《糖果车间》软件V1.0

import Enemy from "./Enemy";

import Candy from "./Candy";

export default class MainSceneControl extends Laya.Script {

public candy: Laya.Prefab;

public candyParent: Laya.Sprite;

public candy\_Explode: Laya.Prefab;

public candy\_ExplodeParent: Laya.Sprite;

public explode: Laya.Prefab;

public roleParent: Laya.Sprite;

public enemy: Laya.Prefab;

public enemyParent: Laya.Sprite;

public enemyBullet: Laya.Prefab;

public background: Laya.Sprite;

public speakBoxParent: Laya.Sprite;

public speakBox: Laya.Prefab;

public bulletParent: Laya.Sprite;

public roleBullet: Laya.Prefab;

public scoreLabel: Laya.FontClip;

public hintWord: Laya.Prefab;

public rewardWords: Laya.Prefab;

public timer: Laya.Sprite;

private timeSchedule: Laya.ProgressBar;

public displays: Laya.Image;

public clickHintSign: Laya.Sprite;

public operating: Laya.Sprite;

public assembly: Laya.Sprite;

public role\_01: Laya.Sprite;

public role\_02: Laya.Sprite;

public resurgence: Laya.Prefab;

public settlement: Laya.Prefab;

public ranking: Laya.Prefab;

public score: Laya.Prefab;

public enemyHint: Laya.Prefab;

private role\_01speak: Laya.Sprite;

private role\_02speak: Laya.Sprite;

private enemyAppear: boolean;

private enemyTagRole: Laya.Sprite;

private enemyCount: number;

private candy\_interval: number;

private creatTime: number;

private creatOnOff: boolean;

private nameArr: Array<string>;

private candyCount: number;

private rescueNum: number;

private timerControl: number;

private enemyProperty: any;

private enemyInterval\_01: number;

private enemyTime\_01: number;

private enemySwitch\_01: boolean;

private enemyInterval\_02: number;

private enemyTime\_02: number;

private enemySwitch\_02: boolean;

private posArr\_left: Array<Array<number>>;

private posArr\_right: Array<Array<number>>;

private candyNameArr: Array<string>;

private startRow: number;

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private suspend: boolean;

private launchTemp\_01: Laya.Templet;

private candyLaunch\_01: Laya.Skeleton;

private launchTemp\_02: Laya.Templet;

private candyLaunch\_02: Laya.Skeleton;

private gameOver: boolean;

constructor() { super(); }

onEnable(): void {

this.noStarted();

this.createStartInterface('start');

this.wxPostInit();

this.adaptive();

}

adaptive(): void {

this.assembly.y = Laya.stage.height / 2 - 150;

this.role\_01.y = Laya.stage.height \* 0.71;

this.role\_02.y = Laya.stage.height \* 0.71 - 24;

this.operating.y = Laya.stage.height \* 0.85;

this.owner.scene.height = Laya.stage.height;

}

startGame(): void {

this.enemyAppear = false;

this.enemyTagRole = null;

this.enemyCount = 0;

this.enemyProperty = {

blood: 200,

attackValue: 10,

attackSpeed: 1500,

defense: 10,

moveSpeed: 10,

creatInterval: 8000

}

this.enemyInterval\_01 = 500;

this.enemyTime\_01 = Date.now();

this.enemySwitch\_01 = false;

this.enemyInterval\_02 = 500;

this.enemyTime\_02 = Date.now();

this.enemySwitch\_02 = false;

this.candy\_interval = 1000;

this.creatTime = Date.now();

this.creatOnOff = true;

this.nameArr = [];

this.candyCount = 0;

this.scoreLabel.value = '0';

this.timeSchedule = this.timer.getChildByName('timeSchedule') as Laya.ProgressBar;

this.rescueNum = 0;

Laya.MouseManager.multiTouchEnabled = false;

this.timerControl = 0;

this.suspend = false;

this.startRow = 4;

this.gameOver = false;

this.assembly['Assembly'].pipeAnimation('flow');

this.roleAppear();

this.candyMoveToDisplay();

this.createEnemyHint();

}

noStarted(): void {

this.self = this.owner as Laya.Scene;

this.owner['MainSceneControl'] = this;

this.gameOver = true;

this.role\_01.x = -500;

this.role\_02.x = 1834;

this.createLaunchAni();

}

createStartInterface(type): void {

Let startInterface = Laya.Pool.getItemByCreateFun('startInterface', this.startInterface.create, this.startInterface) as Laya.Sprite;

this.owner.addChild(startInterface);

startInterface.pivotX = startInterface.width / 2;

startInterface.pivotY = startInterface.height / 2;

startInterface.x = Laya.stage.width / 2;

startInterface.y = Laya.stage.height / 2;

startInterface['startGame'].aniTypeInit(type);

}

createLaunchAni(): void {

this.launchTemp\_01 = new Laya.Templet();

this.launchTemp\_01.on(Laya.Event.COMPLETE, this, this.parseComplete\_01);

this.launchTemp\_01.on(Laya.Event.ERROR, this, this.onError);

this.launchTemp\_01.loadAni("candy/糖果机器/candyLaunch.sk");

this.launchTemp\_02 = new Laya.Templet();

this.launchTemp\_02.on(Laya.Event.COMPLETE, this, this.parseComplete\_02);

this.launchTemp\_02.on(Laya.Event.ERROR, this, this.onError);

this.launchTemp\_02.loadAni("candy/糖果机器/candyLaunch.sk");

}

onError(): void {

}

parseComplete\_01(): void {

this.candyLaunch\_01 = this.assembly.getChildByName('candyLaunch\_01') as Laya.Skeleton;

this.candyLaunch\_01.play('static', false);

this.candyLaunch\_01.on(Laya.Event.LABEL, this, this.candyLaunchListen\_01);

}

parseComplete\_02(): void {

this.candyLaunch\_02 = this.assembly.getChildByName('candyLaunch\_02') as Laya.Skeleton;

this.candyLaunch\_02.play('static', false);

this.candyLaunch\_02.on(Laya.Event.LABEL, this, this.candyLaunchListen\_02);

}

candyLaunchListen\_01(e): void {

if (e.name === 'launch') {

console.log('发射！');

} else if (e.name === 'getReady') {

this.candyMoveToDisplay();

this.timeSchedule.value = 1;

}

}

candyLaunchListen\_02(e): void {

if (e.name === 'launch') {

} else if (e.name === 'getReady') {

}

}

candyMoveToDisplay(): void {

let delayed = 10;

let candyHeiht = 100;

let spacing = -1;

let startX\_01 = Laya.stage.width / 2 + 60;

let startX\_02 = Laya.stage.width / 2 - 50;

let displaysPar = this.displays.parent as Laya.Sprite;

let worldDevY = displaysPar.y - displaysPar.pivotY;

let startY = this.displays.y + worldDevY + 4 \* (candyHeiht + spacing) - 35;

for (let i = 0; i < this.startRow; i++) {

Laya.timer.frameOnce(delayed, this, function () {

for (let j = 0; j < 2; j++) {

let candy = this.createCandy();

candy['Candy'].group = i;

candy.zOrder = this.startRow - i;

let BirthY = this.candyLaunch\_01.y + worldDevY - 20;

if (j === 0) {

candy.pos(this.candyLaunch\_02.x, BirthY);

candy.scaleX = 0;

candy.scaleY = 0;

this.candyLaunch\_01.play('launchLeft', false);

let targetY = startY - i \* (candyHeiht + spacing);

this.candyFlipTheAni(i, j, candy, startX\_01, targetY);

} else {

candy.pos(this.candyLaunch\_01.x, BirthY);

candy.scaleX = 0.5;

candy.scaleY = 0.5;

this.candyLaunch\_02.play('launchRight', false);

let targetY = startY - i \* (candyHeiht + spacing);

this.candyFlipTheAni(i, j, candy, startX\_02, targetY);

}

}

})

delayed += 15;

}

}

candyFlipTheAni(i, j, candy, targetX, targetY): void {

let timePar = 500 - candy['Candy'].group \* 100;

Laya.Tween.to(candy, { scaleX: 0.8, scaleY: 0.8, y: candy.y - 30 }, timePar / 2, null, Laya.Handler.create(this, function () {

let HalfX;

let distancePer = 3;

if (candy.x > Laya.stage.width / 2) {

HalfX = candy.x - (candy.x - targetX) / distancePer;

} else {

HalfX = candy.x + (targetX - candy.x) / distancePer;

}

let HalfY = candy.y + (targetY - candy.y) / distancePer;

Laya.Tween.to(candy, { x: HalfX, y: HalfY, scaleX: 1.3, scaleY: 1.3 }, timePar \* 3 / 4, null, Laya.Handler.create(this, function () {

Laya.Tween.to(candy, { x: targetX, y: targetY, scaleX: 1, scaleY: 1 }, timePar, null, Laya.Handler.create(this, function () {

this.explodeAni(this.owner, candy.x, candy.y, 'disappear', 8, 1000);

candy['Candy'].playSkeletonAni(1, 'bonbonniere');

candy['Candy'].clicksLabel.alpha = 1;

if (i === 3 && j === 1) {

this.operating['OperationControl'].operateSwitch = true;

this.operating['OperationControl'].initHint();

}

}), 0);

}), 0);

}), 10);

let shadow = candy.getChildByName('shadow') as Laya.Image;

Laya.Tween.to(shadow, {}, timePar / 2, null, Laya.Handler.create(this, function () {

Laya.Tween.to(shadow, { x: -20 + 52, y: 100 + 60, scaleX: 0.5, scaleY: 0.5, }, timePar \* 3 / 4, null, Laya.Handler.create(this, function () {

Laya.Tween.to(shadow, { x: 0 + 52, y: 0 + 60, scaleX: 1, scaleY: 1 }, timePar, null, Laya.Handler.create(this, function () {

}), 0);

}), 0);

}), 10);

}

createCandy(): Laya.Sprite {

let candy = Laya.Pool.getItemByCreateFun('candy', this.candy.create, this.candy) as Laya.Sprite;

let randomNum = Math.floor(Math.random() \* 4);

switch (randomNum) {

case 0:

candy.name = 'yellowCandy' + this.candyCount;

break;

case 1:

candy.name = 'redCandy\_\_\_' + this.candyCount;

break;

case 2:

candy.name = 'blueCandy\_\_' + this.candyCount;

break;

case 3:

candy.name = 'greenCandy\_' + this.candyCount;

break;

default:

break;

}

candy.pos(0, 0);

candy.pivotX = candy.width / 2;

candy.pivotY = candy.height / 2;

this.candyParent.addChild(candy);

candy.rotation = 0;

this.candyCount++;

return candy;

}

roleAppear(): void {

Laya.Tween.to(this.role\_01, { x: 139 }, 1000, null, Laya.Handler.create(this, function () {

this.role\_01['Role'].skeleton.play('speak', true);

}, []), 0);

Laya.Tween.to(this.role\_02, { x: 669 }, 1000, null, Laya.Handler.create(this, function () {

this.role\_02['Role'].skeleton.play('speak', true);

}, []), 0);

}

roleVanish(): void {

Laya.Tween.to(this.role\_01, { x: -500 }, 1000, null, Laya.Handler.create(this, function () {

this.role\_01['Role'].skeleton.play('frontMove', true);

}, []), 0);

Laya.Tween.to(this.role\_02, { x: 1834 }, 1000, null, Laya.Handler.create(this, function () {

this.role\_02['Role'].skeleton.play('frontMove', true);

}, []), 0);

}

roleSpeakBoxs(): void {

for (let i = 0; i < 2; i++) {

let speakBox = Laya.Pool.getItemByCreateFun('speakBox', this.speakBox.create, this.speakBox) as Laya.Sprite;

this.speakBoxParent.addChild(speakBox);

if (i === 0) {

speakBox.pos(this.role\_01.x, this.role\_01.y);

this.role\_01speak = speakBox;

this.role\_01speak.alpha = 0;

let pic = this.role\_01speak.getChildByName('pic') as Laya.Sprite;

let label = this.role\_01speak.getChildByName('label') as Laya.Sprite;

pic.scaleX = -1;

label.x += 30;

} else {

speakBox.pos(this.role\_02.x, this.role\_02.y);

this.role\_02speak = speakBox;

this.role\_02speak.alpha = 0;

}

}

}

roleDeathState(): void {

let len = this.roleParent.\_children.length;

if (len === 0) {

this.enemySwitch\_01 = false;

this.enemySwitch\_02 = false;

return;

} else if (len === 1) {

let speak\_01 = this.role\_01speak.getChildByName('label') as Laya.Label;

let speak\_02 = this.role\_02speak.getChildByName('label') as Laya.Label;

if (this.rescueNum >= 5) {

this.rescueNum = 0;

if (this.roleParent.\_children[0].name === "role\_01") {

this.roleParent.addChild(this.role\_02);

this.role\_02speak.x -= 150;

speak\_02.text = '';

} else {

this.roleParent.addChild(this.role\_01);

speak\_01.text = '';

this.role\_01speak.x += 150;

}

} else {

if (this.roleParent.\_children[0].name === "role\_01") {

this.role\_02speak.alpha = 1;

this.role\_02speak.x = this.role\_02.x;

} else {

this.role\_01speak.alpha = 1;

this.role\_01speak.x = this.role\_01.x;

