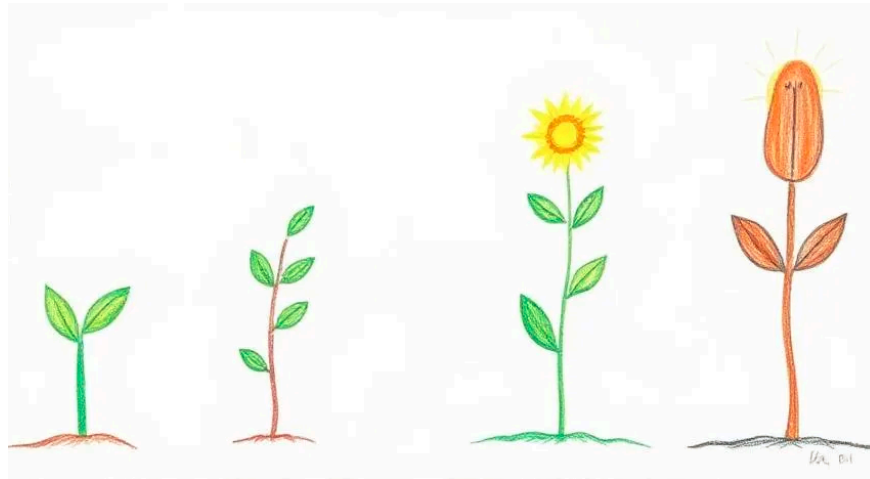


Plantagotchi



Introduction

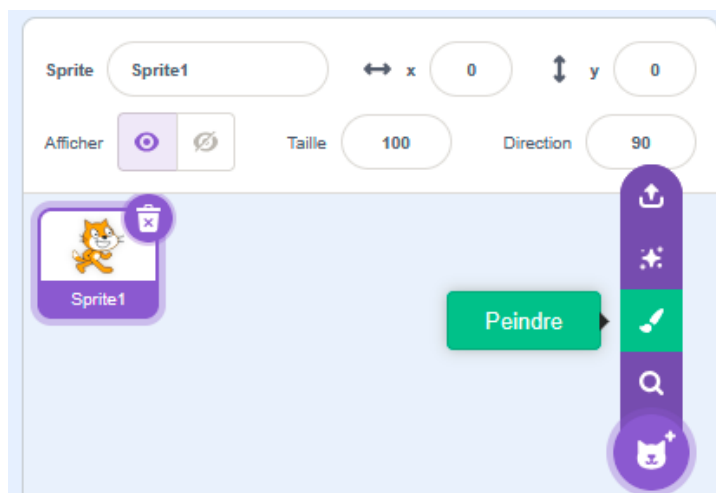
Nous allons créer un jeu de plantes de compagnie, dans le même style que les petits Tamagotchi, mais avec une plante !

Le principe de ce jeu c'est d'avoir une créature qui grandit.

Il faut donc avoir un "sprite" avec plusieurs "costumes", un costume par étape de croissance de notre plante.

1) Créer vos sprites:

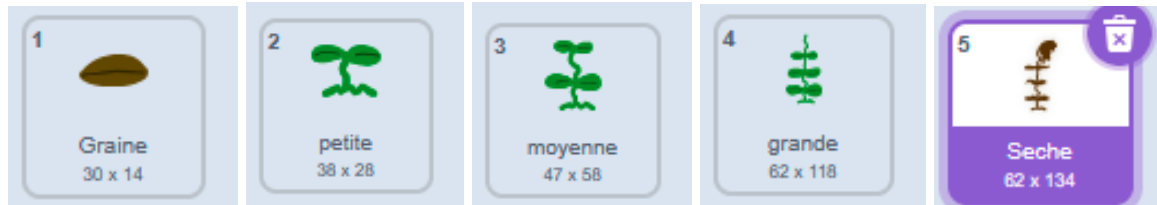
Dans la section des sprites, cliquez sur ajouter un sprite (le chat avec un signe +), puis sur "Peindre"



1.1) La plante

J'ai ajouté un nouveau Sprite que j'ai appelé "Plante".

Sur le canvas, vous pouvez dessiner une plante simple pour le premier costume, puis le dupliquer pour ajouter des feuilles. J'ai fait 5 costumes pour 5 stades de croissance, dont un correspond au stade graine, et le dernier stade c'est la plante sèche.



1.2) La fleur

La fleur peut être dessinée directement sur le stade "grande" de la plante, mais j'ai préféré créer un nouveau sprite pour pouvoir l'animer. Je l'ai appelé "fleur".

