BUVRON Baptiste
LE MOAL Yann

Travail à réaliser pour le 16 mars 2020:

Github du projet : https://github.com/BaptisteBuvron/Monopoly-isn/tree/dev

1. Intitulé du projet

Le projet est un Monopoly remasterisé. Il s'agit d'un Monopoly constitué exclusivement de pays et de planètes à la place de nom de rue que l'on voit dans le jeu traditionnel.

2. Présentation
   - Pourquoi ce projet ?

L'idée de créer un jeu nous est rapidement venu à l'esprit. Étant tous les deux passionné par les jeux vidéos nous nous sommes mis d'accord sur la création de ce jeu vidéo.

Ce projet à été choisie pour ces difficultés technique nécessaire afin de le réaliser ainsi que de son but : créer un jeu disponible gratuitement en ligne.

Nous avons choisi ce projet

- Problématique du projet

Ce projet vise à permettre à des utilisateurs de jouer à un jeu conviviale (Monopoly) sans devoir disposer d'un plateau de Monopoly. Créer un jeu disponible gratuitement et simplement en ligne.

3. Analyse du besoin-recherche d'idée

Il s'agit de présenter :

- Le cahier des charges de l'équipe : quel est le but visé par votre projet ? Quels sont les objectifs à réaliser?

  Le but de ce projet est de créer un jeu de Monopoly multijoueur en local (les joueurs jouent sur le même ordinateur). Les objectifs sont la création des fonctionnalité permettant le bon fonctionnement du jeu : mouvement des joueurs, achat de propriété, payer un loyer, compte en banque des joueurs etc…

- L'environnement de travail utilisé (language, matériel) et les sources d'information.

  langage universel : anglais
  langage utilisé : HTML, CSS, JavaScript
  Source d'information: documentation de Javascript, HTML et CSS.

4. Répartition des tâches et démarche collaborative :

- Distinguer vos tâches de celles de l’équipe

Nous avons d’abord créer le plateau de jeu en HTML et CSS à deux afin de pouvoir partir sur une base solide et identique. Puis nous nous sommes donnés des fonctions à réaliser. Puis quand une fonction est terminé nous nous mettons en commun afin de se répartir les tâches suivantes.

- Présenter un rétroplanning

Etant donné que le projet avance correctement nous n’avons pas défini de dates précises. Cependant nous nous donnons des objectifs pour les semaines à venir.

5. Présenter et documenter votre démarche et votre travail : détailler le rôle et le principe de fonctionnements de vos fonctions, de votre code HTML ou javascript ... Faire des schémas des différents écran, page web … avec leur, contenu, la fonction ou le fichier associés à chaque partie.

Jeu du Monopoly

- Le jeu
- Le Github du projet

Sommaire

HTML

Le plateau de jeu est créer entièrement en html et css :

Chaque case est représenté par une balise `div`.

Chaque `div` peut comporter les classes suivantes:

- `top, left, right, bottom, void, cell` en fonction de leur position
- `property, tax, community-chest, prison, start, free-parking, electric, water` en fonction de leur fonction dans le jeu.

Exemple :

```
<div id="cell21" class="corner cell top free-parking"></div>
<div id="cell22" class="property cell top" data-group="group5">
    <div class="content"></div>
    <div class="title"></div>
</div>
<div id="cell23" class="top cell"></div>
<div id="cell24" class="property cell top" data-group="group5">
    <div class="content"></div>
    <div class="title"></div>
</div>
```

Les propriétés

Chaque propriétés possèdent la classe `available` si elle ne possède pas de propriétaire. Sinon elle possède l'atribut `data-owner=player...` permettant de connaitre le propriétaire de la propriétés.

