École Centrale Lyon

Programmation web
Rapport

Projet web - Abalone

Élèves :
Léo Guislain
Hippolyte Morillon

Enseignant :
Daniel Muller

31 mars 2021

Table des matières

1 Présentation du concept **2**
1.1 Contexte 2
1.2 Présentation du jeu Abalone 2
1.3 Cahier des charges 5

2 Installation et lien vers le dépôt github **5**
2.1 Installation 5
2.2 Lien vers le dépot github 5

3 Détails techniques **5**
3.1 Page d'accueil 5
3.2 Affichage du plateau 6
3.3 Sélection des billes 7
3.4 Dialogue avec le serveur 9
3.5 Autres éléments graphiques 9

1 Présentation du concept

1.1 Contexte

Lors du projet web il est demandé de programmer un jeu multijoueur passant par un serveur. Les informations seront relayées grâce à une connexion permanente au serveur avec le module socket.io.

1.2 Présentation du jeu Abalone

Pour ce projet, nous avons choisi de programmer Abalone. Un jeu de plateau tour par tour de 1v1. Le principe de ce jeu est simple : faire tomber les billes de son adversaire en utilisant les siennes.
Pour se faire, chaque joueur débute avec un total de 14 billes réparties sur le plateau hexagonal comme suit [1]

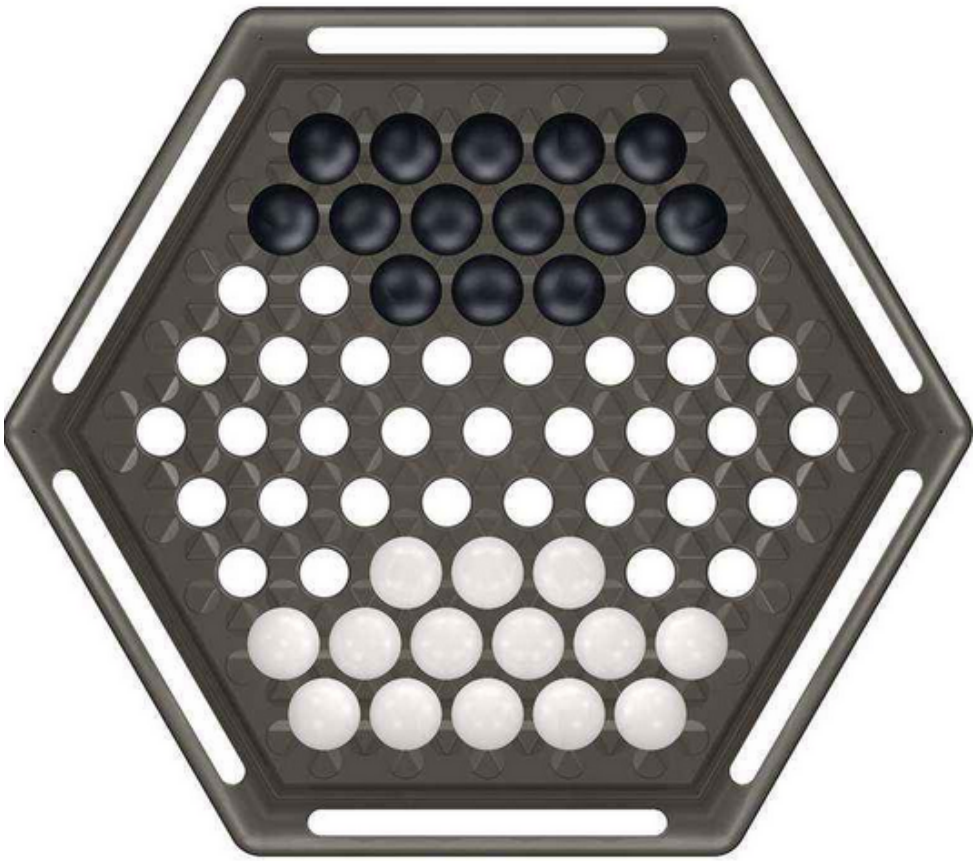

FIGURE 1 – Plateau de jeu et mise en place

Le joueur qui débute peut effectuer un déplacement de 1 à 3 de ses billes adjacentes qui se trouvent sur une même ligne. Ce déplacement doit se faire dans la même direction pour chacune des billes. Quelques exemples pour mieux comprendre [2] [3]

