IUT- Département d'Informatique
G. Rey
2016

Programmation Web – client riche
M4103 - TD n° 3 (séance 4)

1 Objectifs

Ce TD illustre la partie du cours sur la programmation web et en particulier les concepts liés à la conception d'application riches coté client, c'est-à-dire dans le navigateur. Les concepts principaux abordés seront:

- Les principes avancés du langage JavaScript
- L'exploitation du DOM
- Mise en œuvre d'une application en JavaScript

Un compte rendu donnant la réponse à chacune des questions posées devra être rendu à votre responsable de TD par email à la fin de chacune des séances. L'objet (sujet) de l'email devra être le suivant :

[S4T][JS][Gx][TDx]Nom-Prénom / Nom-Prénom

Avec Gx le numéro de votre groupe (exemple G1 pour le groupe 1) et TDx le numéro du TD inscrit sur le sujet du TD (exemple TD1 pour le premier TD).

Vous ne pouvez pas faire 2 séances avec le même binôme. De même, vous ne pouvez pas faire plus de la moitié des séances seul sauf si votre groupe de TD est assez petit pour que chaque étudiant soit sur une machine de l'IUT.

Veuillez, sauf indication contraire de votre responsable de TD, faire l'ensemble des exercices marqués « Obligatoire » dans l'ordre de la feuille de TD. Les exercices « Conseillés » sont à faire si vous avez terminé les précédents ou lors de vos révisions. Les exercices « optionnels » sont là pour les plus passionnés d'entre vous.

Attention, il faut lire l'exercice en entier avant de vous lancer dans sa réalisation.

2 Documentation (obligatoire)

Pour ce TD mais également pour les TD suivant, sauf mention contraire dans l'énoncé ou si un lien vous est fourni, vous ne devez pas aller chercher des réponses sur internet !! Les seuls sites suivants sont autorisés :

- Le site contenant les supports de cours et de TD.
- Le site de Mozilla : Mozilla Developer Network.
- Les sites du W3C : pour la validation HTML5 ou du CSS. Mais aussi pour avoir la documentation sur le DOM.

3 Prise en main de l'environnement (obligatoire)

Pour réaliser ce TD, vous aurez également besoin d'un éditeur de texte pour écrire le code de vos pages. Vous pouvez par exemple utiliser un des logiciels Notepad++, ConText, Quanta+, WebExpert,... Microsoft vous propose de télécharger gratuitement Expression Web.

Pensez à sauvegarder régulièrement votre travail et à commenter votre code. N'hésitez pas à sauvegarder les différentes versions, évolutions d'un même exercice de manière à pouvoir facilement réviser ou le réutiliser par la suite.

4 Exercice 1 : Taquin (obligatoire)

Pour cet exercice vous devez partir d'une page HTML5 ci-dessous et du fichier CSS donné. Aucun de ces deux éléments ne pourra être modifié.

La page HTML :

Programmation Web – client riche
M4103 - TD n° 3 (séance 4)


```
<!DOCTYPE html>
<html lang="fr">
	<head>
		<title>Jeu de taquin</title>
		<link rel="stylesheet" type="text/css" href="TD3.css" />
		<script type="text/javascript" src="./script.js"> </script>
	</head>
	 <body onload="init()">
		<div id="jeu">
			<div style="top:0px;left:0px;" class="case"><h1>6</h1></div>
			<div style="top:0px;left:102px;" class="case"><h1>1</h1></div>
			<div style="top:0px;left:204px;" class="case"><h1>7</h1></div>
			<div style="top:102px;left:0px;" class="case"><h1>3</h1></div>
			<div style="top:102px;left:102px;" class="case"><h1>4</h1></div>
			<div style="top:102px;left:204px;" class="case"><h1>8</h1></div>
			<div style="top:204px;left:0px;" class="case"><h1>5</h1></div>
			<div style="top:204px;left:102px;" class="case"><h1>2</h1></div>
			<div style="top:204px;left:204px;" class="case vide"><h1> </h1></div>
		</div>
	 </body>
 </html>
```

