```
let game;
let gameOptions = {

    // duration of the wall, in milliseconds
    wallDuration: 100,

    // ball start speed, in pixels/second
     ballIntialSpeed: 500,

    // ball speed increase at each successful bounce, in pixels/second
     ballIntialAccelleration: 20
}
window.onload = function() {
    let gameConf = {
        width: 480,
        height: 640,
        scene: gamePlay,
        backgroundColor: 0x222222,
        physics: {
            default: "arcade"
        }
    }
    game = new Phaser.Game(gameConf);
    window.focus()
    resize();
    window.addEventListener("resize", resize, false);
}
class gamePlay extends Phaser.Scene{
    constructor(){
        super("gamePlay");
    }
    preload(){
        this.load.image("ball", "ball.png");
        this.load.image("wall", "wall.png");
    }
    create(){
        this.gameOver = false;
        this.canActivateWall = true;
        this.ballSpeed = gameOptions. ballIntialSpeed;
        this.wallGroup = this.physics.add.group();
        this.theBall = this.physics.add.image(game.config.width / 2, game.config.height * 4 /
5, "ball");
        this.theBall.body.setCircle(25)
        this.theBall.setBounce(1)
        this.createWall(32, game.config.height / 2, 32, game.config.height - 96);
        this.createWall(game.config.width - 32, game.config.height / 2, 32,
game.config.height - 96);
        this.createWall(game.config.width / 2, 32, game.config.width - 32, 32);
        this.lowerWall = this.createWall(game.config.width / 2, game.config.height - 32,
game.config.width - 32, 32);
        this.physics.add.collider(this.theBall, this.wallGroup, function(ball, wall){
            this.canActivateWall = true;
            if(wall.x == this.lowerWall.x && wall.y == this.lowerWall.y){
                this.ballSpeed += gameOptions. ballIntialAccelleration;
                let ballVelocity = this.physics.velocityFromAngle(Phaser.Math.Between(220,
320), this.ballSpeed);
                this.theBall.setVelocity(ballVelocity.x, ballVelocity.y);
            }
        }, null, this);
        this.input.on("pointerdown", this.activateWall, this);
    }
    createWall(posX, posY, width, height){
        let wall = this.physics.add.image(posX, posY, "wall");
        wall.displayWidth = width;
        wall.displayHeight = height;
        this.wallGroup.add(wall);
        wall.setImmovable();
        return wall;
    }
    activateWall(){
        if(this.theBall.body.speed == 0){
            let ballVelocity = this.physics.velocityFromAngle(Phaser.Math.Between(220, 320),
this.ballSpeed)
            this.theBall.setVelocity(ballVelocity.x, ballVelocity.y);
            this.lowerWall.alpha = 0.1;
            this.lowerWall.body.checkCollision.none = true;
            return;
        }
        if(this.canActivateWall){
            this.canActivateWall = false;
            this.lowerWall.alpha = 1;
            this.lowerWall.body.checkCollision.none = false;
            let wallEvent = this.time.addEvent({
                delay: gameOptions.wallDuration,
                callbackScope: this,
                callback: function(){
                    this.lowerWall.alpha = 0.1;
                    this.lowerWall.body.checkCollision.none = true;
                }
            });
        }
    }
    update(){
        if((this.theBall.y > game.config.height || this.theBall.y < 0) && !this.gameOver){
            this.gameOver = true;
            this.cameras.main.shake(800, 0.05);
            this.time.addEvent({
                delay: 800,
                callbackScope: this,
                callback: function(){
                    this.scene.start("gamePlay");
                }
            });
        }
    }
}
function resize() {
    var canvas = document.querySelector("canvas");
    var windowWidth = window.innerWidth;
    var windowHeight = window.innerHeight;
    var windowRatio = windowWidth / windowHeight;
    var gameRatio = game.config.width / game.config.height;
    if(windowRatio < gameRatio){
        canvas.style.width = windowWidth + "px";
        canvas.style.height = (windowWidth / gameRatio) + "px";
    }
    else{
        canvas.style.width = (windowHeight * gameRatio) + "px";
        canvas.style.height = windowHeight + "px";
    }
}
```