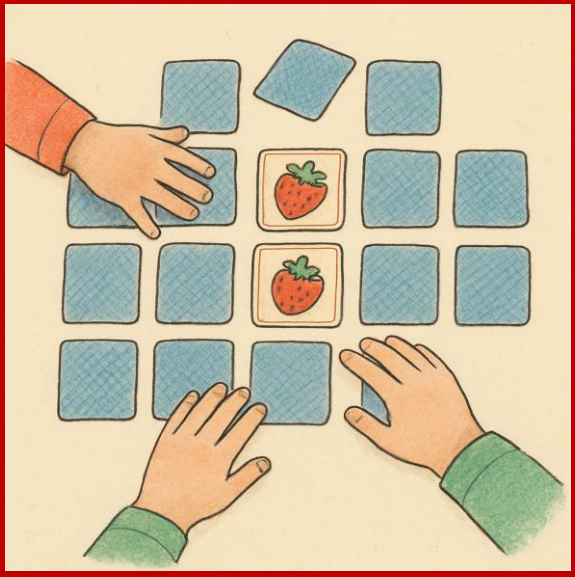
4 Un jeu de Memory



Programmer un jeu de Memory, voilà un beau challenge !

On va utiliser le clonage, des listes et encore bien d'autres techniques de programmation.

Mais ici encore, nous nous efforçons de te proposer une approche simple et logique.

CONSIGNES ET SOLUTIONS

Les consignes

Créer un jeu de mémory

- Un seul joueur
- 8 paires de cartes
- Au démarrage, les cartes sont mélangées et apparaissent retournées
- Le joueur clique sur deux cartes qui révèlent une image
    - Si elles sont identiques, elles disparaissent, sinon, elles se retournent
- Le but est de retourner toutes les paires en le moins d'essais possible.

Vidéo

https://jeunesingenieux.be/vid-s3-4

Proposition de solution

https://scratch.mit.edu/projects/1192700869

Tu te lances en mode Défi ?

Voici une proposition de méthode.

1. Placer 8 paires d'images sur la scène en deux rangées en clonant un sprite doté de 8 costumes.
2. Identifier chaque clone à l'aide du nom de son costume.
3. Choisir deux cartes et enregistrer leurs identifiants.
4. Comparer les identifiants. S'ils sont identiques, les cartes disparaissent, sinon, effacer les identifiants enregistrés.
5. Mélanger les cartes…
6. Retourner les cartes…

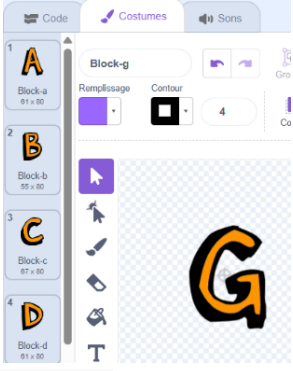
En manque d'inspiration pour aller plus loin?
Il y a quelques idées en fin de ce jeu et ce projet :

https://scratch.mit.edu/projects/1188881835/

4.1 Placer 8 paires de cartes

- Crée un sprite doté de 8 costumes

Nous prenons ici 8 lettres, de A à H.



- Place 8 x 2 clones sur deux rangées

Le résultat devrait ressembler à ceci :

Crée un bloc personnalisé ***Placement des cartes***

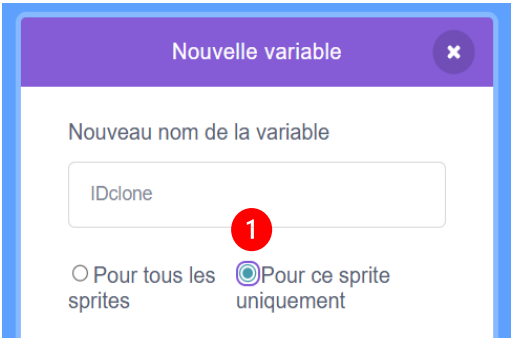
Remplace le script de démarrage par ces deux ci.

Cela rendra le code plus lisible et facilitera les futures améliorations.

4.2 IDENTIFIE LES CLONES

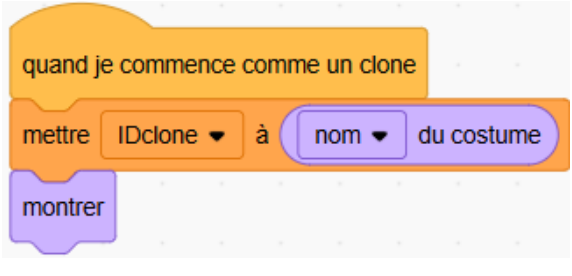
Identifie les clones

Crée une nouvelle variable ***IDclone*** en cochant la case ❶***pour ce sprite uniquement***.