Le fichier CSS

```
html { font-size: 62.5%; }
body { margin: 0px; padding: 0px; }
#jeu {
	border-style:solid;
	border-width:1px;
	background-color:#00008B;
	margin:0px;
	width: 306px;
	height:306px;
	position: absolute;
}
.case {
	border-style:solid;
	border-width:1px;
	background-color:#1E90FF;
	margin:0px;
	padding: 0px;
	width: 100px;
	height:100px;
	position: absolute;
	z-index:1;
	transition-property : top, left;
	transition-duration : 1s;
}
h1{	text-align: center; font-size: 4rem; }
.vide {	background-color:#00008B; z-index:0; border:none}
```

- Pour cet exercice on n'utilisera pas les méthodes getElementByXX et getElementsByXX. A vous de trouver d'autres solutions (on en a vu en cours).

Programmation Web – client riche
M4103 - TD n° 3 (séance 4)

- Ecrivez la fonction *init()* qui ajoute (de manière propre) un écouteur sur l'évènement « click » pour chaque élément HTML de type DIV étant dans le classe CSS « case ».
- Ecrivez une fonction *selection()* qui échangera la position de la case vide avec la case que laquelle on vient de cliquer si et seulement si, ces deux cases ont un coté commun.
- Vous n'avez pas le droit, pour le moment d'écrire d'autres fonctions.

- Que pouvez-vous dire de l'architecture de l'application ?

5 Exercice 3 : 2048 (obligatoire)

Alors que dans le premier exercice on s'est concentré sur la partie graphique de l'application (et on a utilisé les éléments HTML et le DOM pour modéliser la grille de jeu), dans ce deuxième exercice nous allons nous concentrer sur les fonctionnalités et la gestion des données en JavaScript. En effet, le premier exercice mélangeait volontairement la partie « noyau fonctionnel » de l'application et la partie présentation. Dans ce deuxième exercice, nous allons faire l'effort de séparer ces deux parties.

En partant du fichier HTML suivant (que vous n'avez pas le droit de modifier) écrivez le fichier CSS et JavaScript pour réaliser le jeu 2048 (vous devrez suivre les consignes données).

```
<!DOCTYPE html>
<html lang="fr">
	<head>
		<title>Jeu 2048</title>
		<link rel="stylesheet" type="text/css" href="TD3.2.css" />
		<script type="text/javascript" src="./script2.js"> </script>
	</head>
	 <body onload="init()">
		<div id="jeu">
			<div class="ligne">
				<div style="top:0px;left:0px;" class="case"></div>
				<div style="top:0px;left:102px;" class="case"></div>
				<div style="top:0px;left:204px;" class="case"></div>
				<div style="top:0px;left:306px;" class="case"></div>
			</div>
			<div class="ligne">
				<div style="top:102px;left:0px;" class="case"></div>
				<div style="top:102px;left:102px;" class="case"></div>
				<div style="top:102px;left:204px;" class="case"></div>
				<div style="top:102px;left:306px;" class="case"></div>
			</div>
			<div class="ligne">
				<div style="top:204px;left:0px;" class="case"></div>
				<div style="top:204px;left:102px;" class="case"></div>
				<div style="top:204px;left:204px;" class="case"></div>
				<div style="top:204px;left:306px;" class="case"></div>
			</div>
			<div class="ligne">
				<div style="top:306px;left:0px;" class="case"></div>
				<div style="top:306px;left:102px;" class="case"></div>
				<div style="top:306px;left:204px;" class="case"></div>
				<div style="top:306px;left:306px;" class="case"></div>
			</div>
		</div>
```

Programmation Web – client riche
M4103 - TD n° 3 (séance 4)


```
        </body>
</html>
```

Attention, il est conseillé de lire l'ensemble des consignes avant de commencer à coder.

