Jeu Memory : Développement progressif

Dans cette série d'exercices, vous allez créer pas à pas un jeu Memory. Le projet sera développé de manière progressive, chaque exercice s'appuyant sur le précédent pour ajouter de nouvelles fonctionnalités.

Objectifs pédagogiques

- Manipuler le DOM pour créer et modifier des éléments HTML
- Gérer les événements utilisateur (clics)
- Utiliser les variables et structures de données en JavaScript
- Implémenter une logique de jeu avec des conditions
- Utiliser les fonctions setTimeout pour gérer des délais
- Organiser votre code JavaScript de manière structurée

Présentation du jeu

Le Memory est un jeu de cartes où toutes les cartes sont posées face cachée. Le joueur retourne deux cartes à la fois. Si les deux cartes sont identiques, elles restent face visible. Sinon, elles sont retournées face cachée. Le but est de retrouver toutes les paires.

Exercice 1 : Mise en place du jeu

Objectif : Créer la structure de base du jeu avec une grille de cartes.

Consignes :

1. Créez un fichier HTML avec une structure de base
2. Ajoutez un titre et une div qui contiendra la grille de jeu
3. Dans votre script JavaScript, créez un tableau contenant les paires de symboles ou de chiffres (par exemple : ['', '', '', '', '', '', '', '', '', '', '', ''])
4. Écrivez une fonction qui mélange ce tableau (fonction de mélange fournie ci-dessous)
5. Générez dynamiquement les cartes dans la grille HTML, chaque carte étant une div avec une classe "carte"
6. Ajoutez un style CSS de base pour que les cartes apparaissent comme des rectangles face cachée (couleur de fond) disposés en grille

Fonction de mélange à utiliser :

```
function melangerTableau(tableau) {
  for (let i = tableau.length - 1; i > 0; i--) {
    const j = Math.floor(Math.random() * (i + 1));
    [tableau[i], tableau[j]] = [tableau[j], tableau[i]];
  }
  return tableau;
}
```

Exercice 2 : Retourner les cartes

Objectif : Permettre au joueur de retourner les cartes en cliquant dessus.

Consignes :

1. Reprenez le code de l'exercice 1
2. Ajoutez un événement de clic sur chaque carte
3. Lorsqu'une carte est cliquée, ajoutez-lui une classe "retournee" qui modifie son apparence (via CSS)
4. Affichez le symbole de la carte lorsqu'elle est retournée
5. Assurez-vous qu'une carte déjà retournée ne peut pas être cliquée à nouveau

Exercice 3 : Vérifier les paires

Objectif : Implémenter la logique pour vérifier si deux cartes retournées forment une paire.

Consignes :

1. Reprenez le code de l'exercice 2
2. Créez des variables pour stocker la première et la deuxième carte retournées
3. Après avoir retourné deux cartes, vérifiez si elles ont le même symbole
4. Si les cartes forment une paire, laissez-les face visible et ajoutez-leur une classe "paire-trouvee"
5. Si les cartes ne forment pas une paire, retournez-les face cachée après un délai de 1 seconde (utilisez setTimeout)
6. Réinitialisez les variables de carte après chaque vérification

Exercice 4 : Compléter la logique de jeu

Objectif : Finaliser la logique du jeu et détecter la fin de partie.

Consignes :

1. Reprenez le code de l'exercice 3
2. Ajoutez un compteur de paires trouvées
3. Empêchez le joueur de cliquer sur d'autres cartes pendant que deux cartes non correspondantes sont temporairement révélées
4. Détectez quand toutes les paires ont été trouvées (fin de partie)
5. Affichez un message de victoire lorsque le jeu est terminé
6. Ajoutez un bouton "Nouvelle partie" qui réinitialise le jeu

Exercice 5 : Améliorations (optionnel)

Objectif : Améliorer l'expérience de jeu avec des fonctionnalités supplémentaires.

Consignes :

1. Reprenez le code de l'exercice 4
2. Ajoutez un compteur de coups joués
3. Ajoutez un chronomètre qui démarre au premier clic et s'arrête à la fin de la partie
4. Créez un système de niveaux de difficulté (facile : 6 paires, moyen : 8 paires, difficile : 12 paires)
5. Améliorez le design avec des animations de retournement de carte (transitions CSS)
6. Ajoutez un tableau des meilleurs scores utilisant localStorage pour sauvegarder les performances

Conseils

- Testez votre code à chaque étape
- Utilisez la console pour déboguer
- N'hésitez pas à commenter votre code
- Organisez votre code en fonctions pour plus de clarté
- Pensez à désactiver temporairement les clics pendant les animations