La Plateforme

RunTrack JS : Tic Tac Toe

Why did the function get kicked out of school? Because it didn't have class.

Introduction du sujet

La **programmation orientée objet (POO)** n'est pas réservée au **PHP** ou à d'autres langages orientés serveur. Le **JavaScript**, souvent utilisé côté client, possède également des **capacités robustes de POO**. Aujourd'hui, nous allons utiliser ces capacités pour développer un jeu classique : le **Tic Tac Toe (ou Morpion)**.

Nous allons construire ce jeu en utilisant **les classes et les modules ES6** pour organiser et structurer notre code, rendant ainsi le processus de développement plus propre et plus compréhensible.

Vous devriez avoir les classes suivantes dans votre code :

- **Game**
- **Board**
- **Player**

Dans le dossier **runtrack-b3-javascript**, créez un dossier **jour-03**. Assurez-vous de faire des commits réguliers et de respecter les consignes dans le sujet.

Job 01

Avant de plonger dans le JavaScript, il est crucial d'avoir une interface utilisateur pour interagir avec notre jeu. Dans le **dossier jour-03**, créez un fichier **index.html**.
Dans ce fichier, vous allez préparer plusieurs éléments HTML afin de pouvoir afficher les différentes informations à l'utilisateur :

- Un **paragraphe** avec comme **id "current-player"** permettant d'afficher le joueur actuel
- Un **paragraphe** avec comme **id "message"** permettant d'indiquer les message de fin de partie
- Un **paragraphe** avec comme **id "turn"** permettant d'indiquer le nombre de tours
- Un **bouton** avec comme **id "play"** qui permettra de lancer une nouvelle partie
- Une **div** avec comme **id "board"**

Job 02

Dans le dossier **jour-03**, créez un dossier **class** dans lequel vous mettrez toutes les classes de votre projet. Commençons par représenter un joueur. Dans ce dossier, créez un fichier **Player.js**. Cette classe devra avoir les **attributs** suivants:

- **symbol** (soit 'X' ou 'O')
- **isCurrentPlayer** (un booléen indiquant si c'est le tour de ce joueur)

Le constructeur de votre classe devrait permettre d'initialiser ces attributs.

Job 03

Maintenant que vous avez créé votre première **class** en JavaScript, vous allez pouvoir l'**importer dans un fichier script.js dans le dossier jour-03**. Pour ce faire, utilisez le fonctionnement **des modules ES6** pour **exporter le contenu de la class Player** puis l'importer dans votre fichier script.js. Cela implique donc que dans votre fichier **index.html**, vous ne devez avoir **que votre fichier script.js chargé** puisqu'il importera tous les fichiers nécessaires au bon fonctionnement de votre projet.

Job 04

Dans le dossier **class**, créez un fichier **Board.js**. Cette class représentera notre plateau de Tic Tac Toe et devrait avoir les attributs suivants :

- **grid** (un tableau 2D pour représenter le plateau de jeu)
- **hasWinner** (pour savoir si une victoire a été obtenue)

Afin de représenter convenablement le plateau de jeu, nous allons avoir besoin de plusieurs méthodes pour la **class Board** :

1. **initializeBoard()** : Remplit le plateau avec des **chaines de caractères égales à "- "** pour représenter les cases vides et **attribue false à la propriété hasWinner**. Cette méthode ne retourne rien.

2. **displayBoard()** : Affiche le contenu de la grille **dans la div avec l'id "board"**. Pour chacune des lignes, créez une div avec la **class "row"**. Pour chacune des cases, créez un bouton avec la **class "case"** et un **id "btn-Y-X"** où **Y** est compris **entre 0 et 2** et **X** est compris **entre 0 et 2** pour **représenter les coordonnées du bouton**.

3. **placeMove(row, col, symbol)** : Permet de placer le symbole du joueur ('X' ou 'O') à l'emplacement spécifié. Si la case est déjà prise, elle retourne false, sinon elle place le mouvement et retourne true.

4. **checkVictory()** : Vérifie si l'une des combinaisons gagnantes est présente sur le plateau. Si oui, change la valeur de hasWinner et retourne true, sinon, retourne false.

5. **isFull()** : Vérifie si le plateau est plein, c'est-à-dire s'il y a une égalité. Si toutes les cases sont remplies, retourne true, sinon false.

6. **resetBoard()** : Remet à zéro le plateau pour une nouvelle partie, affiche le plateau et ne retourne rien.

Job 05

Dans le dossier **class**, créez un fichier **Game.js**. Cette classe combine **Player** et **Board** pour créer une partie complète de Tic Tac Toe. La class **Game** comprends les attributs suivants :

- **player1** (une instance de la **class Player**)
- **player2** (une instance de la **class Player**)
- **board** (une instance de la **class Board**)
- **currentTurn** (pour suivre le tour actuel)

Méthodes de la classe **Game** :

1. **constructor(player1, player2, board)** : Permet de créer une nouvelle instance de la **class Game** en initialisant ces propriétés. Elle défini le **joueur actuel sur player1 (X)** par défaut.
2. **startNewGame()** : Initialise une nouvelle partie, en créant un nouveau plateau et en déterminant le joueur qui commence.
3. **makeMove(row, col)** : Fait appel à la méthode **placeMove** de la classe **Board**. Si le mouvement est valide, elle affiche le plateau, vérifie si la partie est terminée et change de tour.
4. **registerMove()** : Permet de d'ajouter des events listeners sur chacun des boutons du plateau.
5. **switchTurn()** : Change le tour entre les deux joueurs.
6. **checkGameOver() :** Vérifie si la partie est terminée en consultant la méthode **checkVictory** de la classe **Board** et **isFull**.
7. **announceWinner()** : Affiche le gagnant ou une égalité avec un message **"Le joueur {symbol} a gagné !" ou "Match nul !"**
8. **resetGame()** : Réinitialise le jeu pour une nouvelle partie en utilisant **resetBoard** de la classe **Board** et réinitialise les autres attributs pertinents.

Compétences visées

- **JavaScript :** 15 points
- **Programmation orientée objet :** 10 points
- **Utilisation des modules ES6 :** 10 points

Rendu

Le projet est à rendre sur **https://github.com/prenom-nom/runtrack-b3-javascript**.

C'est le troisième jour de RunTrack, donc normalement, vous n'avez que le dossier **jour-03** à push. Faites attention à **bien respecter la signature de chacune de vos fonctions**, et les **noms de fichiers et de dossiers à la lettre !**