Programmation Web Avancée Côté Client
TP n° 5

Licence Informatique 2ème année
Université de Nice-Sophia Antipolis

Introduction

Maintenant que nous connaissons le mécanismes de base du langage JavaScript et que nous avons fait un tour d'horizon de ses objets intégrés, nous avons tous les prérequis pour affronter des problèmes qui demandent l'utilisation d'algoritmes et de structures de données non triviales.

Pour ne pas disperser nos efforts, nous allons faire un exercice préparatoire au projet du cours de cet année. Notamment, nous allons construir un échiquier et afficher les pièces du jeu d'échecs dans leur position initiale. Nous définirons les pièces comme autants d'objets JavaScript, incorporants leurs règles de déplacement. Nous allons rendre l'échiquier interactif : lorsque l'utilisateur cliquera sur une case contenant une pièce, notre sript devra mettre en évidence l'ensemble des cases dans lesquelles la pièce sélectionnée peut être déplacée.

1 Représentation de l'échiquier

Nous placerons la plupart de notre script JavaScript dans un fichier externe, que nous appellerons « `TP5-cor.js` ».

1. Définissez un constructeur `Echiquier` qui crée une représentation interne d'un échiquier.
2. Nous représenterons l'échiquier par un tableau de 64 cases. Initialisez ce tableau de façon à ce que chaque case soit un objet, pour l'instant sans aucune propriété ni méthode.
3. Dotez le prototype de ce constructeur d'un accesseur `cell`(i, j) qui renvoie l'élément du tableau correspondant à la case avec coordonnés (i, j).

2 Affichage de l'échiquier

Pour l'affichage de l'échiquier, nous utiliserons une table HTML.

1. Préparez une page HTML comme celle montrée en Figure 1.
2. Écrivez une méthode `afficher`(id) qui affiche l'échiquier dans la table HTML ayant l'identifiant *id*. Dotez les cases d'un fond blanc ou marron, selon leur position. La première case en haut à gauche doit être blanche.

```
<html>
<head>
<title>Corrig&eacute; TP 5</title>
<meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
<script src="TP5-cor.js">
</script>
</head>
<body onclick="deselect()">
<h1>Corrigé TP 5</h1>

<table summary="échiquier" id="échiquier" border="1">
</table>
<script>
var echiquier = new Echiquier();
echiquier.afficher("échiquier");
</script>
</body>
</html>
```

FIGURE 1 – Prototype du fichier HTML à créer.

3 Représentation des pièces

Il existe six types de pièces : *roi*, *dame*, *tour*, *fou*, *cavalier* et *pion*. Chaque type de pièce possède ses propres règles de déplacement. En plus, chaque pièce a sa couleur (blanc ou noir).

1. Écrivez un constructeur `Piece` qui crée une pièce d'un type (R, D, T, F, C ou P) et couleur (B ou N) donnés.
2. Une règle de déplacement peut être regardée comme une fonction booléenne $f(e, x_0, y_0, x, y)$, qui prend un échiquier e, une position de départ x_0, y_0 et une position d'arrivée x, y en entrée et répond *vrai* si ce déplacement est légal et *faux* autrement. Écrivez six fonctions de ce type pour chaque type de pièce ; appelez-les `depl`X, où X est le type de pièce. Ces fonctions deviendront des méthodes des objets pièces. Attention : la fonction des pions devra prendre en compte la couleur de la pièce, parce que les pions ne peuvent pas reculer. La raison pour laquelle le paramètre e est nécessaire est que, pour vérifier si un déplacement est valide, il faut savoir si certaines cases sont libres ou occupées. Nous paramétrons par e plutôt que nous référer à une variable globale pour obtenir un code plus général.
3. Modifiez le constructeur `Piece` pour qu'il affecte à la propriété `depl` de la pièce créée la bonne règle de déplacement (qui est, rappelons-nous, une fonction et, donc, deviendra une méthode de la pièce).
4. Modifiez le constructeur de l'échiquier pour disposer sur l'échiquier toutes les pièces des deux joueurs dans leur position initiale. Pour représenter le fait qu'une pièce est positionnée sur una case, il suffira d'ajouter une propriété `piece`, à la quelle on affectera l'objet qui représente la pièce. Une case vide aura `piece` == `undefined`.
5. Modifiez la méthode d'affichage pour afficher la lettre correspondante à chaque pièce en blanc ou noir.

4 Gestion des événements

1. Prédisposez une manière de récupérer les éléments HTML correspondants aux cases de l'échiquier. Il y a au moins deux possibilités : (i) équipper les éléments avec des identifiants qui les rendent accessibles par la méthode `getElementById` ; (ii) ajouter aux objets qui représentent les cases une propriété `td` qui fasse référence à l'élément TD pertinent ;
2. Modifiez le code que vous avez développé pour l'Exercice 2 en sorte qu'il ajoute aux céllules de la table HTML la propriété `onclick`, liée à une fonction de gestion de l'événement, afin de pouvoir réagir lorsque l'utilisateur clique sur une case de l'échiquier.

5 Mise en évidence des cases accessibles

1. Réalisez une fonction de gestion de l'événement emphclick sur une case de l'échiquier de façon à ce que si l'utilisateur clique sur une case contenant une pièce, les cases dans lesquelles cette pièce pourrait être déplacée changent leur fond en jaune.
2. Faire en sorte que si l'utilisateur clique dans une case vide ou hors de l'échiquier les cases en jaune reprennent leur couleur normale.