//游戏逻辑开始

//常量定义 配置加载

//游戏初始化

cc.Class({

extends: cc.Component,

properties: {

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

//程序运行时间

window.NowTime = 0;

// 设置为不抗锯齿

cc.view.enableAntiAlias(false);

var manager = cc.director.getCollisionManager();

manager.enabled = true;

// manager.enabledDebugDraw = true;

cc.director.getPhysicsManager().enabled = true;

cc.director.getPhysicsManager().gravity = cc.v2();

var draw = cc.PhysicsManager.DrawBits;

// cc.director.getPhysicsManager().debugDrawFlags = draw.e\_shapeBit|draw.e\_jointBit;

//全局事件监听

//全局事件

cc.director.GlobalEvent = {

handles\_: {},

//发送事件

emit: function (eventName, data) {

var returns = [] //返回值

data.eventName = eventName//保存一下事件名字

for ( var findEvenName in this.handles\_ ){

if (findEvenName == eventName) {

for (var i = 0; i < this.handles\_[findEvenName].length; i++) {

var returnValue = this.handles\_[findEvenName][i](data)

returns.push(returnValue)

}

}

}

return returns

},

//添加普通事件

on: function (eventName, callback, target) {

// console.log('收到事件', eventName);

this.handles\_[eventName] = this.handles\_[eventName] || []

this.handles\_[eventName].push(callback.bind(target))

},

//通过事件名和target移除一个监听器

off: function (eventName) {

for (var i = 0; i < this.handles\_[eventName].length; i++) {

this.handles\_[eventName][i] = null

}

},

}

},

start () {

},

update (dt) {

window.NowTime += dt

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

cc.Class({

extends: cc.Component,

properties: {

player: {

default: null,

type: cc.Node,

displayName: '玩家',

},

// foo: {

// // ATTRIBUTES:

// default: null, // The default value will be used only when the component attaching

// // to a node for the first time

// type: cc.SpriteFrame, // optional, default is typeof default

// serializable: true, // optional, default is true

// },

// bar: {

// get () {

// return this.\_bar;

// },

// set (value) {

// this.\_bar = value;

// }

// },

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

update (dt) {

},

onBeginContact: function (contact, selfCollider, otherCollider) {

// log("onBeginContact---------------")

// this.node.opacity = 100;

},

// 只在两个碰撞体结束接触时被调用一次

onEndContact: function (contact, selfCollider, otherCollider) {

// this.node.opacity = 255;

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var Lfunction = require('Lfunction');

var HUDManger = cc.Class({

extends: cc.Component,

properties: {

spr\_atlas: {

type: cc.SpriteAtlas,

default: null,

displayName: '通用图集 1',

},

HUD\_root\_node: {

default: null,

type: cc.Node,

displayName: 'HUD根节点',

},

HP\_bar: {

type: cc.ProgressBar,

default: null

},

MP\_bar: {

type: cc.ProgressBar,

default: null

},

level\_lbl: {

type: cc.Label,

default: null

},

part\_info: {

type: cc.Node,

default: null

},

systim\_tip\_layout: {

type: cc.Node,

default: null

},

//面板

lose\_calc\_view: cc.Prefab, //预制件

seting\_view\_pre: cc.Prefab, //预制件

rank\_view\_pre: cc.Prefab, //预制件

achive\_view\_pre: cc.Prefab, //预制件

\_seting\_view: null,

\_rank\_view: null,

\_achive\_view: null,

},

statics:{

\_instance: null

},

onLoad () {

HUDManger.\_instance = this;

this.HP\_bar.progress = 1;

this.MP\_bar.progress = 1;

//事件监听

var self = this;

cc.director.GlobalEvent.on('part\_change', function (data) {

self.setPartNum(data.curr\_part)

}, this.node);

},

start () {

// self.setPartNum(1)

},

update (dt) {

},

getUINodeByName: function (name) {

return this.HUD\_root\_node.getChildByName(name);

},

setHpbarNum: function (value) {

this.HP\_bar.progress = value;

},

setMpbarNum: function (value) {

this.MP\_bar.progress = value;