Ce faisant, chaque clone aura sa variable ***IDclone*** et pourra donc être identifié dans les procédures suivantes.

Utilise le nom du costume comme identifiant



Avec ce code, tu utilises le nom du costume comme identifiant.

Note que, dans ce cas, pour chaque paire de clones (de cartes), les noms seront identiques. Deux cartes s'appelleront ***Block-a***, deux autres ***Block-b***, etc

4.3 Choisir deux cartes

À ce stade, les cartes sont dévoilées. On s'occupera de cacher les lettres plus loin.
La procédure suivante permet de cliquer successivement sur deux cartes et d'enregistrer leurs identités pour pouvoir ensuite les comparer.

Crée deux variables publiques : *choix1* et *choix2*

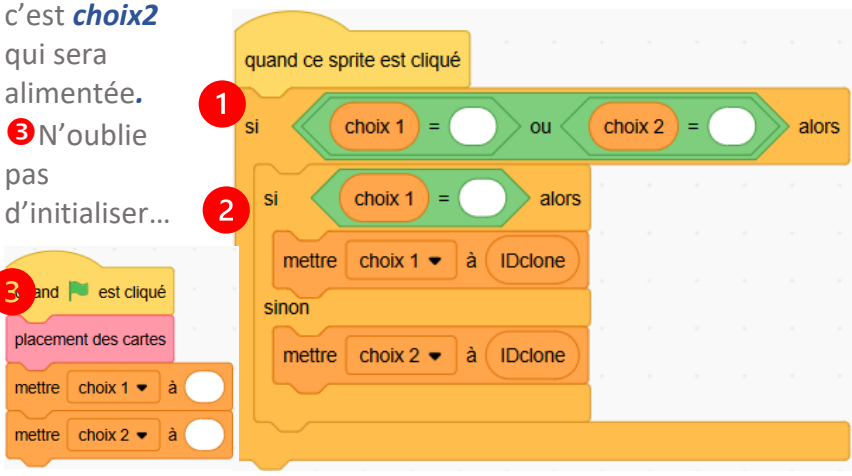
Attention : cocher la case ***pour tous les sprites*** lors de la création des variables !

Ajoute ces instructions

❶ : si au moins une des deux variables est vide

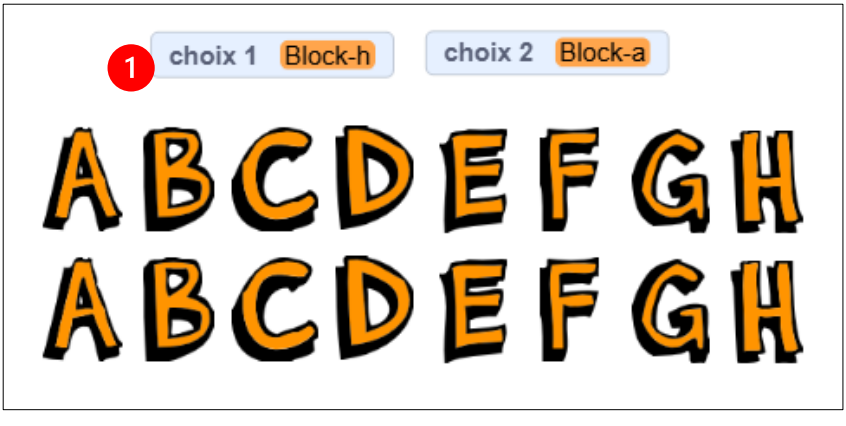
❷ : si ***choix1*** est disponible (vide), enregistre l'***IDclone*** sinon, c'est ***choix2*** qui sera alimentée.

❸ N'oublie pas d'initialiser…



Teste ton code

Fais apparaître les deux variables ***choix1*** et ***choix2*** sur la scène et lance ton programme.
Les lettres s'affichent.
Clique maintenant successivement sur deux cartes et les noms des clones cliqués doivent apparaître ❶ dans les champs des variables.



