialisée à 0 de manière à permettre l'entrée dans la structure répétitive qui suit. Dans cette structure répétitive, on donne à la variable a des numéros de lignes aléatoires. On ne sort de cette structure répétitive que si l'on a trouvé dans la liste *Tableau* une ligne qui contient un 2 (un tuile chemin).

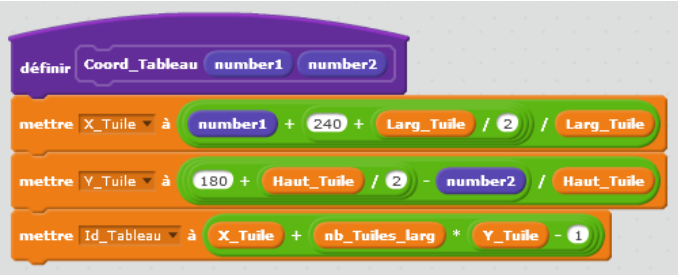
Lorsqu'on a trouvé une telle ligne, le numéro de cette ligne est portée par la variable *a*. Pour obtenir les coordonnées de la $a^{ième}$ tuile dans la fenêtre, on utilise la fonction Coord_Tuile qui a déjà été présentée dans la Partie A.

Dans la fonction Coord_Tuile, les coordonnées de cette tuile sont attribuées aux variables *x* et *y*.
On envoie ensuite un message à la tuile *Garçon* lui demandant de tamponner une copie de *Garçon* au point de coordonnées (x;y), c'est à dire à l'emplacement de la $a^{ième}$ tuile .

5. Une procédure intermédiaire

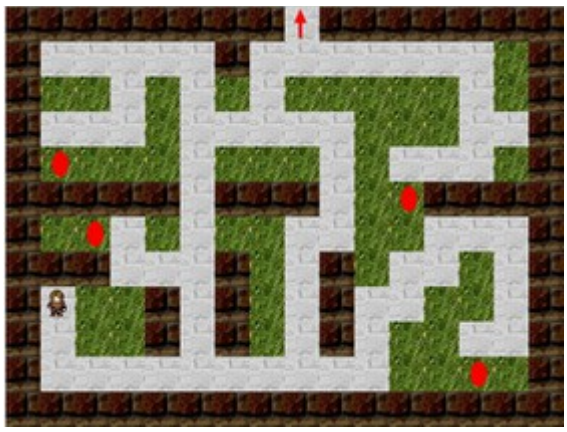
Pour permettre le déplacement du personnage, on a besoin de créer une fonction qui calcule le numéro de la tuile connaissant ses coordonnées dans la fenêtre scratch. C'est le bloc Coord_Tableau décrit ci-dessous.

Les coordonnées, dans la fenêtre, du centre de la tuile scratch sont passées en paramètre :
number1 et number2

