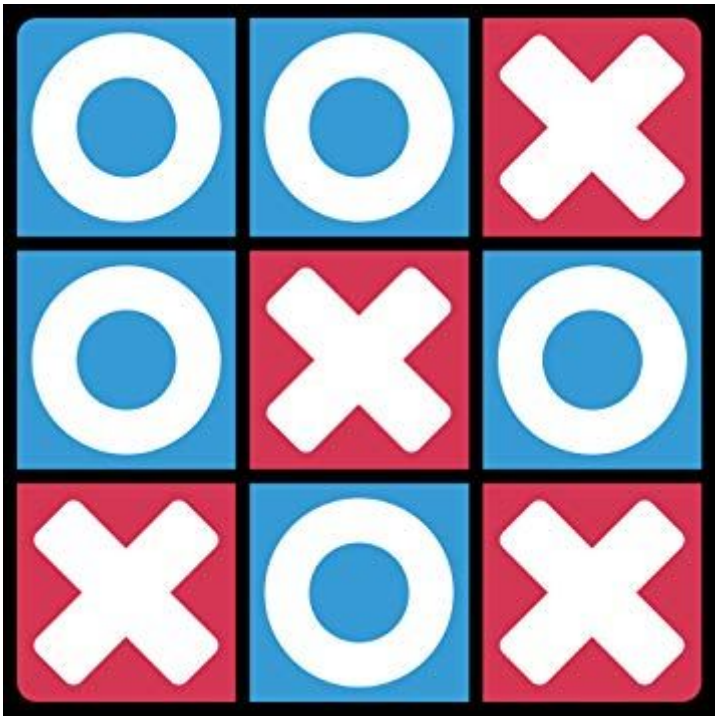
SUMMER SCHOOL

TicTacToe



Auteurs
Lucas Thevenard
Lucas Obadia

Introduction

L'objectif est de réaliser le célèbre jeu vidéo TicTacToe (le morpion en français) dans une version web. Pour réaliser ce jeu, nous allons utiliser le **Javascript** en plus du HTML/CSS.

Qu'est-ce que l'HTML et le CSS ?

L'HTML, acronyme signifiant « HyperText Markup Language », est un **langage de balisage** conçu pour représenter des pages web. On est aujourd'hui à la version 5 de l'html, noté HTML5.

Exemple d'une balise p en HTML

```
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</p>
```

Dans l'exemple ci-dessus, on délimite la phrase « Lorem ipsum dolor sit amet, consectetuer adipiscing elit. » par la balise `p` (définissant un paragraphe) au moyen d'une balise d'ouverture et d'une balise de fermeture .

Le **CSS**, acronyme de « Cascading Style Sheets » est un langage informatique qui décrit la présentation des documents html. Concrètement, l'html est chargé du contenu de la page, et le CSS est chargé de **mettre en forme le contenu**. Actuellement, on est à la version 3 de css, noté CSS3

Exemple d'une feuille de style qui mettra la couleur rouge sur le texte de toutes les balises p

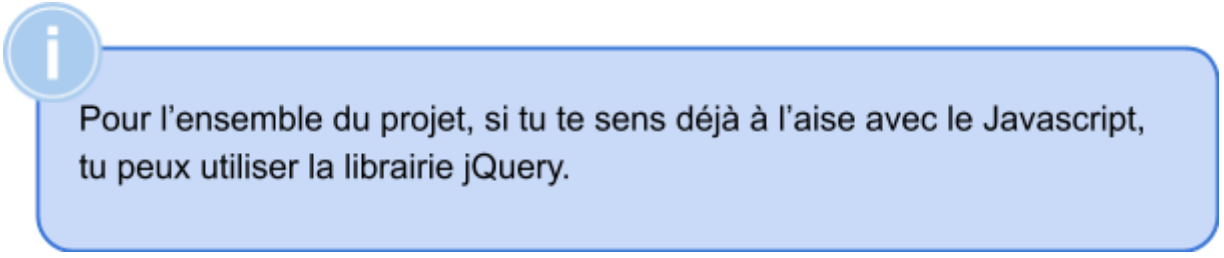
```
p {
  color: red;
}
```

Qu’est-ce que le Javascript ?

Le Javascript est un **langage de programmation**, c’est une forme de code qui permet de dicter à l’ordinateur quoi faire. On trouve la majorité du code Javascript dans les **pages web** pour les rendre **dynamiques** / interactives, même si vous ne le voyez pas s’afficher. En effet, c’est le seul langage qui permet de dicter à un navigateur web (Internet Explorer, Chrome, Firefox…) ce qu’il doit faire sans rien installer préalablement. On peut dire que la majorité des navigateurs web “parlent” le Javascript.