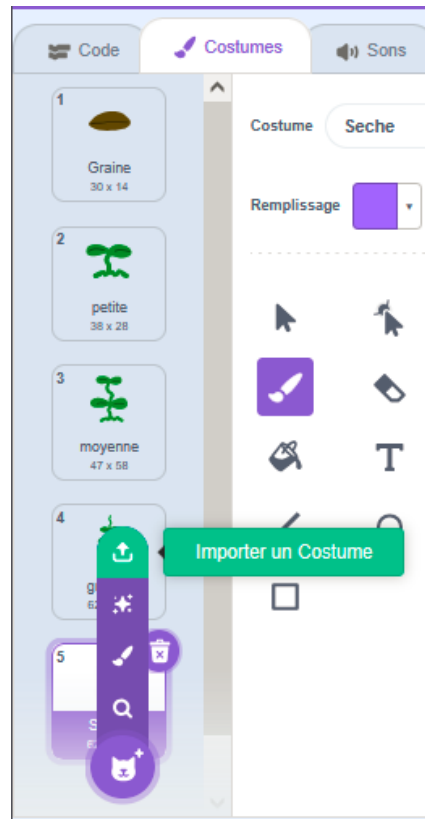
Sur le canvas, vous pouvez dessiner une fleur. J'ai décidé de lui ajouter un visage pour que la plante soit plus attachante



1.3) Importer des images en tant que costumes

Si vous n'aimez pas dessiner, une alternative c'est d'importer les images que vous souhaitez utiliser dans l'onglet costume, après avoir créé un nouveau sprite.

[Vous pouvez trouver mes images dans le lien: comment on partage des images?](#)



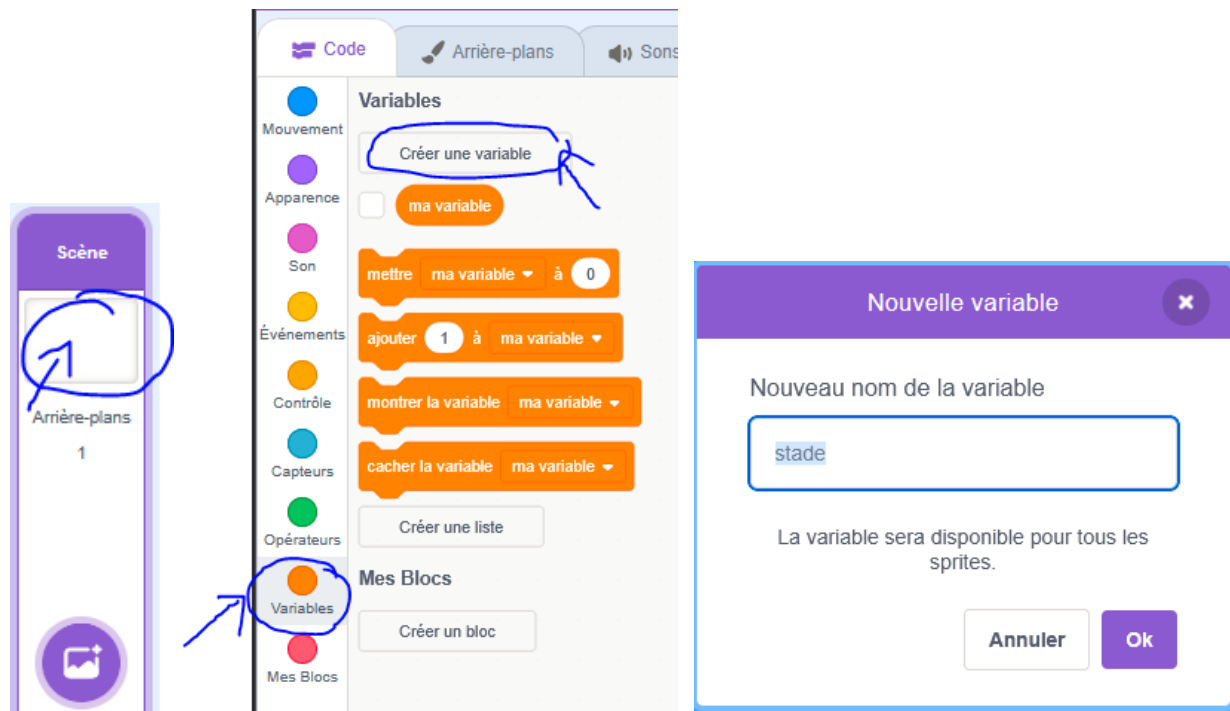
2) Animer la croissance de la plante

Maintenant que nous avons les graphiques nécessaires, passons à la programmation !
Code = Comportement !