}

} else if (len === 2) {

this.rescueNum === 0;

}

}

createEnemyHint(): void {

let enemyHint = Laya.Pool.getItemByCreateFun('enemyHint', this.enemyHint.create, this.enemyHint) as Laya.Sprite;

this.owner.addChild(enemyHint);

enemyHint.zOrder = 1000;

}

createEnemy(mode: string, tagRole: Laya.Sprite, type: string): Laya.Sprite {

this.enemyCount++;

if (tagRole !== null) {

let enemy = Laya.Pool.getItemByCreateFun('enemy', this.enemy.create, this.enemy) as Laya.Sprite;

this.enemyParent.addChild(enemy);

enemy.name = 'enemy' + this.enemyCount;

enemy.pivotX = enemy.width / 2;

enemy.pivotY = enemy.height / 2;

let y = Laya.stage.height \* 0.23;

if (mode === 'left') {

enemy.pos(-50, y);

} else if (mode === 'right') {

enemy.pos(800, y);

} else if (mode === 'target') {

if (tagRole.x < Laya.stage.width / 2 && tagRole.x > 0) {

enemy.pos(-50, y);

} else if (tagRole.x >= Laya.stage.width / 2 && tagRole.x < Laya.stage.width) {

enemy.pos(800, y);

}

}

enemy['Enemy'].slefTagRole = tagRole;

enemy['Enemy'].enemyType = type;

enemy['Enemy'].randomAttackPoint();

enemy['Enemy'].createBoneAni();

let propertyShow = enemy.getChildByName('propertyShow') as Laya.Sprite;

if (!this.suspend) {

propertyShow.alpha = 0;

} else {

propertyShow.alpha = 1;

}

return enemy;

}

}

enemyOrder(): void {

for (let i = 0; i < this.enemyParent.\_children.length; i++) {

this.enemyParent.\_children[i].zOrder = Math.round(this.enemyParent.\_children[i].y);

}

}

enemyPropertyUpdate(): void {

if (this.timerControl % 600 === 0) {

this.enemyProperty.blood += 25;

this.enemyProperty.attackValue += 1;

this.enemyProperty.attackSpeed += 5;

if (this.enemyProperty.attackSpeed < 100) {

this.enemyProperty.attackSpeed = 100;

}

this.enemyProperty.defense += 1;

this.enemyProperty.creatInterval += 25;

if (this.enemyProperty.creatInterval < 500) {

this.enemyProperty.creatInterval = 500;

}

}

}

explodeAni(parent, x, y, type, number, zOrder): void {

for (let i = 0; i < number; i++) {

let explode = Laya.Pool.getItemByCreateFun('explode', this.explode.create, this.explode) as Laya.Sprite;

parent.addChild(explode);

explode.zOrder = zOrder;

explode.pos(x, y);

explode['Explode'].type = type;

explode['Explode'].initProperty(type);

}

}

createResurgence(): void {

this.assembly['Assembly'].pipeAnimation('static');

let resurgence = Laya.Pool.getItemByCreateFun('resurgence', this.resurgence.create, this.resurgence) as Laya.Sprite;

this.self.addChild(resurgence);

resurgence.zOrder = 1000;

resurgence.pos(0, 0);

}

createHintWord(target, x, y, proPertyType, numberValue, scale): void {

let hintWord = Laya.Pool.getItemByCreateFun('hintWord', this.hintWord.create, this.hintWord) as Laya.Sprite;

target.addChild(hintWord);

hintWord.pos(x, y);

hintWord.scale(scale, scale);

hintWord.zOrder = 300;

hintWord['HintWord'].initProperty(proPertyType, numberValue);

}

restart(): void {

let enemyDelayed = 0;

let len1 = this.enemyParent.\_children.length;

for (let i = 0; i < this.enemyParent.\_children.length; i++) {

Laya.timer.frameOnce(enemyDelayed, this, function () {

if (!this.enemyParent.\_children[i]) {

return;

}

this.enemyParent.\_children[i].alpha = 0;

let x = this.enemyParent.\_children[i].x;

let y = this.enemyParent.\_children[i].y;

if (this.enemyParent.\_children[i]['Enemy'] === 'infighting') {

this.explodeAni(this.owner, x, y, 'infighting', 15, 100);

} else {

this.explodeAni(this.owner, x, y, 'range', 15, 100);

}

if (i === len1 - 1) {

this.enemyParent.removeChildren(0, len1 - 1);

}

});

enemyDelayed += 20;

}

let candyDelayed = 0;

let len3 = this.candyParent.\_children.length;

if (len3 === 0) {

this.roleResurgenceAni();

this.candyParent.removeChildren(0, len3 - 1);

return;

}

for (let k = 0; k < len3; k++) {

Laya.timer.frameOnce(candyDelayed, this, function () {

if (!this.candyParent.\_children[k]) {

return;

}

this.candyParent.\_children[k].alpha = 0;

let name = this.candyParent.\_children[k].name.substring(0, 11);

let x = this.candyParent.\_children[k].x;

let y = this.candyParent.\_children[k].y;

this.explodeAni(this.owner, x, y, 'disappear', 8, 1000);

if (k === len3 - 1) {

this.roleResurgenceAni();

this.candyParent.removeChildren(0, len3 - 1);

}

});

candyDelayed += 20;

}

}

roleResurgenceAni(): void {

let skeleton1 = this.role\_01.getChildByName('skeleton') as Laya.Skeleton;

Laya.Tween.to(this.role\_01, { alpha: 1 }, 700, null, Laya.Handler.create(this, function () {

this.restartProperties();

}, []), 0);

let skeleton2 = this.role\_02.getChildByName('skeleton') as Laya.Skeleton;

Laya.Tween.to(this.role\_02, { alpha: 1 }, 700, null, Laya.Handler.create(this, function () {

}, []), 0);

}

restartProperties(): void {

this.startGame();

this.role\_01['Role'].role\_Warning = false;

this.role\_01['Role'].roleDeath = false;

this.role\_01['Role'].initProperty();

this.role\_02['Role'].role\_Warning = false;

this.role\_02['Role'].roleDeath = false;

this.role\_02['Role'].initProperty();

this.operating['OperationControl'].initProperty();

}

returnStartSet(): void {

this.scoreLabel.value = '0';

this.noStarted();

let len1 = this.enemyParent.\_children.length;

this.enemyParent.removeChildren(0, len1 - 1);

let len2 = this.candy\_ExplodeParent.\_children.length;

this.candy\_ExplodeParent.removeChildren(0, len2 - 1);

let len3 = this.candyParent.\_children.length;

this.candyParent.removeChildren(0, len3 - 1);

this.role\_01.alpha = 1;

this.role\_01['Role'].role\_Warning = false;

this.role\_01['Role'].roleDeath = false;

this.role\_01['Role'].initProperty();

this.role\_02.alpha = 1;

this.role\_02['Role'].role\_Warning = false;

this.role\_02['Role'].roleDeath = false;

this.role\_02['Role'].initProperty();

this.operating['OperationControl'].initProperty();

}

wxPostInit() {

if (Laya.Browser.onMiniGame) {

Laya.loader.load(["res/atlas/rank.atlas"], Laya.Handler.create(null, function () {

Laya.MiniAdpter.sendAtlasToOpenDataContext("res/atlas/rank.atlas");

let wx: any = Laya.Browser.window.wx;

let openDataContext: any = wx.getOpenDataContext();

openDataContext.postMessage({ action: 'init' });

}));

}

}

wxPostData() {

if (Laya.Browser.onMiniGame) {

let args = {

type: 'scores', data: { scores: this.scoreLabel.value }

}

let wx: any = Laya.Browser.window.wx;

let openDataContext: any = wx.getOpenDataContext();

openDataContext.postMessage(args);

} else {

}

}

wxShare() {

if (Laya.Browser.onMiniGame) {

let wx: any = Laya.Browser.window.wx;

wx.shareAppMessage({

});

} else {

}

}

addScores(number: number): void {

this.scoreLabel.value = (Number(this.scoreLabel.value) + number).toString();

}

onUpdate(): void {

if (this.gameOver) {

return;

}

if (this.role\_01['Role'].roleDeath && this.role\_02['Role'].roleDeath) {

this.gameOver = true;

this.createResurgence();

return;

}

this.enemyOrder();

this.timerControl += 1;

this.enemyPropertyUpdate();

this.roleDeathState();

if (this.enemySwitch\_01) {

let nowTime = Date.now();

if (nowTime - this.enemyTime\_01 > this.enemyProperty.creatInterval) {

this.enemyTime\_01 = nowTime;

this.enemyTagRole = this.role\_01;

this.createEnemy('left', this.role\_01, 'fighting');

this.createEnemy('left', this.role\_01, 'range');

this.enemyTagRole = null;

}

}

if (this.enemySwitch\_02) {

let nowTime = Date.now();

if (nowTime - this.enemyTime\_02 > this.enemyProperty.creatInterval) {

this.enemyTime\_02 = nowTime;

this.enemyTagRole = this.role\_02;

this.createEnemy('right', this.role\_02, 'fighting');

this.createEnemy('right', this.role\_02, 'range');

this.enemyTagRole = null;

}

}

}

onDisable(): void {

}

}

export default class Assembly extends Laya.Script {

private lamp\_01: Laya.Sprite;

private lamp\_02: Laya.Sprite;

private lamp\_03: Laya.Sprite;

public energyLamp\_02: Laya.Sprite;

private lamp\_04: Laya.Sprite;

private lamp\_05: Laya.Sprite;

private lamp\_06: Laya.Sprite;

private LampSwitch: boolean;

private lampTime: number;

private lampInterval: number;

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private smokeSwitch: boolean;

private smokeInterval: number;

private smokeTime: number;

private mshakeInterval: number;

private mshakeTime: number;

private mshakeSwitch: boolean;

private mshakesTre: number;

private mDirection: string;

private rshakeInterval: number;

private rshakeTime: number;

private rshakeSwitch: boolean;

private rDirection: string;

private initialPX\_Machine: number;

private timer: Laya.Sprite;

private timeSchedule: Laya.ProgressBar;

private pipeSk\_01: Laya.Skeleton;

private pipeSk\_01Tem: Laya.Templet;

private pipeSk\_02: Laya.Skeleton;

private pipeSk\_02Tem: Laya.Templet;

private pipeTime: number;

private pipeIntervel: number;

private pipeSwitch: boolean;

constructor() { super(); }

onEnable(): void {

this.initProperty();

}

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.selfScene = this.self.scene;

this.smokeSwitch = true;

this.smokeTime = Date.now();

this.smokeInterval = 500;

this.initialPX\_Machine = this.machine.x;

this.lamp\_01 = this.energyLamp\_01.getChildByName('lamp\_01') as Laya.Sprite;

this.lamp\_02 = this.energyLamp\_01.getChildByName('lamp\_02') as Laya.Sprite;

this.lamp\_03 = this.energyLamp\_01.getChildByName('lamp\_03') as Laya.Sprite;

this.lamp\_01.alpha = 0.3;

this.lamp\_02.alpha = 0.3;

this.lamp\_04 = this.energyLamp\_02.getChildByName('lamp\_04') as Laya.Sprite;

this.lamp\_05 = this.energyLamp\_02.getChildByName('lamp\_05') as Laya.Sprite;

this.lamp\_06 = this.energyLamp\_02.getChildByName('lamp\_06') as Laya.Sprite;

this.lamp\_04.alpha = 0.3;

this.lamp\_05.alpha = 0.3;

this.lamp\_06.alpha = 0.3;

this.LampSwitch = true;

this.lampTime = Date.now();

this.lampAni\_02();

this.mDirection = Math.random() \* 2 === 1 ? 'left' : 'right';

this.mshakeInterval = 30;

this.mshakeTime = Date.now();

this.mshakesTre = 1;

this.mshakeSwitch = true;

this.rshakeInterval = 30;

this.rshakeTime = Date.now();

this.rshakesTre = 2;

this.rDirection = Math.random() \* 2 === 1 ? 'left' : 'right';

this.timer = this.owner.getChildByName('timer') as Laya.Sprite;

this.timeSchedule = this.timer.getChildByName('timeSchedule') as Laya.ProgressBar;

this.timerShakeNum = 0;

this.pipeSk\_01 = this.machine.getChildByName('pipeline\_01') as Laya.Skeleton;

this.pipeSk\_02 = this.machine.getChildByName('pipeline\_02') as Laya.Skeleton;

this.createPipeSk();

this.pipeSwitch = false;

this.pipeTime = Date.now();

this.pipeIntervel = 2500;

}

createPipeSk(): void {

this.pipeSk\_01Tem = new Laya.Templet();

this.pipeSk\_01Tem.on(Laya.Event.COMPLETE, this, this.parseComplete\_01);

this.pipeSk\_01Tem.on(Laya.Event.ERROR, this, this.onError);

this.pipeSk\_01Tem.loadAni("candy//pipeline\_01.sk");

this.pipeSk\_02Tem = new Laya.Templet();

this.pipeSk\_02Tem.on(Laya.Event.COMPLETE, this, this.parseComplete\_02);

this.pipeSk\_02Tem.on(Laya.Event.ERROR, this, this.onError);

this.pipeSk\_02Tem.loadAni("candy//pipeline\_01.sk");

}

onError(): void {

}

parseComplete\_01(): void {

this.pipeSk\_01.play('static', true);

