Leçon 8.1.2 – HTML5 : Méthodologie

s1 ----------

s2 ----------

Je vous propose de réaliser un petit jeu qui m'a été soumis par un des participants à ce MOOC lors d'une précédente session. Et je le remercie pour cette contribution.

Le principe de ce jeu est le suivant : des bulles ou des balles de taille différentes parcourent l'écran de haut en bas ; la souris contrôle librement la bulle bleue qui doit toucher le maximum de bulles vertes mais en essayant de ne pas toucher les noires ; le score est incrémenté à chaque bulle verte touchée. Le nombre de vies est décrémenté à chaque bulle noire touchée.

Plusieurs niveaux existent, cependant le passage d'un niveau à l'autre se fait sans aucune interruption du jeu ; à chaque changement de niveau, le nombre de bulles croit ainsi que leur vitesse, et donc la difficulté du jeu.

J'espère que le fonctionnement ainsi énoncé est clair.

On devine que ce jeu illustre presque tous les chapitres précédents : un peu d'HTML, bien sûr, du CSS, du code JavaScript, des méthodes de dessin, d'animation et de l'interactivité souris...

s3 ----------

Je vous propose de réaliser ensemble la conception de ce jeu et d'aborder successivement

Le cahier des charges fonctionnel :

- la liste des écrans et leur contenu potentiel,
- la liste des règles et fonctionnalités ;

et la conception détaillée :

- les données importantes,
- les éléments généraux,
- les spécifications détaillées, c'est-à-dire la façon dont l'affichage des écrans et règles et fonctionnalités seront traduites du point de vue technique,
- l'architecture de fichiers.

D'un point de vue pratique, je vous propose de vous munir d'une gomme, d'un porte-mine et d'une feuille A3 (ou de deux feuilles de papier A4 accolées) et de reproduire ce que je vais vous présenter à l'écran. En effet, avant de rédiger une version aboutie du cahier des charges fonctionnel et de la conception détaillée, un travail de construction progressif et au brouillon s'impose à nous.

Faites-le vraiment ! C'est ainsi que personnellement, la plupart du temps, je procède, pour de petits ou même de moyens projets.

Allons-y !

s4 ----------

Commençons par établir la liste des écrans et leur contenu potentiel. Pourquoi potentiel ? Parce que ce contenu peut disparaître ou apparaître après une action utilisateur ou un temps donné.

Dans ce jeu, j'ai envie d'imaginer trois écrans

- un écran d'accueil,
- un écran de jeu,
- un écran de bilan.

L'écran initial étant bien entendu l'écran d'accueil.

Représentons-le par des rectangles. Il est utile de donner d'ores et déjà un nom à ces écrans, pourquoi pas en reprenant la notation CSS ou jQuery qui permet de cibler un élément : #accueil, #jeu, #bilan

Pour pouvoir préciser leur contenu et également les règles et fonctionnalités, il n'y a pas d'autre solution que de s'imaginer en train de jouer à ce jeu.

Commençons par l'écran d'accueil, puisque c'est le premier qui s'affiche à l'utilisateur. Dans l'écran d'accueil, il y aura par exemple :

- un titre
- une image du jeu
- un texte expliquant les règles
- un bouton pour lancer le jeu et passer à l'écran suivant

Dans l'écran de jeu, il y aura :

- un champ de texte qui affichera le score
- un champ de texte qui affichera le niveau atteint
- un élément qui va afficher les bulles (sans doute de type canvas, mais qui sera précisé plus tard)
- un bouton pour quitter le jeu et revenir à l'écran d'accueil

Dans l'écran de bilan, il y aura :

- à nouveau un champ de texte qui affichera le score
- un champ de texte qui affichera le niveau atteint
- un bouton pour revenir à l'écran d'accueil

- un autre pour rejouer sans passer par l'écran d'accueil

Dans tous les écrans, on trouvera un pied de page indiquant par exemple l'auteur et la date.

On reproduit tout cela sur la feuille.

À nouveau, il peut être utile de donner un nom à ces éléments ou conteneurs, pourquoi pas en reprenant la notation CSS ou jQuery : #titre, #image, #texte, etc. Pas de souci à ce stade si certains noms d'éléments se retrouvent dans plusieurs écrans, par exemple le pied de page.

Voilà pour la liste des écrans et leur contenu potentiel. Il est important d'avoir bien identifié tout le contenu potentiel des écrans. À ce stade, vous avez sans doute déjà identifié quelques fonctionnalités.

s5 ----------

On s'intéresse à présent à la liste des règles et fonctionnalités

La distinction opérée ici entre règles et fonctionnalités est importante : j'entends par fonctionnalité quelque chose de nature à changer l'aspect des écrans ou la valeur des variables JavaScript et qui implique l'utilisateur (comme un clic sur un bouton qui permet de changer d'écran) ; j'entends par règle quelque chose de nature à changer l'aspect des écrans ou la valeur des variables JavaScript, mais qui se produit hors utilisateur et doit être testé en continu (comme dans ce jeu, le fait que l'on pourrait passer automatiquement à l'écran de bilan après un certain temps de jeu).