},

setLevel: function (level) {

this.level\_lbl.string = "LV." + level;

},

openLoseView: function () {

var node = cc.instantiate(this.lose\_calc\_view);

node.parent = this.HUD\_root\_node

},

openSetingView: function () {

if (null == this.\_seting\_view){

// log("creat openSetingView")

this.\_seting\_view = cc.instantiate(this.seting\_view\_pre);

this.\_seting\_view.parent = this.HUD\_root\_node

}

this.\_seting\_view.opacity = 255

},

openRankView: function () {

if (null == this.\_rank\_view){

// log("creat openrankView")

this.\_rank\_view = cc.instantiate(this.rank\_view\_pre);

this.\_rank\_view.parent = this.HUD\_root\_node

}

this.\_rank\_view.opacity = 255

},

openAchiveView: function () {

if (null == this.\_achive\_view){

// log("creat openachiveView")

this.\_achive\_view = cc.instantiate(this.achive\_view\_pre);

this.\_achive\_view.parent = this.HUD\_root\_node

}

this.\_achive\_view.opacity = 255

},

onClickSetingBtn: function () {

this.openSetingView()

},

onClickRankBtn: function () {

this.openRankView()

},

onClickAchiveBtn: function () {

this.openAchiveView()

},

setPartNum: function (num) {

var num\_t = Lfunction.\_instance.getSliptNum(num)

this.part\_info.getChildByName("num\_1").getComponent(cc.Sprite).spriteFrame = this.spr\_atlas.getSpriteFrame("num1\_" + num\_t.ten);

this.part\_info.getChildByName("num\_10").getComponent(cc.Sprite).spriteFrame = this.spr\_atlas.getSpriteFrame("num1\_" + num\_t.signle);

},

});

module.exports = HUDManger;

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var UIinstallTool = require('UI\_install\_tool');

cc.Class({

extends: cc.Component,

properties: {

},

// onLoad () {},

start () {

this.\_shadow = UIinstallTool.\_instance.AddShadow()

},

killMe: function () {

this.\_shadow.DeleteMe();

this.node.removeFromParent();

},

update (dt) {

this.\_shadow.Update(this.node)

},

onCollisionEnter: function (other, self) {

if (other.tag == 1){

this.killMe()

this.node.notify(other)

}

},

onCollisionStay: function (other, self) {

},

onCollisionExit: function (other, self) {

}

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var item\_config = require('item\_config');

var ItemManger = cc.Class({

extends: cc.Component,

properties: {

item\_layer: {

default: null,

type: cc.Node,

displayName: '物品道具层',

},

item: cc.Prefab, //预制件

},

statics: {

\_instance: null

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

ItemManger.\_instance = this;

},

start () {

},

// update (dt) {},

fire: function(item\_id){

// log("skill fire" + item\_id)

let node = cc.instantiate(this.item);

var cfg = item\_config[item\_id]

var Url = "item/icon/" + item\_id

cc.loader.loadRes(Url, cc.SpriteFrame, function (err, spriteFrame) {

spriteFrame.getTexture().setFilters(cc.Texture2D.Filter.NEAREST, cc.Texture2D.Filter.NEAREST)

node.getComponent(cc.Sprite).spriteFrame = spriteFrame;

});

node.parent = this.item\_layer;

node.notify = function (target) {

//player

if (target.tag == 1){

window.SysTip.fireSystimTip("get item:" + item\_id)

target.getComponent("player").UpdateInfo(item\_config[item\_id].attr\_list)

}

}

if (cfg.dis\_time){

this.scheduleOnce(function() {

if (null != node)

// 这里的 this 指向 component

node.getComponent("item").killMe()

// log("remove skill fire")

}, cfg.dis\_time);

}

return node

},

});