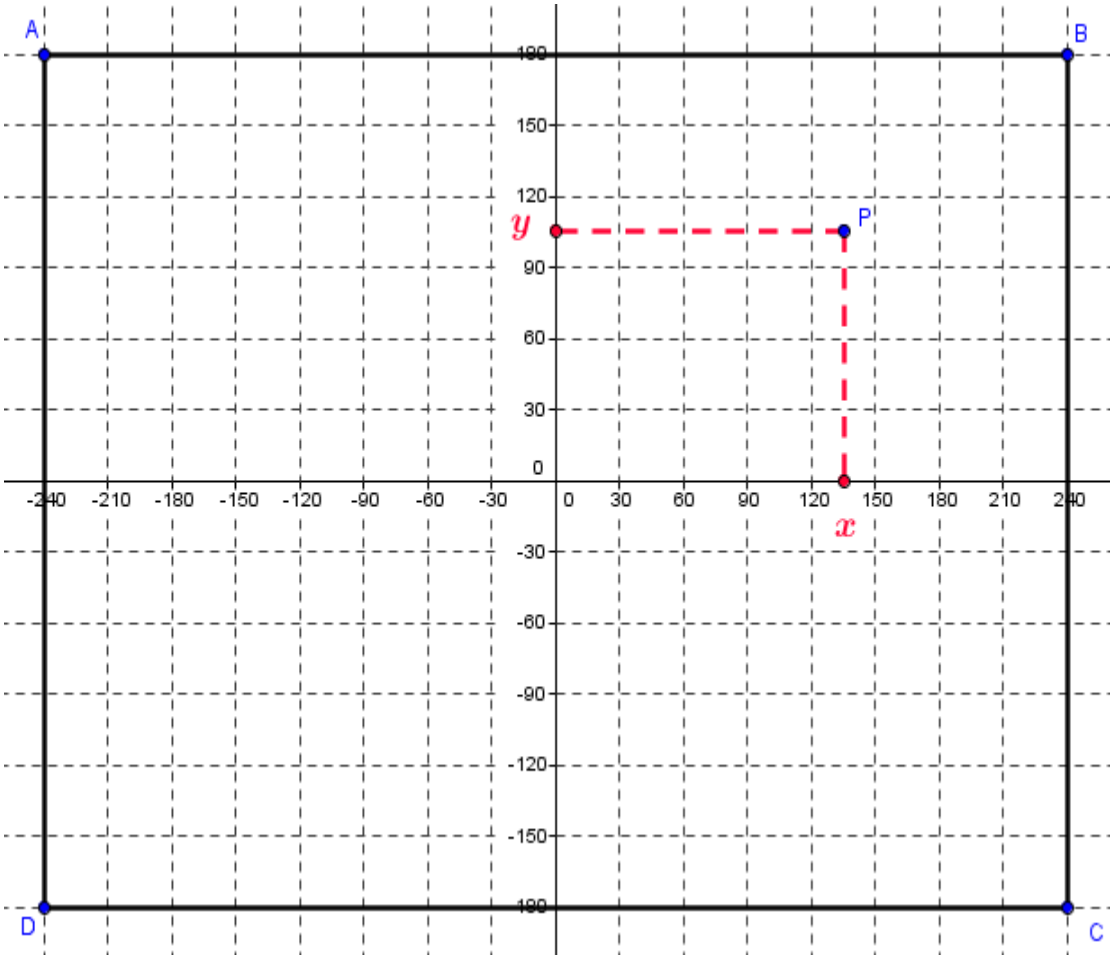


On calcule ensuite, dans le tableau de tuiles, le numéro de la colonne dans laquelle se trouve la tuile. Puis, on calcule le numéro de la ligne dans laquelle elle se trouve. Enfin, on calcule le numéro de la tuile dans le tableau de tuile et on l'attribue à la variable Id_Tableau

Exemple : dans la situation de l'image ci-contre la tuile sur laquelle se trouve le Garçon a pour coordonnées (-195;-75). Elle se trouve dans la 2ième colonne et dans la neuvième ligne du tableau de tuile.



Explications des calculs figurant dans le bloc :



Dans l'exemple ci-dessus, le point P a pour coordonnées (135;105). Il est dans la 13ième colonne (en partant de la droite) et dans la 3ième ligne (en partant du haut).

Le numéro de la colonne est égal à $\frac{x_P+15-(-240)}{30}$. Ce numéro de colonne est égal à la distance entre, le bord droit de la tuile et le bord gauche de la fenêtre, divisé par la largeur d'une tuile.