}

parseComplete\_02(): void {

this.pipeSk\_02.play('static', true);

}

pipeAnimation(type): void {

if (type === 'static') {

this.pipeSk\_01.play('static', true);

this.pipeSk\_02.play('static', true);

} else if (type === 'flow') {

this.pipeSk\_01.play('flow', true);

this.pipeSk\_02.play('flow', true);

}

}

lampAni\_01(): void {

Laya.Tween.to(this.lamp\_01, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_01, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.lamp\_02, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_02, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.lamp\_03, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_03, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

}, []), 0);

}, []), 0);

}, []), 0);

}

lampAni\_02(): void {

Laya.Tween.to(this.lamp\_04, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_04, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.lamp\_05, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_05, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.lamp\_06, { alpha: 1 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

Laya.Tween.to(this.lamp\_06, { alpha: 0.3 }, 500, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

}, []), 0);

}, []), 0);

}, []), 0);

}

moveShake(target) {

if (this.mshakeSwitch) {

let nowTime = Date.now();

if (nowTime - this.mshakeTime > this.mshakeInterval) {

this.mshakeTime = nowTime;

let initialPX; if (target === this.machine) {

initialPX = this.initialPX\_Machine;

}

let shakeX = this.mshakesTre; if (this.mDirection === "left") {

target.x -= this.mshakesTre;

if (this.machine.x < initialPX) {

this.mDirection = "right";

}

} else if (this.mDirection === "right") {

target.x += this.mshakesTre;

if (this.machine.x > initialPX) {

this.mDirection = "left";

}

}

}

}

}

timerShake() {

if (this.timeSchedule.value > 0 && this.timeSchedule.value <= 0.15) {

this.rshakeInterval = 40;

this.rshakesTre = 2;

} else if (this.timeSchedule.value > 0.15 && this.timeSchedule.value <= 0.4) {

this.rshakeInterval = 50;

this.rshakesTre = 1.5;

} else if (this.timeSchedule.value > 0.4 && this.timeSchedule.value <= 0.7) {

this.rshakeInterval = 60;

this.rshakesTre = 1;

} else if (this.timeSchedule.value > 0.7 && this.timeSchedule.value <= 1) {

this.rshakeInterval = 70;

this.rshakesTre = 0.5;

} else {

this.rshakeInterval = 70;

this.rshakesTre = 0.5;

}

if (this.rshakeSwitch) {

let nowTime = Date.now();

if (nowTime - this.rshakeTime > this.rshakeInterval) {

this.rshakeTime = nowTime;

if (this.rDirection === "left") {

this.timer.rotation = -this.rshakesTre;

if (this.timer.rotation < 0) {

this.rDirection = "right";

}

} else if (this.rDirection === "right") {

this.timer.rotation = this.rshakesTre;

if (this.timer.rotation > 0) {

this.rDirection = "left";

}

}

}

}

}

onUpdate(): void {

if (this.selfScene['MainSceneControl'].gameOver) {

return;

}

if (this.smokeSwitch) {

let nowTime = Date.now();

if (nowTime - this.smokeTime > this.smokeInterval) {

this.smokeTime = nowTime;

let random = Math.floor(Math.random() \* 300) + 100;

this.smokeInterval = 600 - random;

this.selfScene['MainSceneControl'].explodeAni(this.machine, 650, 190, 'smokeEffects', 1, 10);

}

}

this.LongPointer.rotation += 10;

this.moveShake(this.machine);

this.timerShake();

if (this.LampSwitch) {

let time = Date.now();

if (time - this.lampTime > this.lampInterval) {

this.lampTime = time;

this.lampAni\_01();

this.lampAni\_02();

}

}

if (this.pipeSwitch) {

let time = Date.now();

if (time - this.pipeTime > this.pipeIntervel) {

this.pipeTime = time;

this.pipeSk\_01.play('flow', false);

this.pipeSk\_02.play('flow', false);

}

}

}

onDisable(): void {

}

}

import MainSceneControl from "./MainSceneControl";

export default class Candy extends Laya.Script {

private self: Laya.Sprite;

private clicksLabel: Laya.FontClip;

private shadow: Laya.Image;

private selfScene: Laya.Scene;

private mainSceneControl;

private selfSpeed: number;

private candyTagRole: Laya.Sprite;

private roleParent: Laya.Sprite;

private scoreLabel: Laya.FontClip;

private timerControl: number;

private hintWord: Laya.Prefab;

private group: any;

private templet: Laya.Templet;

private skeleton: Laya.Skeleton;

constructor() { super(); }

onEnable(): void {

this.initProperty();

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.selfScene = this.self.scene as Laya.Scene;

this.candyTagRole = null;

this.selfSpeed = 10;

this.spaceY = 5;

this.shadow = this.self.getChildByName('shadow') as Laya.Image;

this.hintWord = this.selfScene['MainSceneControl'].hintWord;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

this.group = null;

this.skeleton = this.self.getChildByName('skeleton') as Laya.Skeleton;

this.self['Candy'] = this;

this.randomCandyClicks(3, 1);

this.createBoneAni();

}

randomCandyClicks(randomNum, baseNum): void {

let clicksNum = Math.floor(Math.random() \* randomNum) + baseNum;

this.clicksLabel = this.self.getChildByName('clicksLabel') as Laya.FontClip;

this.clicksLabel.value = clicksNum.toString();

}

createBoneAni(): void {

this.templet = new Laya.Templet();

this.templet.on(Laya.Event.COMPLETE, this, this.parseComplete);

this.templet.on(Laya.Event.ERROR, this, this.onError);

this.templet.loadAni("candy/糖果/candyCompilations.sk");

}

onError(): void {

}

parseComplete(): void {

this.playSkeletonAni(2, 'bonbonniere');

}

playSkeletonAni(speed: number, type: string): void {

if (type === 'bonbonniere') {

this.shadow.skin = 'candy//.png';

} else {

this.shadow.skin = 'candy//.png';

}

switch (this.self.name.substring(0, 11)) {

case 'yellowCandy':

this.skeleton.play('yellow\_' + type, true);

break;

case 'redCandy\_\_\_':

this.skeleton.play('red\_' + type, true);

break;

case 'blueCandy\_\_':

this.skeleton.play('blue\_' + type, true);

break;

case 'greenCandy\_':

this.skeleton.play('green\_' + type, true);

break;

default:

break;

}

this.skeleton.x = this.self.width / 2 + 3;

this.skeleton.y = this.self.height / 2;

this.skeleton.playbackRate(speed);

}

candyFlyToRole(): void {

if (this.self.x < Laya.stage.width / 2) {

this.candyTagRole = this.selfScene['MainSceneControl'].role\_01;

} else {

this.candyTagRole = this.selfScene['MainSceneControl'].role\_02;

}

this.playSkeletonAni(2, 'turnDown');

let targetX = this.candyTagRole.x - 50;

let targetY = this.candyTagRole.y;

let HalfX;

let distancePer = 2;

if (this.self.x > Laya.stage.width / 2) {

HalfX = this.self.x + (this.candyTagRole.x - this.self.x) \* 3 / 5;

} else {

HalfX = this.self.x - (this.self.x - this.candyTagRole.x) \* 3 / 5;

}

let HalfY = this.self.y + (this.candyTagRole.y - this.self.y) / distancePer;

Laya.Tween.to(this.self, { x: HalfX, y: HalfY, scaleX: 1.5, scaleY: 1.5 }, timePar \* 3 / 4, null, Laya.Handler.create(this, function () {

Laya.Tween.to(this.self, { x: targetX, y: this.candyTagRole.y, scaleX: 0.6, scaleY: 0.6 }, timePar / 2, null, Laya.Handler.create(this, function () {

this.hintWordMove();

this.roleAddProperty();

this.self.removeSelf();

this.selfScene['MainSceneControl'].addScores(100);

}), 0);

}), 0);

Laya.Tween.to(this.shadow, { x: - 20 + 52, y: 100 + 60, scaleX: 0.8, scaleY: 0.8, }, timePar \* 3 / 4, null, Laya.Handler.create(this, function () {

Laya.Tween.to(this.shadow, { x: -10 + 52, y: 60, scaleX: 0.7, scaleY: 0.7 }, timePar, null, Laya.Handler.create(this, function () {

}), 0);

}), 0);

}

hintWordMove(): void {

let MainSceneControl = this.selfScene['MainSceneControl'];

if (this.self.x < Laya.stage.width / 2) {

this.candyTagRole = MainSceneControl.role\_01;

} else {

this.candyTagRole = MainSceneControl.role\_02;

}

let x = 80;

let y = -20;

switch (this.self.name.substring(0, 11)) {

case 'yellowCandy':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 10, 1);

break;

case 'redCandy\_\_\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 5, 1);

break;

case 'blueCandy\_\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 10, 1);

break;

case 'greenCandy\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 5, 1);

break;

default:

}

}

roleAddProperty(): void {

this.self.name = this.self.name.substring(0, 11);

switch (this.self.name) {

case 'yellowCandy':

this.candyTagRole['Role'].role\_property.attackValue += 10;

break;

case 'redCandy\_\_\_':

this.candyTagRole['Role'].role\_property.blood += 5;

break;

case 'blueCandy\_\_':

this.candyTagRole['Role'].role\_property.attackSpeed += 10;

break;

case 'greenCandy\_':

this.candyTagRole['Role'].role\_property.defense += 5;

break;

default:

break;

}

}

asExplodeCandy(): void {

if (this.self.x < Laya.stage.width / 2) {

this.candyTagRole = this.selfScene['MainSceneControl'].role\_01;

} else {

this.candyTagRole = this.selfScene['MainSceneControl'].role\_02;

}

Laya.Tween.to(this.self, { x: this.candyTagRole.x, y: this.candyTagRole.y }, 800, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, this.self.x, this.self.y, this.self.name.substring(0, 11), 15, 100);

this.propertyHintWord();

this.roleReduceProperty();

this.self.removeSelf();

}), 0);

propertyHintWord(): void {

let MainSceneControl = this.selfScene['MainSceneControl'];

let x = 80;

let y = -20;

switch (this.self.name.substring(0, 11)) {

case 'yellowCandy':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 10, 1);

break;

case 'redCandy\_\_\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 5, 1);

break;

case 'blueCandy\_\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 10, 1);

break;

case 'greenCandy\_':

MainSceneControl.createHintWord(this.candyTagRole, x, y, '', 5, 1);

break;

default:

}

}

roleReduceProperty(): void {

this.self.name = this.self.name.substring(0, 11);

let role\_01 = this.selfScene['MainSceneControl'].role\_01;

let role\_02 = this.selfScene['MainSceneControl'].role\_02;

switch (this.self.name) {

case 'yellowCandy':

if (this.candyTagRole === role\_01) {

role\_01['Role'].role\_property.attackValue -= 10;

} else {

role\_02['Role'].role\_property.attackValue -= 10;

}

break;

case 'redCandy\_\_\_':

if (this.candyTagRole === role\_01) {

role\_01['Role'].role\_property.blood -= 5;

} else {

role\_02['Role'].role\_property.blood -= 5;

}

break;

case 'blueCandy\_\_':

if (this.candyTagRole === role\_01) {

role\_01['Role'].role\_property.attackSpeed -= 10;

} else {

role\_02['Role'].role\_property.attackSpeed -= 10;

}

break;

case 'greenCandy\_':

if (this.candyTagRole === role\_01) {

role\_01['Role'].role\_property.defense -= 5;

} else {

role\_02['Role'].role\_property.defense -= 5;

}

break;

default:

break;

}

}

onUpdate(): void {

if (this.candyTagRole !== null && this.candyTagRole['Role'].roleDeath) {

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, this.self.x, this.self.y, this.self.name.substring(0, 11), 15, 100);

this.self.removeSelf();

this.candyFlyToRole = null;

}

}

onDisable(): void {

if (this.skeleton) {

this.skeleton.removeSelf();

}

Laya.Tween.clearAll(this);

if (this.self.name === 'yellowCandy') {

Laya.Pool.recover('yellowCandy', this.self);

} else if (this.self.name === 'redCandy\_\_\_') {

Laya.Pool.recover('redCandy\_\_\_', this.self);

} else if (this.self.name === 'blueCandy\_\_') {

Laya.Pool.recover('blueCandy\_\_', this.self);

} else if (this.self.name === 'greenCandy\_') {

Laya.Pool.recover('greenCandy\_', this.self);

}

}

}

export default class Candy extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private selfHealth: Laya.ProgressBar;

private selfSpeed: number;

private pic: Laya.Sprite;

private explodeTarget: Laya.Sprite;

private roleParent: Laya.Sprite;

private scoreLabel: Laya.FontClip;

private timerControl: number;

private spaceY: number;

private posYArr: Array<number>;

private hintWord: Laya.Prefab;

private explode: Laya.Prefab;

private templet: Laya.Templet;

private skeleton: Laya.Skeleton;

constructor() { super(); }

onEnable(): void {

this.initProperty();