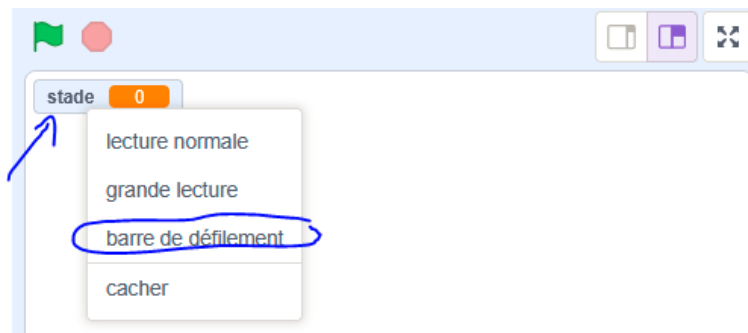
2.1) Ajouter une variable

Rappelons nous que le principe de ce jeu c'est d'avoir une créature qui grandit.
Il faut commencer par créer une variable pour connaître le stade de croissance de notre "Plantagotchi".

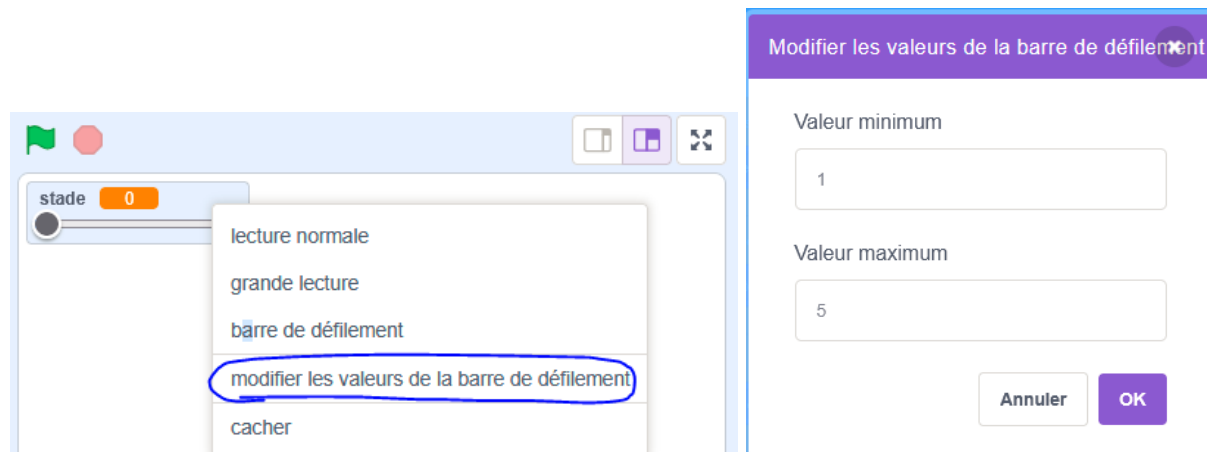
Nous allons cliquer sur l'arrière plan, puis dans la catégorie "variable" -> créer une nouvelle variable. Nous allons l'appeler "stade"



Notre nouvelle variable apparaît en haut de l'écran du jeu. Avec click droit, nous allons changer l'affichage en barre de défilement pour pouvoir mieux la contrôler.



Puis, nous allons modifier les valeurs pour aller de 1 à 5

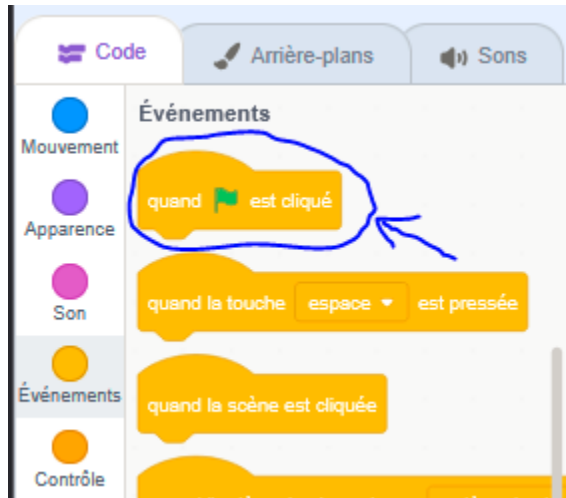


2.2) Démarrer le code

Pour faire cela, on clique dans le sprite de la plante, puis on ajoute le code qui la contrôle.

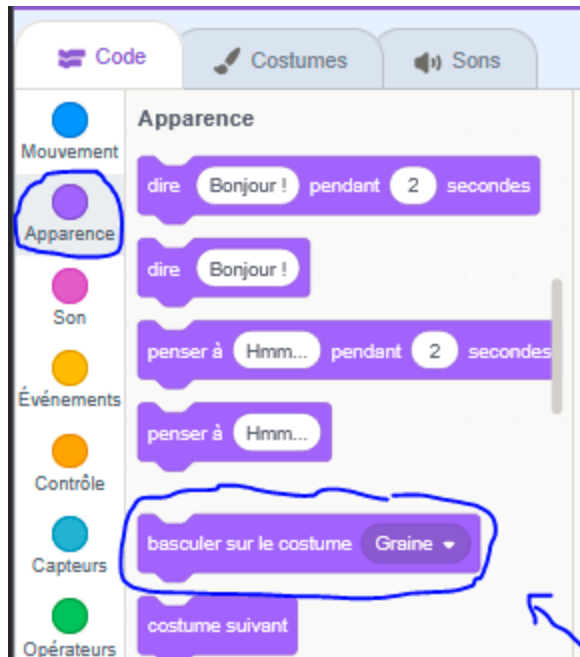
Le code commence toujours par l'événement qui le déclenche. Dans notre cas, nous voulons que la plante change d'état à partir du moment où notre jeu démarre.