Le numéro de la ligne est égal à $\frac{180-(y_P-15)}{30}$. Ce numéro de ligne est égal à la distance entre, le bord supérieur de la fenêtre et le bord inférieur de la tuile, divisé par la hauteur d'une tuile

Pour calculer le numéro de la colonne désigné par la variable X_Tuile, il suffit d'effectuer le calcul suivant :

$$X\_tuile=\frac{x+\frac{Larg\_Tuile}{2}-(-240)}{Larg\_Tuile}$$

$$\text{donc } X\_tuile=\frac{x+\frac{Larg\_Tuile}{2}+240}{Larg\_Tuile}$$

Pour calculer le numéro de la ligne désigné par la variable Y_Tuile, il suffit d'effectuer le calcul suivant :

$$Y\_tuile=\frac{180-\left(y-\frac{Haut\_Tuile}{2}\right)}{Haut\_Tuile}$$

$$\text{donc } Y\_tuile=\frac{180+\frac{Haut\_Tuile}{2}-y}{Haut\_Tuile}$$

Pour calculer le numéro de la tuile, connaissant le numéro de la colonne et de la ligne de cette tuile dans le tableau des tuiles, il suffit de compter les lignes complètes avant cette tuile. Il y en a :

nb_Tuile_Larg*(Y_Tuile-1)

et d'ajouter le nombres des tuiles sur la ligne incomplète. Il y en a X_Tuile

Finalement le numéro de la tuile est égal à : X_Tuile+nb_Tuile_Larg*(Y_Tuile-1)

6. Déplacements du personnage et récupération des œufs

Il faut maintenant déplacer le personnage et lui permettre de récupérer les 5 œufs.

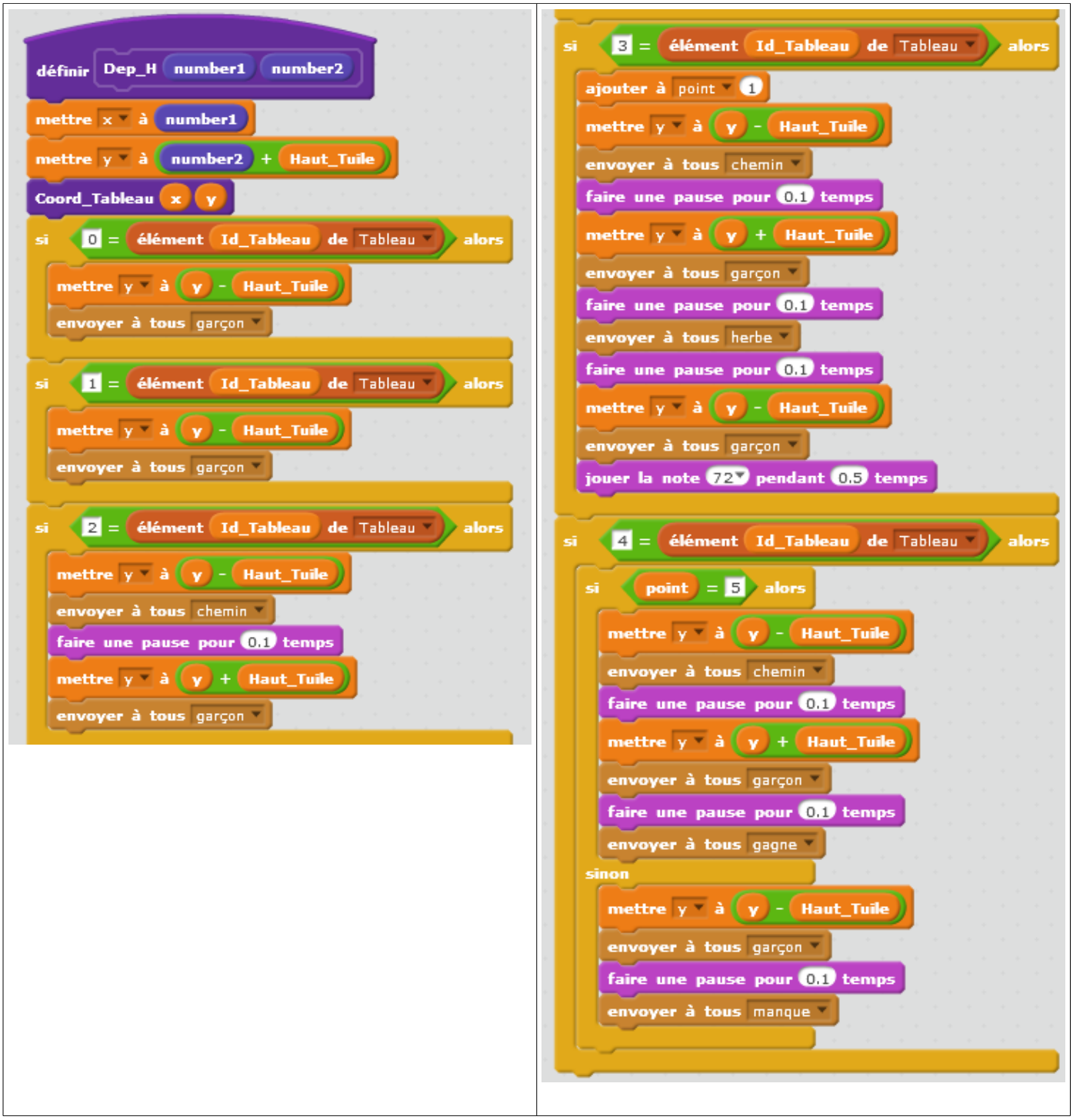
Pour le déplacer dans les 4 directions, on crée les 4 procédures suivantes :