}

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.pic = this.self.getChildByName('pic') as Laya.Sprite;

this.selfScene = this.self.scene as Laya.Scene;

this.explodeTarget = null;

this.roleParent = this.selfScene['MainSceneControl'].roleParent;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

this.selfSpeed = 10;

this.selfHealth = this.self.getChildByName('health') as Laya.ProgressBar;

this.timerControl = 0;

this.explode = this.selfScene['MainSceneControl'].explode;

this.spaceY = 5;

this.hintWord = this.selfScene['MainSceneControl'].hintWord;

this.selfScene['MainSceneControl'].role\_01['Role'].role\_Warning = true;

this.selfScene['MainSceneControl'].role\_02['Role'].role\_Warning = true;

this.self['Candy\_Explode'] = this;

this.skeleton = this.self.getChildByName('skeleton') as Laya.Skeleton;

this.createBoneAni();

}

createBoneAni(): void {

this.templet = new Laya.Templet();

this.templet.on(Laya.Event.COMPLETE, this, this.parseComplete);

this.templet.on(Laya.Event.ERROR, this, this.onError);

this.templet.loadAni("candy/糖果/candyCompilations.sk");

}

onError(): void {

}

parseComplete(): void {

var skeleton: Laya.Skeleton;

switch (this.self.name.substring(0, 11)) {

case 'yellowCandy':

this.skeleton.play('yellow\_explode', true);

break;

case 'redCandy\_\_\_':

this.skeleton.play('red\_explode', true);

break;

case 'blueCandy\_\_':

this.skeleton.play('blue\_explode', true);

break;

case 'greenCandy\_':

this.skeleton.play('green\_explode', true);

break;

default:

break;

}

this.skeleton.playbackRate(1);

}

flyToRole(): void {

if (this.selfScene['MainSceneControl'].gameOver) {

return;

}

if (this.explodeTarget !== null) {

let point = new Laya.Point(this.explodeTarget.x - this.self.x, this.explodeTarget.y - this.self.y);

point.normalize();

this.self.x += point.x \* this.selfSpeed;

this.self.y += point.y \* this.selfSpeed;

let differenceX = Math.abs(this.self.x - this.explodeTarget.x);

let differenceY = Math.abs(this.self.y - this.explodeTarget.y);

if (differenceX < 50 && differenceY < 50) {

this.self.removeSelf();

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, this.self.x, this.self.y, this.self.name.substring(0, 11), 15, 100);

this.propertyHintWord();

this.roleReduceProperty();

this.selfScene['MainSceneControl'].role\_01['Role'].role\_Warning = true;

this.selfScene['MainSceneControl'].role\_02['Role'].role\_Warning = true;

this.explodeTarget = null;

}

}

}

propertyHintWord(): void {

let MainSceneControl = this.selfScene['MainSceneControl'];

switch (this.self.name.substring(0, 11)) {

case 'yellowCandy':

MainSceneControl.createHintWord(this.explodeTarget, '', 10);

break;

case 'redCandy\_\_\_':

MainSceneControl.createHintWord(this.explodeTarget, '', 5);

break;

case 'blueCandy\_\_':

MainSceneControl.createHintWord(this.explodeTarget, '', 10);

break;

case 'greenCandy\_':

MainSceneControl.createHintWord(this.explodeTarget, '', 5);

break;

default:

}

}

roleReduceProperty(): void {

this.self.name = this.self.name.substring(0, 11);

let role\_01 = this.selfScene['MainSceneControl'].role\_01;

let role\_02 = this.selfScene['MainSceneControl'].role\_02;

switch (this.self.name) {

case 'yellowCandy':

if (this.explodeTarget === role\_01) {

role\_01['Role'].role\_property.attackValue -= 10;

} else {

role\_02['Role'].role\_property.attackValue -= 10;

}

break;

case 'redCandy\_\_\_':

if (this.explodeTarget === role\_01) {

role\_01['Role'].role\_property.blood -= 5;

} else {

role\_02['Role'].role\_property.blood -= 5;

}

break;

case 'blueCandy\_\_':

if (this.explodeTarget === role\_01) {

role\_01['Role'].role\_property.attackSpeed -= 10;

} else {

role\_02['Role'].role\_property.attackSpeed -= 10;

}

break;

case 'greenCandy\_':

if (this.explodeTarget === role\_01) {

role\_01['Role'].role\_property.defense -= 5;

} else {

role\_02['Role'].role\_property.defense -= 5;

}

break;

default:

break;

}

}

onUpdate(): void {

if (this.selfHealth.value <= 0) {

this.self.removeSelf();

this.selfScene['MainSceneControl'].explodeAni(this.self.x, this.self.y, this.self.name.substring(0, 11));

}

}

onDisable(): void {

Laya.Tween.clearAll(this);

if (this.self.name === 'yellowCandy') {

Laya.Pool.recover('yellowCandy', this.self);

} else if (this.self.name === 'redCandy\_\_\_') {

Laya.Pool.recover('redCandy\_\_\_', this.self);

} else if (this.self.name === 'blueCandy\_\_') {

Laya.Pool.recover('blueCandy\_\_', this.self);

} else if (this.self.name === 'greenCandy\_') {

Laya.Pool.recover('greenCandy\_', this.self);

}

}

}

import MainSceneControl from "./MainSceneControl";

import Role from "./Role";

import { tools } from "./Tool";

export default class Enemy extends Laya.Script {

private self: Laya.Sprite;

private selfHealth: Laya.ProgressBar;

private selfSpeed: number;

private pic: Laya.Sprite;

private selfScene: Laya.Scene;

private mainSceneControl;

private roleParent: Laya.Sprite;

private slefTagRole: Laya.Sprite;

private tagHealth: Laya.ProgressBar;

private speakBox: Laya.Prefab;

private attackTnterval: number;

private recordTime: number;

private scoreLabel: Laya.FontClip;

private enemyProperty: any;

private bloodLabel: any;

private propertyShow: Laya.Image;

private repelTimer: number;

private hintWord: Laya.Prefab;

private enemyBullet: Laya.Prefab;

private attackX: number;

private attackY: number;

private templet: Laya.Templet;

private skeleton: Laya.Skeleton;

private role\_01: Laya.Sprite;

private role\_02: Laya.Sprite;

constructor() { super(); }

onEnable(): void {

this.initProperty();

this.enemyPropertySet();

}

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.self.alpha = 1;

this.selfScene = this.owner.scene as Laya.Scene;

this.mainSceneControl = this.selfScene.getComponent(MainSceneControl);

this.selfHealth = this.self.getChildByName('health') as Laya.ProgressBar;

this.selfHealth.value = 1;

this.selfSpeed = 3.5;

this.propertyShow = this.self.getChildByName('propertyShow') as Laya.Image;

this.attackTnterval = 100;

this.speakBox = this.mainSceneControl.speakBox;

this.scoreLabel = this.mainSceneControl.scoreLabel;

this.repelTimer = 0;

this.hintWord = this.mainSceneControl.hintWord as Laya.Prefab;

this.enemyBullet = this.mainSceneControl.enemyBullet as Laya.Prefab;

this.self['Enemy'] = this;

if (this.skeleton) {

this.skeleton.removeSelf();

}

this.role\_01 = this.selfScene['MainSceneControl'].role\_01;

this.role\_02 = this.selfScene['MainSceneControl'].role\_02;

}

createBoneAni(): void {

this.templet = new Laya.Templet();

this.templet.on(Laya.Event.COMPLETE, this, this.parseComplete);

this.templet.on(Laya.Event.ERROR, this, this.onError);

if (this.enemyType === 'fighting') {

this.templet.loadAni("candy/敌人/fightingEnemy.sk");

} else if (this.enemyType === 'range') {

this.templet.loadAni("candy/敌人/rangeEnemy.sk");

}

}

onError(): void {

}

parseComplete(): void {

var skeleton: Laya.Skeleton;

this.skeleton = this.templet.buildArmature(0);

this.skeletonListen();

this.self.addChild(this.skeleton);

this.skeleton.zOrder = -1;

if (this.enemyType === 'fighting') {

this.self.width = 120;

this.self.height = 180;

this.self.pivotX = this.self.width / 2;

this.self.pivotY = this.self.height / 2;

this.skeleton.x = 63;

this.skeleton.y = 108;

this.skeleton.play('move', true);

this.skeleton.playbackRate(1);

} else if (this.enemyType === 'range') {

this.self.width = 120;

this.self.height = 180;

this.self.pivotX = this.self.width / 2;

this.self.pivotY = this.self.height / 2;

this.skeleton.x = 64;

this.skeleton.y = 86;

if (this.self.x < Laya.stage.width / 2) {

this.skeleton.play('moveRight', true);

} else {

this.skeleton.play('moveLeft', true);

}

this.skeleton.playbackRate(1);

}

}

skeletonListen(): void {

this.skeleton.on(Laya.Event.LABEL, this, function (e) {

if (this.enemyType === 'fighting') {

if (e.name === 'hitOut') {

this.enemyAttackRules();

}

} else if ('range') {

if (e.name === 'hitOut') {

this.creatBullet();

}

}

});

}

enemyPropertySet(): void {

this.enemyProperty = {

level: '',

blood: '',

moveSpeed: '',

defense: '',

}

this.enemyProperty.blood = this.mainSceneControl.enemyProperty.blood;

this.enemyProperty.attackValue = this.mainSceneControl.enemyProperty.attackValue;

this.enemyProperty.attackSpeed = this.mainSceneControl.enemyProperty.attackSpeed;

this.enemyProperty.defense = this.mainSceneControl.enemyProperty.defense;

this.enemyProperty.moveSpeed = this.mainSceneControl.enemyProperty.moveSpeed;

}

enemyPropertyUpdate(): void {

this.bloodLabel = this.selfHealth.getChildByName('bloodLabel') as Laya.Label;

let str = Math.round(this.enemyProperty.blood \* this.selfHealth.value).toString();

let subStr\_01 = str.substring(0, str.length - 1);

let subStr\_02 = subStr\_01 + 0;

this.bloodLabel.text = subStr\_02;

let blood = this.propertyShow.getChildByName('blood') as Laya.Label;

blood.text = ": " + this.enemyProperty.blood;

let attackValue = this.propertyShow.getChildByName('attackValue') as Laya.Label;

attackValue.text = ": " + this.enemyProperty.attackValue;

let attackSpeed = this.propertyShow.getChildByName('attackSpeed') as Laya.Label;

attackSpeed.text = ": " + this.enemyProperty.attackSpeed;

let defense = this.propertyShow.getChildByName('defense') as Laya.Label;

defense.text = ": " + this.enemyProperty.defense;

let roleDefense = this.slefTagRole['Role'].role\_property.defense;

if (this.mainSceneControl.enemyParent.\_children.length > 20 && this.enemyProperty.attackValue < roleDefense) {

this.enemyProperty.attackValue = roleDefense + 5;

}

}

randomAttackPoint(): void {

let difference;

let number = Math.floor(Math.random() \* 2);

if (number === 1) {

difference = -Math.floor(Math.random() \* 50);

} else {

difference = Math.floor(Math.random() \* 50);

}

this.attackX = this.slefTagRole.x + difference;

if (this.enemyType === 'fighting') {

this.attackY = this.slefTagRole.y - Math.floor(Math.random() \* 50);

} else if ('range') {

this.attackY = this.slefTagRole.y - Math.floor(Math.random() \* 50) - 450;

}

}

replaceTarget(): void {

if (this.slefTagRole['Role'].roleDeath) {

if (this.slefTagRole.name === 'role\_01') {

this.slefTagRole = this.mainSceneControl.role\_02;

this.tagHealth = this.mainSceneControl.role\_02.getChildByName('health');

} else if (this.slefTagRole.name === 'role\_02') {

this.slefTagRole = this.mainSceneControl.role\_01;

this.tagHealth = this.mainSceneControl.role\_01.getChildByName('health');

}

if (this.enemyType === 'fighting') {

this.randomAttackPoint();

this.skeleton.play('move', true);

}

}

}

fightingEnemyMove(): void {

let point = new Laya.Point(this.attackX - this.self.x, this.attackY - this.self.y);

point.normalize();

if (this.slefTagRole.x > Laya.stage.width / 2) {//右边

this.mainSceneControl.role\_01['Role'].role\_Warning = true;

} else if (this.slefTagRole.x < Laya.stage.width / 2) {//左边

this.mainSceneControl.role\_02['Role'].role\_Warning = true;

}

if (this.repelTimer > 0) {

this.repelTimer--;

this.self.x -= point.x \* 4;

this.self.y -= point.y \* 4;

} else {

this.self.x += point.x \* this.selfSpeed;

this.self.y += point.y \* this.selfSpeed;

}

}

rangeEnemyMove(): void {

let point = new Laya.Point(this.attackX - this.self.x, this.attackY - this.self.y);

point.normalize();

this.self.x += point.x \* this.selfSpeed;

this.self.y += point.y \* this.selfSpeed;

}

enemyAttackRules(): void {

let damage = this.enemyProperty.attackValue - this.slefTagRole['Role'].role\_property.defense;

if (damage > 0) {

this.slefTagRole['Role'].role\_property.blood -= damage;

} else {

damage = 0;

}

this.createHintWord(damage);

}

createHintWord(damage: number): void {

let hintWord = Laya.Pool.getItemByCreateFun('candy', this.hintWord.create, this.hintWord) as Laya.Sprite;

hintWord.pos(0, -150);

this.slefTagRole.addChild(hintWord);

let proPertyType: string = '';

let numberValue: number;

if (!this.slefTagRole.parent) {

return;

}

hintWord['HintWord'].initProperty(proPertyType, damage);