// ItemManger.\_instance = new ItemManger();

module.exports = ItemManger;var Common = require('JoystickCommon');

var JoystickBG = require('JoystickBG');

cc.Class({

extends: cc.Component,

properties: {

dot: {

default: null,

type: cc.Node,

displayName: '摇杆节点',

},

ring: {

default: null,

type: JoystickBG,

displayName: '摇杆背景节点',

},

stickX: {

default: 0,

displayName: '摇杆X位置',

},

stickY: {

default: 0,

displayName: '摇杆Y位置',

},

touchType: {

default: Common.TouchType.DEFAULT,

type: Common.TouchType,

displayName: '触摸类型',

},

directionType: {

default: Common.DirectionType.ALL,

type: Common.DirectionType,

displayName: '方向类型',

},

sprite: {

default: null,

type: cc.Node,

displayName: '操控的目标',

},

\_stickPos: {

default: null,

type: cc.Node,

displayName: '摇杆当前位置',

},

\_touchLocation: {

default: null,

type: cc.Node,

displayName: '摇杆当前位置',

}

},

onLoad: function () {

this.\_createStickSprite();

//当触摸类型为FOLLOW会在此对圆圈的触摸监听

if(this.touchType == Common.TouchType.FOLLOW){

this.\_initTouchEvent();

}

},

\_createStickSprite: function()

{

//调整摇杆的位置

this.ring.node.setPosition(this.stickX, this.stickY);

this.dot.setPosition(this.stickX, this.stickY);

},

\_initTouchEvent: function()

{

var self = this;

self.node.on(cc.Node.EventType.TOUCH\_START, self.\_touchStartEvent, self);

self.node.on(cc.Node.EventType.TOUCH\_MOVE, self.\_touchMoveEvent, self);

// 触摸在圆圈内离开或在圆圈外离开后，摇杆归位，player速度为0

self.node.on(cc.Node.EventType.TOUCH\_END, self.\_touchEndEvent,self);

self.node.on(cc.Node.EventType.TOUCH\_CANCEL, self.\_touchEndEvent,self);

},

\_touchStartEvent: function(event) {

// 记录触摸的世界坐标，给touch move使用

this.\_touchLocation = event.getLocation();

var touchPos = this.node.convertToNodeSpaceAR(event.getLocation());

// 更改摇杆的位置

this.ring.node.setPosition(touchPos);

this.dot.setPosition(touchPos);

// 记录摇杆位置，给touch move使用

this.\_stickPos = touchPos;

},

\_touchMoveEvent: function(event) {

// 如果touch start位置和touch move相同，禁止移动

if (this.\_touchLocation.x == event.getLocation().x && this.\_touchLocation.y == event.getLocation().y){

return false;

}

// 以圆圈为锚点获取触摸坐标

var touchPos = this.ring.node.convertToNodeSpaceAR(event.getLocation());

var distance = this.ring.\_getDistance(touchPos,cc.v2(0,0));

var radius = this.ring.node.width / 2;

// 由于摇杆的postion是以父节点为锚点，所以定位要加上touch start时的位置

var posX = this.\_stickPos.x + touchPos.x;

var posY = this.\_stickPos.y + touchPos.y;

if(radius > distance)

{

this.dot.setPosition(cc.v2(posX, posY));

}

else

{

//控杆永远保持在圈内，并在圈内跟随触摸更新角度

var x = this.\_stickPos.x + Math.cos(this.ring.\_getRadian(cc.v2(posX,posY))) \* radius;

var y = this.\_stickPos.y + Math.sin(this.ring.\_getRadian(cc.v2(posX,posY))) \* radius;

this.dot.setPosition(cc.v2(x, y));

}

//更新角度

this.ring.\_getAngle(cc.v2(posX,posY));

//设置实际速度

this.ring.\_setSpeed(cc.v2(posX,posY));

},

\_touchEndEvent: function(){

this.dot.setPosition(this.ring.node.getPosition());

this.ring.\_speed = 0;

},

});

var Common = require('JoystickCommon');

cc.Class({

extends: cc.Component,

properties: {

dot: {

default: null,

type: cc.Node,

displayName: '摇杆节点',

},

\_joyCom: {

default: null,

displayName: 'joy Node',

},

\_playerNode: {

default: null,

displayName: '被操作的目标Node',

},

\_angle: {

default: null,

displayName: '当前触摸的角度',

},

\_radian: {

default: null,

displayName: '弧度',

},

\_speed: 0, //实际速度

\_speed1: 1, //一段速度

\_speed2: 2, //二段速度

\_opacity: 0, //透明度

},

onLoad: function()