Chacune d'entre elle est activée, si on appuie sur une des flèches du clavier.
Détaillons, par exemple, le déplacement vers le haut. Si on appuie sur la flèche du haut, la procédure Dep_H est activée avec comme paramètres (*x*;*y*) les coordonnées du lutin Garçon.

On veut tester si on peut se déplacer sur la tuile qui se trouve au dessus de la tuile sur laquelle se trouve le Garçon. Pour cela, on calcule le numéro de la tuile correspondant à la tuile du dessus, c'est à dire à la tuile de coordonnées (x;y+haut_Tuile) en utilisant la procédure Coord_Tableau qui a été présentée ci-dessus.

Quatre cas se produisent alors :
- si c'est une tuile Mur ou une tuile Herbe (0 ou 1) , alors on dessine la tuile garçon en (*x;y*) c'est à dire que le lutin Garçon ne bouge pas
- si c'est une tuile Chemin (2), alors on dessine en (*x;y*) un lutin chemin (pour effacer le garçon) puis on dessine en (*x;y+haut_Tuile*) un lutin Garçon.
- si c'est une tuile Œuf (3), alors on donne successivement les instructions suivantes :
  on incrémente de 1 la variable Point qui compte le nombre d'œufs ramassés ;
  on dessine un lutin chemin en (*x;y*)
  on dessine un lutin garçon par dessus le lutin Œuf donc en (x;y+haut_Tuile)
  on dessine par dessus un lutin herbe
  on dessine un lutin Garçon en (x;y)
- si c'est une tuile Sortie (4), alors
  si la variable Point est égale à 5, c'est à dire si tous les œufs sont récupérés, alors on dessine le lutin Chemin en (*x;y*), on dessine le lutin Garçon en (*x;y+haut_Tuile*) c'est à dire sur la tuile *Sortie* et on envoie le message Gagne qui permet au lutin Garçon d'émettre le son correspondant à la victoire et de faire apparaître 3 secondes le message « vous avez gagné » sur un nouveau Background.
  si le nombre de Point est inférieur strictement à 5, alors on dessine le lutin garçon en (*x;y*) et on lui envoie le message « manque » qui déclenche le son correspondant à l'échec.

Dans ce qui suit le bloc de droite et la suite du bloc de gauche.



<u>Remarques</u> : Les valeurs des dimensions des tuiles et du labyrinthe auraient pu aussi être mises dans le tableur et extraite par Scratch pour être affectées aux variables :

Larg_Tuile ; *Haut_Tuile* ; *nbTuile_Larg* ; *nb_Tuile_Haut*