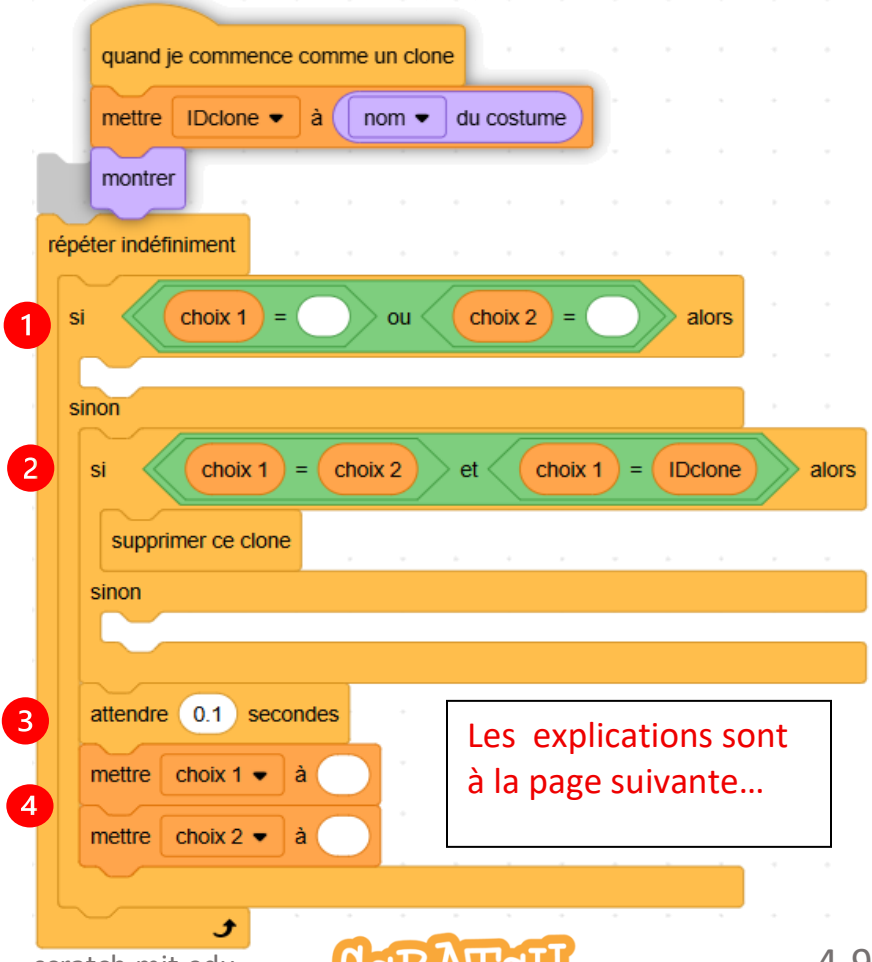
Maintenant que les identifiants des deux clones cliqués sont capturés, on va pouvoir les comparer et :

- Faire disparaître les cartes si elles sont identiques
- Remettre les variables ***choix1*** et ***choix*** 2 à « » (vide…)

Et recommencer…

Ceci est l'étape cruciale. Prends le temps de bien comprendre la méthode utilisée.

- Ajoute ce code au script ***quand je commence…***



Les explications sont à la page suivante…

Quelques explications

Rappelons que cette procédure va être exécutée par tous les clones simultanément !

❶ Avec ce premier test, tant que au moins un des deux choix sera vide, il ne se passera rien. Il faudra donc que le joueur clique sur deux cartes pour passer au ***sinon***…

❷ dès que cette condition sera rencontrée, le clone sera supprimé et disparaîtra.

Avant la suppression, tu peux ajouter une petite pause de 1 seconde.
Le résultat de ***sinon*** reste vide mais on s'en servira plus loin…

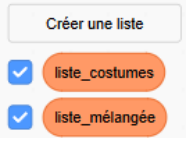
❸ une petite pause est indispensable pour laisser le temps au deuxième clone cliqué de vérifier qu'il est également concerné avant de réinitialiser les deux variables ***choix1*** et ***choix2.***
Essaie le code sans cette petite pause, observe bien le comportement des variables ***choix1*** et ***choix2*** et tu comprendras ce qu'il se passe***…***

❹ Une fois la comparaison terminée, le variables sont réinitialisées et on peut recommencer un tirage…