Voici des exemples de ce que le Javascript nous permet de faire

- Faire bouger, apparaitre disparaitre des éléments de la page (titre, menu, image…)
- Mettre à jour des éléments de la page sans la recharger
- Faire une action spécifique au moment d’un événement de l’utilisateur (cliquer sur un élément, entrée clavier, bouger la souris…)



Environnement de travail

Nous t'avons donné préalablement une clé usb avec un dossier **TicTacToe** qui contient un fichier html, un fichier css, et un dossier img contenant toutes les images de ton site web.

A la racine de ce dossier, commence par créer un fichier Javascript qui te permettra de coder la logique de ton jeu. Un fichier javascript est un fichier dont l'extension est **.js**, tu peux évidemment nommer ton fichier comme tu veux tant que tu respecte l'extension.

Si tu veux rajouter d'autres ressources, penses bien à **organiser tes dossiers**. Par exemple si tu rajoutes une musique à ton jeu, mets la dans un dossier "Musique".

Pour éditer ton fichier Javascript, on te recommande d'utiliser un IDE, tu peux voir ça comme un gros éditeur de texte avec plein d'outils spécialement pensés par et pour les développeurs. Il en existe plusieurs tels que Visual Studio Code, Sublime Text ou encore WebStorm. Tu peux très bien utiliser aussi un éditeur de texte classique, mais ton expérience en tant que développeur sera biaisé.

Maintenant que tu as créé le fichier Javascript, il faut l'importer depuis le fichier html. Les imports js se font toujours à la fin de la balise **body** du fichier html. Je te laisse compléter cette ligne du fichier html en ajoutant le nom de ton fichier dans **src** :

```
<script src="monfichier.js" type="text/javascript"></script>
```

Sans cet import, ton fichier javascript ne sera jamais pris en compte dans ta page web.

Crée ton jeu

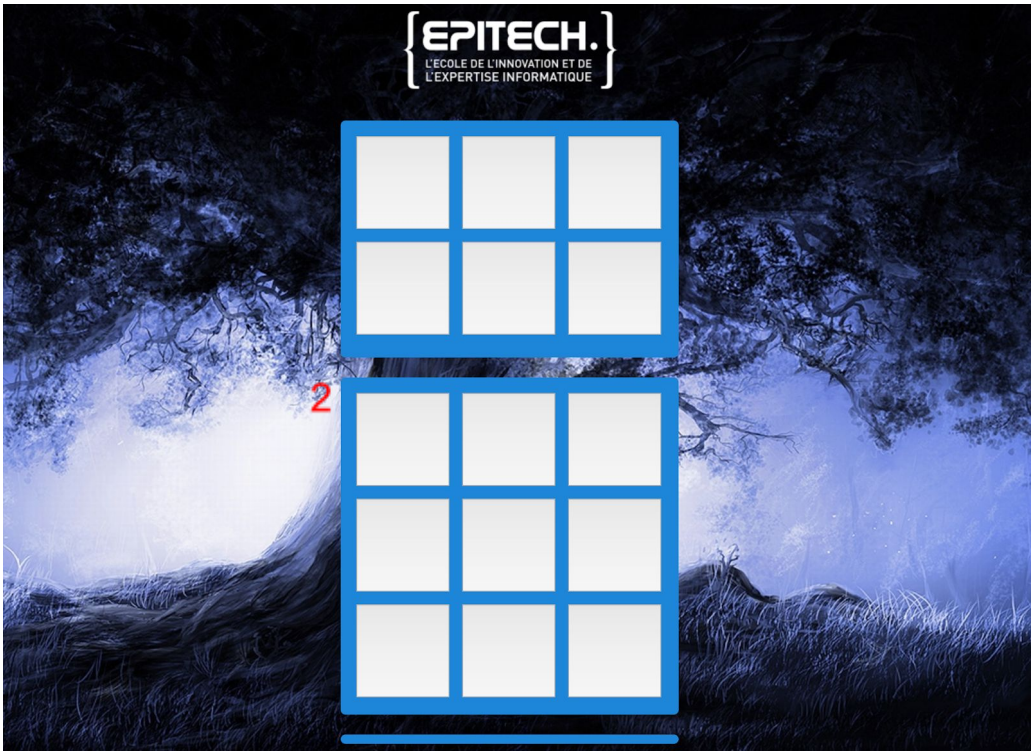
Scénario

Dans un premier temps, les 2 joueurs devront choisir les personnages avec qui jouer (2 avatars différents). Une fois que les joueurs ont sélectionné leur avatar, le menu de sélection disparaît pour laisser place au “plateau” du TicTacToe. La logique classique du jeu démarre avec 2 finalités différentes (victoire ou match nul), sachant qu’une défaite d’un joueur entraîne forcément la victoire de son adversaire. Une fois la partie terminée, mettre un bouton pour pouvoir rejouer.