Numéro de case

Chaque case possède un id permettant de savoir qu'elle est son numéro de case :

```
<div id="cell2" class="property bottom" data-group="group1">
            <div class="content"></div>
            <div class="title"></div>
</div>
<div id="cell1" class="corner start"></div>
```

CSS

Chaque case du jeu est stylisé en fonction de sa classe. Ainsi par exemple :

Pour les éléments de classe `right` et `left` enfant d'un élément possédant l'id `game` :

```
/* Style d'une div étant soit à gauche soit à droite */
#game .right, #game .left{
    width: 140px;
    height: 65px;
}
```

Les propriétés

Chaque propriétés possèdent une div `content` et `title`. Afin de pouvoir ajouter les joueurs lors du déplacement et également pour différencier les différents groupes de propriétés.

```
/* Style des div title propriété en haut */
#game .cell.top .title {
    height: 20%;
    border-top: 1px solid black;
    width: 100%;
}

/* Style des div content propriété en haut */
#game .cell.top .content {
    height: 80%;
    width: 100%;
}
```

Chaque propriété possède un groupe, ce groupe est répéré avec l'attribut `data-group="group1"`

Ainsi la couleur de fond d'écran est définit en fonction de ce `data-group`

```
#game [data-group='group1'] .title{
    background-color: brown;
```

```
}

#game [data-group='group2'] .title{
    background-color: aquamarine;
}

...
```

Les cases à fonctions

Les cases à fonctions sont repéré avec leur classe, en fonction de cette classe une image de fond d'écran leur est attribuée.

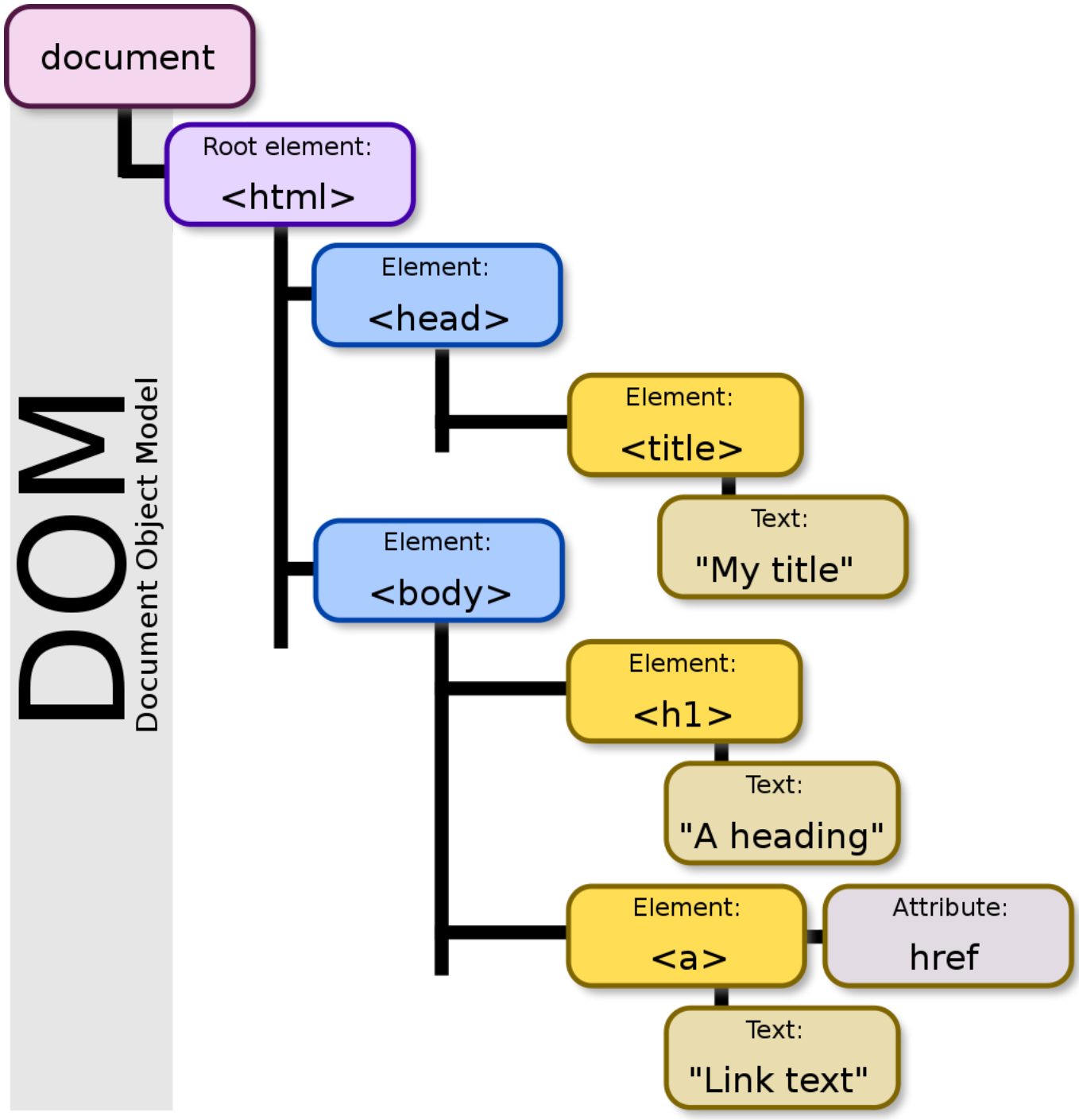
```
<div class="corner prison"></div>
```

```
#game .prison{
    background-image: url('pictures/jail.jpg');
    background-size: contain;
    background-repeat: no-repeat;
}
```