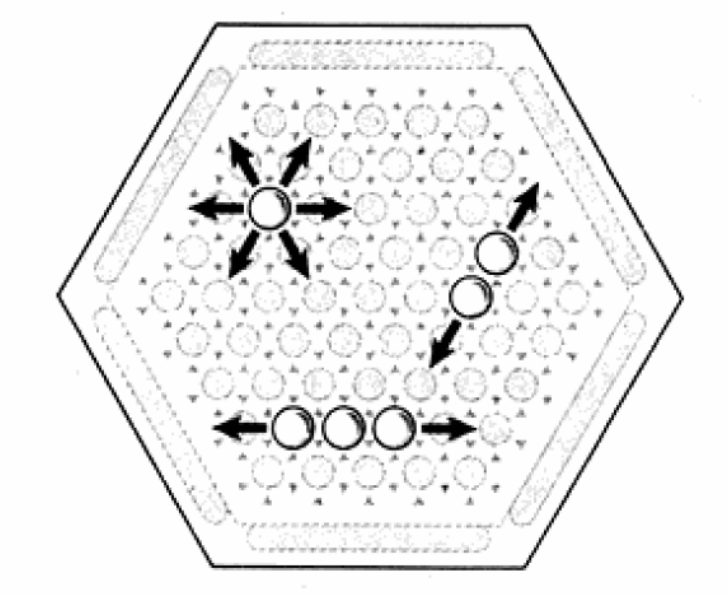

FIGURE 2 – Description du mouvement en ligne pour 1 à 3 billes

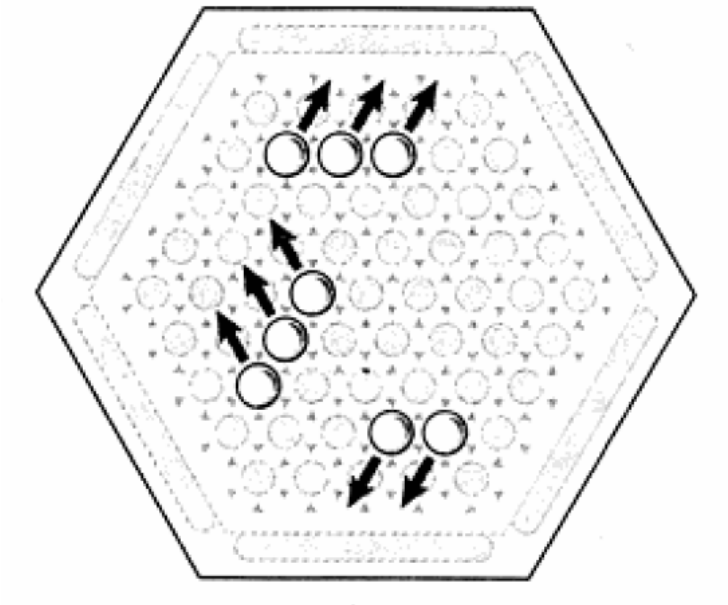

FIGURE 3 – Description du mouvement en flèche pour 2 à 3 billes

Ainsi chaque joueur à tour de rôle doit effectuer un déplacement d'une ou plusieurs de ses billes.
Puisque l'objectif du jeu est de faire tomber hors du plateau les billes de son adversaire, il faut pouvoir les déplacer. Pour se faire un joueur qui effectue un déplacement en ligne peut avoir recours au sumito, un déplacement qui permet de pousser les billes de son adversaire. Le sumito est assez simple à comprendre, si le joueur qui effectue le déplacement déplace strictement plus de billes lui appartenant que de billes de son opposant, alors le déplacement est autorisé. Voici un exemple pour y voir plus clair [4]

