const canvas = document.querySelector("canvas");

const c = canvas.getContext("2d");

canvas.width = 1024;

canvas.height = 576;

c.fillRect(0, 0, canvas.width, canvas.height);

const gravity = 0.7

class sprite {

    constructor({ position, velocity, color = 'red', offset }) {

        this.position = position

        this.velocity = velocity

        this.width = 50

        this.height = 150

        this.lastkey

        this.attackBox = {

            position: {

                x: this.position.x,

                y: this.position.y

            },

            offset,

            width: 100,

            height: 50,

        }

        this.color = color

        this.isAttacking

        this.health = 100

    }

    draw() {

        c.fillStyle = this.color

        c.fillRect(this.position.x, this.position.y, this.width, this.height)

        if (this.isAttacking) {

            c.fillStyle = "green"

            c.fillRect(this.attackBox.position.x,

                this.attackBox.position.y,

                this.attackBox.width,

                this.attackBox.height)

        }

    }

    update() {

        this.draw()

        this.attackBox.position.x = this.position.x - this.attackBox.offset.x

        this.attackBox.position.y = this.position.y

        this.position.x += this.velocity.x

        this.position.y += this.velocity.y

        if (this.position.y + this.height + this.velocity.y >= canvas.height) {

            this.velocity.y = 0

        } else {

            this.velocity.y += gravity

        }

    }

    attack() {

        this.isAttacking = true

        setTimeout(() => {

            this.isAttacking = false

        }, 100)

    }

}

const player = new sprite({

    position: {

        x: 0,

        y: 0

    },

    velocity: {

        x: 0,

        y: 0

    },

    offset: {

        x: 0,

        y: 0

    }

})

const enemy = new sprite({

    position: {

        x: 400,

        y: 100

    },

    velocity: {

        x: 0,

        y: 0

    },

    color: 'blue',

    offset: {

        x: 50,

        y: 0

    }

})

const keys = {

    a: {

        pressed: false

    },

    d: {

        pressed: false

    },

    w: {

        pressed: false

    },

    ArrowRight: {

        pressed: false

    },

    ArrowLeft: {

        pressed: false

    },

    ArrowUp: {

        pressed: false

    }

}

function rectangularCollision({ rectangle1, rectangle2 }) {

    return (

        rectangle1.attackBox.position.x + rectangle1.attackBox.width >= rectangle2.position.x &&

        rectangle1.attackBox.position.x <= rectangle2.position.x + rectangle2.width &&

        rectangle1.attackBox.position.y + rectangle1.attackBox.height >= rectangle2.position.y &&

        rectangle1.attackBox.position.y <= rectangle2.position.y + rectangle2.height

    )

}

function determineWinner({ player, enemy, timerId }) {

    clearTimeout(timerId)

    document.querySelector('#displayText').style.display = 'flex'

    if (player.health === enemy.health) {

        document.querySelector('#displayText').innerHTML = 'Tie'

    } else if (player.health > enemy.health) {

        document.querySelector('#displayText').innerHTML = 'Player 1 win'

    } else {

        document.querySelector('#displayText').innerHTML = 'player 2 win'

    }

}

let timer = 60

let timerId

function decreaseTimer() {

    if (timer > 0) {

        timerId = setTimeout(decreaseTimer, 1000)

        timer--

        document.querySelector('#timer').innerHTML = timer

    }

    if (timer === 0) {

        determineWinner({ player, enemy, timerId })

    }

}

decreaseTimer()

function animate() {

    window.requestAnimationFrame(animate)

    c.fillStyle = "black"

    c.fillRect(0, 0, canvas.width, canvas.height)

    player.update()

    enemy.update()

    player.velocity.x = 0

    enemy.velocity.x = 0

    if (keys.a.pressed && player.lastkey === 'a') {

        player.velocity.x = -5

    } else if (keys.d.pressed && player.lastkey === 'd') {

        player.velocity.x = 5

    }

    if (keys.ArrowLeft.pressed && enemy.lastkey === 'ArrowLeft') {

        enemy.velocity.x = -5

    } else if (keys.ArrowRight.pressed && enemy.lastkey === 'ArrowRight') {

        enemy.velocity.x = 5

    }

    if (rectangularCollision({

            rectangle1: player,

            rectangle2: enemy

        }) &&

        player.isAttacking) {

        player.isAttacking = false

        enemy.health -= 20

        if (player.health < 100 && player) { player.health += 10 }

        document.querySelector('#enemy-health').style.width = enemy.health + '%'

        document.querySelector('#player-health').style.width = player.health + '%'

    }

    if (rectangularCollision({

            rectangle1: enemy,

            rectangle2: player

        }) &&

        enemy.isAttacking) {

        enemy.isAttacking = false

        player.health -= 20

        if (enemy.health < 100) { enemy.health += 10 }

        document.querySelector('#player-health').style.width = player.health + '%'

        document.querySelector('#enemy-health').style.width = enemy.health + '%'

    }

    if (player.health === 0 || enemy.health === 0) {

        determineWinner({ player, enemy, timerId })

    }

}

animate()

window.addEventListener('keydown', (event) => {

    switch (event.key) {

        case 'd':

            keys.d.pressed = true

            player.lastkey = 'd'

            break

        case 'a':

            keys.a.pressed = true

            player.lastkey = 'a'

            break

        case 'w':

            player.velocity.y = -20

            break

        case ' ':

            player.attack()

            break

        case 'ArrowRight':

            keys.ArrowRight.pressed = true

            enemy.lastkey = 'ArrowRight'

            break

        case 'ArrowLeft':

            keys.ArrowLeft.pressed = true

            enemy.lastkey = 'ArrowLeft'

            break

        case 'ArrowUp':

            enemy.velocity.y = -20

            break

        case 'ArrowDown':

            enemy.isAttacking = true

            break

    }

})

window.addEventListener('keyup', (event) => {

    switch (event.key) {

        case 'd':

            keys.d.pressed = false

            break

        case 'a':

            keys.a.pressed = false

            break

        case 'w':

            keys.w.pressed = false

            break

        case 'ArrowRight':

            keys.ArrowRight.pressed = false

            break

        case 'ArrowLeft':

            keys.ArrowLeft.pressed = false

            break

        case 'ArrowUp':

            keys.ArrowUp.pressed = false

            break

    }

})