Kick Tournament

Par Niels Terese (1G6) & Maël Peron (1G9)

Partie Niels

Installation du serveur NodeJS

Tout se passe dans le terminal !

Installation de NodeJS :

- curl -sL https://deb.nodesource.com/setup_12.x | sudo bash -
- sudo apt install nodejs

Copier le dépôt du projet :

- git clone git@github.com:NielsTRS/Kick_Tournament.git

Installation des dépendances :

- cd Kick_Tournament
- npm install

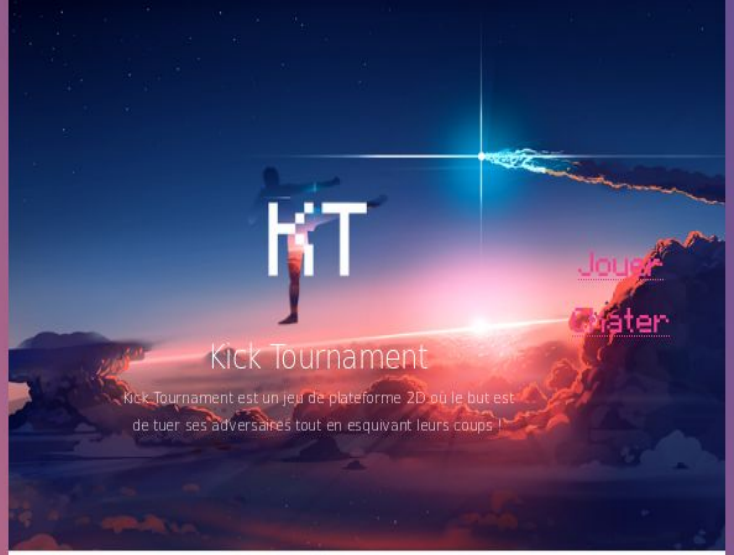
Lancement du serveur :

- node app.js

Vous pouvez maintenant accéder au jeu en démarrant votre navigateur web et en vous rendant à l'adresse : http://127.0.0.1:2000

Création du design

SASS / Compass / CSS



Chatez avec d'autres joueurs

Texte

Envoyer

Messages

Création des pages HTML + chat

HTML / NodeJS / jQuery

```html
<section class="main" id="chat">
    <h2>Chatez avec d'autres joueurs</h2>
    <form action="#" method="post">
        <div class="row gtr-uniform">
            <div class="col-12">
                <input id="message" name="message" placeholder="Texte" type="text" value=""/>
                <button class="primary">Envoyer</button>
            </div>
        </div>
    </form>
</section>
```

HTML

NodeJS

```nodejs
socket.on('chat', function (msg,id) {
    io.emit('chat', msg, id);
});
```

jQuery

```jquery
$(function () {
    var socket = io();
    $('form').submit(function (e) {
        e.preventDefault(); // prevents page reloading
        socket.emit('chat', $('#message').val(), socket.id);
        $('#message').val('');
        return false;
    });
    socket.on('chat', function (msg, id) {
        $('#messages').prepend($('<li>').text(id + ": " + msg));
    });
});
```

Partie Maël

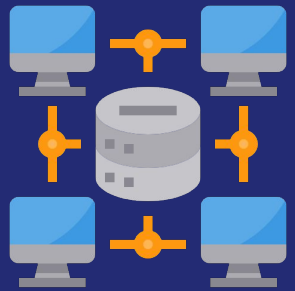
À la recherche des bibliothèques

Processing,
Benjamin Fry et Casey Reas

En Java
(https://fr.wikipedia.org/wiki/Processing)

p5js,
Processing Foundation

En Javascript
(p5js.org)



L’idée de jeu

Kick Tournament

Description du projet

Jeu de plateforme en 2D où le but de tuer ses adversaires tout en esquivant ses coups.

Gameplay

Vous incarnez J1 (abréviation de Joueur n°1), lors de votre apparition sur le terrain, votre personnage apparaît vers l'une des extrémités

Mécaniques

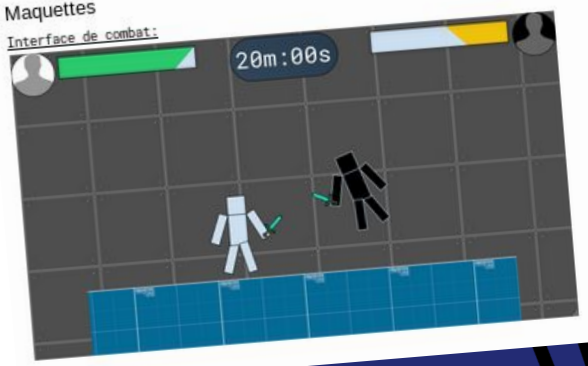
- Les joueurs ont chacun une vie qui, quand elle diminue, change de couleur et va du vert au rouge (Vert,Jaune,Orange,Rouge)
- double jump ? (Capacités en fonction des héros choisis)
- punch/kick pour frapper les autres joueurs (Cooldown 0.2sec)
- punch chargé mais agilité réduite (charge limite de 3/4sec)
- Lorsqu'un joueur est éjecté et passe sous la plateforme, la gravité est modifiée.
- ~~pourcentage smash bros~~

