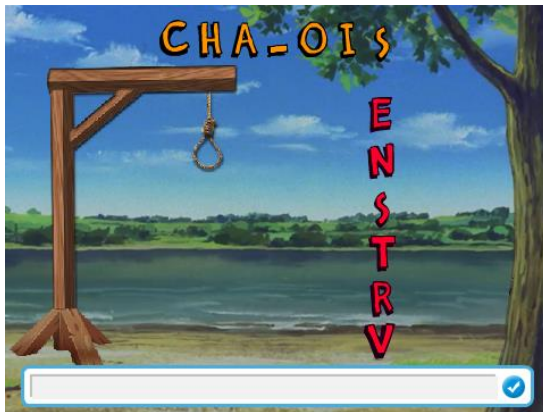


# Jeu du pendu

## Objectif secondaire « L'attaque des clones »

- ⇒ Pour aller plus loin, une fois les objectifs principaux réalisés.
- ⇒ Les objectifs secondaires peuvent être réalisés dans n'importe quel ordre.
  - ⇒ Les objectifs secondaires sont de niveaux différents
- ⇒ Tous les objectifs secondaires peuvent être réalisés dans un même projet.

Utiliser des lutins représentant des caractères alphabétiques pour représenter le mot à chercher et les erreurs commises.





Niveau or

### Fichier requis

[Alphabet.sprite2](#) : inclut 29 costumes (26 lettres, tiret, apostrophe et tiret du bas).



En cliquant sur  (Scratch 2) ou sur  (Scratch 3), importer le lutin et le nommer **Mot**, puis le réimporter et nommer le nouveau lutin **Erreurs**.

### Réalisation

#### Idée générale :

Alors qu'on pourrait penser qu'il suffirait de créer autant de lutins que de lettres de l'alphabet, on se rend compte que cette hypothèse n'est pas réalisable dès lors que le mot à trouver contient plusieurs fois la même lettre. Il faut envisager une autre stratégie : l'utilisation des clones.

En effet, chaque lutin peut être dupliqué (cloné) durant l'exécution du programme, et chaque clone va se comporter exactement de la même façon que le lutin d'origine.

Ainsi, en dupliquant le lutin **Mot**, on peut représenter sur la scène n'importe quel mot souhaité (une lettre = un clone), le tout étant de faire apparaître le bon costume (= la bonne lettre) au bon endroit.

Durant le fonctionnement du programme, on a besoin de modifier certains clones spécifiques et pas d'autres, lorsqu'on veut transformer un « \_ » en « S » par exemple (cas où le joueur a découvert une lettre) : pour ce faire, l'envoi de messages ne suffit pas car tous les clones reçoivent le même message. Il

faut donc pouvoir identifier chaque clone pour savoir la lettre qu'il contient et ainsi réagir convenablement à la réception du message : c'est au moment de la création du clone qu'on va devoir définir ses propriétés.

Enfin, il faut être conscient que lorsque, durant l'exécution du programme, on crée par exemple 2 clones du même lutin, on a non pas 2, mais 3 lutins identiques : celui d'origine, et ses deux clones. Dans ce qui suit, on considère que le lutin d'origine reste masqué tout au long de la partie et ne sert qu'à être cloné : on fait le choix de ne se servir que des clones.

#### Application :

La création d'un clone se fait au moyen de l'instruction **créer un clone de** **Mot**. Ceci lance la création du clone, mais il faut savoir que l'instruction placée juste après est susceptible d'être exécutée avant la fin de la création du clone en question, ce qui peut créer des problèmes de synchronisation. Pour éviter ce problème, il suffit de créer une variable **Clone créé** de type booléen (« oui » / « non ») et de l'utiliser ainsi :

Bloc appelant la création d'un clone	Création du clone
<p>La variable <b>Clone créé</b> est initialisée à « non » et l'on attend qu'elle devienne « oui » avant de placer d'autres instructions.</p> <p>C'est à la toute dernière instruction de la procédure de création du clone que <b>Clone créé</b> est passée « oui » comme montré dans l'exemple ci-contre.</p>	

Pour créer une propriété de chaque clone, on sélectionne le lutin **Mot**, puis on crée une nouvelle variable pour ce lutin uniquement :



La variable **Costume** ainsi créée pourra prendre une valeur différente pour chaque clone de **Mot**.