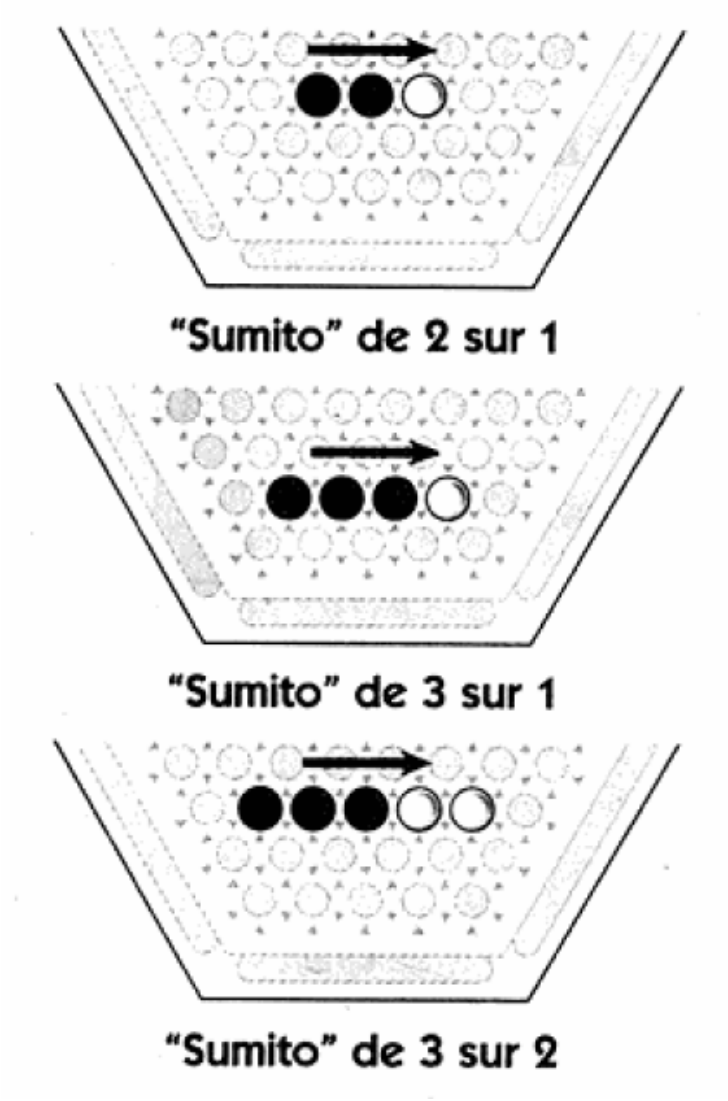

FIGURE 4 – Description du sumito

Mais attention, un sumito ne peut avoir lieu que si les billes adverses ne sont pas bloqué par des billes appartenant au joueur qui effectue l'action. Ce qui veut dire que si les billes adverses sont entourées par des billes appartenant au joueur qui effectue l'action, elles ne peuvent pas bouger [5]

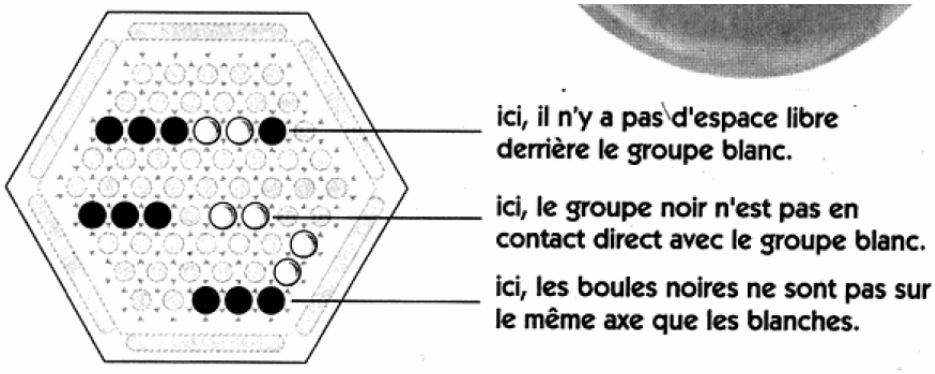

FIGURE 5 – Cas particulier

1.3 Cahier des charges

Maintenant que les règles du jeu son claires, il nous faut définir le cahier des charges de l'application. L'application doit simuler le jeu Abalone en 1 contre 1 par l'intermédiaire d'un serveur. La connexion permanente au serveur se fera grâce au module socket.io. Pour créer le serveur, nous utiliserons express. L'application devra comporter une page d'accueil expliquant les règles et qui renvoie vers le jeu.

2 Installation et lien vers le dépôt github

2.1 Installation

Pour installer l'application, il suffit de cloner le répertoire github, puis de se placer dans le répertoire PAWeb_jeu en ligne de commande. Il faut exécuter "npm update" pour installer les dépendances la première fois, puis "node app.js" pour lancer l'application côté serveur. Ensuite, il faut ouvrir un navigateur et taper http ://localhost :8080/page_accueil.html dans la barre de recherche. Les règles sont alors décrites sur la page web. Pour lancer une partie il faut cliquer sur lancer la partie en bas de la page, puis ouvrir un second navigateur (ou onglet) et effectuer la même opération. Enfin sur les deux pages, il faut cliquer sur "Prêt" et la partie devrait se lancer directement.