Commandes

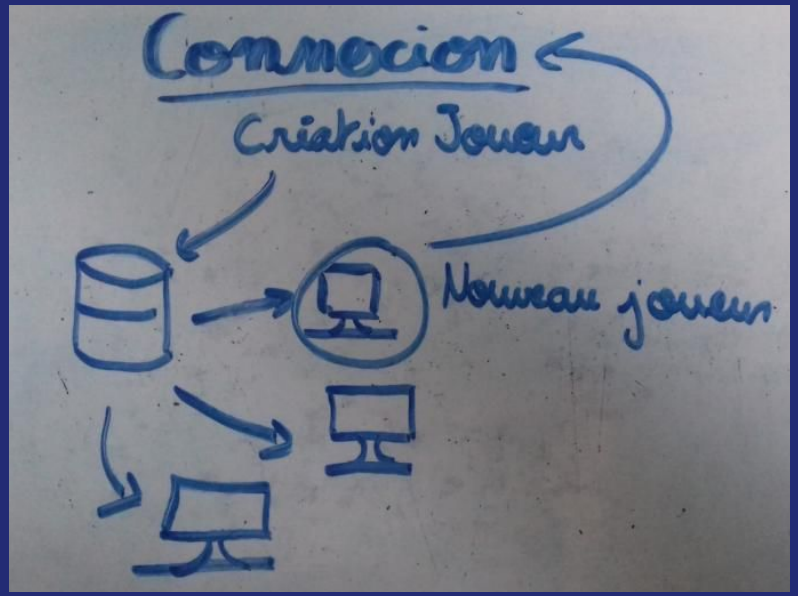
- Les déplacements se feraient soit grâce aux touches directionnelles, soit grâce aux touches "qd" (elles-mêmes personnalisables selon les envies de l'utilisateur).
- La barre espace (ou la touche "z") permettrait au joueur de faire un saut, et quand elle est rappuyée durant ce dernier, pour en faire un deuxième.
- Pour frapper des joueurs: "Clic gauche"
- Pour charger une frappe et donc attaquer plus violemment un joueur: "Maintenir pendant quelques temps et relâcher le clic gauche"
- Pour lancer un grappin à l'endroit où se trouve la souris: "Clic droit".
- Touche pour esquiver les grappins Touche "Ctrl".
- Interagir avec des éléments du décor. Touche "F" ou "E".
- Blocs boost/bonus. Touches situées sur le "pavé numérique supérieur".

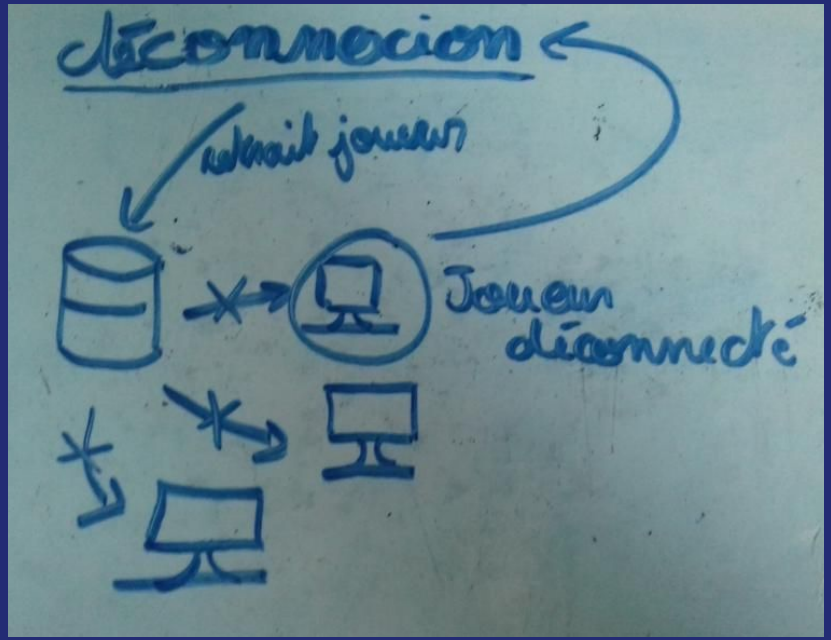
Maquettes

Interface de combat:



Concevoir l’algorithme


Connecion
Criation Joueur
Nouveau joueur


déconnecion
retrait joueur
Joueur
déconnecté

Etc...