}

creatBullet(): void {

let bullet = Laya.Pool.getItemByCreateFun('enemyBullet', this.enemyBullet.create, this.enemyBullet) as Laya.Sprite;

let bulletParent = this.mainSceneControl.bulletParent;

bulletParent.addChild(bullet);

bullet.pos(this.self.x - 100, this.self.y);

bullet['EnemyBullet'].attackValue = this.enemyProperty.attackValue;

bullet['EnemyBullet'].bulletTarget = this.slefTagRole;

}

playSpeedAdjust(): void {

let playSpeed;

if ((500 - this.enemyProperty.attackSpeed) / 500 > 0) {

playSpeed = 1 + (500 - this.enemyProperty.attackSpeed) / 500;

} else {

playSpeed = 1;

}

this.skeleton.playbackRate(playSpeed);

}

enemyDeath(): void {

this.mainSceneControl.role\_01['Role'].role\_Warning = false;

this.mainSceneControl.role\_02['Role'].role\_Warning = false;

this.self.removeSelf();

if (this.enemyType === 'fighting') {

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, this.self.x, this.self.y, 'fighting', 15, 100);

} else {

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, this.self.x, this.self.y, 'range', 15, 100);

}

this.selfScene['MainSceneControl'].createHintWord(this.selfScene, this.self.x, this.self.y, '杀', 100, 0.5);

}

launchAttack(): void {

let nowTime = Date.now();

if (nowTime - this.recordTime > this.enemyProperty.attackSpeed) {

this.recordTime = nowTime;

if (!this.slefTagRole['Role'].roleDeath) {

this.skeleton.play('attack', false);

this.playSpeedAdjust();

} else {

this.replaceTarget();

}

}

}

onUpdate(): void {

if (this.selfScene['MainSceneControl'].gameOver) {

return;

}

if (this.slefTagRole === null) {

return;

}

if (this.enemyProperty.blood < 0) {

this.enemyDeath();

return;

}

this.enemyPropertyUpdate();

if (this.enemyType === 'range') {

this.rangeEnemyMove();

let differenceX = Math.abs(this.self.x - this.attackX);

let differenceY = Math.abs(this.self.y - this.attackY);

if (differenceX < 10 && differenceY < 10) {

this.role\_01['Role'].role\_Warning = true;

this.role\_02['Role'].role\_Warning = true;

this.selfSpeed = 0;

this.launchAttack();

}

} else if (this.enemyType === 'fighting') {

this.fightingEnemyMove();

let differenceX = Math.abs(this.self.x - this.attackX);

let differenceY = Math.abs(this.self.y - this.attackY);

if (differenceX < 100 && differenceY < 100) {

this.selfSpeed = 0;

this.launchAttack();

} else {

this.selfSpeed = 4;

}

}

}

onDisable(): void {

Laya.Pool.recover('enemy', this.self);

}

}

import MainSceneControl from "./MainSceneControl";

export default class EnemyBullet extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private roleParent: Laya.Sprite;

private role\_01: Laya.Sprite;

private role\_02: Laya.Sprite;

private enemyParent: Laya.Sprite;

private attackValue: number;

private bulletTarget: Laya.Sprite;

private selfSpeed: number;

private belongEnemy: Laya.Sprite;

private hintWord: Laya.Prefab;

constructor() { super(); }

onEnable(): void {

this.init();

}

init() {

this.self = this.owner as Laya.Sprite;

this.selfScene = this.self.scene;

this.selfSpeed = 15;

this.enemyParent = this.selfScene['MainSceneControl'].enemyParent;

this.roleParent = this.selfScene['MainSceneControl'].roleParent;

this.self['EnemyBullet'] = this;

}

bulletMove(): void {

if (this.bulletTarget && this.bulletTarget.parent) {

let point = new Laya.Point(this.bulletTarget.x - this.self.x, this.bulletTarget.y - this.self.y);

point.normalize();

this.self.x += point.x \* this.selfSpeed;

this.self.y += point.y \* this.selfSpeed;

} else {

let point = new Laya.Point(this.self.x - this.belongEnemy.x, this.self.y - this.belongEnemy.y);

point.normalize();

this.self.x += point.x \* this.selfSpeed;

this.self.y += point.y \* this.selfSpeed;

}

}

bulletAttackRules(role): void {

let numberValue: number;

let damage = this.attackValue - role['Role'].role\_property.defense;

if (damage > 0) {

role['Role'].role\_property.blood -= damage;

numberValue = damage;

} else {

numberValue = 0;

}

}

onUpdate(): void {

this.bulletMove();

if (this.self.x > 750 + this.self.width + 50 || this.self.x < -this.self.width) {

this.self.removeSelf();

}

for (let i = 0; i < this.roleParent.\_children.length; i++) {

let role = this.roleParent.\_children[i] as Laya.Sprite;

let differenceX = Math.abs(role.x - this.self.x);

let differenceY = Math.abs(role.y - this.self.y);

if (differenceX < 10 && differenceY < 10) {

this.bulletAttackRules(role);

this.self.removeSelf();

}

}

}

onDisable(): void {

}

}

export default class EnemyHint extends Laya.Script {

public baseboard: Laya.Sprite;

public seconds: Laya.FontClip;

constructor() { super(); }

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private numInterval: number;

private numTime: number;

private numSwitch: boolean;

onEnable(): void {

this.init();

this.appear();

}

init(): void {

this.self = this.owner as Laya.Sprite;

this.self.rotation = 0;

this.self.x = -20;

this.self.scale(0, 0);

this.self.alpha = 0;

this.self.y = Laya.stage.height / 3;

this.selfScene = this.owner.scene as Laya.Scene;

this.numTime = Date.now();

this.numInterval = 1000;

this.numSwitch = false;

}

appear(): void {

Laya.Tween.to(this.self, { alpha: 1, scaleX: 1, scaleY: 1 }, 400, null, Laya.Handler.create(this, function () {

}), 0);

Laya.Tween.to(this.baseboard, { rotation: 720 }, 400, null, Laya.Handler.create(this, function () {

this.baseboard.rotation = 0;

this.seconds.value = '10' + 's';

this.numSwitch = true;

}), 0);

}

vanish(): void {

Laya.Tween.to(this.self, { x: -500 }, 400, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

}), 0);

Laya.Tween.to(this.baseboard, { rotation: -720 }, 400, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

this.baseboard.rotation = 0;

this.seconds.value = '10' + 's';

this.self.removeSelf();

}), 0);

}

onUpdate(): void {

let time = Date.now();

if (this.numSwitch) {

if (time - this.numTime > this.numInterval) {

this.numTime = time;

let value;

if (this.seconds.value.length === 3) {

value = this.seconds.value.substring(0, 2);

} else {

value = this.seconds.value.substring(0, 1);

}

this.seconds.value = (Number(value) - 1).toString() + 's';

if ((Number(value) - 1) < 0) {

this.seconds.value = '0' + 's';

this.selfScene['MainSceneControl'].enemySwitch\_01 = true;

this.selfScene['MainSceneControl'].enemySwitch\_02 = true;

this.numSwitch = false;

this.vanish();

}

}

}

}

onDisable(): void {

Laya.Pool.recover(this.self.name, this.self);

Laya.Tween.clearAll(this);

}

}

import tools from "./Tool";

export default class Explode extends Laya.Script {

private self: Laya.Sprite;

private timer: number;

private initialAngle: number;

private accelerated: number;

private randomSpeed: number;

private moveSwitch: boolean;

private scale: number;

private vinshTime: number;

private startAlpha: number;

private rotationD: number;

private effectsType: string;

private img: Laya.Image;

constructor() { super(); }

onEnable(): void {

this.timer = 0;

this.accelerated = 0.1;

this.self = this.owner as Laya.Sprite;

this.img = this.self.getChildByName('img') as Laya.Image;

this.self['Explode'] = this;

this.self.pivotX = this.self.width / 2;

this.self.pivotY = this.self.height / 2;

}

initProperty(type): void {

this.effectsType = type;

switch (type) {

case 'starShining':

this.starShiningProperty();

break;

case 'disappear':

this.disappearProperty();

break;

case 'appear':

this.appearProperty();

break;

case 'fireworks':

this.fireworksProperty();

break;

case 'smokeEffects':

this.smokeProperty();

break;

case 'fighting':

this.fightingProperty();

break;

case 'range':

this.rangeProperty();

break;

case 'redCandy\_\_\_':

this.redCandyProperty();

break;

case 'greenCandy\_':

this.greenCandyProperty();

break;

case 'blueCandy\_\_':

this.blueCandyProperty();

break;

case 'yellowCandy':

this.yellowCandyProperty();

break;

default:

break;

}

this.img.pivotX = this.img.width / 2;

this.img.pivotY = this.img.height / 2;

}

commonEnmeyAndCandy(): void {

this.moveSwitch = true;

this.randomSpeed = Math.floor(Math.random() \* 5) + 10;

this.initialAngle = Math.floor(Math.random() \* 360);

this.scale = Math.floor(Math.random() \* 4) + 6;

this.self.scaleX = this.scale / 10;

this.self.scaleY = this.scale / 10;

this.vinshTime = Math.floor(Math.random() \* 5) + 2;

this.startAlpha = (Math.floor(Math.random() \* 10) + 5) / 10;

this.self.alpha = this.startAlpha;

this.rotationD = Math.floor(Math.random() \* 2) === 1 ? -20 : 20;

}

fightingProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy//.png';

}

rangeProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy//.png';

}

redCandyProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy/特效/红色单元.png';

}

yellowCandyProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy//.png';

}

blueCandyProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy//.png';

}

greenCandyProperty(): void {

this.commonEnmeyAndCandy();

this.img.skin = 'candy.png';

}

commonExplosion(): void {

this.accelerated += 0.2;

if (this.timer > 0 && this.timer <= 5) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed);

} else if (this.timer > 5 && this.timer < 10) {

this.self.alpha -= 0.02;

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed);

} else if (this.timer >= 10) {

this.vinshTime -= 0.1;

if (this.vinshTime < 0) {

this.self.removeSelf();

}

}

}

fireworksProperty(): void {

this.moveSwitch = true;

this.randomSpeed = Math.floor(Math.random() \* 15) + 2;

this.initialAngle = Math.floor(Math.random() \* 360);

this.scale = Math.floor(Math.random() \* 8) + 4;

this.self.scaleX = this.scale / 10;

this.self.scaleY = this.scale / 10;

this.vinshTime = Math.floor(Math.random() \* 5) + 2;

this.startAlpha = 1;

this.self.alpha = this.startAlpha;

this.rotationD = Math.floor(Math.random() \* 2) === 1 ? -10 : 10;

let number = Math.floor(Math.random() \* 7) + 1;

switch (number) {

case 1:

this.img.skin = 'candy/特效/星星1.png';

break;

case 2:

this.img.skin = 'candy/特效/星星2.png';

break;

case 3:

this.img.skin = 'candy/特效/星星3.png';

break;

case 4:

this.img.skin = 'candy/特效/星星4.png';

break;

case 5:

this.img.skin = 'candy/特效/星星5.png';

break;

case 6:

this.img.skin = 'candy/特效/星星6.png';

break;

case 7:

this.img.skin = 'candy/特效/星星7.png';

break;

default:

break;

}

}

fireworksExplosion(): void {

this.img.rotation += this.rotationD;

this.accelerated += 0.1;

if (this.timer > 0 && this.timer <= 15) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed + 5);

} else if (this.timer > 15 && this.timer < 18) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed - 5);

} else if (this.timer >= 18) {

this.self.removeSelf();

}

}

smokeProperty(): void {

this.moveSwitch = true;

this.randomSpeed = Math.floor(Math.random() \* 10) + 5;

this.initialAngle = 90;

this.scale = Math.floor(Math.random() \* 4) + 2;

this.self.scale(this.scale / 10, this.scale / 10);

this.vinshTime = Math.floor(Math.random() \* 5) + 2;

this.startAlpha = 1;

this.self.alpha = this.startAlpha;

this.img.skin = 'candy/特效/白色单元.png';

}

smokeEffects(): void {

if (this.timer > 0 && this.timer <= 10) {

this.self.scaleX += 0.08;

this.self.scaleY += 0.08;

this.self.y -= 0.01;

} else if (this.timer > 10 && this.timer < 40) {

this.self.y -= 2;

} else if (this.timer > 40) {

this.self.y -= 2;

this.startAlpha -= 0.025;

if (this.startAlpha < 0) {

this.self.removeSelf();

}

}

}

disappearProperty(): void {

this.moveSwitch = true;

this.randomSpeed = Math.random() \* 2 + 2;

this.initialAngle = Math.floor(Math.random() \* 360);

this.scale = 7;

this.self.scale(this.scale / 10, this.scale / 10);

this.vinshTime = Math.floor(Math.random() \* 5) + 2;

this.startAlpha = (Math.floor(Math.random() \* 6) + 4) / 10;

this.self.alpha = this.startAlpha;

this.rotationD = Math.floor(Math.random() \* 2) === 1 ? -5 : 5;

this.img.skin = 'candy/特效/白色单元.png';

this.img.rotation = this.initialAngle - 90;

let number = Math.floor(Math.random() \* 2);

switch (number) {

case 0:

this.img.skin = 'candy/特效/白色单元.png';

break;

case 1:

this.img.skin = 'candy/特效/白色单元\_02.png';

break;

default:

break;