{

// joy下的Game组件

this.\_joyCom = this.node.parent.getComponent('joystick');

// game组件下的player节点

this.\_playerNode = this.\_joyCom.sprite;

if(this.\_joyCom.touchType == Common.TouchType.DEFAULT){

//对圆圈的触摸监听

this.\_initTouchEvent();

}

},

//对圆圈的触摸监听

\_initTouchEvent: function()

{

var self = this;

self.node.on(cc.Node.EventType.TOUCH\_START, this.\_touchStartEvent, self);

self.node.on(cc.Node.EventType.TOUCH\_MOVE, this.\_touchMoveEvent, self);

// 触摸在圆圈内离开或在圆圈外离开后，摇杆归位，player速度为0

self.node.on(cc.Node.EventType.TOUCH\_END, this.\_touchEndEvent, self);

self.node.on(cc.Node.EventType.TOUCH\_CANCEL, this.\_touchEndEvent, self);

},

//更新移动目标

update: function(dt)

{

switch (this.\_joyCom.directionType)

{

case Common.DirectionType.ALL:

this.\_allDirectionsMove();

break;

default :

break;

}

},

//全方向移动

\_allDirectionsMove: function()

{

this.\_playerNode.x += Math.cos(this.\_angle \* (Math.PI/180)) \* this.\_speed;

this.\_playerNode.y += Math.sin(this.\_angle \* (Math.PI/180)) \* this.\_speed;

},

//计算两点间的距离并返回

\_getDistance: function(pos1, pos2)

{

return Math.sqrt(Math.pow(pos1.x - pos2.x, 2) +

Math.pow(pos1.y - pos2.y, 2));

},

/\*角度/弧度转换

角度 = 弧度 \* 180 / Math.PI

弧度 = 角度 \* Math.PI / 180\*/

//计算弧度并返回

\_getRadian: function(point)

{

this.\_radian = Math.PI / 180 \* this.\_getAngle(point);

return this.\_radian;

},

//计算角度并返回

\_getAngle: function(point)

{

var pos = this.node.getPosition();

this.\_angle = Math.atan2(point.y - pos.y, point.x - pos.x) \* (180/Math.PI);

return this.\_angle;

},

//设置实际速度

\_setSpeed: function(point)

{

//触摸点和遥控杆中心的距离

var distance = this.\_getDistance(point, this.node.getPosition());

//如果半径

if(distance < this.\_radius)

{

this.\_speed = this.\_speed1;

}

else

{

this.\_speed = this.\_speed2;

}

},

\_touchStartEvent: function(event) {

// 获取触摸位置的世界坐标转换成圆圈的相对坐标（以圆圈的锚点为基准）

var touchPos = this.node.convertToNodeSpaceAR(event.getLocation());

//触摸点与圆圈中心的距离

var distance = this.\_getDistance(touchPos,cc.v2(0,0));

//圆圈半径

var radius = this.node.width / 2;

// 记录摇杆位置，给touch move使用

this.\_stickPos = touchPos;

var posX = this.node.getPosition().x + touchPos.x;

var posY = this.node.getPosition().y + touchPos.y;

//手指在圆圈内触摸,控杆跟随触摸点

if(radius > distance)

{

this.dot.setPosition(cc.v2(posX, posY));

return true;

}

return false;

},

\_touchMoveEvent: function(event){

var touchPos = this.node.convertToNodeSpaceAR(event.getLocation());

var distance = this.\_getDistance(touchPos,cc.v2(0,0));

var radius = this.node.width / 2;

// 由于摇杆的postion是以父节点为锚点，所以定位要加上ring和dot当前的位置(stickX,stickY)

var posX = this.node.getPosition().x + touchPos.x;

var posY = this.node.getPosition().y + touchPos.y;

if(radius > distance)

{

this.dot.setPosition(cc.v2(posX, posY));

