【参考したサイト等の主な説明】

私の作品は以下の動画の途中部分までを参考に作成しました。

<https://www.youtube.com/watch?v=v2pS8e22-Fw>

参考動画におけるライセンス関係についての記述はありませんでした。

また背景に設定している画像につきましては、以下のサイトよりダウンロードし、そのまま設定させていただいています。こちらに関しましても、ライセンス関係についてはわかりませんでした。

<http://www.bing.com/images/search?q=%e7%94%bb%e5%83%8f+%e3%83%96%e3%83%a9%e3%83%83%e3%82%af%e3%83%9b%e3%83%bc%e3%83%ab&view=detailv2&&id=CD57DA29E1E18ABBE313F8784F76A1671261F9C9&selectedIndex=149&ccid=XA44PBCD&simid=608004114186306432&thid=OIP.M5c0e383c1083803f3a70b636396716edo0&ajaxhist=0>

【私の作品のコード】

ArrayList<Block> blocks = new ArrayList<Block>();

//ブロックをランダムに80個配置　ArrayListでブロックを管理し、画面いっぱいに適当に配置

// for関数のiで数を指定

Player player;

PVector basePos = new PVector();

//setup()は初期化、drawは繰り返し実行

void setup(){

size(1000,600);

blocks.add(player = new Player(100,100));

for(int i=0;i<80; i++){

blocks.add(new Block(random(width),height - random(height \* 7),random(50,350),10,color(255)));

} //上方向に画面7つ分ランダムに配置し、ブロックをまばらにする

}

void draw(){

PImage img;

img = loadImage("hole2.jpg"); //背景

image(img,0,0);

basePos.lerp(new PVector(-player.pos.x + width/2, -player.pos.y + height/2),0.02f);

translate(basePos.x,basePos.y);

for(Block b:blocks){

b.update();

b.draw();

}

}

//class 「型」のこと

class Block{

PVector pos = new PVector();

PVector size = new PVector();

PVector vel = new PVector();

PVector prevPos = new PVector();

//ステージに配置するブロッククラス、位置、大きさ、速度、前の位置

color col;

Block(float x, float y,float w, float h, color col){

//X,Y位置、横幅、縦幅、色でブロックを生成

pos.set(x,y,0);

size.set(w/2,h/2,0);

this.col = col;

}

void update(){

prevPos.set(pos);

pos.add(vel);

}

void draw(){

fill(col);

rectMode(CENTER);

ellipse(pos.x, pos.y, size.x\*2, size.y\*2);

}

boolean syototu(Block b){

return abs(pos.x - b.pos.x)<abs(size.x + b.size.x)&&

abs(pos.y - b.pos.y)<abs(size.y + b.size.y);

//プレイヤーとブロックの衝突判定

//衝突していたらsyototu()がtrueを返すように

}

}

class Player extends Block{

boolean landing;

int jumpingFrames;

Player(float x, float y){

//プレイヤークラスを用意し、ブロッククラスを継承しつつ、位置だけで初期化をできるように

super(x,y,20,20,color(245,250,10));

}

void update(){

super.update();

float x = mouseX - pos.x - basePos.x;

if(abs(x)>size.x)vel.x += x<0 ? -0.2f : 0.2f;

//マウスの横移動

vel.y += 0.5f; //重力 +方向がY画面下

if(mousePressed){

if(landing){

vel.y -= 2.0f;

jumpingFrames =30; //滞空時間30フレーム

}if(jumpingFrames-->0){

vel.y -= 1.0f;

//

}

}else jumpingFrames = 0;

vel.mult(1f);

vel.limit(10.0f);

//少し止まる

if(pos.y>height)pos.y= -height;

//下に消えたら上から降ってくる

landing = false;

for(Block b: blocks){

if(b==this)continue;

if(syototu(b)){

if(prevPos.y > b.pos.y && b.pos.y - b.size.y <= b.pos.y + b.size.y){

pos.y = size.y + b.pos.y + b.size.y;

}else if(pos.y + size.y >=b.pos.y - b.size.y){

pos.y = -size.y + b.pos.y - b.size.y;

landing = true; //landing 地面接地

}

}

}

}

}

＜参考にしたサイトの作品のコード＞

ArrayList<Block> blocks = new ArrayList<Block>();

//ブロックをランダムに100個配置　ArrayListでブロックを管理し、画面いっぱいに適当に配置

// for関数のiで数を指定

Player player;

PVector basePos = new PVector();

void setup(){

size(1200,600);

blocks.add(player = new Player(100,100));

for(int i=0;i<100; i++){

blocks.add(new Block(random(width),height - random(height \* 6),random(50,350),10,color(0)));

}

}

void draw(){

background(255); //白背景

basePos.set(-player.pos.x + width/2, -player.pos.y + height/2,0);

translate(basePos.x,basePos.y);

for(Block b:blocks){

b.update();

b.draw();

}

}

class Block{

PVector pos = new PVector();

PVector size = new PVector();

PVector vel = new PVector();

PVector prevPos = new PVector();

//ステージに配置するブロッククラス、位置、大きさ、速度、前の位置

color col;

Block(float x, float y,float w, float h, color col){

//X,Y位置、横幅、縦幅、色でブロックを生成

pos.set(x,y,0);

size.set(w/2,h/2,0);

this.col = col;

}

void update(){

prevPos.set(pos);

pos.add(vel);

}

void draw(){

fill(col);

rectMode(CENTER);

rect(pos.x, pos.y, size.x\*2, size.y\*2);

}

boolean isHit(Block b){

return abs(pos.x - b.pos.x)<abs(size.x + b.size.x)&&

abs(pos.y - b.pos.y)<abs(size.y + b.size.y);

//ブロック同士（プレイシャーもブロック）のあたりを判定

}

}

class Player extends Block{

boolean islanding;

int jumpingFrames;

Player(float x, float y){

//プレイヤークラスを用意し、ブロッククラスを継承しつつ、位置だけで初期化をできるように

super(x,y,50,50,color(255,0,0));

}

void update(){

super.update();

float x = mouseX - pos.x - basePos.x;

if(abs(x)>size.x)vel.x += x<0 ? -0.2f : 0.2f;

//マウスの横移動

vel.y += 0.2f;

//重力 +方向がY画面下

if(mousePressed){

if(islanding){

vel.y -= 2.0f;

jumpingFrames =12;

}if(jumpingFrames-->0){

vel.y -= 1.0f;

//ジャンプ　１２フレームの滞空時間で

}

}else jumpingFrames = 0;

vel.mult(0.98f);

vel.limit(10.0f);

//少し止まる

if(pos.y>height)pos.y= -height;

//下に消えたら上から降ってくる

islanding = false;

for(Block b: blocks){

if(b==this)continue;

if(isHit(b)){

if(prevPos.y > b.pos.y && b.pos.y - b.size.y <= b.pos.y + b.size.y){

pos.y = size.y + b.pos.y + b.size.y;

}else if(pos.y + size.y >=b.pos.y - b.size.y){

pos.y = -size.y + b.pos.y - b.size.y;

islanding = true;

}

}

}

}

}