Dans la catégorie "événements", nous ajoutons l'événement "quand le drapeau vert est cliqué".

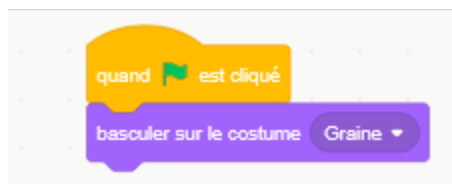


2.3) Changer l'apparence

Ensuite, nous voulons changer l'apparence de la plante. Dans la catégorie "Apparence", on prend la brique "basculer sur le costume Graine".



Notre code ressemble a ca:



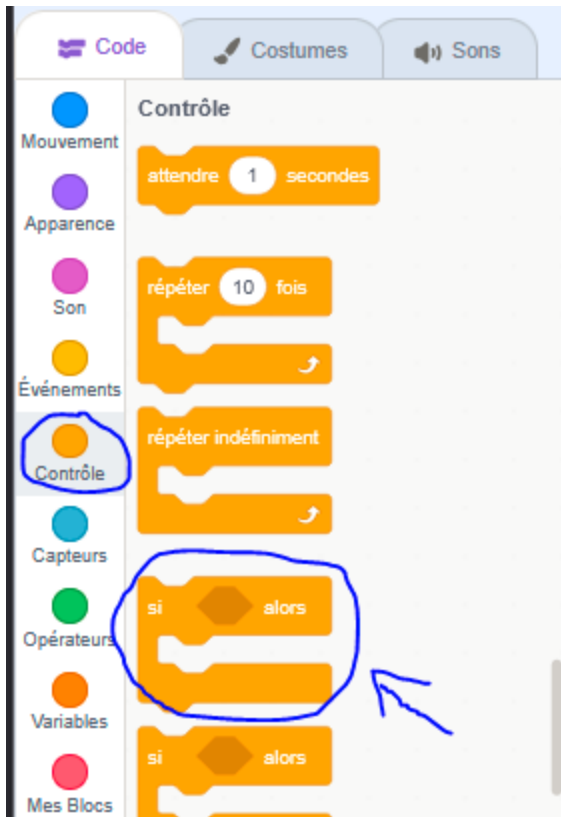
On clique sur le drapeau vert pour l'exécuter !

Comportement observé: peu importe le stade de la plante dans la fenêtre du jeu, quand on clique sur le drapeau vert, la plante devient une graine. Mais rien d'autre ne se passe. Même si on bouge la valeur du stade.

Ce dont nous avons besoin c'est que l'ordre de "basculer sur le costume Graine" soit conditionnée par la valeur de la variable stade.

2.4) Contrôle conditionnel

On va utiliser le bloc de "Contrôle" "Si quelque-chose, alors on bascule sur le costume...". Le "quelque-chose" s'appelle "condition" et nous allons le rajouter après.



Voici à quoi cela ressemble:



Il ne reste plus qu'à remplir la valeur de la "condition". Pour cela il faut réfléchir dans quel cas nous voulons basculer sur le costume Graine?

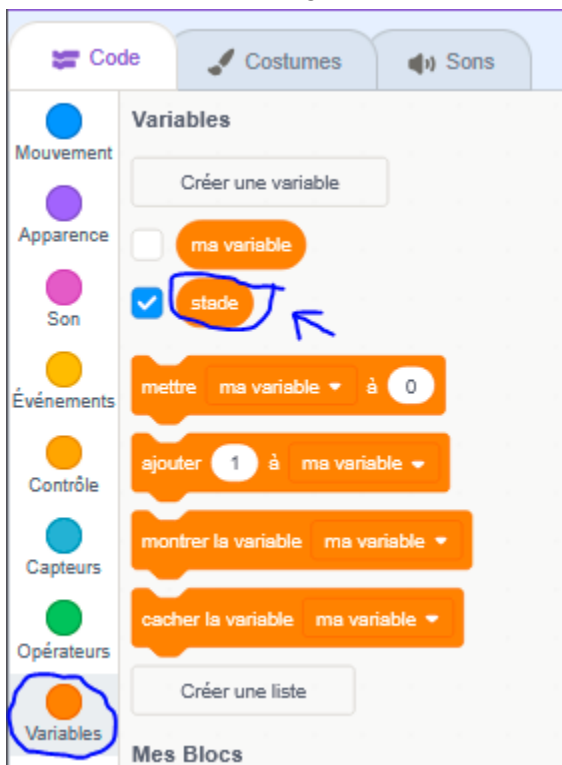
Alors, la condition serait que la variable "stade" soit 1. Pour comparer la variable stade, nous utiliserons un "opérateur"

2.5) Operateur "égal à"

Dans la catégorie opérateur, on veut comparer si la valeur est exactement égale à 1.