2.2 Lien vers le dépot github

Cliquer ici pour accéder au dépôt github,
ou copier le lien : https ://github.com/JCSaucisse/PAWeb_Jeu

3 Détails techniques

3.1 Page d'accueil

Le code de cette page est un code standard html, il créé une table qui contient les règles et les affiches à la suite. De plus, il possède des liens cliquables href qui permettent d'accéder au pdf hébergé sur l'application contenant les règles détaillées, un lien vers la photo pour les droits d'images et un lien pour lancer la partie.

```
            <th> Pour avoir les règles plus en détail: <a href=/pdf/regle.pdf> cliquer ici </a>
        </tr>
    </table>
</h3>
<table id="2">
    <h4>Plateau de jeu:</h4>
    <tr>
        <th> <img src=/images/abalone.PNG title="plateau de jeu" height="400px" width="430px"/> </th>
    </tr>
    <tr>
        <th> <a href=https://www.cogitoys.fr/1195-home_default/abalone-nouvelle-edition-jeu-de-strategie.jpg> https://www.cogitoys.fr/1195-home_default/abalone-n
    </tr>
</table>
<h5>Maintenant à vous de jouer: <a href=hexagone.html> lancer la partie </a></h5>
```

FIGURE 6 – Partie du code

Une fois l'application lancée, la page d'accueil donne ceci [7]

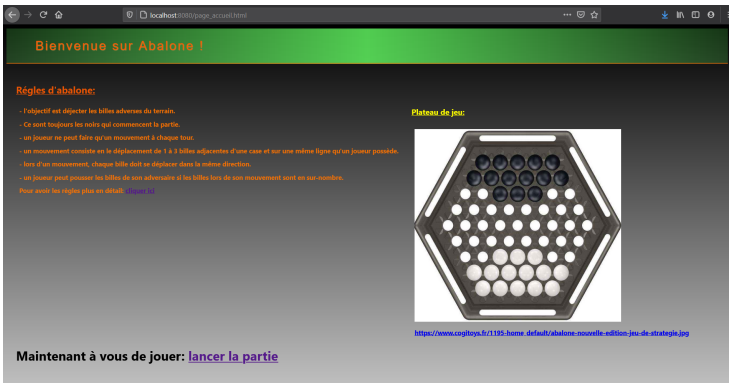

FIGURE 7 – Page d'accueil

3.2 Affichage du plateau

Le jeu Abalone est basé sur une grille hexagonale : les cases individuelles, tout comme la forme générale du plateau sont des hexagones [8].

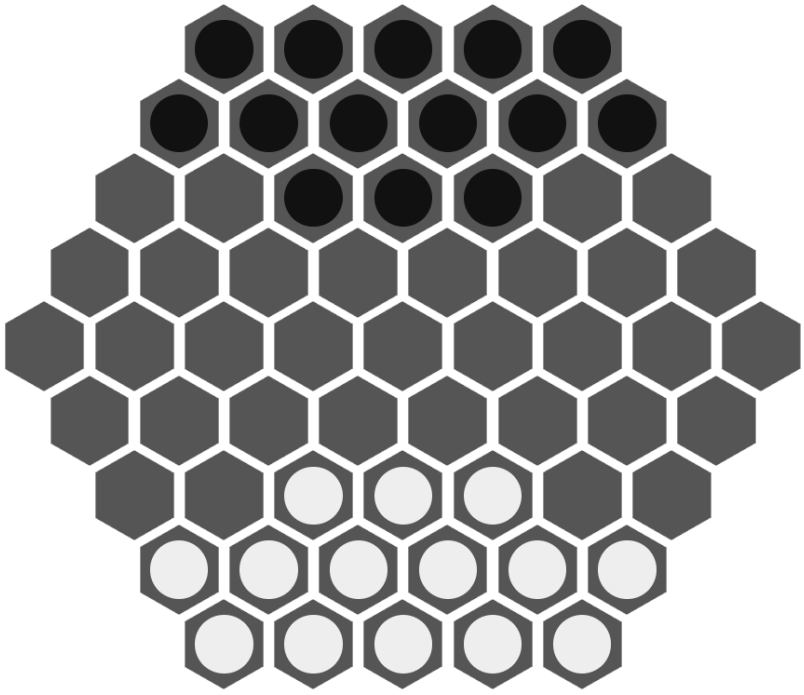

FIGURE 8 – Plateau de jeu : des hexagones dans un hexagone

Pour afficher un hexagone seul, on crée une balise *div* et on lui attribue la classe "hexagon". La majeure partie du travail se situe dans le fichier hexagone.css. Celui-ci met en forme un rectangle, mais aussi deux triangles en haut et en bas en utilisant les pseudo-éléments css " : :before" et " : :after".

