# Exam JavaScript Avancé

Vous avez 2 exercices à réaliser : Le jeu d'échec et l'utilisation de l'API Popular Movies.

A rendre un fichier compressé (zip, 7z, rar) contenant :

- Le projet contenant les fichiers html et les fichiers js.

# Popular Movies

Vous allez développer une page qui affichera une liste des films les plus populaire en utilisant l'API Popular Movies. Cette API fournit des informations sur les films.

API utilisée :
https://developers.themoviedb.org/3/movies/get-popular-movies

API Key : 3f951fde1f94ff23e3aebbd24b292474

API url :
https://api.themoviedb.org/3/movie/popular?api_key=3f951fde1f94ff23e3aebbd24b292474&language=fr-FR&page=1

1) Créer un fichier popular_movies.html
2) Depuis un fichier JS, écrire le code pour faire appel à l'API
3) Récupérer le titre et l'image de chaque film et les injecter dans un div pour voir un résultat similaire à :


**Spider-Man : No Way Home**


**The Batman**

# Jeu d'échec

Vous allez devoir développer en Javascript une partie des fonctionnalités du jeu.

Vous allez utiliser le fichier Jeu_echec.html et les images qui vous sont fournies pour développer le jeu.

## Moteur du jeu

### Tour par tour

Définir le mécanisme du jeu tour par tour entre le joueur 1 et le joueur 2.

Il faudra donc mettre en place une variable numérique pour désigner le joueur courant (déjà mise en place par **currentPlayer**).

Utilisez correctement la fonction **changePlayer** pour mettre en place le changement de joueur lors du clic.

### Initialisation/Réinitialisation du plateau

L'objectif ici est de faire une fonction qui va initialiser votre plateau de jeu et une autre qui le réinitialisera et utilisera ensuite votre fonction d'initialisation.

Dans un premier temps, faites une fonction **getElementsChildren** qui prendra en paramètre un élément HTML et qui renverra sa liste d'éléments enfant par le biais de childNodes.

Dans un second temps, selon l'index de vos cases, récupérez successivement tous les éléments à partir de childNodes (qui se comporte comme un tableau) et ajoutez les classes des pièces qui conviennent.

## Gestion des pièces

Sélection

Faire en sorte que lorsque l'on click sur une case qui contient une pièce celle-ci est entourée de :

- rouge pour le joueur 2
- bleu pour le joueur 1

Pour ce faire, il vous faudra utiliser au bon endroit la fonction addSelectClassByPlayer.

Dans le script moteur.js fourni les variables suivantes sont déjà définies :

- **currentSelection** : boolean qui permet de savoir si une sélection a déjà été faite.
  - Sa valeur par défaut est false.
  - Lors du premier click sa valeur va déterminer du traitement à suivre
    - false : on va récupérer les données qui nous intéressent sur l'élément cible **selectedHTML**, .
    - true : il s'agit du deuxième clic et donc du déplacement de notre pièce. Vous traiterez alors les différents cas dans la prochaine section.
- **selectedHTML** : fait référence à l'élément HTML sélectionné.
- **selectedHTMLClasses** : fait référence à la string correspondant à l'attribut Class de votre élément HTML sélectionné.

Ces variables vont vous permettre de garder en mémoire les données de votre précédent click afin d'agir sur l'élément précédemment sélectionné lors du deuxième click.

Un détail est à remarquer, les joueurs pourront alors sélectionner toutes les cases (toutes les couleurs de pièces et les cases vides).

Créez une fonction **isCaseEmpty** qui prend en paramètre l'élément HTML cible et qui à l'aide du retour de la fonction **getCaseClass** va renvoyer true si la case est vide, false sinon.

Pour les pions réservés aux joueurs, créez deux tableaux **arrayPlayer1Pieces** et **arrayPlayer2Pieces** dans le fichier JS.

Le tableau du joueur 1 contient tous les noms de classes faisant référence à des pièces blanches.

Le tableau du joueur 2 contient tous les noms de classes faisant référence à des pièces noires.

Créez une fonction **isCaseAllowed** qui prend en paramètre l'élément HTML cible et qui à l'aide du retour de la fonction **getCaseClass** va renvoyer true si la case contient une pièce autorisée pour le joueur courant, false sinon. Vous utiliserez la fonction include des tableaux en Javascript qui permet de tester la présence d'un élément.

Une fois votre fonction **isCaseAllowed** terminée faites en sorte que, selon son résultat, la sélection se fasse.