- La grille sera modélisée sous la forme d'un tableau de tableau et initialisée dans une fonction init().
- Chaque case de la grille contiendra un objet maCase ayant une valeur et un booléen (false à l'initialisation) qui servira à indiquer si la valeur contenue est une nouvelle valeur insérée dans la grille par la fonction insertionValeur().
- La fonction init() :
  - Créera la grille, l'initialisera (chaque case aura pour valeur une chaîne vide).
  - Abonnera la fenêtre à l'action « appui sur une touche du clavier » (la fonction appelée sera actionClavier()).
  - Ajoutera 2 valeurs aléatoirement dans la grille.
  - Affichera la grille via la fonction afficherGrille().
- La fonction actionClavier() appelera les fonctions deplacementVersHaut() , deplacementVersBas(), deplacementVersGauche() et deplacementVersDroite() en fonction de la touche de direction pressée. Les autres touches seront ignorées. Les touches qui nous intéressent ont un keyCode entre 37 et 40.
- La fonction afficherGrille() mettra à jour le DOM en fonction de la grille. La dernière case ajoutée sera en rouge. Pour le moment on ne s'occupe pas des autres effets graphiques (couleur des cases, animations, …).
- Chacune des fonctions deplacementVersXXX fonctionnera de la fonction suivante :
  - Pour toutes les lignes (ou les colonnes en fonction du sens),
    - si la ligne courante (ou la colonne) n'est pas vide
      - tasser les cases dans le sens demandé
      - fusionner les cases de même valeur
      - tasser à nouveau les cases pour supprimer les blancs générés par la fonction de fusion
  - S'il y a eu au moins un mouvement ou une fusion dans les actions précédentes
    - ajouter une nouvelle valeur dans la grille
    - afficher la grille
- L'ajout d'une valeur dans la grille sera faites par la fonction insertionValeur(valeur, coordonnée)
  - la valeur sera déterminée par une fonction obtenirNouvelleValeur() qui retournera un 2 avec une probabilité de 0.9 et un 4 dans le reste des cas. La fonction Math.random() vous retourne une valeur dans l'intervalle [0 ; 1[.
  - la coordonnée sera un objet composé de deux attributs, ligne et colonne. Elle correspondra à une case vide de la grille, déterminée aléatoirement par la fonction obtenirCaseVide()
- Les fonctions tasserVersXXX(ligne) ou tasserVersXXX(colonne) déplaceront l'ensemble des éléments de la ligne (ou de la colonne) dans la direction spécifiées pour supprimer les cases vides. Il n'y a pas de fusion de nombre dans ce cas.
- Les fonctions fusionnerVersXXX(ligne) ou fusionnerVersXXX(colonne) fusionneront les éléments de même valeur qui sont cote à cote. En respectant la direction spécifiée. Attention, ici on ne déplace pas de cases. Certaines doublent de valeur, d'autres deviennent vide.
- Les fonctions estLigneVide(ligne) et estColoneVide(colonne) retourne vrai si la ligne (ou la colonne) ne contient que des éléments de valeur vide.

6 Exercice 3 : Taquin, refonte (conseillé)

Maintenant que vous avez vu comment écrire une application en JavaScript indépendante du DOM, à vous de modifier le jeu de taquin pour respecter cela.

Programmation Web – client riche
M4103 - TD n° 3 (séance 4)

- Modifiez votre jeu de taquin pour avoir un ensemble de fonction qui représente le modèle de la grille (initialisation aléatoire de la grille, victoire au jeu, …)
- Liez votre modèle aux DOM via un ensemble de fonctions qui ne font que modifier l'affichage en fonction de l'état du modèle.

7 Exercice 4 : Amélioration du jeu 2048 (optionnel)

Même si nous avons mis en place, lors de l'exercice 2, la mécanique principale d'un jeu de type 2048, il reste encore pas mal de travail à faire pour s'approcher de l'original.

- Ajoutez une fonction estVictoire() qui retourne vrai si le joueur a obtenu une case d'une valeur égale à 2048 pour la première fois.
- Ajoutez une fonction estDefaite() qui retourne vrai s'il n'y a plus de case vide dans la grille et si aucun des déplacements ne provoque une fusion.
- Ajoutez une fonction, qui lors de la mise à jour de l'affichage, modifie la couleur (via le css) dans case en fonction de leur valeur.
- Ajoutez une fonction qui permet d'animer le déplacement des cases.

8 Conclusion

J'espère que ce troisième TD vous a permis de découvrir ou redécouvrir le JavaScript ainsi que son lien avec le CSS et le HTML (via le DOM) et vous a donné une idée des possibilités qu'offre ce langage.