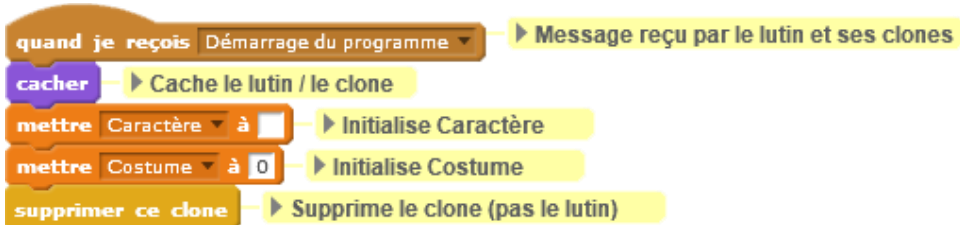
On aura besoin de deux propriétés : **Caractère** et **Costume**.

- **Caractère** *Variable chaîne de caractère*  
Contient le caractère (lettre) que le clone représente.
- **Costume** *Variable numérique*

Contient le numéro de costume du clone cachant la lettre à deviner (au début de la partie, le costume est celui représentant le « \_ », puis change lorsque la lettre est devinée). Ainsi, par exemple, si **Mot à deviner** est « GLAIVE », et que l'on considère le clone représentant le « I » :

- **Caractère** vaut « I »,
- **Costume** vaut 9.

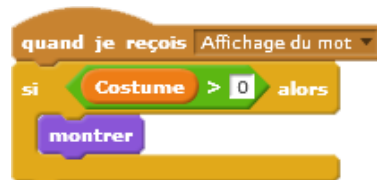
Le script de **Mot** se décompose en 4 parties :



```

quand je reçois Démarrage du programme
  Message reçu par le lutin et ses clones
  cacher
  mettre Caractère à 
  mettre Costume à 0
  supprimer ce clone
  
```

La suppression du clone permet de s'assurer, en début de partie, qu'aucun clone d'une partie précédente n'est plus présent.



```

quand je reçois Affichage du mot
  si Costume > 0 alors
    montrer
  
```


Cette partie doit s'exécuter au moment où **Abby** envoie un message juste avant que le jeu ne débute. **Costume** vaut 0 uniquement pour le lutin initial : il ne doit pas se montrer.



```

quand je reçois Lettre choisie
  si Costume > 0 et Caractère = réponse et costume # = 29 alors
    basculer sur le costume Costume
  
```

Cette partie s'exécute après que le joueur a saisi une lettre.



```

quand je commence comme un clone
  Instructions réalisées pour chaque clone, mais pas pour le lutin d'origine
  
```



```

définir Initialisation
  
```

Dans cette dernière partie, appelée par **Initialisation**, on attribue à chaque clone lors de sa création ses propriétés **Caractère** et **Costume**.

Chaque clone reste caché, mais se met à sa place définitive, prêt à s'afficher : pour se placer, on crée une variable **Position horizontale** initialisée par Abby à  $\text{longueur de Mot à deviner} * -15 + 15$  (formule de centrage du mot horizontalement dans la scène), puis augmentée de 30 après chaque création de clone. Il suffit de faire exécuter **aller à x: Position horizontale y: 150** par le clone pour qu'il se place convenablement.

Le script de **Erreurs** est plus court : seule la propriété **Costume** devra être créée et il n'y aura besoin que de 2 parties :

quand je reçois Démarrage du programme ▼

Cette partie est identique à celle de **Mot**.

Pour modifier la couleur du texte sans retoucher chaque image, on pourra utiliser

mettre l'effet couleur ▼ à 175

quand je commence comme un clone

La création du clone est appelée à chaque fois qu'une lettre perdante est proposée par le joueur. Dans cette procédure, le clone se place (une variable **Position verticale** pourra être utile), puis on calcule le numéro du costume correspondant à **réponse** avant d'afficher le clone.