}

else

{

//控杆永远保持在圈内，并在圈内跟随触摸更新角度

var x = this.node.getPosition().x + Math.cos(this.\_getRadian(cc.v2(posX,posY))) \* radius;

var y = this.node.getPosition().y + Math.sin(this.\_getRadian(cc.v2(posX,posY))) \* radius;

this.dot.setPosition(cc.v2(x, y));

}

//更新角度

this.\_getAngle(cc.v2(posX,posY));

//设置实际速度

this.\_setSpeed(cc.v2(posX,posY));

},

\_touchEndEvent: function(){

this.dot.setPosition(this.node.getPosition());

this.\_speed = 0;

},

});

module.exports = {

TouchType : cc.Enum({

DEFAULT: 0,

FOLLOW: 1,

}),

DirectionType : cc.Enum({

FOUR: 4,

EIGHT: 8,

ALL: 0,

}),

};

var Lfunction = cc.Class({

// 成员变量

name : "",

age : 0,

ctor () {

},

statics: {

\_instance: null

},

randomNum: function(minNum,maxNum){

switch(arguments.length){

case 1:

return parseInt(Math.random()\*minNum+1,10);

break;

case 2:

return parseInt(Math.random()\*(maxNum-minNum+1)+minNum,10);

break;

default:

return 0;

break;

}

},

updatePosRigidNode: function(node, pos){

},

getSliptNum: function(num){

var a = parseInt(num % 10); // 个位数

var b = parseInt((num % 100) / 10); // 十位数

return {ten:a, signle:b}

},

});

Lfunction.\_instance = new Lfunction();

module.exports = Lfunction;// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

cc.Class({

extends: cc.Component,

properties: {

// foo: {

// // ATTRIBUTES:

// default: null, // The default value will be used only when the component attaching

// // to a node for the first time

// type: cc.SpriteFrame, // optional, default is typeof default

// serializable: true, // optional, default is true

// },

// bar: {

// get () {

// return this.\_bar;

// },

// set (value) {

// this.\_bar = value;

// }

// },

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

onCollisionEnter: function (other, self) {

// 碰撞系统会计算出碰撞组件在世界坐标系下的相关的值，并放到 world 这个属性里面

var world = self.world;

// 碰撞组件的 aabb 碰撞框

var aabb = world.aabb;

// 节点碰撞前上一帧 aabb 碰撞框的位置

var preAabb = world.preAabb;

// 碰撞框的世界矩阵

var t = world.transform;

// 以下属性为圆形碰撞组件特有属性

var r = world.radius;

var p = world.position;

// 以下属性为 矩形 和 多边形 碰撞组件特有属性

var ps = world.points;

},

onCollisionStay: function (other, self) {

this.node.parent.opacity = 100;

},

/\*\*

\* 当碰撞结束后调用

\* @param {Collider} other 产生碰撞的另一个碰撞组件

\* @param {Collider} self 产生碰撞的自身的碰撞组件

\*/

onCollisionExit: function (other, self) {

this.node.parent.opacity = 255;

}

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var UIinstallTool = require('UI\_install\_tool');

cc.Class({

extends: cc.Component,

properties: {

\_sign:{

default: null,

type: cc.Node,

},

\_is\_move:false,

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

},

start () {

this.\_sign = cc.director.getScene().getChildByName('Canvas').getChildByName('root\_node').getChildByName('player');

this.\_bind\_node = UIinstallTool.\_instance.AddShadow()

},

setMoveEnd: function () {

this.\_is\_move = false;

},

killMe: function () {

cc.director.GlobalEvent.emit('mos\_dead', {mos\_id:1});

this.\_bind\_node.DeleteMe();

this.node.removeFromParent();

},

update (dt) {

this.\_bind\_node.Update(this.node)

if(false == this.\_is\_move && null != this.\_sign){

this.\_is\_move = true;

var callback = cc.callFunc(this.setMoveEnd, this);

var actionTo = cc.moveTo(2, cc.v2(this.\_sign.x, this.\_sign.y));

this.node.runAction(cc.sequence(actionTo, callback))

}

},

onBeginContact: function (contact, selfCollider, otherCollider) {

// log("monster onBeginContact---------------")

},

onCollisionEnter: function (other, self) {

if (other.tag == 1000){

this.killMe()

