SCIENCES SORBONNE UNIVERSITÉ

LICENCE INFORMATIQUE

RAPPORT DU PROJET MPIL

O'Rush Hour – Partie 2

3i008, Licence d'Informatique L3

REALISER PAR:
DJEDDAL Hanane.

2018/2019

INTRODUCTION

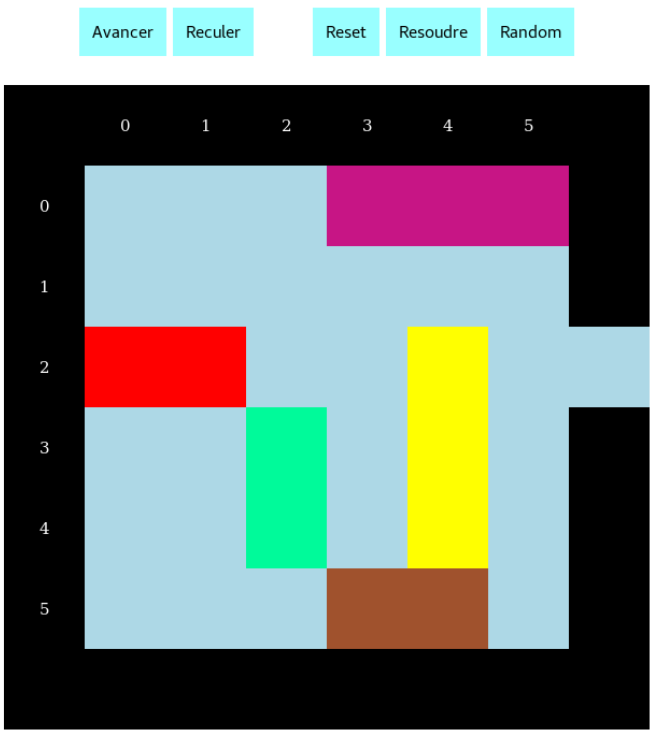
Rush Hour est un casse-tête dont le but est de faire sortir d'un terrain de jeu rectangulaire une voiture, alors que le chemin vers la sortie est encombré par d'autres véhicules.

Après avoir programmé la logique du jeu dans une première partie en Ocaml, l'objectif est maintenant de réaliser une présentation graphique de ce jeu dans un langage adéquat et de lier ces deux partie grâce à interopérabilité.

1. Vue générale:

Au lancement de l'application, une grille représentant une configuration du jeu est affichée, ainsi qu'un ensemble des buttons permettant à l'utilisateur de jouer.

L'objectif est donc, de réaliser l'interface graphique ci-dessous:



2. Choix Techniques :

L'application est développé en utilisant le traducteur du byte-code Ocaml en JavaScript: Js_of_Ocaml

3. Structure de l'application :

Un fichier index,html contient les éléments html de l'application explicités ci-après, avec un fichier style.css pour la représentation. Le fichier orush,ml permet par la suite, d'écrire les fonctions javaScript en ocaml pour ajouter l'aspect dynamique. Ces fonctions exploitent les modules crées dans la première partie du projet.

4. Composants Statiques:

2.1. La Grille:

La grille est un ensemble des cases: des balises avec comme attribut id leur position. C'est à dire, la case ij a pour identifiant html la chaine «ij» ce qui permet aux fonctions Ocaml de récupérer la case facilement en utilisant les champs x et y d'un bateau.

2.2. Un Bateau:

Un bateau est représenté par l'ensemble des cases qu'il occupe. Pour cela, au lancement de l'application, on initialise l'attribut 'class' des cases concernées à l'identifiant du bateau. Donc les cases occupées par un bateau 'A' sont toutes de la classe 'A' .

2.3. Contrôle:

Un panneau de contrôle permet à l'utilisateur de: réinitialiser le jeu, générer une nouvelle configuration et manipuler le bateau sélectionné.

5. Fonctions dynamiques :

Pour l'aspect dynamique, on déclare un ensemble de fonctions et de variables qui permettent de mettre à jour la grille et manipuler les bateaux:

On garde à tout moment une référence vers le bateau sélectionné ainsi que l'état courant. En cliquant sur un bateau ou en exécutant un déplacement (avancer/ reculer) on met à jour ces références et on effectue le traitement qui correspond. Ces traitement sont des fonctions qu'on déclare en Ocaml et on les associe à l'attribut «Onclick» des composants; ce qui permet un affichage qui évolue en jouant.

Après chaque mise à jour, on teste si c'est un état gagnant, dans lequel on affiche un message et le nombre de déplacements effectués.

CONCLUSION

Comme tous développement d'application bien organisé, il est très important de bien séparer les partie indépendantes et de trouver le meilleur langage pour chaque partie. Une telle approche minimise le temps de développement ainsi que le nombre des erreurs possibles. À la fin, il faut bien choisir la façon dont on fusionne les différentes parties pour minimiser les modifications supplémentaires.