Introduction

Description brève

Ce projet a eu pour but de recréer un monopoly personnalisable. C'est comme un monopoly classique mais on peut choisir n'importe quel type de case pour chaque case. On pourrait donc par exemple mettre que des stations de train sur toute la carte.

Répartition du travail (Cahier des charges)

Marco:

- Créé le plateau
- backend (création des classes)

Cyril:

- Monopoly personnalisable
- Partie graphique côté jeu

Technologies choisies

html/css/canvas
Librairies:
jQuery pour rendre les formulaires dynamiques
EaselJS pour rendre le canvas interactif
Bootstrap pour la mise en page des pages et des formulaires

Cyril - Réalisation individuelle

Buts

Monopoly modifiable
Partie graphique du jeu

Travail réalisé

Le point 1 a été réalisé avec la librairie easelJS qui permet de définir des objets cliquables(rectangle, cercle, image). Un clic permet d'arriver sur un formulaire avec les choix de case. Le formulaire est dynamique il se modifie en fonction de la catégorie de la case. Une fois la confirmation de la case choisie, cela va modifier l'aspect graphique ainsi que les données.

J'ai ensuite fait la partie graphique du jeu en reprenant le code de Marco pour dessiner le plateau mais en l'adaptant avec la librairie. J'ai ajouté le mouvement des pions, les boutons d'actions, les changements de propriétaires et l'abandon.

Questions:

Qu'est ce qui vous a positivement surpris ?

De réaliser assez vite une interface graphique toute simple sans trop de soucis.

Qu'est ce qui vous a pris plus de temps ?

Comprendre comment fonction easelJS et l'utilisation du canvas.

Quel est le pire bug que vous avez eu ?

Avec easelJS on doit ajouter chaque élément dans une stage, mais j'oublie parfois d'y ajouter et d'y mettre à jours. Ce qui ne produisait rien pour l'affichage même si le code était juste.

Au final si vous deviez recommencer que changeriez-vous ?

Ne pas utiliser le canvas mais en utilisant plutôt des balises div.

Marco - Réalisation individuelle

Buts

1. Dessiner le monopoly avec les cases, les propriétaires de chaque case, la famille des cases (couleur) et des images selon le type (gare, impôts, caisse communautaire, chance, etc).
2. Créer les classes gérant le jeu (backend).

Travail réalisé

Le point 1 a été totalement fini. Mise en place de bootstrap pour les formulaires (échanges, configuration du jeu, enchères, actions sur les cases) plutôt que du html simple. Le point 2 gère presque tout excepté l'achat des maisons/hypothèque.

Questions:

Qu'est ce qui vous a positivement surpris ?

La simplicité des actions groupées, par exemple filter, forEach. Le local storage que je n'avais jamais utilisé.

Qu'est ce qui vous a pris plus de temps ?

Mettre en place la fonction jouer avec tous les états du jeux.

Quel est le pire bug que vous avez eu ?

Lors d'un échange il y avait des erreurs car les valeurs récupérées étaient des chaînes de caractères et pas des entiers.

Au final si vous deviez recommencer que changeriez-vous ?

Ne pas utiliser le canvas mais plutôt dessiner le jeu grâce à des divs ou un tableau. L'interaction serait bien plus simple et pour changer le type de case il suffirait de changer la classe.

Conclusion

Description du travail dans l'état actuel (avec les problèmes connus)

Le monopoly s'affiche correctement et il est totalement configurable. Concernant le jeu tout fonctionne (jouer, abandonner, acheter une case, aller en prison, sortir de prison, échanger, enchères, payer le loyer d'une case) excepté la gestion des maisons/hypothèque.

Proposition d'amélioration

- Finir ce qu'on n'a pas eu le temps de faire
- Ajouter un mode ordinateur(bot)
- Mettre le jeu en réseau
- Jouer avec de l'argent réel