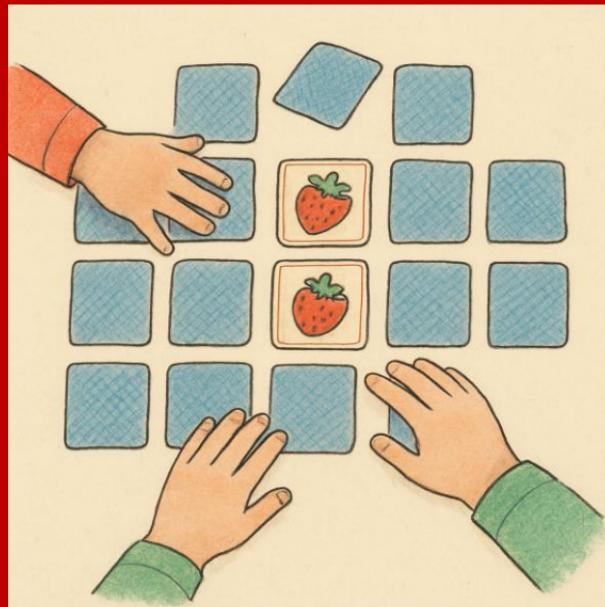




## 4 Un jeu de Memory



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
  - o 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>



## EN MODE DÉFI

### ■ 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/>

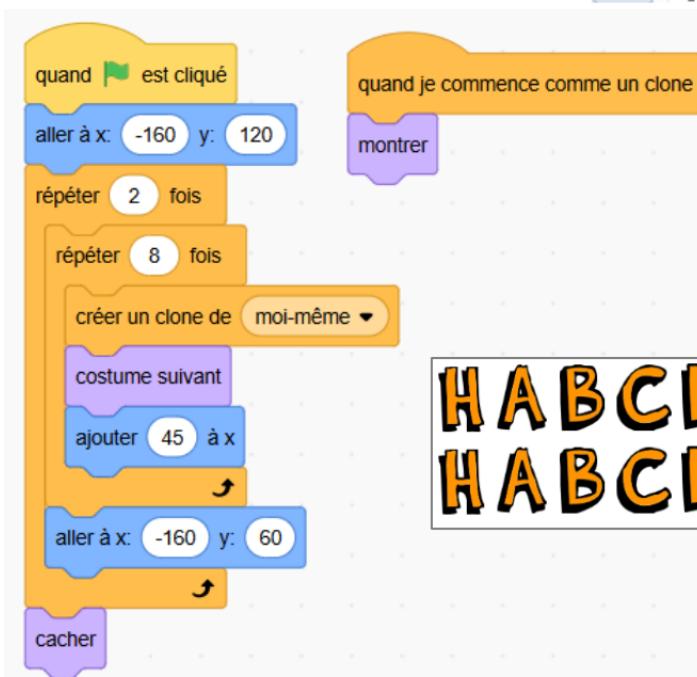


## 4.1 PLACER 8 PAIRES DE CARTES

- 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 :





# PLACER 8 PAIRES DE CARTES

## ▼ 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.



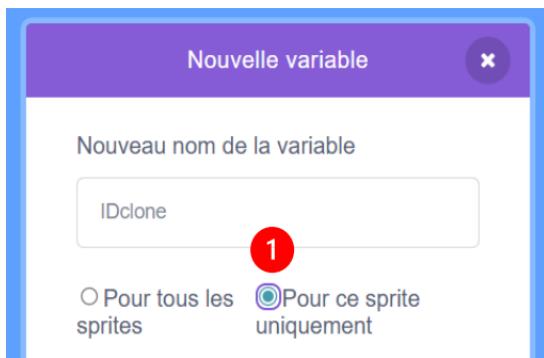


## 4.2 IDENTIFIE LES CLONES

### Identifie les clones

Crée une nouvelle variable **IDclone** en cochant la case

**1 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.

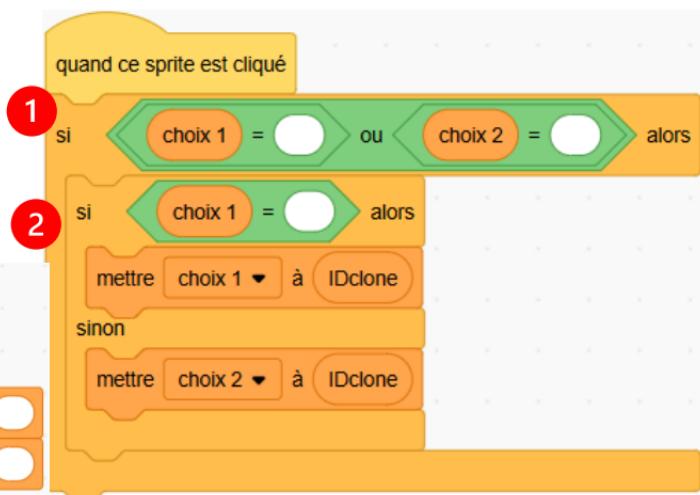
### ► Crée deux variables publiques : *choix1* et *choix2*

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

### ► Ajoute ces instructions

- ① : si au moins une des deux variables est vide
- ② : si *choix1* est disponible (vide), enregistre l'*IDclone* sinon, c'est *choix2*

qui sera alimentée.  
③ N'oublie pas d'initialiser...





