# Leçon 3-2 – Premier projet : le jeu des pays

s1 ----------

Cette leçon se propose de vous guider pas à pas dans le développement d'un petit jeu. Allez-y progressivement.

s2 ----------

Voilà pour une introduction rapide à JavaScript.
Revenons au quiz sur les pays.
Quel est le fonctionnement souhaité pour ce jeu ?
Une carte est affichée ; elle représente plusieurs pays ; un numéro est associé à chaque pays ; sur le côté gauche de la page, on retrouve ces numéros avec en regard une liste déroulante de noms de pays ; il s'agit donc d'associer à chaque numéro le nom du pays qui lui correspond
deux boutons sont également présents sur la page : un bouton qui permet de valider notre choix ; un bouton qui permet de réinitialiser le jeu

Comment allons-nous procéder pour développer ce jeu ?
Nous allons suivre le cheminement suivant

- création du fichier HTML
- développement du code JavaScript
- enfin, introduction d'une feuille de style

Bien entendu, il y plusieurs façons de développer ce jeu. J'ai fait ici des choix pédagogiques qui vous permettront à la fois de découvrir de nouvelles balises HTML et les instructions JavaScript les plus communes. Cet exemple appelle d'ailleurs des améliorations qui feront l'objet de la prochaine leçon.

s3 ----------

Je vais vous demander de reproduire les fichiers HTML et JS suivants, de tester le fonctionnement du jeu, pas à pas. Je vous donnerai des explications sur le comportement du jeu et la nature du code écrit. Cet exemple nous permettra de retrouver les notions vues précédemment : les Variables, les Tableaux, les Chaînes de caractères, les Structures conditionnelles (de type if else) et les Fonctions. En outre, nous découvrirons ici comment prendre en compte des événements souris (un clic sur un bouton notamment).

s4 ----------

Première étape : on s'occupe du fichier HTML

On commence par créer un dossier : "HTML5_Javascript_ex2"
On ouvre éditeur de bas niveau
On crée d'abord le fichier HTML, on le sauve, par exemple jeu_pays.html
On écrit la première ligne indiquant qu'il s'agit d'un fichier HTML et on commence à structurer le document : en-tête, corps
On prépare le lien vers la feuille de style

On introduit les différents éléments dont on a besoin :
le texte du titre
puis un élément initialement vide repéré par une balise form (c'est-à-dire un élément de type formulaire - nous verrons pourquoi plus tard) et qui va contenir toutes les listes déroulantes
autre élément l'image de la carte
enfin, les derniers éléments de cette page : les deux boutons.

Quelques remarques :

- l'élément titre est vide pour l'instant : c'est le code JavaScript qui va le remplir
- les deux éléments de type input sont des boutons auxquels est attaché le nom d'une fonction JavaScript qui est la fonction qui sera lancée au moment du clic. Nous définirons ces fonctions plus tard.
- l'élément form sera défini plus loin
- le placement de la carte à droite des listes déroulantes sera réalisé plus tard grâce à une feuille de style

Tout à la fin, on déclare l'appel au fichier qui contiendra le code JavaScript. Ce code va dans notre exemple initialiser le titre ; il doit donc être lancé une fois que l'élément titre est créé.
Cependant, on a beau placer son code en fin de body, il se peut parfois que ce code n'ait pas d'effet, car les éléments de la page web n'ont pas encore eu le temps d'être identifiés par le navigateur au moment où le code s'exécute.

Nous verrons plus tard comment contourner ce problème.

s5 ----------

Dans l'élément form, on introduit les 7 éléments select (c'est-à-dire les 7 listes déroulantes) précédés d'éléments span qui nous permettent d'isoler le numéro afin plus tard de lui affecter un style.
Dans l'élément select, name représente le nom associé au champ, c'est le nom qui permettra d'identifier la liste ; option représente chaque entrée de la liste, value le nom associé à une entrée, c'est le nom qui permettra d'identifier cette entrée

La carte fait référence à un fichier qui est présent dans le même dossier que le fichier HTML/

On peut d'ores et déjà tester ce fichier en l'ouvrant dans un navigateur.
Comme on pouvait s'y attendre, la carte est pour l'instant placée en dessous des listes déroulantes. On se rappelle en effet qu'en l'absence d'information de positionnement, le navigateur place par défaut les éléments de type div les uns en dessous des autres dans l'ordre dans lequel ils se trouvent dans le fichier HTML. Vous pouvez d'ailleurs faire quelques tests pour vous en convaincre.

s6 ----------

Deuxième étape : le fichier Javacript

On utilise le même éditeur de bas niveau. On ouvre un nouveau fichier et on le sauve sous le nom code.js
Première chose : on va écrire les instructions JavaScript qui permettent de définir le titre (on aurait pu évidemment le définir directement dans le fichier HTML, mais cette façon de faire nous donne l'occasion de réutiliser des instructions connues).

document.getElementById("titre").innerHTML
se lit de la façon suivante
pour ce document (c'est-à-dire le fichier HTML dans lequel ce code est inséré), applique la méthode getElementById (c'est-à-dire, recherche l'élément appelé "titre") et à élément, donne à l'attribut innerHTML (c'est-à-dire son contenu) la valeur située de l'autre côté du signe égale (c'est-à-dire ici un élément de paragraphe).