On revient dans la catégorie “variable” et on glisse “stade” dans l’emplacement blanc



Notre code ressemble à ça maintenant:








Cet opérateur va comparer la variable stade avec le chiffre 50, et si stade est 50, alors il va exécuter le code à l'intérieur ("basculer sur le costume Graine").

Quelle valeur faut-il mettre à la place de 50?

Réponse:

Ce que nous voulons obtenir comme comportement, c'est une animation qui montre le costume de la graine quand nous sommes au stade 1, puis la petite plante au stade 2, ainsi de suite jusqu'à la plante adulte au stade 4, puis la plante sèche au stade 5.

				
graine	petite	moyenne	grande	sèche
Stade = 1	Stade = 2	Stade = 3	Stade = 4	Stade = 5

Donc nous avons besoin de comparer stade a la valeur 1 pour le costume "Graine"

graine	petite	moyenne	grande	sèche
Stade = 1	Stade = 2	Stade = 3	Stade = 4	Stade = 5

Maintenant, c'est à toi de jouer :

Enchaîne à la suite autant de blocs “si ..., alors ...” que nécessaire pour chaque valeur de la variable stade.

N'oublie pas d'ajuster le bon costume pour la bonne valeur !

Voici à quoi doit ressembler ton code a la fin :



Super ! On peut cliquer sur le drapeau vert de l'écran pour exécuter notre code et voir si la plante évolue quand on bouge la valeur de la variable stade.

Oh non! Ça marche seulement une fois, selon l'état de la variable quand on clique sur le drapeau !

Que pouvons-nous ajouter pour vérifier en continu l'état de la variable et réagir tout le temps? (et pas seulement une fois au début ?)



Piste: dans la catégorie “Contrôle”

2.6) Répéter indéfiniment

Solution:



Et maintenant on peut vérifier le fonctionnement à nouveau en appuyant sur le drapeau vert. Chouette ! Quand la variable stade est 1, on voit bien la graine, puis la plante évolue pour les autres valeurs de stade.

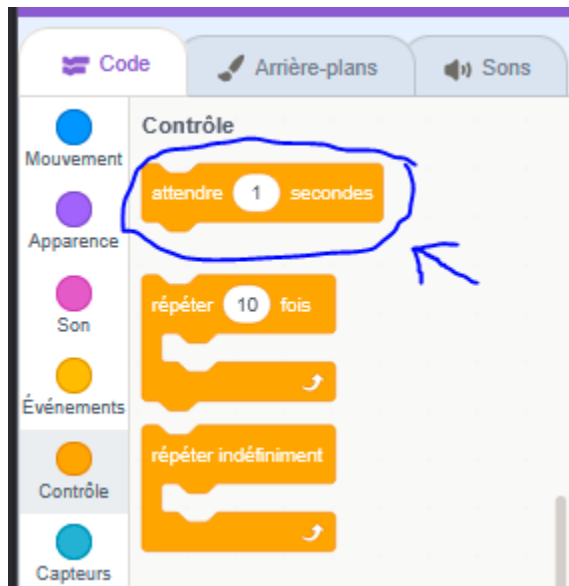
3) Ajouter le passage du temps

Maintenant que la plante change de costume selon la valeur de la variable “stade”, nous aimerions faire évoluer le stade de croissance avec le passage du temps.

3.1) Savoir attendre =)

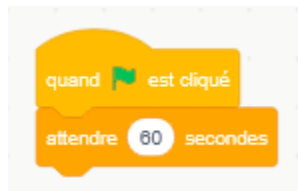
Nous pouvons imaginer que la plante grandit un stade chaque minute.

Le bloc qui permet d'attendre se trouve dans la catégorie "Contrôle".