}

}

disappearEffects(): void {

this.accelerated += 0.01;

if (this.timer > 0 && this.timer <= 15) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed);

} else if (this.timer > 15 && this.timer < 25) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed - 2);

} else if (this.timer >= 25) {

this.self.alpha -= 0.02;

if (this.self.alpha <= 0) {

this.self.removeSelf();

}

}

}

appearProperty(): void {

this.moveSwitch = true;

this.randomSpeed = Math.random() \* 2 + 2;

this.initialAngle = Math.floor(Math.random() \* 360);

this.scale = 7;

this.self.scale(this.scale / 10, this.scale / 10);

this.vinshTime = Math.floor(Math.random() \* 5) + 2;

this.startAlpha = (Math.floor(Math.random() \* 6) + 4) / 10;

this.self.alpha = this.startAlpha;

this.rotationD = Math.floor(Math.random() \* 2) === 1 ? -5 : 5;

this.img.skin = 'candy/特效/白色单元.png';

this.img.rotation = this.initialAngle - 90;

let number = Math.floor(Math.random() \* 2);

switch (number) {

case 0:

this.img.skin = 'candy/特效/白色单元.png';

break;

case 1:

this.img.skin = 'candy/特效/白色单元\_02.png';

break;

default:

break;

}

}

appearEffects(): void {

this.accelerated += 0.01;

if (this.timer > 0 && this.timer <= 20) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed);

} else if (this.timer > 20 && this.timer < 30) {

this.commonSpeedXYByAngle(this.initialAngle, this.randomSpeed - 2);

} else if (this.timer >= 30) {

this.self.alpha -= 0.02;

if (this.self.alpha <= 0) {

this.self.removeSelf();

}

}

}

starShiningProperty(): void {

this.moveSwitch = true;

this.randomSpeed = Math.floor(Math.random() \* 15) + 2;

this.scale = Math.floor(Math.random() \* 8) + 4;

this.self.scale(this.scale / 10, this.scale / 10);

this.startAlpha = 0;

this.self.alpha = this.startAlpha;

this.rotationD = Math.floor(Math.random() \* 2) === 1 ? -5 : 5;

let number = Math.floor(Math.random() \* 7) + 1;

switch (number) {

case 1:

this.img.skin = 'candy/特效/星星1.png';

break;

case 2:

this.img.skin = 'candy/特效/星星2.png';

break;

case 3:

this.img.skin = 'candy/特效/星星3.png';

break;

case 4:

this.img.skin = 'candy/特效/星星4.png';

break;

case 5:

this.img.skin = 'candy/特效/星星5.png';

break;

case 6:

this.img.skin = 'candy/特效/星星6.png';

break;

case 7:

this.img.skin = 'candy/特效/星星7.png';

break;

default:

break;

}

}

starShiningEffects(): void {

if (this.timer > 0 && this.timer <= 15) {

this.self.alpha += 0.1;

this.self.rotation += this.rotationD;

this.self.scaleX += 0.01;

this.self.scaleY += 0.01;

} else if (this.timer > 15 && this.timer < 25) {

this.self.rotation += this.rotationD;

} else if (this.timer >= 25) {

this.self.rotation -= this.rotationD;

this.self.alpha -= 0.015;

this.self.scaleX -= 0.01;

this.self.scaleY -= 0.01;

if (this.self.alpha <= 0) {

this.self.removeSelf();

}

}

}

move(): void {

if (this.effectsType === 'fireworks') {

this.fireworksExplosion();

} else if (this.effectsType === 'smokeEffects') {

this.smokeEffects();

} else if (this.effectsType === 'disappear') {

this.disappearEffects();

} else if (this.effectsType === 'appear') {

this.appearEffects();

} else if (this.effectsType === 'starShining') {

this.starShiningEffects();

} else {

this.commonExplosion();

}

}

commonSpeedXYByAngle(angle, speed) {

this.self.x += tools.speedXYByAngle(angle, speed + this.accelerated).x;

this.self.y += tools.speedXYByAngle(angle, speed + this.accelerated).y;

}

onUpdate(): void {

if (this.moveSwitch) {

this.timer += 1;

this.move();

}

}

onDisable(): void {

Laya.Pool.recover('explode', this.self);

}

}

export default class HintWord extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Sprite;

private propertyType: Laya.Image;

private addNumber: Laya.FontClip;

private subNumber: Laya.FontClip;

private addScore: Laya.FontClip;

private scoreLabel: Laya.FontClip;

constructor() { super(); }

onEnable(): void {

this.self = this.owner as Laya.Sprite;

this.self['HintWord'] = this;

this.selfScene = this.self.scene;

this.self.alpha = 0;

this.self.pivotX = this.self.width / 2;

this.self.rotation = 0;

this.self.scale(1, 1);

this.propertyType = this.self.getChildByName('propertyType') as Laya.Image;

this.addNumber = this.self.getChildByName('addNumber') as Laya.FontClip;

this.subNumber = this.self.getChildByName('subNumber') as Laya.FontClip;

this.addScore = this.self.getChildByName('addScore') as Laya.FontClip;

this.propertyType.skin = null;

this.addNumber.value = null;

this.subNumber.value = null;

this.addScore.value = null;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

}

initProperty(propertyType: string, numberValue: number): void {

switch (propertyType) {

case '':

this.propertyType.skin = 'candy/ui/速度图标.png';

this.addNumber.value = "+" + numberValue;

this.propertyMove();

break;

case '':

this.propertyType.skin = 'candy/ui/攻击力图标.png';

this.addNumber.value = "+" + numberValue;

this.propertyMove();

break;

case '':

this.propertyType.skin = 'candy/ui/生命图标.png';

this.addNumber.value = "+" + numberValue;

this.propertyMove();

break;

case '':

this.propertyType.skin = 'candy/ui/防御图标.png';

this.addNumber.value = "+" + numberValue;

this.propertyMove();

break;

case '':

this.propertyType.skin = 'candy/ui/速度图标.png';

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '':

this.propertyType.skin = 'candy/ui/攻击力图标.png';

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '减少生命':

this.propertyType.skin = 'candy/ui/生命图标.png';

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '减少防御力':

this.propertyType.skin = 'candy/ui/防御图标.png';

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '主角掉血':

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '敌人掉血':

this.subNumber.value = "-" + numberValue;

this.propertyMove();

break;

case '额外奖励分数':

this.addScore.value = "+" + numberValue;

this.rewardScoreMove(numberValue);

break;

case '杀怪分数':

this.addScore.value = "+" + numberValue;

this.killEnemy(numberValue);

break;

default:

break;

}

}

propertyMove(): void {

let timeLine = new Laya.TimeLine;

timeLine.addLabel('appear', 0).to(this.self, { y: this.self.y - 60, alpha: 1 }, 100, null, 0)

.addLabel('pause', 0).to(this.self, { y: this.self.y - 120 }, 800, null, 0)

.addLabel('vanish', 0).to(this.self, { y: this.self.y - 150, alpha: 0 }, 100, null, 0)

timeLine.play('appear', false);

timeLine.on(Laya.Event.COMPLETE, this, function () {

this.self.removeSelf();

});

}

rewardScoreMove(numberValue): void {

this.self.scale(1, 1);

let scoreLabel\_p = this.scoreLabel.parent as Laya.Sprite;

let scoreLabel\_p\_p = scoreLabel\_p.parent as Laya.Sprite;

let stageX = this.scoreLabel.x + scoreLabel\_p.x + scoreLabel\_p\_p.x;

let stageY = this.scoreLabel.y + scoreLabel\_p.y + scoreLabel\_p\_p.y;

let timeLine = new Laya.TimeLine;

timeLine.addLabel('appear', 0).to(this.self, { y: this.self.y - 60, alpha: 1 }, 100, null, 0)

.addLabel('pause', 0).to(this.self, { y: this.self.y - 120 }, 600, null, 0)

.addLabel('moveUp', 0).to(this.self, { y: this.self.y - 150, alpha: 1 }, 100, null, 0)

.addLabel('moveTo', 0).to(this.self, { scaleX: 0.5, scaleY: 0.5, rotation: -360, x: stageX, y: stageY, alpha: 1 }, 600, Laya.Ease.cubicIn, 0)

.addLabel('vanish', 0).to(this.self, { y: stageY - 100, alpha: 0 }, 300, Laya.Ease.cubicIn, 0)

timeLine.play('appear', false);

timeLine.on(Laya.Event.COMPLETE, this, function () {

this.self.removeSelf();

this.selfScene['MainSceneControl'].addScores(numberValue);

});

}

killEnemy(numberValue): void {

let timeLine = new Laya.TimeLine;

timeLine.addLabel('appear', 0).to(this.self, { y: this.self.y - 60, alpha: 1 }, 100, null, 0)

.addLabel('pause', 0).to(this.self, { y: this.self.y - 120 }, 600, null, 0)

.addLabel('moveUp', 0).to(this.self, { y: this.self.y - 150, alpha: 0 }, 100, null, 0)

timeLine.play('appear', false);

timeLine.on(Laya.Event.COMPLETE, this, function () {

this.self.removeSelf();

this.selfScene['MainSceneControl'].addScores(numberValue);

});

}

onDisable(): void {

Laya.Pool.recover('hintWord', this.self);

Laya.timer.clearAll(this);

Laya.Tween.clearAll(this);

}

}

import MainSceneControl from "./MainSceneControl";

import Candy from "./Candy";

import tools from "./Tool";

export default class OperationButton extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private candyParent: Laya.Sprite;

private operateSwitch: boolean;

public btnGroup: Laya.Sprite;

public redButton: Laya.Sprite;

public yellowButton: Laya.Sprite;

public greenButton: Laya.Sprite;

public blueButton: Laya.Sprite;

private timer: Laya.Sprite;

private timeSchedule: Laya.ProgressBar;

public clickHintSign: Laya.Sprite;

private scoreLabel: Laya.FontClip;

private rewardWords: Laya.Prefab;

private clicksGroup: number;

private zeroCount: number;

private erroCount: number;

constructor() { super(); }

onEnable(): void {

this.initProperty();

this.buttonClink();

}

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.selfScene = this.self.scene;

this.candyParent = this.selfScene['MainSceneControl'].candyParent;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

this.timer = this.selfScene['MainSceneControl'].timer;

this.rewardWords = this.selfScene['MainSceneControl'].rewardWords;

this.clickHintSign.alpha = 0;

this.timeSchedule = this.timer.getChildByName('timeSchedule') as Laya.ProgressBar;

this.self['OperationControl'] = this;

this.clicksGroup = 0;

this.zeroCount = 0;

this.erroCount = 0;

}

buttonClink(): void {

for (let i = 0; i < this.btnGroup.\_children.length; i++) {

this.btnGroup.\_children[i].on(Laya.Event.MOUSE\_DOWN, this, this.down);

this.btnGroup.\_children[i].on(Laya.Event.MOUSE\_MOVE, this, this.move);

this.btnGroup.\_children[i].on(Laya.Event.MOUSE\_UP, this, this.up);

this.btnGroup.\_children[i].on(Laya.Event.MOUSE\_OUT, this, this.out);

console.log(this.btnGroup.\_children[i]);

}

}

down(event): void {

if (!this.operateSwitch) {

return;

}

let btn\_name = event.currentTarget.name;

this.clickJudge(btn\_name);

event.currentTarget.scale(0.9, 0.9);

}

clickJudge(btn\_name): void {

let pairName;

switch (btn\_name) {

case 'redButton':

pairName = 'redCandy\_\_\_';

break;

case 'yellowButton':

pairName = 'yellowCandy';

break;

case 'greenButton':

pairName = 'greenCandy\_';

break;

case 'blueButton':

pairName = 'blueCandy\_\_';

break;

default: break;

}

let nameCount = 0;

let rightCount = 0;

let mainSceneControl = this.selfScene['MainSceneControl'];

for (let i = 0; i < this.candyParent.\_children.length; i++) {

let candy = this.candyParent.\_children[i];

let group = candy['Candy'].group;

let name = candy.name.substring(0, 11);

let clicksLabel = candy.getChildByName('clicksLabel') as Laya.FontClip;

if (group === this.clicksGroup && name === pairName) {

nameCount++;

if (nameCount === 1 || nameCount === 2) {

if (Number(clicksLabel.value) > 0) {

rightCount++;

clicksLabel.value = (Number(clicksLabel.value) - 1).toString();

this.clicksRightAni(candy);

if (Number(clicksLabel.value) === 0) {

mainSceneControl.explodeAni(this.selfScene, candy.x, candy.y, 'disappear', 8, 100);

candy['Candy'].playSkeletonAni(1, 'static');

clicksLabel.value = ' ';

this.zeroCount++;

}

break;

}

}

}

}

if (nameCount === 0 || rightCount === 0) {

this.erroCount += 2;

for (let i = 0; i < this.candyParent.\_children.length; i++) {

let candy = this.candyParent.\_children[i];

let group = candy['Candy'].group;

if (group === this.clicksGroup) {

mainSceneControl.explodeAni(this.selfScene, candy.x, candy.y, 'disappear', 8, 1000);

candy['Candy'].group = 'error';

candy['Candy'].playSkeletonAni(1, 'explode');

let clicksLabel = candy.getChildByName('clicksLabel') as Laya.FontClip;

clicksLabel.value = ' ';

