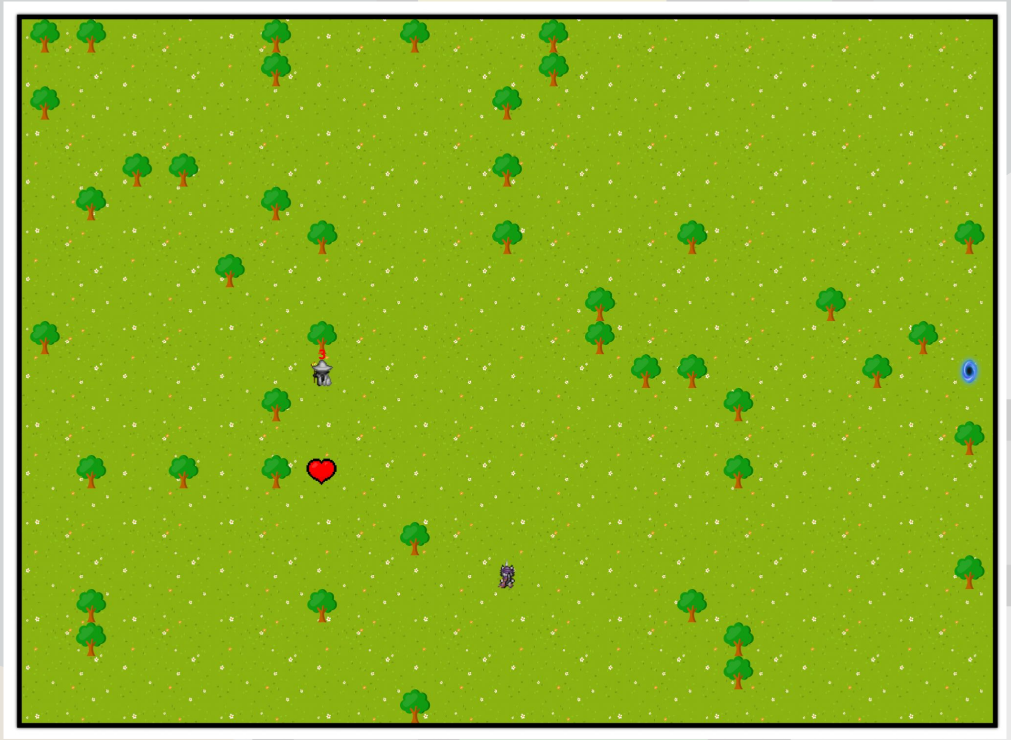
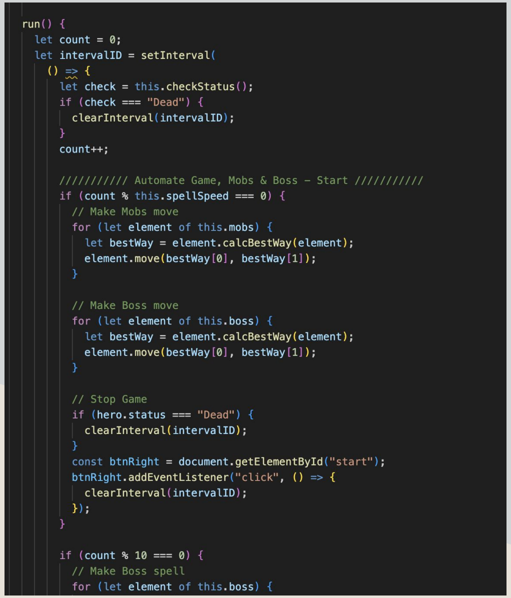
The Adventure

IronHack project

Project Elevator Pitch

- Zelda like game
- Needs several algorithms to make monsters move and attack
- Quite “infiny” to develop.