Les composants

Je t’invite à ouvrir le fichier index.html dans un navigateur web. On remarque que la page contient une image de fond, un logo Epitech, ainsi que 3 encadrés bleus qui sont l’un en dessous de l’autre.

Le premier encadré bleu va nous servir à la sélection d’avatar. (1)
Le second encadré bleu est le plateau du TicTacToe. (2)
Le troisième barre est une barre de status, c’est là où l’on va mettre des messages pour avertir le joueur que c’est à lui de jouer ou encore qu’on va mettre le bouton pour rejouer. (3)



On ne veut pas avoir les 3 composants bleus au même moment, au tout début, on souhaite afficher seulement le composant de sélection avec la barre de status. Il faut donc le dire explicitement à notre site grâce au… Javascript !

Sélectionner son avatar

Les sélecteurs

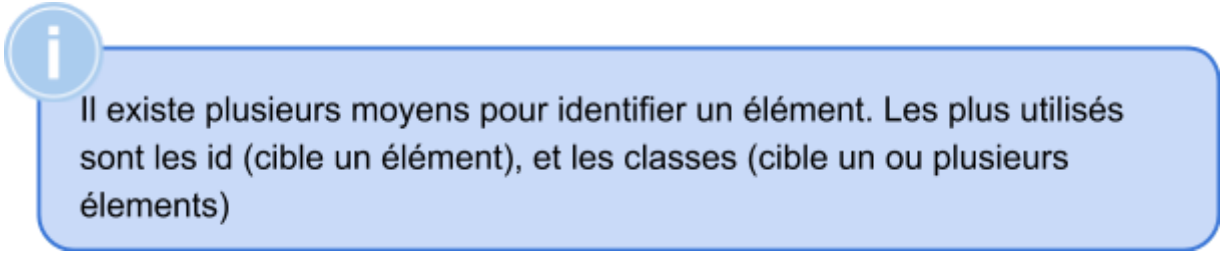
On va stocker dans des variables les éléments de notre page. Dans la page html, on a donné des **id** sur les **div**. Par exemple la div responsable d'afficher l'élément de sélection d'avatar a pour id "selecteur".

Un id permet d'identifier, grâce au nom donné, **un et un seul élément** de la page web. Voici comment récupérer un élément html en Javascript :

```
var maVariable = document.querySelector("#monId");
```

La variable **document** est une variable globale qui existe dès lors le chargement de votre page, et qui stocke l'ensemble de votre page web sous forme d'objet. On peut donc récupérer à partir de cette variable, n'importe quel élément de notre page, n'est-ce pas magnifique ?

Cette variable a ce qu'on appelle des **méthodes** (fonctions), telle que **querySelector** qui est responsable de retourner l'élément cibler grâce à l'id par exemple.



Dans le fichier js, créer une variable **game** qui aura l'élément html qui a comme id 'game'.
Faire de même pour l'élément qui a l'id 'gameStatus'.

Les variables créées juste avant doivent être dans aucune fonction, c'est ce que qu'on appelle des variables globales, elles pourront être utilisées dans n'importe quelle fonction.

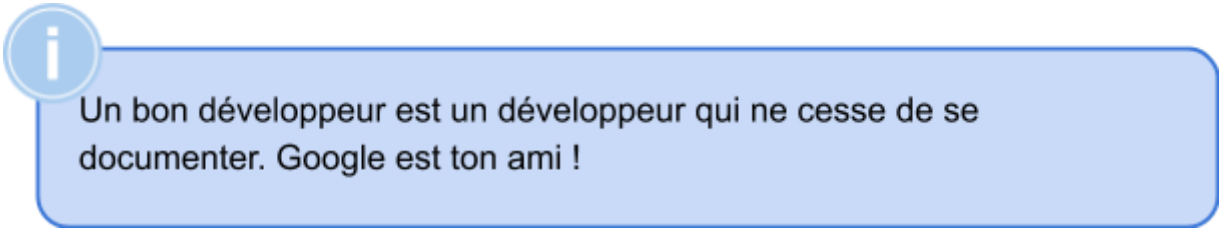
Crée ensuite une fonction avec le nom que tu veux.
Tu dois avoir quelque chose comme ceci :

```
// Variables globales
var game = document.querySelector("#game");
var afficheur = document.querySelector("#gameStatus");

function selection() {
  // Do something
}

selection(); // Execute la fonction 'selection'
```