La grille est un élément dans lequel on ajoute les hegaxones comme enfants dans le code javascript. Celle-ci ordonne l'affichage des hexagones grâce au css. Cependant, pour avoir une grille hexagonale et non rectangulaire, il a fallu décaler les hexagones vers la droite une ligne sur deux, puis masquer les hexagones qui ne forment pas le plateau.

Enfin, pour placer une bille dans une case, on ajoute une balise *div* à l'hexagone et on lui attribue la bonne classe. On atttribue à chaque case jouable un identifiant unique pour gérer les déplacements, et on contrôle la couleur de la case (pour sélectionner, déplacer) avec des classes et des feuilles de style.

3.3 Sélection des billes

Un joueur, lors de son tour, peut déplacer jusqu'à trois billes adjacentes et alignées dans une même direction. Pour se faire il doit d'abord sélectionner les trois billes à déplacer. Pour se faire on utilise la fonction hexagonClicked. Le code se trouve dans le fichier hexagon.js dans le dossier public de l'application. Une partie du code se trouve ci-dessous [9].

```
function hexagonClicked(hexagonId) {
    console.log("clic on : ");
    console.log(hexagonId);
    hexagon = document.getElementById(hexagonId);
    if(hexagon.classList.contains("select") && gamePhase == 'playerSelect'){
        // Selection
        unselectAll();
        hexagon.classList.add("select");
        let hexagonInt = parseInt(hexagonId.replace("hexagon",""));
        selectedHexagonInt = hexagonInt;
        selectedHexagonIntList.push(hexagonInt);
        let targets = getPossibleTargets(hexagonInt);
            for(let i = 0; i < targets.length; i++){
                let targetIntStr = targets[i].toString();
                targetI = document.getElementById("hexagon"+targetIntStr)
                if(isSameColor(hexagon, targetI)){
                    document.getElementById("hexagon"+targetIntStr).classList.add("same");
                }
                else{
                    document.getElementById("hexagon"+targetIntStr).classList.add("target");
                }
        }
        gamePhase = 'playerTarget';
    }
    else if(hexagon.classList.contains("select") && gamePhase == 'playerTarget'){
        // Deselection
        unselectAll();
        untargetAll();
        unsameAll();
        let hexagonInt = parseInt(hexagonId.replace("hexagon",""));
        selectedHexagonInt = hexagonInt;
        let selectedHexagonIntListBis = [];
        let isOk = false;
        while(!isOk){
            potentialSelectedHexagonInt = selectedHexagonIntList.pop();
            selectedHexagonIntListBis.push(potentialSelectedHexagonInt);
            if(potentialSelectedHexagonInt == selectedHexagonInt){
                selectedHexagonIntListBis.pop();
                isOk = true;
            }
```

FIGURE 9 – partie de code de hexagonClicked

Cette fonction prend en argument l'hexagone sur lequel l'utilisateur a cliqué et le traite de différentes manières en fonction des attributs qu'il possède dans sa classe. Tout d'abord on vérifie dans quelle phase de jeu on se trouve et on regarde si l'hexagone possède un attribut select (vert), target (rouge) ou same (bleu) qui représente si l'hexagone peut être sélectionné, si on peut se déplacer dans l'hexagone sur lequel l'utilisateur a cliqué ou si il y a déjà un hexagone de sélectionné et on peut en sélectionner un deuxième. A chaque fois l'hexagone sur lequel on aura cliqué sera traité différemment. Si on clique plusieurs fois sur une même bille nous appartenant, elle sera d'abord sélectionnée puis déselectionnée. De même si on a sélectionné plusieurs billes, l'appui sur une bille sélectionnée déselectionne toute les billes qui ont été sélectionnées après.