Déplacement

Au vue de la complexité des déplacements dans le jeu des échecs, nous allons poursuivre sur une version très simplifiée. Après la sélection d'un élément, vous pouvez vous déplacer sur la case de votre choix sans restriction.

Lorsque la sélection a déjà été faite vous allez :

- vérifier si la case cible est la même que celle sélectionnée
  - dans ce cas-là il faudra faire en sorte qu'elle ne soit plus sélectionnée en utilisant la fonction **removeSelectedClassByPlayer** et passer **currentSelection** à false.
- vérifier si la case cible contient bien une pièce de l'autre couleur (vous servir des tableaux de la sélection).
  - Faire en sorte que votre case de départ ne contient plus votre pièce et que la case cible soit effacée et remplacée par la pièce de votre case de départ. On implémente ici le fait de prendre/remplacer une pièce de son adversaire.
- dans le cas où la case cible contient une pièce de la même couleur, annuler la sélection et passer au joueur suivant.

## Condition de victoire

Comme pour les déplacements les conditions de victoire seront simplifiées.

Si un joueur a pris toutes les pièces du joueur adverse, donc que le plateau ne contient plus de pièce de ce joueur. Vous pourrez alors essayer de récupérer l'ensemble des éléments par nom de classe et vérifier si le tableau retourné est vide ou non.

Si un joueur prend le roi du joueur adverse alors il gagne.

## Gestion du score

Le score est affiché sur la page, et est incrémenté lors d'une victoire.

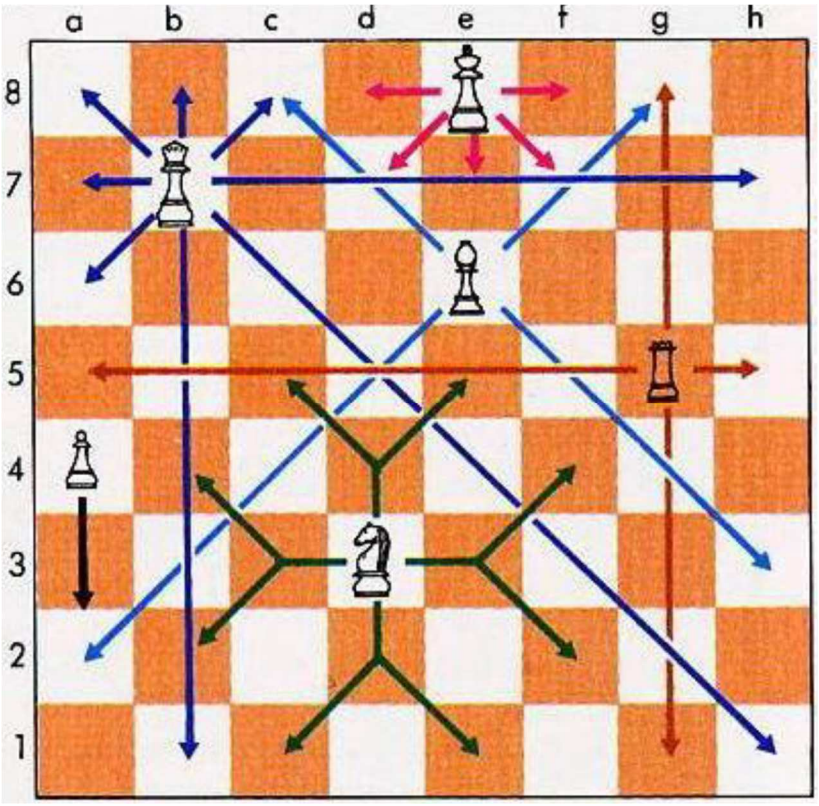
Faites une fonction qui appelle la fonction de réinitialisation du plateau et une nouvelle fonction qui réinitialise le score.

Mettez en place un bouton permettant de réinitialiser la partie.

## Pour aller plus loin (bonus)

### Gestion avancée des déplacements

Les pièces dans le jeu d'échecs ont chacun une manière de se déplacer :




- **pion**: ne peut qu'avancer d'une case.
- **reine** : peut avancer dans toutes les directions sans limite.
- **roi** : peut avancer d'une case dans n'importe quelle direction.
- **tour** : peut avancer horizontalement et verticalement depuis sa position sans limite.
- **fou** : peut avancer sur les diagonales sans limite.
- **cavalier** : peut avancer en "L" verticalement et horizontalement
    - exemple : le cavalier va avancer de deux cases puis pourra aller une case à gauche ou une case à droite.

Note : Une pièce ne peut pas aller sur une case si entre lui et cette case il existe une autre pièce.