}

}

this.groupHint();

}

if (this.zeroCount === 2) {

this.groupHint();

}

if (this.clicksGroup === 4) {

this.settlement('finished');

}

}

clicksRightAni(candy): void {

Laya.Tween.to(candy, { scaleX: 1.1, scaleY: 1.1 }, 50, null, Laya.Handler.create(this, function () {

Laya.Tween.to(candy, { scaleX: 1, scaleY: 1 }, 50, null, Laya.Handler.create(this, function () {

}, []), 0);

}, []), 0);

}

groupHint(): void {

this.zeroCount = 0;

this.clicksGroup++;

let maskYArr = [753, 651.5, 550, 449];

this.clickHintSign.alpha = 1;

this.clickHintSign.y = maskYArr[this.clicksGroup];

}

initHint(): void {

this.zeroCount = 0;

this.clicksGroup = 0;

let maskYArr = [753, 651.5, 550, 449];

this.clickHintSign.alpha = 1;

this.clickHintSign.y = maskYArr[this.clicksGroup];

this.erroCount = 0;

}

settlement(type): void {

this.clickHintSign.alpha = 0;

this.operateSwitch = false;

if (this.erroCount === 0) {

this.additionAward();

}

this.settlementAni();

} else if ('unfinished') {

let delayed = 0;

for (let i = this.candyParent.\_children.length - 1; i >= 0; i--) {

let candy = this.candyParent.\_children[i];

let group = candy['Candy'].group;

if (group !== 'error') {

let clicksLabel = candy.getChildByName('clicksLabel') as Laya.FontClip;

if (clicksLabel.value !== ' ') {

delayed += 15;

Laya.timer.frameOnce(delayed, this, function () {

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, candy.x, candy.y, 'disappear', 8, 1000);

clicksLabel.value = ' ';

candy['Candy'].group = 'error';

candy['Candy'].playSkeletonAni(1, 'explode');

if (i === 1) {

this.settlementAni();

}

})

}

}

}

}

}

settlementAni(): void {

let delayed = 10;

for (let i = this.candyParent.\_children.length - 1; i >= 0; i--) {

delayed += 15;

Laya.timer.frameOnce(delayed, this, function () {

let candy = this.candyParent.\_children[i];

let group = candy['Candy'].group;

if (group === 'error') {

this.explodeCandyFlyToRole(candy);

} else {

candy['Candy'].candyFlyToRole();

}

if (i === 0) {

this.selfScene['MainSceneControl'].candyLaunch\_01.play('prepare', false);

this.selfScene['MainSceneControl'].candyLaunch\_02.play('prepare', false);

}

})

}

}

explodeCandyFlyToRole(candy: Laya.Sprite): void {

let point;

let explodeTarget;

let HalfX;

let HalfY;

let distancePer = 2;

if (candy.x < Laya.stage.width / 2) {

explodeTarget = this.selfScene['MainSceneControl'].role\_01;

point = new Laya.Point(candy.x - 200, candy.y + 80);

HalfX = candy.x - (candy.x - point.x) / distancePer;

} else {

explodeTarget = this.selfScene['MainSceneControl'].role\_02;

point = new Laya.Point(candy.x + 200, candy.y + 80);

HalfX = candy.x + (point.x - candy.x) / distancePer;

}

HalfY = candy.y - 100;

Laya.Tween.to(candy, { x: HalfX, y: HalfY, scaleX: 1.3, scaleY: 1.3 }, 250, null, Laya.Handler.create(this, function () {

Laya.Tween.to(candy, { x: point.x, y: point.y, scaleX: 0.9, scaleY: 0.9 }, 300, null, Laya.Handler.create(this, function () {

candy.scale(0.9, 0.9);

this.selfScene['MainSceneControl'].explodeAni(this.selfScene, candy.x, candy.y, 'disappear', 8, 1000);

candy['Candy'].asExplodeCandy();

}, []), 0);

}, []), 0);

let shadow = candy.getChildByName('shadow') as Laya.Image;

Laya.Tween.to(shadow, { x: -20 + 52, y: 80 + 60, scaleX: 0.8, scaleY: 0.8, }, 300, null, Laya.Handler.create(this, function () {

Laya.Tween.to(shadow, { x: 0 + 52, y: 0 + 60, scaleX: 1, scaleY: 1 }, 300, null, Laya.Handler.create(this, function () {

}), 0);

}), 0);

}

timerControl(): void {

if (this.timeSchedule.value > 0 && this.operateSwitch) {

this.timeSchedule.value -= 0.0015;

} else if (this.timeSchedule.value <= 0 && this.operateSwitch) {

this.settlement('unfinished');

}

}

additionAward(): void {

if (this.timeSchedule.value > 0.8) {

this.creatRewardWords('干得漂亮');

} else if (this.timeSchedule.value > 0.6) {

this.creatRewardWords('太棒了');

} else if (this.timeSchedule.value > 0.4) {

this.creatRewardWords('牛皮');

}

}

creatRewardWords(word): void {

let rewardWords = Laya.Pool.getItemByCreateFun('rewardWords', this.rewardWords.create, this.rewardWords) as Laya.Sprite;

this.selfScene.addChild(rewardWords);

rewardWords['RewardWords'].createWordsAni(word);

}

move(event): void {

event.currentTarget.scale(1, 1);

}

up(event): void {

event.currentTarget.scale(1, 1);

}

out(event): void {

event.currentTarget.scale(1, 1);

}

onUpdate(): void {

if (this.selfScene['MainSceneControl'].gameOver) {

this.operateSwitch = false;

return;

}

this.timerControl();

}

onDisable(): void {

}

}

export default class Ranking extends Laya.Script {

public but\_Close: Laya.Sprite;

public content: Laya.Sprite;

public background: Laya.Sprite;

private self;

constructor() { super(); }

onEnable(): void {

this.init();

}

init(): void {

this.self = this.owner as Laya.Sprite;

this.background.alpha = 0;

this.background.width = Laya.stage.width;

this.background.height = Laya.stage.height;

this.content.x = Laya.stage.width / 2;

this.content.y = Laya.stage.height / 2;

this.content.scale(0, 0);

this.content.alpha = 0;

this.appear();

}

onAwake() {

if (Laya.Browser.onMiniGame) {

let wx: any = Laya.Browser.window.wx;

let openDataContext: any = wx.getOpenDataContext();

openDataContext.postMessage({ action: 'ranking' });

}

}

appear(): void {

Laya.Tween.to(this.content, { rotation: 720, alpha: 1, scaleX: 1, scaleY: 1 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

this.rotation = 0;

this.btnClikClink();

}, []), 0);

Laya.Tween.to(this.background, { alpha: 0.5 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

}

vanish(): void {

this.self.pivotX = Laya.stage.width / 2;

this.self.pivotY = Laya.stage.height / 2;

this.self.x = Laya.stage.width / 2;

this.self.y = Laya.stage.height / 2;

Laya.Tween.to(this.self, { rotation: -720, alpha: 0, scaleX: 0, scaleY: 0, x: 1500 }, 700, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

this.self.removeSelf();

if (Laya.Browser.onMiniGame) {

let wx: any = Laya.Browser.window.wx;

let openDataContext: any = wx.getOpenDataContext();

openDataContext.postMessage({ action: 'close' });

}

}, []), 0);

Laya.Tween.to(this.background, { alpha: 0 }, 450, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

}, []), 0);

}

btnClikClink(): void {

this.but\_Close.on(Laya.Event.MOUSE\_DOWN, this, this.down);

this.but\_Close.on(Laya.Event.MOUSE\_MOVE, this, this.move);

this.but\_Close.on(Laya.Event.MOUSE\_UP, this, this.up);

this.but\_Close.on(Laya.Event.MOUSE\_OUT, this, this.out);

}

btnCloseClink(): void {

this.but\_Close.off(Laya.Event.MOUSE\_DOWN, this, this.down);

this.but\_Close.off(Laya.Event.MOUSE\_MOVE, this, this.move);

this.but\_Close.off(Laya.Event.MOUSE\_UP, this, this.up);

this.but\_Close.off(Laya.Event.MOUSE\_OUT, this, this.out);

}

down(event): void {

this.but\_Close.scale(0.9, 0.9);

}

move(event): void {

this.but\_Close.scale(0.9, 0.9);

}

up(event): void {

this.but\_Close.scale(1, 1);

this.vanish();

}

out(event): void {

this.but\_Close.scale(1, 1);

}

onDisable(): void {

}

}

export default class Resurgence extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private background: Laya.Sprite;

private content: Laya.Sprite;

private resurgence\_Btn: Laya.Image;

private digitalPlate: Laya.Image;

private timeLine: number;

private countdown: boolean;

private hintWord: Laya.Prefab;

private settlement: Laya.Prefab;

constructor() { super(); }

onEnable(): void {

this.init();

}

init(): void {

this.self = this.owner as Laya.Sprite;

this.self.pivotX = 0;

this.self.pivotY = 0;

this.self.x = this.self.pivotX;

this.self.y = this.self.pivotY;

this.selfScene = this.self.scene as Laya.Scene;

this.background = this.self.getChildByName('background') as Laya.Sprite;

this.background.alpha = 0;

this.content = this.self.getChildByName('content') as Laya.Sprite;

this.content.alpha = 0;

this.resurgence\_Btn = this.content.getChildByName('resurgence\_Btn') as Laya.Image;

this.resurgence\_Btn.x = -1200;

this.digitalPlate = this.content.getChildByName('digitalPlate') as Laya.Image;

this.digitalPlate.x = 1200;

this.digital = this.content.getChildByName('digital') as Laya.FontClip;

this.digital.scaleX = 0;

this.digital.scaleY = 0;

this.digital.alpha = 0;

this.digital.value = '5';

this.timeLine = 0;

this.countdown = false;

this.hintWord = this.selfScene['MainSceneControl'].hintWord;

this.settlement = this.selfScene['MainSceneControl'].settlement;

Laya.timer.frameOnce(100, this, function () {

this.appearAni();

})

this.adaptive();

}

adaptive(): void {

this.background.width = Laya.stage.width;

this.background.height = Laya.stage.height;

this.content.x = Laya.stage.width / 2;

this.content.y = Laya.stage.height / 2;

this.self.width = Laya.stage.width;

this.self.height = Laya.stage.height;

this.self.alpha = 1;

this.self.scale(1, 1);

this.self.rotation = 0;

this.self.x = 0;

this.self.y = 0;

}

appearAni(): void {

Laya.Tween.to(this.content, { alpha: 1 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.resurgence\_Btn, { x: 375, rotation: 720 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

this.resurgence\_Btn.rotation = 0;

}, []), 0);

Laya.Tween.to(this.digitalPlate, { x: 375, rotation: 720 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

this.digitalPlate.rotation = 0;

this.btnonClink();

}, []), 0);

Laya.Tween.to(this.background, { alpha: 0.7 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

Laya.Tween.to(this.digital, { scaleX: 1.2, scaleY: 1.2, alpha: 1 }, 700, null, Laya.Handler.create(this, function () {

this.countdown = true;

}, []), 0);

}

cutTnterface(type): void {

this.btnOffClink();

this.self.pivotX = Laya.stage.width / 2;

this.self.pivotY = Laya.stage.height / 2;

this.self.x = this.self.pivotX;

this.self.y = this.self.pivotY;

Laya.Tween.to(this.self, { x: 1500, rotation: 720, scaleX: 0, scaleY: 0, alpha: 0 }, 700, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

if (type === 'main') {

this.roleResurgenceAni();

this.self.removeSelf();

} else {

this.self.removeSelf();

}

}, []), 0);

Laya.Tween.to(this.background, { alpha: 0 }, 450, Laya.Ease.expoIn, Laya.Handler.create(this, function () {

}, []), 0);

}

roleResurgenceAni(): void {

let role\_01 = this.selfScene['MainSceneControl'].role\_01;

let role\_02 = this.selfScene['MainSceneControl'].role\_02;

Laya.Tween.to(role\_01, { alpha: 1 }, 700, null, Laya.Handler.create(this, function () {

this.hintWordMove();

}, []), 0);

Laya.Tween.to(role\_02, { alpha: 1 }, 700, null, Laya.Handler.create(this, function () {

}, []), 0);

}

hintWordMove(): void {

let delayed = 0;

let MainSceneControl = this.selfScene['MainSceneControl'];

let role\_01 = MainSceneControl.role\_01;

let role\_02 = MainSceneControl.role\_02;

for (let i = 0; i < 4; i++) {

Laya.timer.frameOnce(delayed, this, function () {

switch (i) {

case 0:

MainSceneControl.createHintWord(role\_01, 100, -10, '攻击里', 20, 1);

MainSceneControl.createHintWord(role\_02, 100, -10, '攻击里', 20, 1);

role\_01['Role'].role\_property.attackValue += 20;

role\_02['Role'].role\_property.attackValue += 20;

break;

case 1:

MainSceneControl.createHintWord(role\_01, 100, -10, '生命', 1000, 1);

MainSceneControl.createHintWord(role\_02, 100, -10, '生命', 1000, 1);

role\_01['Role'].role\_property.blood = 1000;

role\_02['Role'].role\_property.blood = 1000;

break;

case 2:

MainSceneControl.createHintWord(role\_01, 100, -10, '速度', 20, 1);