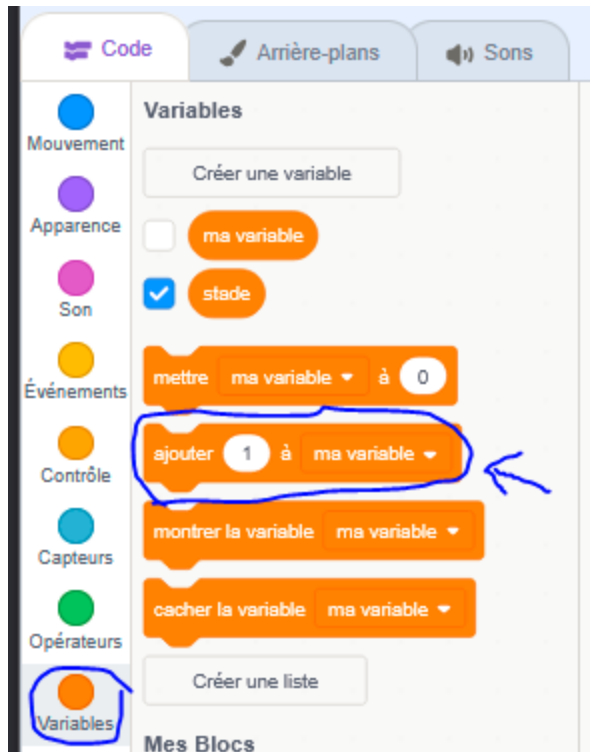
Ajoutons ce bloc dans l'arrière-plan et on change la valeur à 60 secondes (une minute). N'oubliez pas d'ajouter "quand le drapeau vert est cliqué", pour que le temps commence à couler quand le programme commence.

Vous pouvez ajuster le temps à votre convenance, mais si vous avez choisi 60 secondes, votre code de l'arrière plan ressemble à ça :



3.2) Grandir la plante

Le comportement que nous voulons obtenir c'est: une fois passés 60 secondes, le stade évolue. Pour cela nous allons dans la catégorie variable et le bloc "ajouter 1 a ma variable" semble pertinent pour ça:



Notre code ressemble à ça:



Alors, chaque fois qu'on ajoute un comportement, on peut vérifier si ça fait ce qu'on veut.

Petit coup de drapeau vert et...

C'est trop long pour tester facilement... on va réduire le temps à 5 secondes juste pendant les tests !

Petit coup de drapeau vert et...

Top ! la plate grandit d'un cran !!

Mais seulement d'un cran... que devons nous ajouter pour que cela se passe tout le temps?

Piste: nous l'avons déjà fait dans 2.6 =)

Réponse: le code ressemble a ca:



Petit coup de drapeau vert :

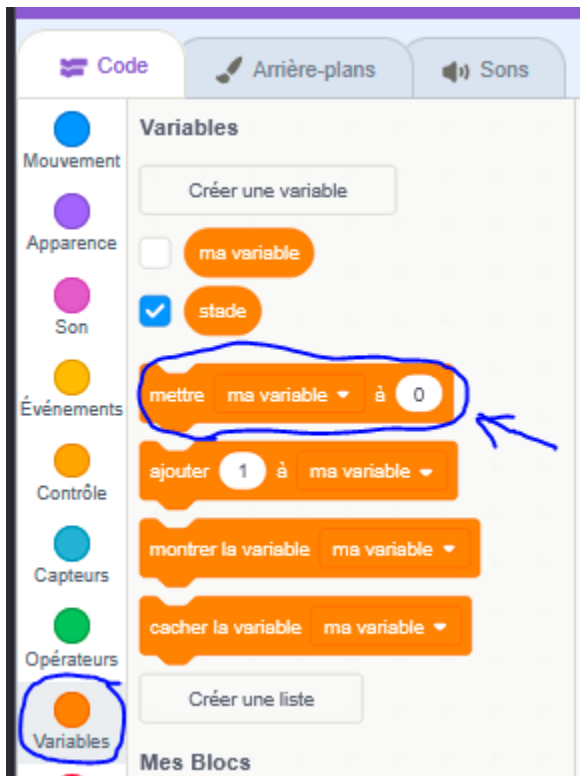
On constate 2 problèmes :

- La plante commence dans le stade où elle était
- La variable stade grandit indéfiniment, tant qu'on n'arrête pas l'exécution et dépasse notre valeur maximale.

3.3) Initialiser la variable

On a besoin de remettre le stade à sa valeur initiale au début.

Pour cela nous allons utiliser le bloc "mettre ma variable à 0" dans la catégorie "variables"



Mais, où l'ajouter? Et aussi, est-ce que la valeur 0 nous convient?

Si tu n'as pas oublié de mettre la bonne variable, la bonne valeur et le tout au bon endroit, ton code devrait ressembler à ça:



Est-ce que ça marche? -> Drapeau Vert -> yes !!

Mais il nous reste à corriger le problème de l'incrément de stade à l'infini

3.4) Savoir quand s'arrêter !

