PROJET DEV

Yasser Youssouf Mohamed & Enzo Pinot

SOMMAIRE

PRÉSENTATION DU SUJET

PRÉSENTATION DU SUJET

POO

MORPION BDD

SERVEUR

MATCHMAKING

OUTILS UTILISÉS

VISUAL STUDIO CODE

node JS ™

RÉPARTITION DES TÂCHES

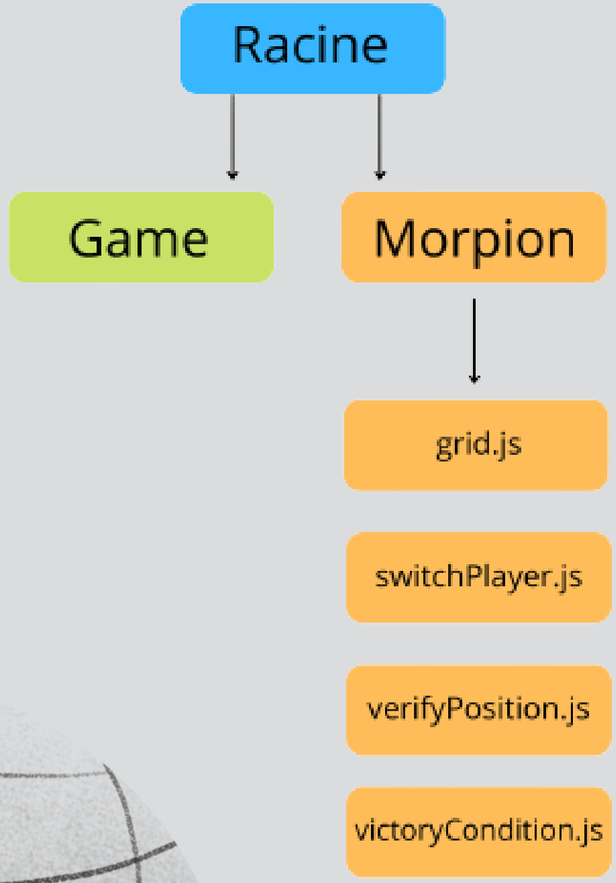
Yasser

Partie réseaux du jeux
BDD
Fusion des codes

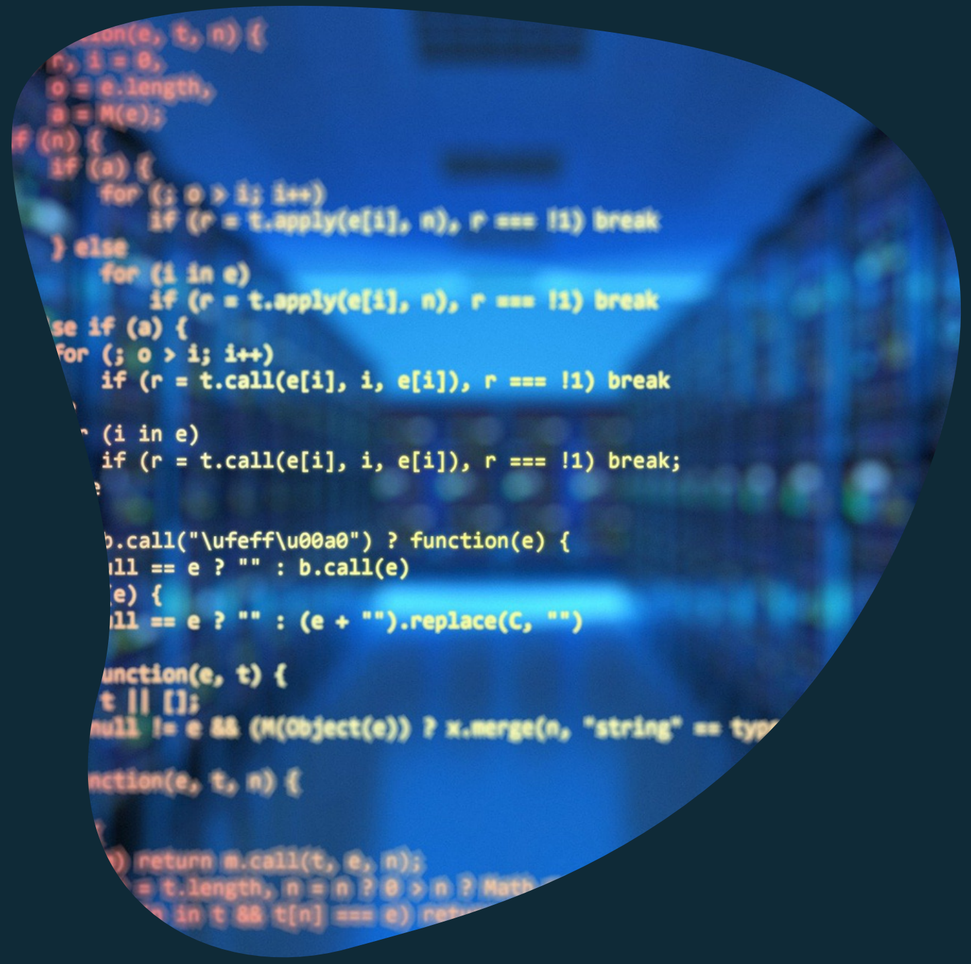
Enzo

Conception du jeux, UML
Annexe :
Powerpoint

Architecture logiciel & modèle de données



DÉMONSTRATION DU JEUX



PRÉSENTATION DU CODE

Grid.js

```
let line1=[0,0,0]
let line2=[0,0,0]
let line3=[0,0,0]

export let tab=[line1,line2,line3]; // tableau à double dimension

export function grid(tab)
{
    for(let i=0; i < tab.length; i++) //accès aux lignes 1, 2, 3
    {
        let design ="";

        for(let j=0; j < tab[i].length; j++) // | | --> à chaque case (0)
        {

            if (tab[i][j] === 0) //condition qui vérifie si des valeur on été
            {
                design +="| |"; //concaténation
            }

            else if (tab[i][j] === 1) //condition qui vérifie si des valeur on
            {
                design +="|X|";
            }

            else if (tab[i][j] === 2)
            {
                design +="|O|";
            }
        }

        console.log(design);

    }
}
```

https://www.youtube.com/watch?v=LHahCqmmvLY

Game.js

```
class Game {
    constructor(end, nbTour) {
        this.end = end;
        this.nbTour = nbTour;
    }
    async play() {
        this.grid = new grid_1.Grid();
        this.grid.buildGrid();
        while (!this.end) {
            let position = false;
            while (!position) {
                console.log("Où placez-vous votre morpion ?");
                var inputLine =   await rl.question('Line:', (answer) => {
                    inputLine = parseInt(answer);
                    rl.close();
                });;

                var inputColumn=await rl.question("Column : ", (answer) => {
                    inputColumn = parseInt(answer);
                    rl.close();
                });;