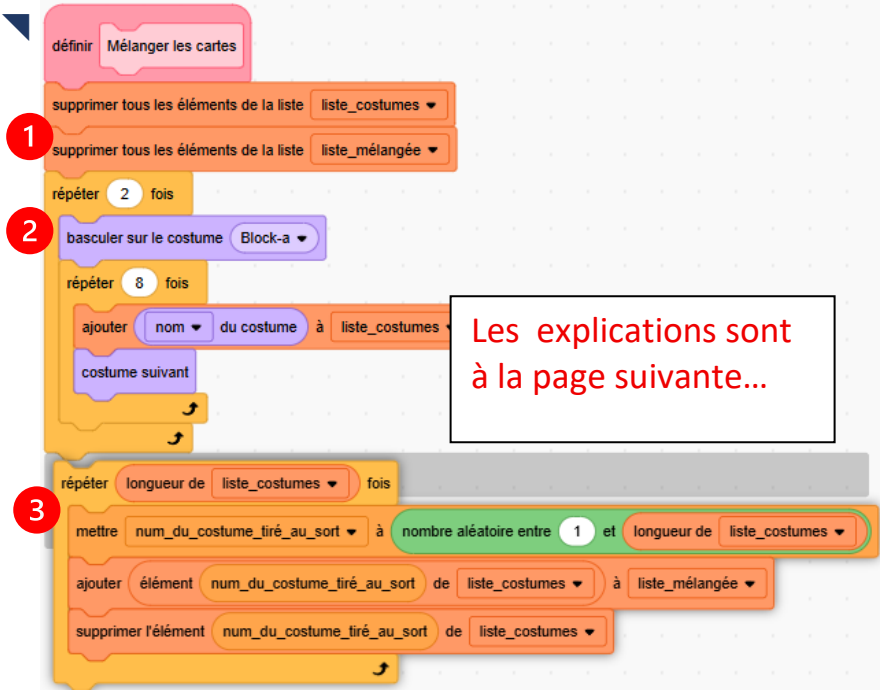
4.5 MÉLANGER LES CARTES

Restent deux étapes… mélanger les cartes et les retourner…
La procédure de mélange des cartes est liée à un bloc personnalisé.

- Crée un bloc personnalisé « ***mélanger les cartes*** » et deux listes : ***liste_costumes*** et ***liste_mélangée***



- Le code pour mélanger les cartes :

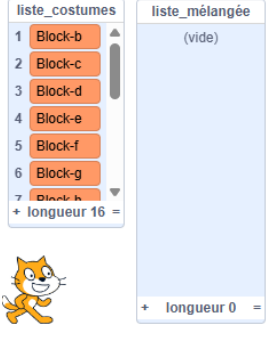


Quelques explications

Cette procédure sera exécutée qu'une fois en début de partie.

❶ Par précaution, on vide les deux listes au départ.

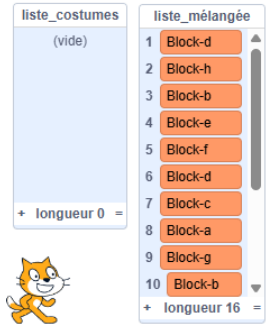
❷ À deux reprises, on copie le nom des 8 costumes dans la ***liste_costumes***, pour obtenir 8 paires d'identifiants.
Avant de passer à la procédure ❸, affiche les deux listes et teste le code.



❸ Ajoute cette procédure.
À 16 reprises, on va

- choisir aléatoirement un numéro compris entre 1 et la longueur de la liste,
- recopier l'élément correspondant de la liste dans la ***liste_mélangée***
- et supprimer cet élément de la ***liste_costumes***

Ajoute une petite pause dans la boucle ❸ et teste la procédure. Tu devrais voir la liste_costumes se vider progressivement et alimenter la liste_mélangée.