Comportement que nous voulons corriger:



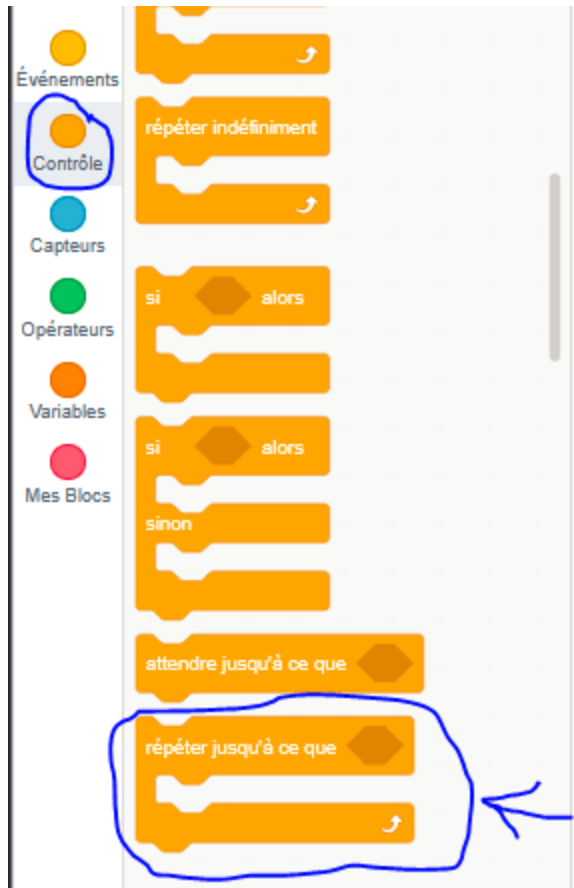
Il y a souvent plusieurs facon de faire les choses.

Voici deux possibilités:

- Si on dépasse 5, on remet 5 dans la variable
- On arrête de répéter indéfiniment l'ajout quand le stade est 5

Pour la première idée, tu connais déjà tous les blocs nécessaires.

Pour la deuxième possibilité, nous allons remplacer le bloque "répéter indéfiniment" par "répéter jusqu'à ce que"



Le code:



Quand il y a un trou en losange c'est parce qu'il manque une condition...

Nous avons fait cela dans 2.4.

Quelle est la condition que nous devons ajouter ici?

Réponse : nous voulons répéter jusqu'à ce que stade = 5.

Le code :



Le drapeau vert pour vérifier si le comportement obtenu correspond au comportement que nous voulons -> comportement observé:

- on commence par la graine,
- avec le passage de temps, la plante grandit,
- puis elle meurt et la variable stade s'arrête a la bonne valeur (5).

4) Ajouter des besoins

Dans les Tamagotchis, la créature grandit avec le temps. Mais il faut s'occuper d'elle, satisfaire ses besoins, pour qu'elle puisse arriver au stade adulte.

4.1) Arroser la plante

Nous pouvons ajouter un besoin évident pour une plante : de l'eau.

Traçons d'abord le chemin à parcourir pour ajouter ce besoin.

Avons-nous besoin de sprites?

Des variables?

Quelles comportements voulons nous obtenir?

Quel est le défi ajouté pour l'utilisateur du jeu?

4.1.1) Plan d'action proposé pour l'eau

Nous pouvons ajouter un spirit "goute"

Nous pouvons ajouter une variable "eau"

Nous pouvons ajouter une visualisation du niveau d'eau dans l'écran de jeu

Comportement proposé :

Quand on clique sur la goutte, on augmente le niveau d'eau (on arrose la plante).

La plante consomme une goutte d'eau tous les 10 secondes.

Si la plante est à 0 eau, elle meurt (stade 5, game over).
Si la plante est à 5 eau, elle meurt aussi (stade 5, game over)

Défi: Le joueur doit maintenir le niveau d'eau de 1 à 4 pour que la plante grandisse correctement

Tu as déjà découvert et utilisé tous les blocs dont tu as besoin.

Nous restons à disposition pour toute question ou pour donner un petit coup de main.
Cette section n'est pas obligatoire si tu ne souhaites pas ajouter ce besoin pour ton plantagotchi.

4.2) Des besoins similaires

D'autres besoins que la plante peut avoir, comme de l'engrais ou de la lumière peuvent être ajoutés.

A toi de proposer un plan d'action et d'essayer !
Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Tous ces exemples, comme l'eau, représentent des besoins permanents qui se consomment avec le temps. Dans la comparaison avec un Tamagotchi, ces besoins sont équivalents à la nourriture ou le jeu.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter des besoins pour ton plantagotchi.

5) Ajouter des dangers

Dans les Tamagotchis, la créature grandit avec le temps, et même si on satisfait ses besoins permanents, il y a des besoins qui peuvent arriver d'un coup, de façon "événement". Ces besoins sont équivalents au caca ou la maladie, et demandent une attention de l'utilisateur pour que la plante puisse arriver au stade adulte.

5.1) Danger = escargots !