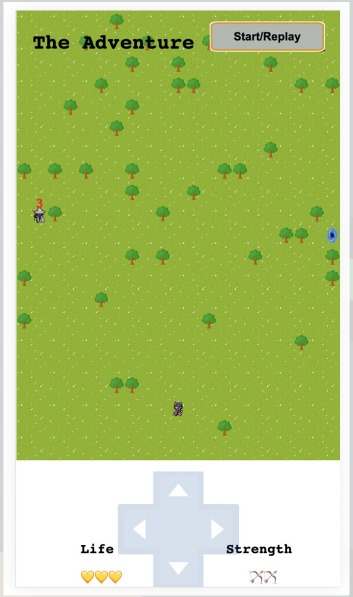
```
run() {
  let count = 0;
  let intervalID = setInterval(
    () => {
      let check = this.checkStatus();
      if (check === "Dead") {
        clearInterval(intervalID);
      }
      count++;

      /////////// Automate Game, Mobs & Boss - Start ///////////
      if (count % this.spellSpeed === 0) {
        // Make Mobs move
        for (let element of this.mobs) {
          let bestWay = element.calcBestWay(element);
          element.move(bestWay[0], bestWay[1]);
        }

        // Make Boss move
        for (let element of this.boss) {
          let bestWay = element.calcBestWay(element);
          element.move(bestWay[0], bestWay[1]);
        }

        // Stop Game
        if (hero.status === "Dead") {
          clearInterval(intervalID);
        }
        const btnRight = document.getElementById("start");
        btnRight.addEventListener("click", () => {
          clearInterval(intervalID);
        });
      }

      if (count % 10 === 0) {
        // Make Boss spell
        for (let element of this.boss) {
```