Que se passe-t-il ici en termes de règles et fonctionnalités ? À nouveau, pas d'autre solution que de s'imaginer en train de jouer à ce jeu.

Première chose, l'écran d'accueil s'affiche au chargement (et masque les autres, bien entendu). C'est si l'on veut la première fonctionnalité : elle est liée au fait que l'utilisateur charge la page HTML.

Dans l'écran d'accueil, sans condition, tous les éléments potentiels s'affichent : le titre, l'image, le texte et le bouton.

Quelles sont les règles et fonctionnalités de l'écran d'accueil ? Dans cet écran pas de règle, mais une seule fonctionnalité :

- si on clique sur le bouton #boutonJeu alors on passe à l'écran #jeu (et on masque les autres écrans).

Dans cet écran de jeu, sans condition, tous les éléments potentiels s'affichent : le score, le niveau, l'animation avec les bulles et le bouton quitter. Cependant, le score, le niveau et l'animation sont

soumis à plusieurs règles ou fonctionnalités :

Premier élément a priori de type fonctionnalité (mais ici, tout à fait franchement, la distinction entre règle et fonctionnalité n'est pas évidente) :

- si on déplace la souris alors la bulle bleue suit le mouvement ;

De plus

- si la bulle bleue touche une bulle verte (ceci n'implique pas forcément un mouvement de l'utilisateur) alors cette dernière disparaît et le score est augmenté
- si la bulle bleue touche une bulle noire alors cette dernière disparaît et le nombre de vies est diminué

Autres règles

- au bout d'un certain temps de jeu, le niveau change et la vitesse des bulles augmente
- si le temps de jeu dépasse un certain nombre de secondes alors on passe à l'écran de bilan

Une dernière fonctionnalité

- si on clique sur le bouton #boutonQuitter alors on revient à l'écran d'accueil et le jeu est réinitialisé

Nous verrons plus loin comment écrire ces règles de façon plus cohérence et logique. À ce stade, il s'agit simplement de décrire le mieux possible le fonctionnement. De la même façon, ce n'est pas encore ici que l'on essaie de réfléchir à la mise en œuvre de ces règles, c'est-à-dire comment on va s'y prendre pour faire descendre les bulles.

Intéressons-nous à l'écran de bilan. Dans cet écran, au départ tous les éléments potentiels s'affichent : le score, le niveau atteint et les deux boutons.

Quelles sont les règles et fonctionnalités de l'écran de bilan. Pas de règle, mais deux fonctionnalités :

- si on clique sur le bouton #boutonRejouer alors on repasse à l'écran #jeu (et on masque les autres écrans) ;
- si on clique sur le bouton #boutonAccueil alors on passe à l'écran #accueil (et on masque les autres écrans).

On reproduit tout cela sur la feuille. Voilà pour les règles et fonctionnalités.

s6 ----------

À ce stade, avec ces écrans, leur contenu, ces règles et fonctionnalités, on a déjà une bonne idée du fonctionnement du jeu. Pour l'instant nous nous sommes cantonnés à décrire le fonctionnement du jeu. Nous passons à la Conception détaillée qui consiste à essayer de traduire du point de vue technique la conception générale. Commençons par identifier les données importantes.

Dans ce jeu, il y a plusieurs niveaux de difficulté variable. Essayons de nous représenter le plus clairement possible la relation entre jeu, niveau et bulles.

La figure ci-dessous est une façon de représenter ces relations : un jeu possède plusieurs niveaux, pour chaque niveau on trouve un certain nombre de bulles qui sont utilisées seulement pour ce niveau. Les traits lorsqu'ils sont suivis de gauche vers la droite traduisent une relation de possession ; lorsqu'ils sont suivis de la droite vers la gauche traduisent une relation d'appartenance.

Voici un autre type de dépendance : un jeu possède 5 niveaux ; chaque niveau possède un certain nombre de bulles, mais à la différence de la représentation précédente, certaines bulles sont utilisées par un seul niveau, d'autres par plusieurs niveaux.

Voici un troisième type de dépendance : un jeu possède 5 niveaux ; mais ici toutes les bulles sont jouées par chaque niveau ; chaque niveau possède et affichera toutes les bulles.

De façon plus abstraite, nous pouvons représenter (ou modéliser) ces relations de dépendance de la façon suivante.