MainSceneControl.createHintWord(role\_02, 100, -10, '速度', 20, 1);

role\_01['Role'].role\_property.attackSpeed += 20;

role\_02['Role'].role\_property.attackSpeed += 20;

break;

case 3:

MainSceneControl.createHintWord(role\_01, 100, -10, '防御力', 10, 1);

MainSceneControl.createHintWord(role\_02, 100, -10, '防御力', 10, 1);

role\_01['Role'].role\_property.defense += 10;

role\_02['Role'].role\_property.defense += 10;

break;

default:

break;

}

if (i === 3) {

this.resurgenceProperties();

}

})

delayed += 25;

}

}

resurgenceProperties(): void {

let MainSceneControl = this.selfScene['MainSceneControl'];

MainSceneControl.gameOver = false;

MainSceneControl.role\_01['Role'].roleDeath = false;

MainSceneControl.role\_02['Role'].roleDeath = false;

MainSceneControl.operating['OperationControl'].operateSwitch = true;

}

btnonClink(): void {

this.resurgence\_Btn.on(Laya.Event.MOUSE\_DOWN, this, this.down);

this.resurgence\_Btn.on(Laya.Event.MOUSE\_MOVE, this, this.move);

this.resurgence\_Btn.on(Laya.Event.MOUSE\_UP, this, this.up);

this.resurgence\_Btn.on(Laya.Event.MOUSE\_OUT, this, this.out);

}

btnOffClink(): void {

this.resurgence\_Btn.off(Laya.Event.MOUSE\_DOWN, this, this.down);

this.resurgence\_Btn.off(Laya.Event.MOUSE\_MOVE, this, this.move);

this.resurgence\_Btn.off(Laya.Event.MOUSE\_UP, this, this.up);

this.resurgence\_Btn.off(Laya.Event.MOUSE\_OUT, this, this.out);

}

down(event): void {

event.currentTarget.scale(0.95, 0.95);

}

move(event): void {

event.currentTarget.scale(1, 1);

}

up(event): void {

this.btnOffClink();

event.currentTarget.scale(1, 1);

this.countdown = false;

this.cutTnterface('main');

}

out(event): void {

event.currentTarget.scale(1, 1);

}

createSettlement(): void {

this.selfScene['MainSceneControl'].wxPostData();

Laya.timer.frameOnce(20, this, function () {

let settlement = Laya.Pool.getItemByCreateFun('settlement', this.settlement.create, this.settlement) as Laya.Sprite;

this.selfScene.addChild(settlement);

this.selfScene['MainSceneControl'].roleVanish();

settlement.pos(0, 0);

})

}

onUpdate(): void {

if (this.countdown) {

this.timeLine++;

if (this.timeLine % 60 == 0) {

this.digital.value = (Number(this.digital.value) - 1).toString();

this.digital.scale(1, 1);

if (this.digital.value === '0') {

this.countdown = false;

this.createSettlement();

this.cutTnterface('settlement');

}

} else {

this.digital.scaleX -= 0.003;

this.digital.scaleY -= 0.003;

}

}

}

onDisable(): void {

Laya.Tween.clearAll(this);

Laya.timer.clearAll(this);

Laya.Pool.recover('resurgence', this.self);

}

}

import MainSceneControl from "./MainSceneControl";

import Candy from "./Candy";

import Bullet from "./RoleBullet";

export default class Role extends Laya.Script {

public bulletParent: Laya.Sprite;

public roleBullet: Laya.Prefab;

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private mainSceneControl;

private candyParent: Laya.Sprite;

private selfHealth: Laya.ProgressBar;

private bloodLabel: Laya.Label;

private propertyShow: Laya.Image;

private role\_property: any;

private role\_Warning: boolean;

private nowTime: number;

public scoreLabel: Laya.FontClip;

private templet: Laya.Templet;

private skeleton: Laya.Skeleton;

private roleDeath: boolean;

constructor() { super(); }

onEnable(): void {

this.initProperty();

this.bucketClink();

this.createBoneAni();

}

initProperty(): void {

this.self = this.owner as Laya.Sprite;

this.selfHealth = this.self.getChildByName('health') as Laya.ProgressBar;

this.selfHealth.value = 1;

this.propertyShow = this.self.getChildByName('propertyShow') as Laya.Image;

this.propertyShow.alpha = 0;

this.bloodLabel = this.selfHealth.getChildByName('bloodLabel') as Laya.Label;

this.self['Role'] = this;

this.selfScene = this.self.scene as Laya.Scene;

this.candyParent = this.selfScene['MainSceneControl'].candyParent;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

this.nowTime = Date.now();

this.roleDeath = false;

this.rolePropertySet();

}

createBoneAni(): void {

this.templet = new Laya.Templet();

this.templet.on(Laya.Event.COMPLETE, this, this.parseComplete);

this.templet.on(Laya.Event.ERROR, this, this.onError);

if (this.self.name === 'role\_01') {

this.templet.loadAni("candy/主角/role\_01.sk");

} else if (this.self.name === 'role\_02') {

this.templet.loadAni("candy/主角/role\_02.sk");

}

}

onError(): void {

}

parseComplete(): void {

this.skeleton = this.templet.buildArmature(0);

this.skeletonListen();

this.self.addChild(this.skeleton);

this.skeleton.play('frontMove', true);

this.skeleton.playbackRate(1);

this.skeleton.name = 'skeleton';

if (this.self.name === 'role\_01') {

this.skeleton.x = 60;

this.skeleton.y = 72;

} else if (this.self.name === 'role\_02') {

this.skeleton.x = 60;

this.skeleton.y = 72;

}

}

skeletonListen(): void {

this.skeleton.on(Laya.Event.LABEL, this, function (e) {

if (e.name === 'hitOut') {

this.createBullet();

}

});

this.skeleton.on(Laya.Event.COMPLETE, this, function (e) {

console.log(e);

});

}

createBullet(): void {

let bullet = Laya.Pool.getItemByCreateFun('roleBullet', this.roleBullet.create, this.roleBullet) as Laya.Sprite;

this.bulletParent.addChild(bullet);

bullet.pos(this.self.x, this.self.y);

let pic = bullet.getChildByName('pic') as Laya.Image;

if (this.self.name === 'role\_01') {

pic.skin = 'candy/主角/主角1子弹.png';

} else if (this.self.name === 'role\_02') {

pic.skin = 'candy/主角/主角2子弹.png';

}

this.lockedBulletTarget(bullet);

bullet['Bullet'].belongRole = this.self;

}

playSpeedAdjust(): void {

let playSpeed;

if ((500 - this.role\_property.attackSpeed) / 500 > 0) {

playSpeed = 1 + (500 - this.role\_property.attackSpeed) / 500;

} else {

playSpeed = 1;

}

this.skeleton.playbackRate(playSpeed);

}

onsetAttack() {

this.skeleton.play('attack', false);

this.playSpeedAdjust();

}

rolePropertySet(): void {

if (this.self.name === 'role\_01') {

this.role\_property = {

blood: 100,

attackValue: 5000,

attackSpeed: 1500,

defense: 5,

};

} else if (this.self.name === 'role\_02') {

this.role\_property = {

blood: 100,

attackValue: 5000,

attackSpeed: 1500,

defense: 5,

};

}

}

updateProperty(): void {

this.bloodLabel.text = this.role\_property.blood;

let blood = this.propertyShow.getChildByName('blood') as Laya.Label;

blood.text = "血量: " + this.role\_property.blood;

let attackValue = this.propertyShow.getChildByName('attackValue') as Laya.Label;

attackValue.text = "攻击力: " + this.role\_property.attackValue;

let attackSpeed = this.propertyShow.getChildByName('attackSpeed') as Laya.Label;

attackSpeed.text = "攻击速度: " + this.role\_property.attackSpeed;

let defense = this.propertyShow.getChildByName('defense') as Laya.Label;

defense.text = "防御力: " + this.role\_property.defense;

}

bucketClink(): void {

this.self.on(Laya.Event.MOUSE\_DOWN, this, this.down);

this.self.on(Laya.Event.MOUSE\_MOVE, this, this.move);

this.self.on(Laya.Event.MOUSE\_UP, this, this.up);

this.self.on(Laya.Event.MOUSE\_OUT, this, this.out);

}

down(event): void {

}

move(event): void {

}

up(): void {

this.self.scale(1, 1);

}

out(): void {

this.self.scale(1, 1);

}

lockedBulletTarget(bullet): void {

let distanceArr: Array<any> = [];

let enemyParent = this.selfScene['MainSceneControl'].enemyParent;

for (let i = 0; i < enemyParent.\_children.length; i++) {

let enemy = enemyParent.\_children[i] as Laya.Sprite;

let dx: number = enemy.x - this.self.x;

let dy: number = enemy.y - this.self.y;

let distance: number = Math.sqrt(dx \* dx + dy \* dy);

let object = {

distance: distance,

name: enemy.name

}

distanceArr.push(object);

}

var compare = function (obj1, obj2) {

var val1 = obj1.distance;

var val2 = obj2.distance;

if (val1 < val2) {

return -1;

} else if (val1 > val2) {

return 1;

} else {

return 0;

}

}

distanceArr.sort(compare);

if (distanceArr.length > 0) {

let target = enemyParent.getChildByName(distanceArr[0].name) as Laya.Sprite;

bullet['Bullet'].bulletTarget = target;

bullet['Bullet'].bulletTargetName = target.name;

}

}

onUpdate(): void {

if (this.roleDeath) {

return;

}

if (this.role\_property.blood <= 0) {

this.role\_property.blood = 0;

if (this.self.name === 'role\_01') {

this.roleDeath = true;

this.self.alpha = 0;

} else if (this.self.name === 'role\_02') {

this.roleDeath = true;

this.self.alpha = 0;

}

}

let nowTime = Date.now();

if (this.skeleton && this.role\_Warning) {

if (nowTime - this.nowTime > this.role\_property.attackSpeed) {

if (this.role\_Warning) {

this.onsetAttack();

this.nowTime = nowTime;

}

}

}

}

onDisable(): void {

}

}

export default class Settlement extends Laya.Script {

private self: Laya.Sprite;

private selfScene: Laya.Scene;

private background: Laya.Sprite;

private content: Laya.Sprite;

private operation: Laya.Sprite;

private btn\_Again: Laya.Image;

private btn\_Return: Laya.Image;

private GOLogo: Laya.Image;

private timeLine: number;

private scoreLabel: Laya.FontClip;

private score: Laya.Prefab;

private btnAniSwich: boolean;

private btnAniTime: number;

private btnAniInterval: number;

constructor() { super(); }

onEnable(): void {

this.init();

}

init(): void {

this.self = this.owner as Laya.Sprite;

this.selfScene = this.self.scene as Laya.Scene;

this.background = this.self.getChildByName('background') as Laya.Sprite;

this.content = this.self.getChildByName('content') as Laya.Sprite;

this.background.alpha = 0;

this.operation = this.content.getChildByName('operation') as Laya.Sprite;

this.operation.x = -1200;

this.operation.alpha = 0;

this.btn\_Return = this.operation.getChildByName('btn\_Return') as Laya.Image;

this.btn\_Again = this.operation.getChildByName('btn\_Again') as Laya.Image;

this.GOLogo = this.content.getChildByName('GOLogo') as Laya.Image;

this.GOLogo.x = 1200;

this.GOLogo.alpha = 0;

this.scoreLabel = this.selfScene['MainSceneControl'].scoreLabel;

this.score = this.selfScene['MainSceneControl'].score;

this.timeLine = 0;

this.btnAniSwich = true;

this.btnAniTime = Date.now();

this.btnAniInterval = 3000;

this.adaptive();

this.appearAni();

}

adaptive(): void {

this.background.width = Laya.stage.width;

this.background.height = Laya.stage.height;

this.content.x = Laya.stage.width / 2;

this.content.y = Laya.stage.height / 2 - 50;

this.self.width = Laya.stage.width;

this.self.height = Laya.stage.height;

this.self.x = 0;

this.self.y = 0;

}

appearAni(): void {

Laya.Tween.to(this.operation, { x: 375, rotation: 720, alpha: 1 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

this.operation.rotation = 0;

}, []), 0);

Laya.Tween.to(this.GOLogo, { x: 375, rotation: 720, alpha: 1 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

this.GOLogo.rotation = 0;

this.GOLogoAni();

}, []), 0);

Laya.Tween.to(this.background, { alpha: 0.8 }, 550, Laya.Ease.cubicOut, Laya.Handler.create(this, function () {

}, []), 0);

this.scoreAni();

}

move(event): void {

event.currentTarget.scale(1, 1);

}

up(event): void {

this.btnOffClink();

event.currentTarget.scale(1, 1);

if (event.currentTarget.name === 'btn\_Again') {

this.cutTnterface('restart');

} else if (event.currentTarget.name === 'btn\_Return') {

this.cutTnterface('returnStart');

}

Laya.timer.resume();

Laya.Tween.clearTween(this.GOLogo); }

out(event): void {

Laya.timer.resume();

event.currentTarget.scale(1, 1);

}

onUpdate(): void {

let time = Date.now();

if (this.btnAniSwich) {

if (time - this.btnAniTime > this.btnAniInterval) {

this.btnAniTime = time;

this.btnAni();

}

}

}

onDisable(): void {

Laya.Tween.clearAll(this);

}

}