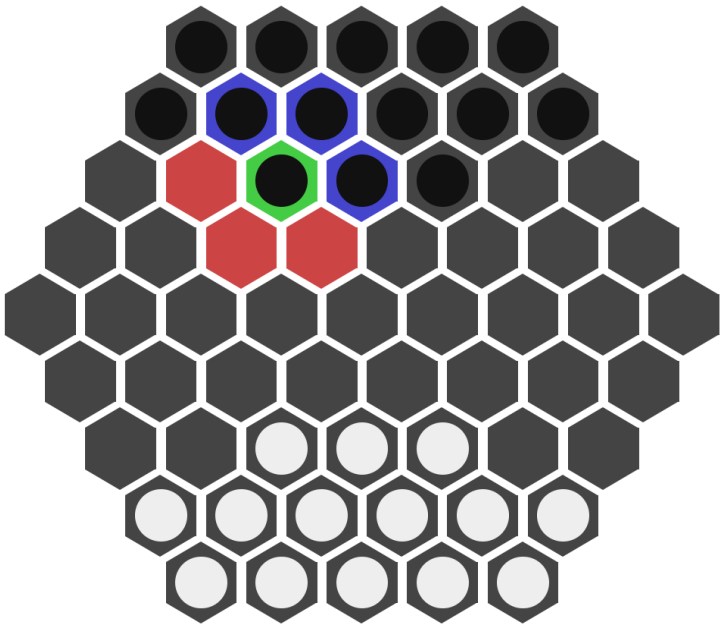

FIGURE 10 – Sélection d'une bille

Par exemple, sur la figure [10], une bille a été sélectionnée en vert, les cases en bleues sont les cases contenant des billes qui peuvent être sélectionnées en plus et les cases en rouges sont celles où la bille peut se déplacer.

De plus, les déplacements se font en fonction de la première bille sélectionnée, toute les billes se déplaceront dans le même sens mais l'affichage se fera en fonction de la première bille [11].

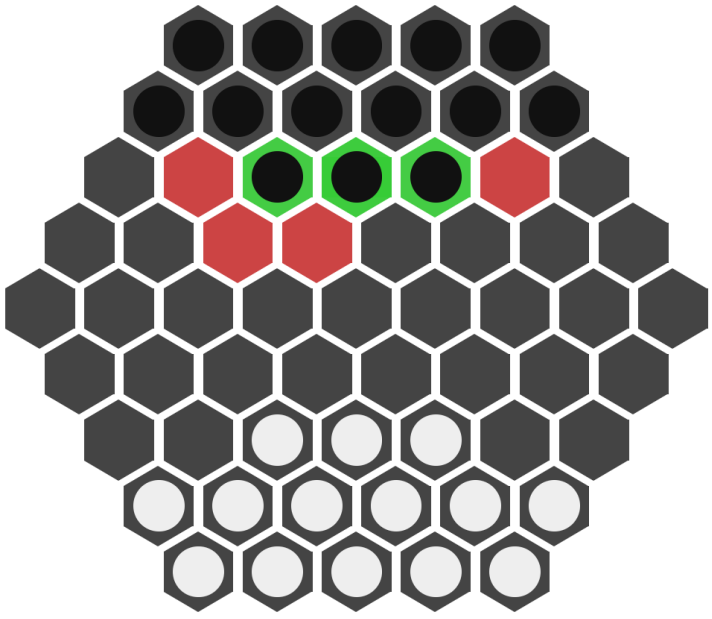

FIGURE 11 – La première bille sélectionnée est celle de gauche, les déplacements sont indiquées par rapport à elle

3.4 Dialogue avec le serveur

La partie serveur de l'application se trouve dans le code app.js. Une connexion avec socket.io est établie dès le chargement de la page de jeu, et un bouton prêt permet de signaler au serveur qu'on peut lancer la partie ou attendre l'autre joueur. Si un des joueurs se déconnecte ou quitte la partie, celle-ci est annulée et on peut en recommencer une nouvelle. Pendant le déroulement du jeu, les informations sur les billes à déplacer transitent du joueur vers le serveur puis vers l'autre joueur.

3.5 Autres éléments graphiques

Plusieurs éléments permettent de mieux comprendre le déroulement de la partie. Au-dessus du plateau se trouvent la couleur des billes jouées par le joueur sur la page, le joueur qui est en train de déplacer des billes, ainsi que le score. Lorsque le score d'un des joueurs dépasse 6, c'est à dire qu'il a expulsé 6 billes adverses, le jeu est fini et l'affichage de victoire/défaite apparaît [12]. On peut alors quitter la partie en cliquant sur le bouton "Quitter la partie", puis en relancer une autre si les deux joueurs sont prêts.

FIGURE 12 – Écran de victoire d'une partie