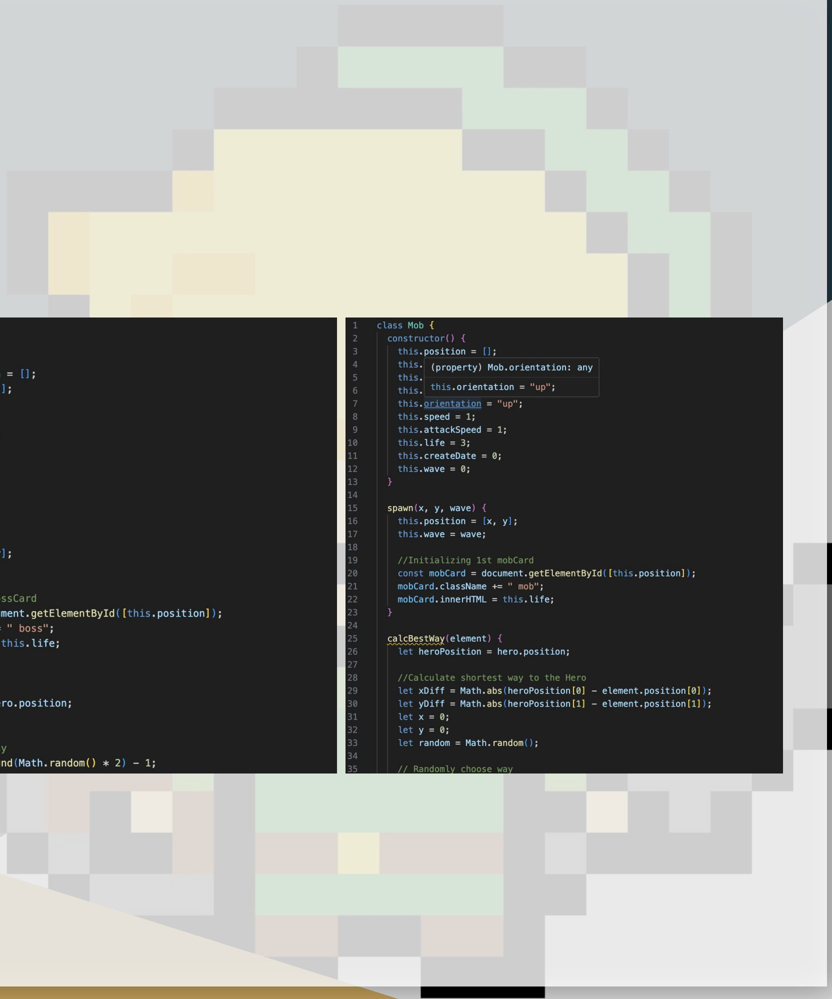


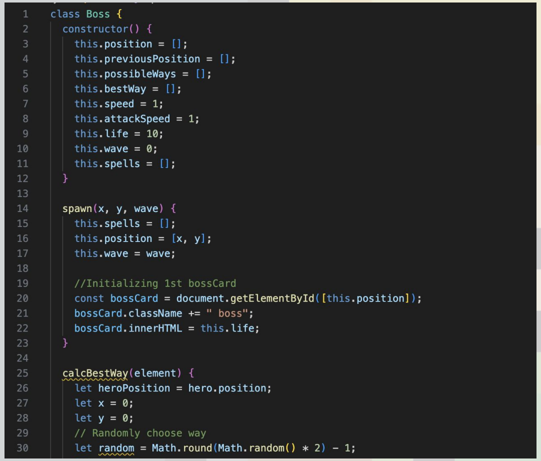
Technical Challenge

- Create the “engine” to permanently calculate all coordinates, movement and actions.
- Different timeIntervental to manage.
- Generate random maps
- Making it playable on mobile

Big Mistake

- Not thinking about the whole game before starting to code!
- Add the build, unbuild, rebuild several time several class because I wanted to add new functionalities.



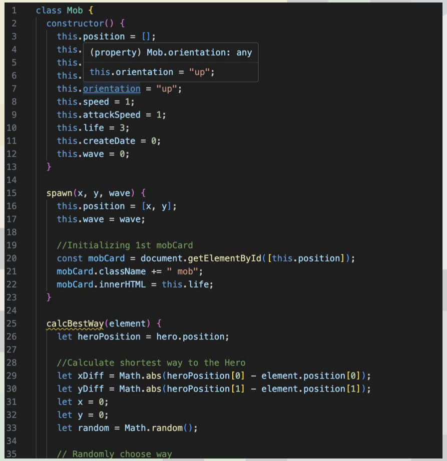


```
class Boss {
  constructor() {
    this.position = [];
    this.previousPosition = [];
    this.possibleWays = [];
    this.bestWay = [];
    this.speed = 1;
    this.attackSpeed = 1;
    this.life = 10;
    this.wave = 0;
    this.spells = [];
  }

  spawn(x, y, wave) {
    this.spells = [];
    this.position = [x, y];
    this.wave = wave;

    //Initializing 1st bossCard
    const bossCard = document.getElementById([this.position]);
    bossCard.className += " boss";
    bossCard.innerHTML = this.life;
  }

  calcBestWay(element) {
    let heroPosition = hero.position;
    let x = 0;
    let y = 0;
    // Randomly choose way
    let random = Math.round(Math.random() * 2) - 1;
```



```
class Mob {
  constructor() {
    this.position = [];
    this.  (property) Mob.orientation: any
    this.
    this.  this.orientation = "up";
    this.orientation = "up";
    this.speed = 1;
    this.attackSpeed = 1;
    this.life = 3;
    this.createDate = 0;
    this.wave = 0;
  }

  spawn(x, y, wave) {
    this.position = [x, y];
    this.wave = wave;

    //Initializing 1st mobCard
    const mobCard = document.getElementById([this.position]);
    mobCard.className += " mob";
    mobCard.innerHTML = this.life;
  }

  calcBestWay(element) {
    let heroPosition = hero.position;

    //Calculate shortest way to the Hero
    let xDiff = Math.abs(heroPosition[0] - element.position[0]);
    let yDiff = Math.abs(heroPosition[1] - element.position[1]);
    let x = 0;
    let y = 0;
    let random = Math.random();

    // Randomly choose way
```

Demo Slide

DEMO

The Adventure

Thank you

Alexandre Smadja