

# Développement web – A3

## TP n°3 : JavaScript – Jeu de Morpion

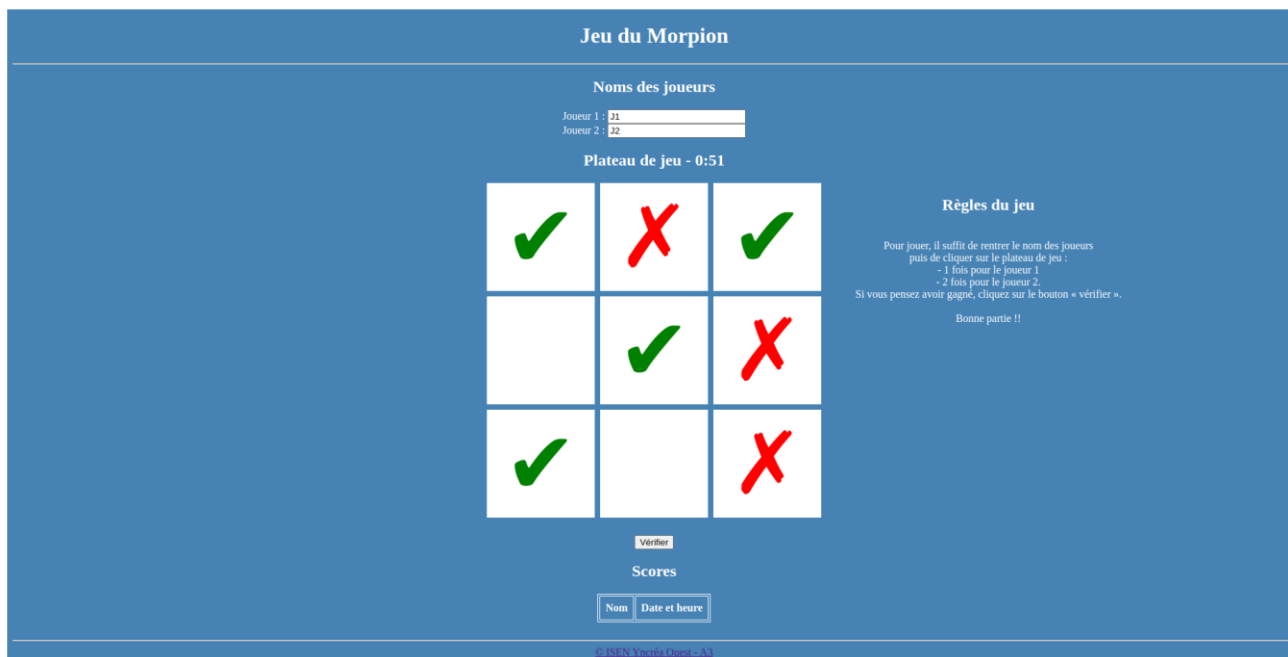


### Notions abordées

- Inclusion d'un script JavaScript
- Console JavaScript
- Algorithmique en JavaScript
- Fonctions
- Gestion des événements
- Gestion des timers
- Sélecteurs d'éléments par id et par nom

### Objectifs du TP

Dans ce TP vous allez créer toute la partie dynamique d'un jeu de Morpion à deux joueurs grâce au langage JavaScript. En particulier, vous allez gérer les événements permettant de jouer une case et de vérifier si un joueur a gagné, d'afficher les scores sur la page HTML (c.-à-d. de manière non permanente) et de mettre en place un timer de jeu. Le visuel final du jeu ressemblera à celui présenté dans la figure suivante.



### 1. Téléchargement des fichiers

1. Téléchargez et extrayez l'archive **tp3Input.zip**
2. Créez un fichier JavaScript « tic-tac-toe.js » dans le dossier « js »
3. Pensez à inclure le fichier JavaScript « tic-tac-toe.js » dans la page HTML.