Dans le premier cas, un jeu a un titre unique (c'est une chaîne de caractères) et possède 5 niveaux ; ces niveaux n'appartiennent qu'à ce jeu ; un niveau contient un certain nombre de bulles (d'où le petit n que l'on voit sur la droite du trait qui mène à la boite Balle) ; ces bulles sont différentes d'un niveau à l'autre ; elles n'appartiennent qu'à ce niveau (d'où le chiffre 1 que l'on voit sur la gauche du trait qui retourne vers la boite Niveau ; dans cet exemple, les bulles ont même une couleur choisie dans un ensemble de trois couleurs (du gris clair au noir et le vert pour les bulles qu'il faut toucher) et une taille choisie dans un ensemble de trois tailles.

Dans le deuxième cas, un jeu possède 5 niveaux ; ces niveaux n'appartiennent qu'à ce jeu ; un niveau contient un certain nombre de bulles (d'où le petit n que l'on voit sur la droite du trait qui mène à la boite Balle) ; ces bulles peuvent appartenir à plusieurs niveaux (d'où le chiffre n que l'on voit sur la gauche du trait qui retourne vers la boite Niveau.

Dans le troisième cas, un jeu possède 5 niveaux ; ces niveaux n'appartiennent qu'à ce jeu ; un niveau contient la même liste de bulles. Les bulles ont même une couleur choisie dans un ensemble de couleurs (du gris clair au noir plus le vert pour les bulles qu'il faut toucher) et une taille donnée.

Nous n'allons pas poursuivre dans la voie consistant à modéliser de façon abstraite – grâce à des langages tels que UML, notamment –, les données importantes du jeu, car cela dépasse le cadre de ce MOOC. Mon intention ici est simplement de vous sensibiliser à l'importance de bien réfléchir avant toute programmation à la structure de ces données. Pourquoi ? Pour deux raisons :

- ces choix de structure initiaux ont une grande influence sur la plus ou moins grande facilité à traduire en code JavaScript les règles et fonctionnalités du jeu ;
- de ces choix initiaux dépend également la plus ou moins grande facilité à faire évoluer le jeu : à rajouter un niveau, à rajouter une balle à un niveau, à changer une consigne, à changer les couleurs des bulles, etc.

s7 ----------

Dans ce jeu, il y aura a priori un grand nombre de bulles (quelques centaines). Nous avons le choix entre les décrire toutes (par exemple dans une liste) ou les générer aléatoirement (c'est-à-dire leur attribuer aléatoirement une couleur, une taille et une vitesse)

Décrire toutes les bulles est fastidieux, mais nous permet de contrôler la difficulté du jeu. Générer aléatoirement les bulles est relativement facile grâce à JavaScript. Cependant, il sera plus difficile – mais pas impossible – de garantir que le jeu soit jouable. C'est c'est solution que nous allons mettre en place.

s8 ----------

Dans cette partie, rappelons-le, il nous faut penser aux éléments généraux tels que la façon dont l'application est organisée à l'écran ; il faut également penser à l'esthétique générale, aux choix des couleurs, des polices, etc. ; il faut penser aux dimensions générales de notre application.

Ici aussi, un petit croquis est utile pour rassembler la plus grande partie de ces informations. Je vous invite à le reproduire sur votre feuille.

Éléments communs aux écrans :

- fond de page gris clair #ccc

- un pied de page : Arial, noir, 12 pts

Pour l'écran d'accueil :

- le titre : Arial, rouge, 16 pts

- une image du jeu 450px *300px

- le texte expliquant les règles : Arial, noir, 12 pts

- le bouton pour lancer le jeu : au milieu

Pour l'écran de jeu :

- le texte pour la consigne : à nouveau Arial, noir, 12 pts

- un élément de type canvas 450 *500 à l'intérieur duquel évolueront les bulles de couleurs sur un fond, par exemple blanc : #fff

- le bouton pour quitter le jeu : à droite

Dans l'écran de bilan, il y aura :

- le texte du bilan : à nouveau Arial, noir, 12 pts

- un bouton pour revenir à l'écran d'accueil à droite

- un autre pour rejouer sans passer par l'écran d'accueil : à gauche du précédent

Ce travail nous permet de lister certaines classes CSS, on crée par exemple la classe paragraphe – on peut aussi prendre tout simplement comme sélecteur la balise p – pour les textes de règles, de consigne, de bilan :

```
.paragraphe {$
        font...
        color...
}
```

Ce croquis peut être encore plus précis : nous pourrions préciser les marges et bien d'autres éléments. Ce que nous ne ferons pas ici pour alléger le propos. Mais n'oubliez pas une chose : tout ce à quoi vous réfléchissez pendant la phase de conception ne sera plus à définir !

Dans une autre vidéo, nous allons réfléchir à la façon dont le code JavaScript va créer la structure des écrans et leur contenu et permettre toutes les règles et fonctionnalités que nous avons identifiées.