Dans la fonction 'selection' et en te documentant sur internet, il te faut modifier le style de l'élément game et lui dire qu'on ne veut pas l'afficher maintenant. Aller je te donne un indice, tu dois modifier la propriété **display** du style de l'élément...



Maintenant que le plateau de jeu n'est plus affiché au début, on peut se concentrer sur la sélection des avatars.

Remplir les cases de nos avatars

Ajoutons une variable dans notre fonction *selection*, comme ceci :

```
var avatars = document.querySelectorAll("#selecteur button");
```

Cette variable contient une liste de chaque bouton de notre div qui a comme id “selecteur”. Vu qu’il y a 6 boutons (cases blanches) dans cette div, notre variable contiendra les 6 boutons. L’avantage est qu’on peut manipuler directement la variable est donc les 6 boutons en même temps, ou manipuler seulement un seul bouton. Par exemple pour accéder au 3e bouton, on doit écrire :

```
avatars[2] // Correspond au 3e bouton de notre liste
```

Tu es peut être en train de te demander pourquoi on appelle le 2e index de la liste “avatars” pour avoir le 3e bouton. La raison est que tous les tableaux/listes commencent par l’index 0.

Je veux le...	1	2	3
Je dois utiliser l’index...	0	1	2

Il faut maintenant créer 2 tableaux, l'un qui contient les balises <img> avec le chemin du gif de l'avatar et l'autre qui contient le nom de chacuns de ces avatars.

Voici ce que tu devrai avoir dans la fonction *selection* :

```
function selection() {
   var avatars = document.querySelectorAll("#selecteur button");
   var images = ['<img src="img/ts.gif">', '<img src="img/rd.gif">', '<img
src="img/aj.gif">', '<img src="img/pp.gif">', '<img src="img/r.gif">', '<img
src="img/f.gif">'];
   var noms = ['Twilight Sparkle', 'Rainbow Dash', 'Applejack', 'Pinkie Pie',
'Rarity', 'Fluttershy'];

   game.style.display = 'none'; // Cache le plateau de jeu
}
```

Si tu le souhaites, tu peux utiliser tes propres gif ;)

Tu te rappelles, tout à leur je t'ai dis qu'on pouvait manipuler la liste *avatars*, élément par élément. L'objectif ici est de remplir le contenu de chaque bouton pour afficher un gif et un nom. Pour l'instant le bouton en html ressemble à ceci :

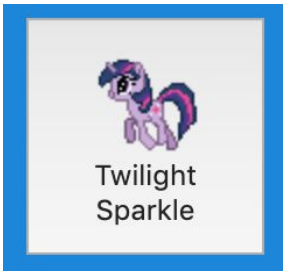
```
<button></button>
```

Ce qu'il donne comme résultat… une case blanche

Il faut faire une boucle qui itère sur la liste *avatars* pour que chaque bouton ressemble à :

```
<button>
  <img src="img/ts.gif">
  Twilight Sparkle
</button>
```

Résultat :



Pour cela je te conseille d'utiliser une **boucle for**, et de regarder de plus près la [propriété innerHTML] d'un élément en Javascript.

> N'hésites pas à demander de l'aide !

Dans cette même boucle for que tu viens de créer, rajoute un eventListener pour que lors du clique sur l'une des cases/boutons, la fonction *setAvatar* soit appelée.

Tu dois donc créer une fonction *setAvatar,* tu peux la laisser vide pour l'instant.

Message de statut

On peut voir à présent les 6 avatars (gif + nom) dans chacune des cases.
Une personne qui joue pour la première fois à votre jeu ne sait pas forcément ce qu'il doit faire pour commencer la partie. Il faut donc lui dire explicitement qu'il doit choisir un avatar.
Pour cela, on va écrire une fonction *modifyStatut* qui prendra un paramètre *message* et qui sera responsable de changer le texte qui est à l'intérieur de notre barre de statut (pense à utiliser la variable globale afficheur…).