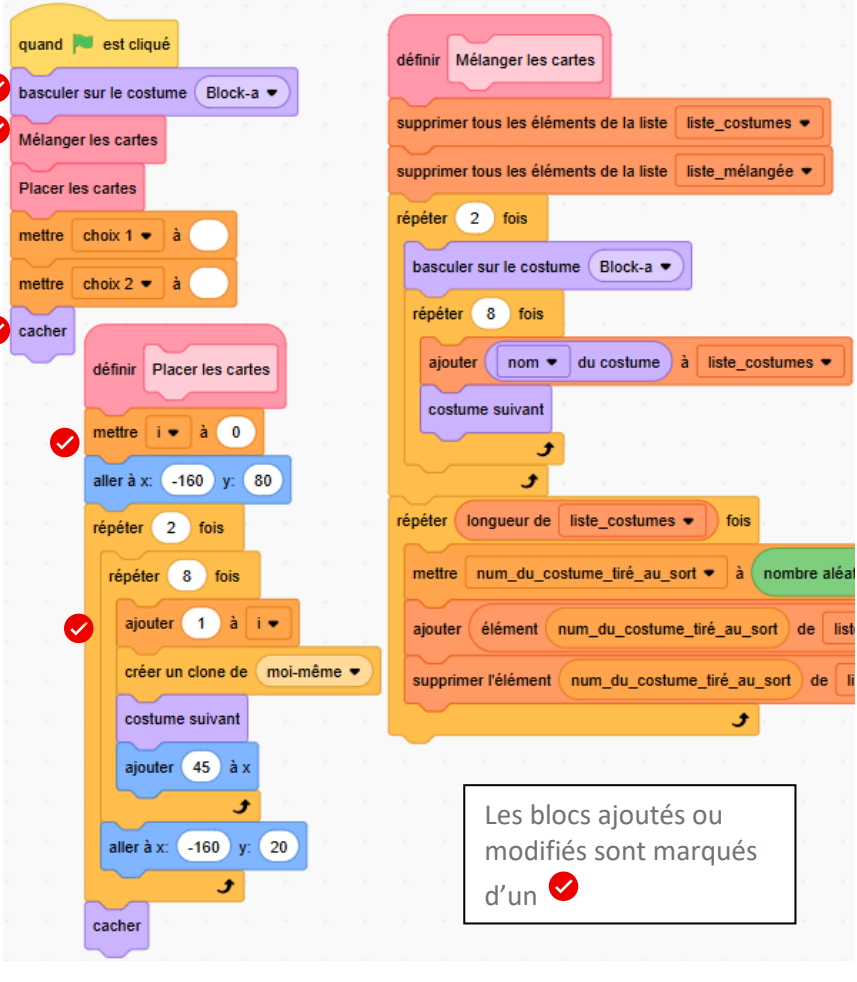
Il reste à adapter les scripts pour utiliser cette liste mélangée au moment de la création de clones. Nous le faisons à la page suivante.

6 Mélanger et Retourner les cartes

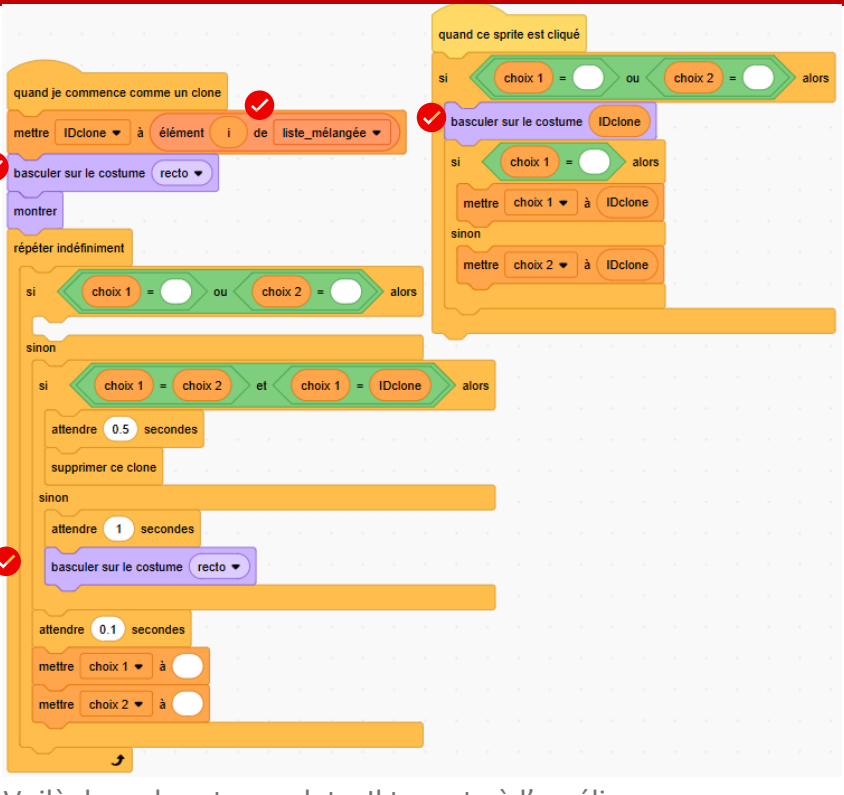
Les cartes sont invisibles au départ du jeu, se retournent quand on clique dessus ou, si elles ne forment pas une paire.

Crée un costume ***recto,*** crée la variable compteur ***i***

Sur cette page et la suivante se trouve le programme complet.



Les blocs ajoutés ou modifiés sont marqués d'un ✔



Voilà, le code est complet… Il te reste à l'améliorer :

- Compter le nombre d'essais,
- Créer et intégrer plusieurs séries de cartes,
- Programmer un jeu pour deux ou trois joueurs.

Par exemple comme ceci :
https://scratch.mit.edu/projects/1188881835