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 labyrinthe aléatoire, l'afficher sur une page HTML en utilisant une table et le rendre interactif : lorsque l'utilisateur cliquera sur une case du labyrinthe, notre sript devra mettre en évidence l'ensemble des cases accessibles à partir de la case sélectionnée.

1 Génération du labyrinthe

Le labyrinthe que nous allons générer est composé par un tableau de $n \times n$ cases (nous outiliserons $n = 7$ pour faire des tests). Chaque case peut être :

— soit un couloir orienté Nord-Sud ou Est-Ouest ;
— soit un coin avec deux entrées à Sud et à Ouest, à Est et à Sud, à Nord et à Est ou à Ouest et à Nord ;
— soit un « croisement à T » avec trois entrées : Est-Sud-Ouest, Nord-Est-Sud, Ouest-Nord-Est ou Sud-Ouest-Nord.

Cela donne un total de 10 types de cases possibles.

1. Préparez une page HTML pour contenir le script et la table qu'on construira dans l'Exercice 2 ;
2. Définissez un constructeur de cases du labyrinthe, avec son prototype où vous placerez les méthodes dont vous aurez besoin ;
3. Faites en sorte que, lorsque le constructeur est appelé sans spécifier le type de case souhaité, le type de la case créée soit détérminé de façon aléatoire avec les probabilités suivantes :

type	NS	EO	SO	ES	NE	ON	ESO	NES	ONE	SON
Pr(type)	12%	12%	10%	10%	10%	10%	9%	9%	9%	9%

4. Créez un objet `Labyrinthe`, qui représentera le labyrinthe. Dotez cet objet d'un accesseur à la case de coordonnés (i, j), avec $i, j \in \{0, \ldots, n-1\}$.

2 Affichage du labyrinthe

Pour l'affichage du labyrinthe, nous utiliserons une table HTML. Si on avait plein de temps, ce serait amusant de dessiner avec un éditeur graphique 10 images carrées de la même taille correspondantes aux 10 types de cases dont on a parlé à l'Exercice 1. Pourtant, nous n'avons pas de temps à perdre, donc nous irons utiliser la couleur de fond des cellules de la table HTML pour « dessiner » les cases. Cependant, il nous faudra 9 cellules (3×3) de la table pour dessiner une case, ce qui rend cet exercice un peu moins trivial de ce qu'il pouvait sembler...

Par exemple, une case du type ESO sera représentée par un morceau de table comme le suivant :

noir	noir	noir
blanc	blanc	blanc
noir	blanc	noir

1. Écrivez une méthode de l'objet `Labyrinthe` qui affiche la structure du labyrinthe par une table HTML de $3n \times 3n$ cellules.
2. Utilisez la propriété `style` des éléments HTML correspondants aux cellules de la table pour changer la propriété CSS `background-color`.

3 Gestion des événements

1. Pédisposez une manière de récupérer les éléments HTML correspondants aux cases du labyrinthe. 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 des propriétés qui fassent référence aux éléments HTML pertinents ;
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 du labyrinthe.
3. (Facultatif) Ajoutez un bouton qui permette à l'utilisateur de régénérer aléatoirement le labyrinthe (et, par conséquence, de mettre à jour la table HTML).

4 Calcul de l'ensemble accessible

Deux cases du labyrinthe c_1 de coordonnéss (i_1, j_1) et c_2 de coordonnéss (i_2, j_2) sont *adjacentes* si et seulement si $|i_1 - i_2| + |j_1 - j_2| = 1$.

Deux cases adjacentes c_1 et c_2 sont communiquantes si et seulement si c_1 a une entrée sur le côté en commun avec c_2 et c_2 a une entrée sur le côté en commun avec c_1.

Une case c du labyrinthe est accessible à partir d'une case initiale c_0, s'il existe une suite finie de cases adjacentes communiquantes telle que la première case de la suite est c_0 et la dernière est c.

1. Écrivez une méthode qui, étant données les coordonnées d'une case initiale du labyrinthe, calcule l'ensemble des cases accessibles à partir de la case spécifiée.
2. Concevez une manière de représenter cet ensemble qui vous facilite la tâche que vous devrez affronter dans l'Exercice 5.

5 Mise en évidence des cases accessibles

1. Écrivez une méthode de gestion des événements `onclick` qui mette en évidence dans la table HTML les cases accessibles à partir de la case cliquée par l'utilisateur. Changez la couleur du fond (par exemple, de blanc à jaune) pour mettre une case en évidence.
2. Pensez à remettre à blanc le fond de toutes les cases en évidence avant de gérer un nouveau événement `onclick`.
3. (Facultatif) Faites en sorte que si l'utilisateur clique hors de la table, toutes les cases en évidence soient remises en blanc.