On peut tester pour vérifier que le code est bien exécuté.

Il reste à écrire les fonctions associées aux deux boutons.
Mais avant cela, on crée un simple tableau pour définir quel pays est associé à un numéro de la carte. Ceci de fait de la façon suivante :
l'instruction
tableauPays = new Array();
permet de déclarer la variable tableauPays comme un tableau
les lignes suivantes définissent les différents éléments du tableau
tableauPays[1] = 'Azerbaïdjan';

Testons d'ores et déjà ces premiers éléments. Profitons de cette pause pour bien comprendre le code que nous avons écrit.

s7 ----------

On s'occupe maintenant da fonction de validation qui, on le rappelle, est lancée lorsque l'on clique sur le bouton "Valider".

Que fait cette fonction ? Cette fonction fait une boucle sur les listes et teste pour chaque liste si c'est le bon pays qui a été choisi.

La récupération de l'index choisi se lit de la façon suivante :
on fait une boucle de 1 à 7
pour un i donné, dans le document, dans le tableau (créé automatiquement) qui répertorie touts les formulaires présents (ici un seul formulaire est présent), cibler celui qui a pour nom listes ; parmi tous les éléments constituant ce formulaire, cibler celui qui pour nom listei (c'est-à-dire liste1, puis liste2, etc.), et récupérer la valeur de l'index de l'option choisie : 0 pour la première, 1 pour la seconde, etc.

La récupération du nom du pays choisi se lit de la façon suivante : pour un i donné dans le document, parmi tous les formulaires présents, cibler celui qui a pour nom listes ; parmi tous les éléments constituant ce formulaire, cibler celui qui pour nom listei (c'est-à-dire liste1, puis liste2, etc.), et récupérer l'attribut value de l'option choisie

Si c'est le bon pays qui a été choisi, on change la couleur de fond du numéro correspondant. Pour l'instant, on met le style en dur, ce qui nous permet de vérifier le jeu, mais nous allons changer cette application de style plus tard

Enfin, une fonction de reset.
Cette fonction fait une boucle sur les listes et affiche le premier choix.

Avec ces quelques lignes de code, vous avez découvert très rapidement comment grâce à JavaScript

- cibler un élément d'un document par son identifiant
- définir un tableau
- définir une fonction associée à un bouton
- écrire une boucle
- récupérer la valeur sélectionnée au sein d'une liste déroulante

Déjà beaucoup de choses que vous pouvez d'ores et déjà réutiliser à l'occasion de plusieurs tests

s8 ----------

Troisième étape : fichier CSS

On utilise le même éditeur de bas niveau, on ouvre un nouveau fichier, que l'on sauve sous le nom styles.js, conformément à la déclaration faite plus tôt dans le fichier HTML

Dans ce fichier, on va d'abord définir un style général pour le document et un style pour le titre

On a vu que le positionnement par défaut des éléments de type div en blocks place la carte sous les listes déroulantes. Il y a plusieurs façons de faire que la carte se retrouve à droite des listes - nous en essaierons d'autres plus loin (association de positionnement relatif et absolu, utilisation d'un positionnement flottant que nous avons eu l'occasion de découvrir dans le chapitre précédent...).
Ici, nous modifions simplement le positionnement par défaut des deux éléments
listes et cartes : on le change en éléments de type ligne, grâce à l'instruction display:inline-block;

s9 ----------

On définit aussi un style pour les boutons pour les écarter un peu des listes.

Enfin, on précise les styles OK et NOK pour les numéros des listes.

s10 ----------

On revient sur code JS des fonctions et on change le code JavaScript afin de prendre en compte ces deux derniers styles : OK si le pays choisi est le bon, NOK dans le cas contraire

Voilà, c'est fini pour ce petit exemple.
Reprenez une à une chaque explication afin de comprendre
la structure générale du jeu et comment chaque fonctionnalité du jeu est traduite en code JS.
Ce jeu nous a permis de découvrir plusieurs éléments que vous pouvez approfondir grâce aux liens proposés dans la dernière diapositive.

s11 ----------

On s'en doute, la duplication des éléments liés aux pays dans le fichier HTML et dans le fichier JS n'est pas optimale, en particulier, si on veut faire évoluer le jeu en changeant le nombre de pays, il faudra faire des modifications dans les deux fichiers.

Nous vous proposons comme exercice, de faire évoluer ce premier jeu de la façon suivante : au

départ le formulaire est vide et les 7 listes sont créées par le code JavaScript.
Je vous donne une indication : faire une boucle sur les pays, créer une chaîne contenant le span et le select, introduire cette chaîne dans l'élément listes

Autre évolution : nous avons évoqué le fait qu'il était important que le code s'exécute après que les éléments qu'il cible sont chargés par le navigateur et disponible.

Pour éviter ce genre de problèmes, nous allons faire en sorte que le code JavaScript ne se lance qu'une fois le document prêt. Ceci se fait en plaçant l'ensemble du code Java-Script (attention : pas les fonctions) à l'intérieur de l'expression

window.onload = function(){

ne pas oublier de refermer l'accolade tout à la fin du code

À vous de jouer !