                 //insérer une valeur sur la colonne voulu
                position = gameController_1.GameController.verifyPosition(inputColumn, inputLine, grid_1.tab);
                if (!position) {
                    console.log("----------------------------");
                    console.log("Choisir une position valide");
                    console.log("----------------------------");
                }
            }
            this.nbTour--;
            this.grid.tab[inputLine - 1][inputColumn - 1] = gameController_1.turn; //valeur que le joueur numéro 1 saisie, -1 --> car tableau commence indice 0
            gameController_1.GameController.switchPlayer();
            this.grid.buildGrid(); //on réexécute le tableau pour mettre à jour les valeur
            this.end = gameController_1.GameController.victoryCondition();
            if (this.end) {
                if (gameController_1.turn === 1) {
                    console.log("Joueur 0 vous avez gagné !");
                }
                if (gameController_1.turn === 2) {
                    console.log("Joueur X vous avez gagné !");
                }
                return;
            }
            if (this.nbTour == 0) {
                console.log("Egalité !");
```

verifyPosition.js

```
    static verifyPosition(inputColumn, inputLine, tab) {
        if (inputLine >= 0 && inputLine <= 3)
            if (inputColumn >= 0 && inputColumn <= 3)
                if (tab[inputLine - 1][inputColumn - 1] === 0) {
                    //condition qui verifie si la saisie de la ligne est faite entre 1 et 3
                    //condition qui verifie si la saisie de la colonne est faite entre 1 et 3
                    //verifie si il y a un emplacement sur la position saisie
                    return true;
                }
                else {
                    return false;
                }
    }
}
```

switchPlayer.js

```
class GameController {
    static switchPlayer() {
        if (exports.turn === 1) {
            exports.turn = 2;
        }
        else {
            exports.turn = 1;
        }
    }
```