### 2. Premiers pas

Dans cette partie nous souhaitons prendre en main le langage JavaScript et réaliser son interprétation à l'aide d'un navigateur afin de donner du dynamisme à notre page. Pour cela, la «

console » disponible dans les outils développeurs doit être maîtrisée.

**Page dynamique vs statique** : Dans une page dynamique, le langage JavaScript permet d'afficher/cacher des éléments, de les modifier ou d'en ajouter. En revanche, dans une page statique, les éléments sont figés et le « dynamisme » n'est possible qu'en changeant de page HTML. Ceci amène à des charges réseau et serveur plus importantes ainsi qu'à un manque de fluidité pour l'utilisateur final.

### 2.1. Mode Strict

Il est conseillé d'utiliser le mode « strict » de JavaScript afin de faciliter le débogage. Pour cela, indiquez ce mode au début de votre script « tic-tac-toe.js » avec :

```
'use strict' ;
```

### 2.2. Définition de constantes

Pour commencer, définissez la constante suivante pour les messages à afficher :

```
let MESSAGES = {'message1': 'Bienvenue sur notre jeu.',  
                'message2': ' vous avez gagné !'};
```

### 2.3. Première fonction

Afin de vérifier le bon fonctionnement d'un script et de permettre le débogage, il est possible d'utiliser la console JavaScript. Celle-ci est disponible à travers les outils développeurs du navigateur. Il est possible de l'afficher avec le raccourci clavier **CTRL + MAJ + I** (ou éventuellement en faisant un clic droit sur la page et en ouvrant le menu « Inspecter »). Ensuite, l'onglet « console » permet de visualiser le résultat des instructions `console.log(...)`.

**Console JavaScript** : La console n'est pas qu'un espace d'affichage. Il s'agit également d'un interpréteur interactif dans lequel vous pouvez évaluer des expressions comme `1 + 5`. Une fois les scripts chargés, il est aussi possible d'utiliser les variables ou les fonctions ayant une portée globale comme la fonction « `welcome` ».

Dans votre script, commencez par créer une fonction nommée « `main` » qui :

- affiche dans la console (avec `console.log(...)`) le message n°1 de la constante 'MESSAGES' à savoir « Bienvenue sur notre jeu. » ;
- affiche dans une boîte de dialogue (avec `alert(...)`) le même message.

**Finalement, vérifiez son bon fonctionnement en l'appelant. Elle devrait alors s'exécuter au chargement de votre page HTML.**

## 3. Remplissage des cases

Dans cette partie nous souhaitons pouvoir cliquer sur les cases du plateau de jeu pour les marquer d'un cercle ou d'une croix. En particulier, **un clic devra permettre de marquer la case d'une « coche » tandis que deux clics devront permettre de marquer la case d'une « croix »**. Aussi, nous souhaitons que les « coches » soient verts et que les « croix » soient rouges (voir image de la page 1).

### 3.1. Définition de constantes

Pour commencer, définissez deux constantes :

```
const PLAYER1 = "✓";  
const PLAYER2 = "✗";
```

### 3.2. Gestion des évènements

Dans « index.html », les cases du plateau de jeu ont des id allant de cell0 à cell18. Ajoutez, dans la fonction « main() » du fichier JS, à chacun de ceux-ci un gestionnaire d'évènements pour le « clic » en vous appuyant sur le code suivant :

```
element.addEventListener('eventName', myCallback);
```

**Information sur les callbacks** : Un callback est une fonction, passée en paramètre d'une autre fonction, qui sera appelée dans cette dernière. Afin de passer une fonction comme paramètre d'une autre fonction, on donne son nom sans mettre de paramètre. Exemple :

```
element.addEventListener('submit', myCallback);

function myCallback(event)
{
    console.log(event.target.innerHTML);
}
```

Dans votre cas, la callback sera la fonction nommée « fill ». Créez cette fonction qui contiendra un argument « event » qui contient pour le moment l'instruction suivante :

```
console.log(event.target);
```

Notez que l'argument « event.target » de la fonction stocke la cible de l'évènement (c.-à-d. l'élément HTML cliqué).