JavaScript

Le DOM

Le DOM (Document Object Model) est une interface de notre page WEB. Il va permettre aux javascript de pouvoir lire et manipuler le contenu de la page, sa structure et son style.



-- *Wikipedia*

Jquery

Pour sélectionner les éléments du DOM nous utiliserons la bibliothèque JavaScript qui a été créer pour facilité l'écriture de scripts côté client dans le code HTML des pages web. Le but de la bibliothèque étant le parcours et la modification du DOM.

Ainsi par exemple :

```
document.getElementById("id");
```

Devient :

```
$("#id");
```

Objet Monopoly

Un objet Monopoly a été créer, toutes les fonctions lier au déroulement du jeu du Monopoly seront des propriétés de l'objet Monopoly :

```
var Monopoly = new Object();
```

Les propriétés

Monopoly.start

Cette propriété de l'objet Monopoly est appelée au chargement de la page et appelle les propriétés nécessaire au démarrage de la partie.

```
Monopoly.start = function(){
    Monopoly.getNbrPlayer();
};
```

Monopoly.allowToDice

Cette propriété de l'objet Monopoly est un booléen qui permet de déterminer quand les joueurs ont le droit de lancer les dés.

```
Monopoly.allowToDice = false;
```

Monopoly.dice

Cette propriété de l'objet Monopoly permet au joueur de lancer les dés de maniére aléatoire et indépendante. La propriété `Monopoly.movePlayer` est ensuite appelé pour faire avancer le joueur d'un nombre de case.

```
Monopoly.dice = function () {
    if (Monopoly.allowToDice) {
        var dice_1 = Math.floor(Math.random() * 6) + 1; /* retourne un nombre
compris entre 1 et 6 */
        var dice_2 = Math.floor(Math.random() * 6) + 1; /* retourne un nombre
compris entre 1 et 6 */
        Monopoly.allowToDice = false; /* interdit au joueur de relancer les dés*/

        var total = dice_1 + dice_2;
```

```
        Monopoly.movePlayer(Monopoly.getCurrentPlayer(), total);
    }
```

Monopoly.bankPLayer

Initialisation du dictionnaire qui contiendra le montant du compte de chaque joueur. Clé du dictionnaire = id du joueur, Valeur du dictionnaire = Montant du compte (int)

```
Monopoly.bankPlayer = new Map();
```

Monopoly.listCell

Initialisation du dictionnaire qui contiendra les propriétés de chaque cellule. Clé du dictionnaire = id de la cellule, Valeur = Objet contenant les propriétés de la cellule (objet)

```
Monopoly.listCell = new Map();
```

Modèle de l'objet contenant les propriétés des cellules.

```
function propertiesCell(type, name, picture, group, owner, buy, sell, rent,
upgradePrice, level) {
    this.type = type;
    this.name = name;
    this.picture = picture;
    this.group = group;
    this.owner = owner;
    this.buy = buy;
    this.sell = sell;
    this.rent = rent;
    this.upgradePrice = upgradePrice;
    this.level = level;

}
```

Ajout d'une cellule dans le dictionnaire `Monopoly.listCell` avec l'id de la cellule et l'objet contenant les propriétés de la cellule.

```
Monopoly.listCell.set(2, new propertiesCell("property", "Turkmenistan",
"turkmenistan.png", "Autre", null, 60, 30, 25, 1));
```