Traduction + Correction des bugs

Particules de déplacement

Écran de mort

Deux joueurs connectés en même temps

```
class Player {
    constructor(x, y, w, h, img) {
        this.x = x;
        this.y = y;
        this.w = w;
        this.h = h;
        this.life = 100;
        this.totalLife = 100;
        this.right = false;
        this.left = false;
        this.gravity = 0.2;
        this.vy = 0;
        this.speed = 10;
        this.lockSound = false;
        this.img = img;
        this.jumps = {count: 2, cooldown: 60}
    }
}
```

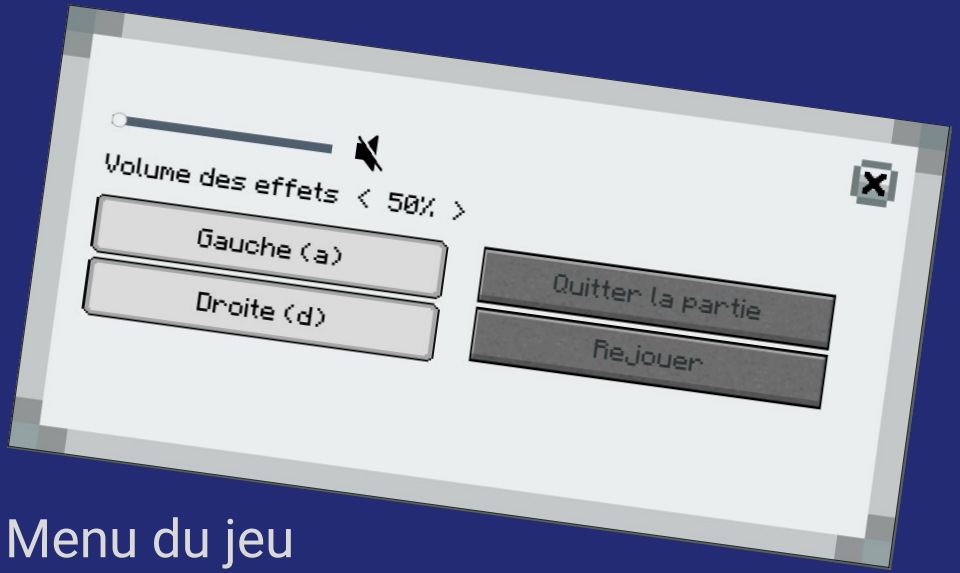
```
constrain() {
    this.x = constrain(this.x, 0, Screen.x - this.w);
    this.y = constrain(this.y, 0, Screen.y - this.h - Floor / 2);
}

show() {
    fill(255);
    image(this.img, this.x, this.y, this.w, this.h)
    fill(149, 165, 166)
    rect(this.x, this.y - 20, this.w, 5)
    if (this.life > 0) fill(colors.red)
    if (this.life > this.totalLife / 4 * 1) fill(colors.orange)
    if (this.life > this.totalLife / 4 * 2) fill(colors.yellow)
    if (this.life > this.totalLife / 4 * 3) fill(colors.green)
    rect(this.x, this.y - 20, this.life / 100 * this.w, 5)

}
```

```
function keyPressed() {
    if (!player.death()) {
        if (keyCode === LEFT_ARROW || key === getCookie("left") && !menuOpened) player.left = true;
        if (keyCode === RIGHT_ARROW || key === getCookie("right") && !menuOpened) player.right = true;
        if (key === " " && !menuOpened) player.jump();
    }
    if (keyCode === ESCAPE) menuUi();
    if (key === "s") player.life = 0;
}

function keyReleased() {
    if (keyCode === LEFT_ARROW || key === getCookie("left") && !menuOpened) player.left = false;
    if (keyCode === RIGHT_ARROW || key === getCookie("right") && !menuOpened) player.right = false;
}

function mousePressed() {
    if (mouseButton == "left" && !player.death() && !menuOpened) {
        try {
            playSound('/public/assets/sound/sweep.wav', volume)
        } catch (err) {
        }
        distance = (player.x + player.w / 2) - mouseX
        for (i of players) {
            point(mouseX, mouseY)
            if (socket.id != i.id && collidePointEllipse(mouseX, mouseY, i.x + i.w / 2, i.y + i.h / 2, i.w, i.h)) {
                socket.emit('hit', {hitted: i.id, hitter: socket.id, damage: 10, x: mouseX, y: mouseY})
            }
        }
        socket.emit('attack', {dist: distance * -1, x: mouseX - 50, y: mouseY - 50})
    } else {

    }
}
```

```
function draw() {
    background(51);

    displayBackground();
    displayParticles();
    displayPlayers(socket.id);

    displaySweeps();
    displayDeconnections();

    pack = {
        x: player.x,
        y: player.y,
        w: player.w,
        h: player.h,
        life: player.life,
        dead: player.death()
    }
    socket.emit('update', pack)

```

Données joueurs

document.cookie.get(“volume”);



Menu du jeu

FIN