Créer une nouvelle variable globale *currentSelection et la mettre à 0.* Appelle la fonction *modifyStatut* dans la fonction *selection*, après ta boucle for, comme ceci :

```
modifyStatut("Joueur " + (currentSelection + 1) + ", choisissez votre avatar");
```

On pourra bien entendu utiliser notre fonction *modifyStatut* à n'importe quel moment dans notre code et plus particulièrement à chaque fois que l'on veut notifier le(s) joueur(s) d'un message.

Sélection des avatars

Grâce à la fonction *addEventListener* que vous avez ajouter sur chaque case, quand un joueur va cliquer sur un avatar à sélectionner, la fonction *setAvatar* sera exécutée.

Avant tout, créer 2 nouvelles variables globales :

```
var gamers = ['', ''];
var gamersNom = ['', ''];
```

La variable *gamers* contiendra la balise img de chaque joueur.
La variable *gamersNom* contiendra le nom d'avatar de chaque joueur.

Voici un exemple des variables une fois remplis :

```
var gamers = ['<img src=”img/aj.gif>', '<img src=”img/r.gif>'];
var gamersNom = ['AppleJack', 'Rarity'];
```

Pour clarifier les deux lignes du dessus :

	Balise img (gif)	**Nom**
Joueur 1	`<img src=”img/aj.gif>`	AppleJack
Joueur 2	`<img src=”img/ar.gif>`	Rarity

Je vais te donner la logique de la fonction *setAvatar*, tu dois l’adapter à ton code Javascript :

```
fonction setAvatar
 | si nom avatar != nom avatar joueur précédent
   | gamersNom[currentSelection] := nom avatar
   | gamers[currentSelection] := balise img
   | alert pour avertir que le joueur currentSelection à choisi tel avatar
   | currentSelection := currentSelection + 1
   | modifyStatut(“Joueur 2, sélectionnez votre avatar”)
   | si currentSelection > 1
     | cacher #selecteur
     | afficher #game
     | lancer la fonction play() // La créer avant
 | sinon
   | alert pour avertir que le joueur doit choisir un avatar différent de son
adversaire
```

Tu auras besoin d’utiliser l’opérateur **this**, check sur google !
Fais un *console.log(this)* dans la fonction setAvatar et regarde ce que t’affiche la console du navigateur, tu pourra peut être en tirer quelque chose ;)

Nous voilà avec le menu de sélection d’avatars ! ;)

Le jeu

Résumons. On a maintenant le design du jeu, le menu de sélection des joueurs. Il nous manque plus qu'une seule chose, le TicTacToe !

Pour cela, nous avons besoin de trouver la logique qui se cache derrière ce jeu. Il faut qu'on se pose les bonnes questions, par exemple :

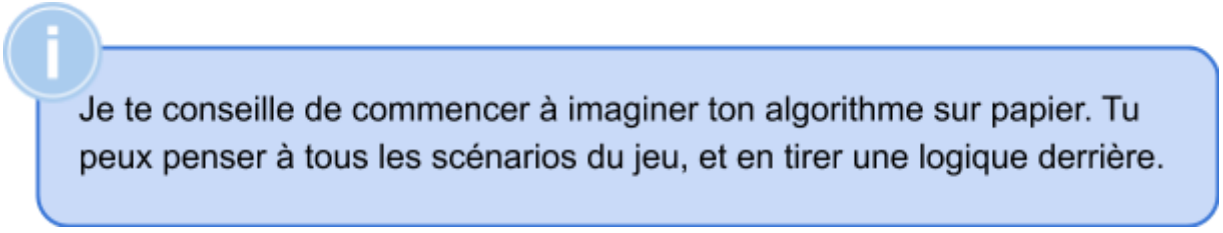
- Dans quelles conditions le jeu se termine ? *(condition d'arrêt d'une boucle for….)*
- Quel joueur commence ? (toujours le joueur 1 ? Au hasard ?)
- Comment savoir si c'est au joueur 1 ou au joueur 2 de jouer ?
- Quand un joueur clique sur une cellule, que se passe-t-il visuellement et qu'est-ce que cela entraîne ?

A toi de jouer !

Dans les parties précédentes, on t'a donné les bases du javascript, ou du moins, les éléments utiles pour coder la logique du jeu TicTacToe.

Apprenti codeur, j'ai une mission pour toi ! Celle d'imaginer un algorithme pour jouer au TicTacToe !

N'oublies pas, **tu peux nous demander de l'aide à tout moment**, on ne mords pas, ou presque pas ! ;)



Lien Git TicTacToe


CONGRATULATIONS