Monopoly.listChance

Initialisation d'une liste qui contiendra le contenu des cases chance sous forme d'objet.

```
Monopoly.listChance = [];
```

Modèle de l'objet contenant les propriétés de la case chance.

```
function addChance(name,presentation,action,number){
    this.name = name;
    this.presentation = presentation;
    this.action = action;
    this.number = number;
};
```

Ajout dans la liste `Monopoly.listChance` l'objet contenant les propriétés d'une carte chance.

```
Monopoly.listChance.push(new addChance("Facture","Vous n'avez pas payer vos
factures ! <br>- Vous devez 50€ à la banque","pay",-50));
```

Monopoly.getNbrPlayer

Cette propriété de l'objet Monopoly est appelée par `Monopoly.start` et demande à l'utilisateur le nombre de joueurs.Tant que le nombre indiqué n'est pas entre 2 et 5 la propriété est rappelée. Sinon la propriété `Monopoly CreatePlayer` est appelée.

Aucun paramètre n'est nécessaire.

```
Monopoly.getNbrPlayer = function () {
    $(document).ready(function () {
        $("#modal-player").modal('show');

    });
    $("#button-nbrPlayer").click(getNbrPlayer);


    function getNbrPlayer() {
        var nbrPlayer = 0;
        nbrPlayer = parseInt($("#nbrPlayer").val());
        if (nbrPlayer <= 5 && nbrPlayer >= 2) {
            $("#modal-player").modal('hide');
            Monopoly.createPlayer(nbrPlayer);
        }
    }

};
```

Monopoly.createPlayer

Cette propriété de l'objet Monopoly est appelé par `Monopoly.getNbrPlayer` et permet de créer le nombre de joueur demandé. Il initialise également le compte en banque des joueurs.

Paramètre :

- Le nombre de joueur a créer (int).

```
Monopoly.createPlayer = function (nbrPlayer) {

    for (let i = 1; i <= nbrPlayer; i++) {
        if (i == 1) {
            $('<div id="player' + String(i) + '" class="player current-turn">
</div>').appendTo('#game .start .content');
        } else {
            $('<div id="player' + String(i) + '" class="player">
</div>').appendTo('#game .start .content');
        }
        Monopoly.bankPlayer.set("player" + String(i), Monopoly.moneyAtStart);


    }
    Monopoly.allowToDice = true;
    Monopoly.dice();

};
```

Monopoly.getCurrentPlayer

Cette propriété de l'objet Monopoly permet de retourner du `DOM` le joueur qui possède la classe curent-turn.

Aucun paramètre n'est nécessaire.

```
Monopoly.getCurrentPlayer = function(){
    return $(".player.current-turn");
};
```

Monopoly.getClosestCell

Cette propriété de l'objet Monopoly permet de retourner du `DOM` la cellule la plus proche d'un joueur et qui possède la classe `cell`.

Les paramètres :

- L'élément du DOM d'un joueur.

```
Monopoly.getClosestCell= function(player){
    return player.closest(".cell");
};
```

Monopoly.getIdCell

Cette propriété de l'objet Monopoly permet de retourner l'id d'une cellule (int). Cette propriété utilise la methode replace qui remplace dans l'id de la cellule cell par une chaine de caractère vide afin de récuperer seulement l'id de la cellule.

Les paramètres :

- L'élément du DOM d'un cellule.

```
Monopoly.getIdCell = function(playerCell){
    return parseInt(playerCell.attr('id').replace("cell",""));
}
```

```
<div id="cell36"></div>
```

Monopoly.getIdplayer

Cette propriété de l'objet Monopoly permet de retourner l'id d'un joueur (int).

Les paramètres :

- L'élément du DOM d'un joueur.

```
Monopoly.getIdPlayer = function (player) {
    return parseInt(player.attr("id").replace("player", ""));
}
```

Monopoly.getNextCell

Cette propriété de l'objet Monopoly permet de retourner la cellule suivante. De plus elle appelle la propriété `Monopoly.addMoneyPlayer` quand un tour a été effectué.