## CHOISIR DEUX CARTES

### ► 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 ① dans les champs des variables.

```
choice1 Block-h
choice2 Block-a
```

A B C D E F G H  
A B C D E F G H

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 **choix2** à « » (vide...)

Et recommencer...



## 4.4 COMPARER LES IDENTIFIANTS

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

► Ajoute ce code au script *quand je commence...*

The Scratch script starts with a **quand je commence comme un clone** hat. It contains the following steps:

1. **si** **choix 1** = **choix 2** **ou** **choix 1** = **IDclone** **alors**
2. **sinon**
  - si** **choix 1** = **choix 2** **et** **choix 1** = **IDclone** **alors**
    - supprimer ce clone**
    - sinon**
3. **attendre** **0.1** **secondes**
4. **mettre** **choix 1** **à** **nom** **du costume**
5. **mettre** **choix 2** **à** **nom** **du costume**

Les explications sont  
à la page suivante...



# COMPARER LES IDENTIFIANTS

## ► Quelques explications

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

- ① 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**...
- ② 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...

- ③ 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...

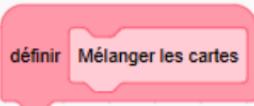
- ④ Une fois la comparaison terminée, les variables sont réinitialisées et on peut recommencer un tirage...



## 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 :

The Scratch script consists of three main parts:

- Step 1:** A **définir** block named "Mélanger les cartes". Inside, two "supprimer tous les éléments de la liste" blocks remove all items from *liste\_costumes* and *liste\_mélangée*.
- Step 2:** A **répéter [2 fois]** loop. Inside, a **basculer sur le costume** block (set costume to Block-a) is followed by another **répéter [8 fois]** loop. Inside this inner loop, a **ajouter** block adds "nom du costume" from *liste\_costumes* to *liste\_mélangée*, followed by a **costume suivant** block.
- Step 3:** A **répéter** loop with a variable of "longueur de liste\_costumes" and "fois". Inside, a **mettre** block sets "num\_du\_costume\_tiré\_au\_sort" to a random number between 1 and the length of *liste\_costumes*. This is followed by an **ajouter élément** block adding the element at index *num\_du\_costume\_tiré\_au\_sort* from *liste\_costumes* to *liste\_mélangée*, and a **supprimer l'élément** block removing the same element from *liste\_costumes*.

Les explications sont  
à la page suivante...



# MÉLANGER LES CARTES

## Quelques explications

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

① Par précaution, on vide les deux listes au départ.

② À deux reprises, on copie le nom des 8 costumes dans la *liste\_costumes*, pour obtenir 8 paires d'identifiants.

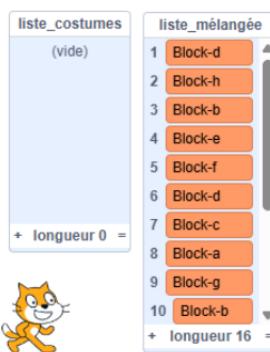
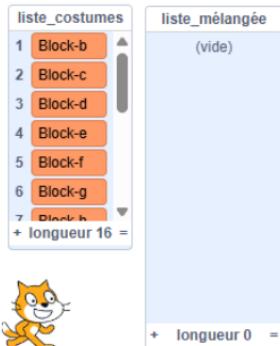
Avant de passer à la procédure ③, affiche les deux listes et teste le code.

③ 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 ③ 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.

The image shows two Scratch scripts side-by-side. The left script, titled 'recto', contains the following blocks:

- when green flag clicked:
  - switch costume to [recto v.]
  - set [i v.] to [0]
  - repeat (8)
    - set [i v.] to [i + 1]
    - create clone of [me]
    - set [x] to [-160 v.]
    - set [y] to [80 v.]
    - repeat (2)
      - switch costume to [costume next v.]
      - add [45 v.] to [x v.]
      - go to [x v.], [y v.]

The right script, titled 'Mélanger les cartes', contains the following blocks:

  - define [Mélanger les cartes]
    - clear [list v.]
    - clear [list v.]
    - repeat (2)
      - switch costume to [Block-a v.]
    - repeat (8)
      - add [costume next v.] to [list v.]
    - repeat (longueur de [list v.])
      - set [num du costume tiré au sort v.] to [random (1) (length of [list v.])]
      - add [element num du costume tiré au sort v. of list v.] to [list v.]
      - remove [element num du costume tiré au sort v. of list v.] from [list v.]

A callout box at the bottom right states: "Les blocs ajoutés ou modifiés sont marqués d'un ✓".



# MÉLANGER ET RETOURNER LES CARTES

The Scratch script consists of two main sections. The first section, triggered by 'when I start as a clone', initializes a variable 'i' to 1 and loops until 'i' reaches the length of the 'liste\_mélangée' list. It then checks if 'choice 1' or 'choice 2' equals the current value of 'i'. If so, it switches costumes to 'recto' and waits 0.5 seconds. If neither matches, it waits 1 second. After each iteration, it increments 'i' by 1. The second section, triggered by 'when this sprite is clicked', contains a nested if-then loop. It checks if 'choice 1' equals 'choice 2' and if 'choice 1' equals the ID of the current clone. If both conditions are true, it waits 0.1 seconds and then sets 'choice 1' and 'choice 2' to their original values.

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>