**Testez si le clic sur une case affiche bien l'élément HTML cliqué dans la console JavaScript.**

### 3.3. Fonction de traitement

Dans la fonction de traitement « fill », créez un code avec les fonctionnalités suivantes :

- Si le code HTML de la case est vide ou s'il contient la valeur PLAYER2 :
  - Remplir la case (c.-à-d. son code HTML) avec la valeur PLAYER1 ;
  - Mettre en vert le texte de la case avec `event.target.style.color`.
- Sinon :
  - Remplir la case (c.-à-d. son code HTML) avec la valeur PLAYER2 ;
  - Mettre en rouge le texte de la case avec `event.target.style.color`

**Testez ce fonctionnement sur votre page HTML.**

## 4. Vérification de fin de partie

Dans cette partie nous souhaitons pouvoir vérifier si un joueur a gagné en cliquant sur le bouton « Vérifier » de la page HTML.

### 4.1. Vérification du plateau de jeu

Avant toutes choses, il va falloir écrire une fonction JavaScript permettant de vérifier si un joueur a gagné. Pour cela, écrivez la fonction suivante :

```
function verifyPlayer(playerMark)
```

playerMark est le symbole (c.-à-d. les symboles de PLAYER1 ou PLAYER2) du joueur pour lequel on souhaite vérifier s'il a gagné. La fonction renverra true si celui-ci a gagné et false sinon. Dans cette fonction il va falloir renvoyer true si :

- L'une des trois lignes est gagnante (c.-à-d. une ligne contient trois fois le symbole du joueur) ;
- L'une des trois colonnes est gagnante ;
- L'une des deux diagonales est gagnante.

et renvoyer false sinon.

## 4.2. Gestion de l'évènement

Pour vérifier si l'un des joueurs ou l'autre à gagner, il va falloir définir un gestionnaire d'évènement associé au « clic » sur le bouton, d'id play, en appelant une fonction nommée « verify ». Celle-ci aura pour rôle de vérifier si le joueur 1 a gagné ou si le joueur 2 a gagné ou si personne n'a encore gagné ou s'il y'avait égalité. Donc, elle va appeler la fonction « verifyPlayer » programmée auparavant.

Après avoir défini votre gestionnaire d'évènement dans la fonction « main », définissez la fonction « verify » dans laquelle il faut afficher « Bravo NOM\_JOUEUR vous avez gagné ! ! » (où NOM\_JOUEUR est le nom que le joueur a entré) si un joueur est gagnant.

Dans le cas où un joueur est gagnant, il serait intéressant de réinitialiser chaque case du plateau de jeu afin de pouvoir relancer une partie.

## 5. Gestion des scores

Dans cette partie nous souhaitons pouvoir enregistrer dans un tableau l'historique des scores pour la session en cours (c.-à-d. temps que la page n'a pas été rechargée ou fermée).

Scores	
Nom	Date et heure
J1	3/30/2023, 10:11:34 PM

Pour cela, écrivez la fonction suivante :

```
function addScore(name)
```

Celle-ci est appelée quand un joueur a gagné et doit ajouter une ligne au tableau d'id scores avec le nom du gagnant ainsi que la date et l'heure de sa victoire.

Pour ajouter la date et l'heure, utilisez :

```
new Date().toLocaleString();
```

## 6. Gestion du timer

Enfin, dans cette partie nous souhaitons pouvoir ajouter un timer de jeu. Pour cela, stockez, dans la fonction « main » la date et l'heure de début de partie en utilisant :

```
let start = new Date();
```

Ensuite, appelez toutes les secondes (en utilisant « setInterval ») une fonction nommée « updateTimer » dans laquelle :

- Vous calculerez le temps écoulé en soustrayant la date et l'heure courante de celle de début de partie ;
- Vous afficherez l'heure au format minutes:secondes dans l'élément HTML d'id timer prévu à cet effet.