victoryCondition.js

```
15      static victoryCondition() {
16          if (grid_1.tab[0][0] === 1 && grid_1.tab[1][0] === 1 && grid_1.tab[2][0] === 1)
17              return true; //vertical gauche
18          if (grid_1.tab[0][1] === 1 && grid_1.tab[1][1] === 1 && grid_1.tab[2][1] === 1)
19              return true; //vertical milieu
20          if (grid_1.tab[0][2] === 1 && grid_1.tab[1][2] === 1 && grid_1.tab[2][2] === 1)
21              return true; //vertical droit
22          if (grid_1.tab[0][0] === 1 && grid_1.tab[0][1] === 1 && grid_1.tab[0][2] === 1)
23              return true; //horizontal haut
24          if (grid_1.tab[1][0] === 1 && grid_1.tab[1][1] === 1 && grid_1.tab[1][2] === 1)
25              return true; //horizontal milieu
26          if (grid_1.tab[2][0] === 1 && grid_1.tab[2][1] === 1 && grid_1.tab[2][2] === 1)
27              return true; //horizontal bas
28          if (grid_1.tab[0][0] === 1 && grid_1.tab[1][1] === 1 && grid_1.tab[2][2] === 1)
29              return true; //diagonal gauche-droite
30          if (grid_1.tab[0][2] === 1 && grid_1.tab[1][1] === 1 && grid_1.tab[2][0] === 1)
31              return true; //diagonal droite-gauche
32          //Conditions pour le joueur 0
33          if (grid_1.tab[0][0] === 2 && grid_1.tab[1][0] === 2 && grid_1.tab[2][0] === 2)
34              return true;
35          if (grid_1.tab[0][1] === 2 && grid_1.tab[1][1] === 2 && grid_1.tab[2][1] === 2)
36              return true;
37          if (grid_1.tab[0][2] === 2 && grid_1.tab[1][2] === 2 && grid_1.tab[2][2] === 2)
38              return true;
39          if (grid_1.tab[0][0] === 2 && grid_1.tab[0][1] === 2 && grid_1.tab[0][2] === 2)
40              return true;
41          if (grid_1.tab[1][0] === 2 && grid_1.tab[1][1] === 2 && grid_1.tab[1][2] === 2)
42              return true;
43          if (grid_1.tab[2][0] === 2 && grid_1.tab[2][1] === 2 && grid_1.tab[2][2] === 2)
44              return true;
45          if (grid_1.tab[0][0] === 2 && grid_1.tab[1][1] === 2 && grid_1.tab[2][2] === 2)
46              return true;
47          if (grid_1.tab[0][2] === 2 && grid_1.tab[1][1] === 2 && grid_1.tab[2][0] === 2)
48              return true;
49          return false;
50      }
```

BDD

```
function create_bdd(){
    const db = mysql_1.default.createConnection({
        host: "localhost",
        user: "root",
        password: "root"
      });

      db.connect(function(err) {
        if (err) {
            console.log("Erreur de connexion");
        };
        console.log("Connecté à la base de données MySQL!");
       db.query("CREATE DATABASE mabddjs", function (err, result) {

            if (err) {
                console.log("base de données existe déja");


            } else {
                console.log("Base de données créée !");
            };

            db.end();
          });
      });
}
```

```
function connectdb() {

    const db = mysql_1.default.createConnection({
        host: "localhost",
        user: "root",
        password: "root",
        database: "mabddjs",
    });
    let pseudo = process.argv[3];
    db.connect(function (err) {
        if (err)
            throw err;
            let sql = "CREATE TABLE IF NOT EXISTS game  ( pseudo Text,date TIMESTAMP DEFAULT CURRENT_TIMESTAMP)";
        db.query(sql);
        let sql_3 = "INSERT INTO `game`(`pseudo`) VALUES (?)";
        db.query(sql_3, pseudo);
                // db.query("SELECT * From game", function(err,result){
                //     console.log(result);
                // })

            console.log("Bienvenue " + pseudo);
            db.end();
        });

}
function delete_data() {
    const db = mysql_1.default.createConnection({
        host: "localhost",
        user: "root",
        password: "root",
        database: "mabddjs",
    });
    let pseudo = process.argv[3];
    db.connect(function (err) {
        if (err)
            throw err;
        if (err) {
            console.log("La base de données MySQL existe déjà");
            let sql = "DELETE FROM `game` WHERE pseudo = ?";
            db.query(sql, pseudo);
            db.end();
        }
    });
}
```

Réseau

```
////////////////////////////// Réseau //////////////////////////////////////////
// Serveur
if (process.argv[2] == "server") {
    const http_s = http_1.default.createServer();
    const io = new socket_io_1.Server(http_s);
    const port = 3000;
    let connectionsLimit = 2;
    create_bdd();
    http_s.listen(port, () => console.log(`Serveur en ecoute sur le port : ${port}`));
    io.on('connection', (socket) => {
        // limiter le nombre de connexion
        if (io.engine.clientsCount > connectionsLimit) {
            // @ts-ignore
            socket.emit({ message: 'reach the limit of connections' });
            socket.disconnect();
            console.log('Disconnected...');
            return;
        }
        else {
            console.log('connected');
        }
        socket.on('message', (evt) => {
            console.log(evt);
            socket.broadcast.emit('message', evt);
        });
    });
    io.on('disconnect', (evt) => {
        console.log('disconnected');
        delete_data();
    });

}
```

```
//Client
if (process.argv[2] == "client") {
    let server = 'http://localhost:3000';
    let socket = (0, socket_io_client_1.default)(server);
    this.game = new game_1.Game(false, 9).play();
    // let username=null;
    socket.on('connect', () => {

        connectdb();
        console.log('# Le jeu a commencé #');
        // username = process.argv[3]
    });
    socket.on('disconnect', function () {
        socket.emit('disconnect');
    });
    socket.on('message', data => {
        console.log(data.split('\n')[0]);
        console.log("à vous de jouer");
        this.game;

    });
    repl_1.default.start({
        prompt: '',
        eval: (data) => {
            socket.send(data);
            console.log("Attendez votre tour");
            this.game;


        }
    });
}
```

AXE D'AMÉLIORATION

Coté jeux:
-Proposer une version web du jeux

Coté réseau:
-Synchroniser le jeux

CONCLUSION

MERCI !