Un autre besoin possible c'est de surveiller la plante et d'enlever les escargots à proximité (qui essayent de la manger).

A différence de l'eau, l'engrais ou le soleil. Les escargots peuvent apparaître de façon aléatoire et se diriger vers la plante avec le temps.

Traçons d'abord le chemin à parcourir pour ajouter ce besoin.

Avons-nous besoin de sprites?

Des variables?

Quelles comportements voulons nous obtenir?

Quel est le défi ajouté pour l'utilisateur du jeu?

5.1.1) Plan d'action proposé pour les escargots

Nous pouvons ajouter un spirit "escargot"

Comportement proposé :

On attend un temps aleatoire et on rend visible le sprite de l'escargot a une position aleatoire.

TODO: voir les blocs nécessaires

Tous les secondes, l'escargot marche vers la plante

TODO: voir les blocs nécessaires

Quand on clique sur l'escargot, il disparaît en faissant un son (ouch !)

Si l'escargot arrive sur la plante, elle meurt

Défi: Le joueur doit cliquer sur l'escargot avant qu'il n'arrive pas sur la plante, cela demande de surveiller, même si tous les besoins de la plante sont comblés.

Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter ce besoin pour ton plantagotchi.

5.2) Des besoins similaires

D'autres elements peuvent endommager une plante et demandent une action protectrice événementielle: un ballon, les petits frères, les oiseaux, d'autres plantes qui poussent à côté et qui boivent l'eau 3 fois plus vite...

A toi de proposer un plan d'action et d'essayer !

Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter des besoins pour ton plantagotchi.

5) Notifications

Dans notre jeu, la plante meurt assez facilement sans que l'utilisateur en soit conscient. Nous pourrions prévenir l'utilisateur que la plante a un besoin avant qu'elle ne meure.

5.1) Plan d'action proposé pour les notifications

Comme d'hab, on trace d'abord le plan pour ajouter des notifications.

Avons-nous besoin des variables?

Quelles comportements voulons nous obtenir?

Quel est le bénéfice ajouté pour l'utilisateur du jeu?

Est-ce qu'on lui laisse la possibilité de désactiver le son?

6) Ajouter de la vie

Tous les jeux aujourd'hui bougent en permanence. Quand un visuel est statique, nous avons tendance à penser que le jeu est figé. Il y a donc un intérêt à ajouter du mouvement à notre plante pour donner un effet "vivant" agréable à l'utilisateur.

6.1) Ajouter du mouvement

Nous pouvons utiliser la réflexion horizontale de nos sprites plante pour créer deux costumes par stade et les basculer par intermittence pour donner du mouvement à notre plante.

TODO: blocs nécessaires?

6.2) La fleur, des émotions

Nous pouvons ajouter un sprite fleur, qui se montre seulement à l'estade adulte (dernier stade avant mourir, et qui donne quelque chose en plus pour motiver à l'utilisateur à arriver à ce stade.

6.2.1) Animer l'apparition de la fleur

Le comportement souhaité c'est que la fleur se montre dans le stade 4, en haut de la tige. Ensuite elle doit disparaître pour les autres stades.

Tu as déjà découvert et utilisé tous les blocs dont tu as besoin pour faire cela.

Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter une fleur pour ton plantagotchi.

6.2.2) Ajouter un effet aléatoire à l'apparence de la fleur adulte

Dans les Tamagotchis, la forme adulte peut varier. Cela donne une petite motivation au joueur à s'occuper de cet animal virtuel pour voir en quoi il va se transformer chaque fois. Puis, une petite déception quand il n'arrive pas au stade adulte.

Nous pourrions ajouter cet effet très facilement en rendant aléatoire la couleur de la fleur quand la plante passe au stade 4. Une autre façon de donner cet effet "diverse" c'est d'ajouter différents types de fleur et en choisir un de façon aléatoire, mais cela est plus difficile.

Ajouter les blocs nécessaires pour faire le changement de couleurs aléatoires au début du stade 4

Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter des couleurs différentes pour ton plantagotchi.

6.2.3) Ajouter les émotions de la fleur

Dans le sprite, nous pouvons ajouter un costume par émotion, puis nous pouvons afficher

- La joie quand les besoins sont satisfaits
- La peur quand des dangers sont présents
- La tristesse ou la colère quand la plante est sur le point de mourir
- Tout ce dont tu pourrais imaginer et avoir envie...

Tu as déjà découvert et utilisé tous les blocs dont tu as besoin pour faire cela.

Nous restons à disposition pour toute question ou pour donner un petit coup de main.

Cette section n'est pas obligatoire si tu ne souhaites pas ajouter des émotions pour ton plantagotchi.

7) Merci

Kidikod vous remercie d'avoir partagé avec nous ce coding goûter, votre temps, vos créations, et le goûter, miam !