Les paramètres :

- L'id de la cellule précédente (int).

```
Monopoly.getNextCell = function (idCell) {
    if (idCell == 40) {
        idCell = 0;
```

```
Monopoly.updateMoneyPlayer(Monopoly.getIdPlayer(Monopoly.getCurrentPlayer()),
200);
    }
    var nextIdCell = idCell + 1;
    return $("#game .cell#cell" + nextIdCell);


}
```

Monopoly.movePlayer

Cette propriété de l'objet Monopoly permet de faire avancer un joueur de `number` case.

Les paramètres :

- L'élement du DOM correspondant au joueur a déplacer,
- Le nombre de case a avancé (int).

```
Monopoly.movePlayer = function (player, number) {
    Monopoly.allowToDice = false;

    var movePlayerInterval = setInterval(movePLayer, 500);

    function movePLayer() {

        var cellPlayer = Monopoly.getClosestCell(player);
        var idCell = Monopoly.getIdCell(cellPlayer);
        var nextCell = Monopoly.getNextCell(idCell);

        nextCell.find('.content').append(player);
        number--;
        if (number == 0) {
            clearInterval(movePlayerInterval);
            cellPlayer = Monopoly.getClosestCell(player);
            Monopoly.action(player, cellPlayer);
        }
    }

};
```

Monopoly.updateMoneyPlayer

Cette propriété de l'objet Monopoly permet de modifier le montant d'un compte d'un joueur.

Les paramètres :

- L'id d'un joueur (int),
- Le montant a enlever ou ajouter sur le compte (int).

```
Monopoly.addMoneyPlayer = function (playerId, amount) {
    var money = Monopoly.getMoneyPlayer(playerId);
    var newMoney = money + amount;
    Monopoly.bankPlayer.set("player" + String(playerId), newMoney);

};
```

Monopoly.getMoneyPlayer

Cette propriété de l'objet Monopoly permet de retourner le montant d'un compte d'un joueur.

Les paramètres :

- L'id du joueur (int).

```
Monopoly.getMoneyPlayer = function (playerId) {
    return parseInt(Monopoly.bankPlayer.get("player" + String(playerId)));
};
```

Monopoly.calcRent

Cette propriété de l'objet Monopoly permet de retourner le montant du loyer d'une propriété en fonction de son level.

Les paramètres :

- L'id de la cellule (int).

```
Monopoly.calcRent = function (idCell) {
    var rent = Monopoly.listCell.get(idCell)["rent"];
    var level = Monopoly.listCell.get(idCell)["level"];
    switch (level) {
        case 0:
            rent = rent;
            break;
        case 1:
            rent = rent * 5;
            break;
        case 2:
            rent = rent * 15
            break;
        case 3:
            rent = rent * 45;
            break;
        case 4:
            rent = rent * 80;
            break;
        case 5:
```

```
            rent = rent * 125;
            break;
    }
    return rent;
};
```

Monopoly.payRent

Cette propriété de l'objet Monopoly permet de faire payer le montant du loyer d'une propriété.

Les paramètres :

- L'id du propriétaire de la propriété(int),
- L'id du joueur qui doit payer le loyer(int),
- L'id de la cellule de la propriété(int).

```
Monopoly.payRent = function (idOwner, idPlayer, idCell) {
    var rent = Monopoly.calcRent(idCell);
    if (Monopoly.verifbank(idPlayer), rent) {
        Monopoly.updateMoneyPlayer(idOwner, rent);
        Monopoly.updateMoneyPlayer(idPlayer, -rent);
    } else {
        Monopoly.sellProperty(rent);
    }
};
```

Monopoly.verifBank

Cette propriété de l'objet Monopoly permet de vérifier si un joueur dispose suffisament d'argent pour payer un loyer par exemple. True est retourné si le compte contient suffisament d'argent sinon False est retourné.

Les paramètres :

- L'id du joueur qui doit payer le loyer(int),
- Le montant à payer(int).

```
Monopoly.verifBank = function (idPlayer, amount) {
    var money = Monopoly.getMoneyPlayer(idPlayer);
    if (money - amount >= 0) {
        return true;
    } else {
        return false;
    }
}
```

Sources

Les images

Les images du plateau de monopoly proviennent du site commons Wikipedia, il s'agit d'une bibliothèque de média libre de droit.