}

},

onCollisionStay: function (other, self) {

},

onCollisionExit: function (other, self) {

}

});// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var Lfunction = require('Lfunction');

cc.Class({

extends: cc.Component,

properties: {

moster: cc.Prefab, //预制件

time\_limit: 0,

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

window.MosterManger = this

},

// start () {},

loadMoster: function(mos\_id){

let node = cc.instantiate(this.moster);

node.parent = this.node.getChildByName("root\_node").getChildByName("moster\_layer");

return node

},

});

cc.Class({

extends: cc.Component,

properties: {

size: cc.size(0, 0),

mouseJoint: true

},

// use this for initialization

onLoad: function () {

let width = this.size.width || this.node.width;

let height = this.size.height || this.node.height;

let node = new cc.Node();

let body = node.addComponent(cc.RigidBody);

body.type = cc.RigidBodyType.Static;

if (this.mouseJoint) {

// add mouse joint

let joint = node.addComponent(cc.MouseJoint);

joint.mouseRegion = this.node;

}

this.\_addBound(node, 0, height/2, width, 20);

this.\_addBound(node, 0, -height/2, width, 20);

this.\_addBound(node, -width/2, 0, 20, height);

this.\_addBound(node, width/2, 0, 20, height);

node.parent = this.node;

},

\_addBound (node, x, y, width, height) {

let collider = node.addComponent(cc.PhysicsBoxCollider);

collider.offset.x = x;

collider.offset.y = y;

collider.size.width = width;

collider.size.height = height;

}

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var Lfunction = require('Lfunction');

var SkillManger = require('skill\_manger');

var UIinstallTool = require('UI\_install\_tool');

var HUD\_manger = require('HUD\_manger');

var role\_config = require('role\_config');

var part\_config = require('part\_config');

var COLOR = require('color\_def');

var OnFileData = require('onfile\_data');

cc.Class({

extends: cc.Component,

properties: {

// 主角跳跃高度

jumpHeight: 0,

// 主角跳跃持续时间

jumpDuration: 0,

// 最大移动速度

maxMoveSpeed: 0,

//属性

HP: 0,

MP: 0,

EXP: 0,

level: 1,

weapon\_id: 1,

//当前动画

\_animate\_state: "stand",

//引用技能管理器

\_SkillPlay: null,

\_weapon\_view: null,

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

var self = this;

//事件监听

cc.director.GlobalEvent.on('mos\_dead', function (data) {

self.onMosDead(data)

}, this.node);

},

start () {

if (null != OnFileData.\_instance.GetOnFileList()[1]){

this.level = OnFileData.\_instance.GetOnFileList()[1].level

HUD\_manger.\_instance.setLevel(this.level)

}

// 初始化属性

this.HP = role\_config[this.level].HP;

this.MP = role\_config[this.level].MP;

// ui 绑定

this.\_bind\_node = UIinstallTool.\_instance.AddShadow()

this.\_weapon\_view = UIinstallTool.\_instance.CreatAndBindWeapon(this.weapon\_id)

//技能释放

this.schedule(function() {

var ranpos\_x = Lfunction.\_instance.randomNum(-20, 20)

var ranpos\_y = Lfunction.\_instance.randomNum(-20, 20)

SkillManger.\_instance.fire(1, cc.v2(this.node.x + ranpos\_x, this.node.y + ranpos\_y))

}, 1);

},

update (dt) {

this.\_bind\_node.Update(this.node)

this.\_weapon\_view.Update(this.node)

},

onMosDead: function (info) {

this.UpdateInfo({EXP:1});

//等级提升调用

if (null != role\_config[this.level + 1] && this.EXP >= role\_config[this.level].EXP){

this.UpdateInfo({level: 1, HP:role\_config[this.level].HP, MP:role\_config[this.level].MP})

}

},

UpdateInfo: function (change\_attr\_list) {

for(var key in change\_attr\_list){

//更新玩家数据

var value = change\_attr\_list[key]

this[key] += value

//默认字符串

var sign = ""

if(value > 0){sign = "+"};

var tip\_str = key + sign + value

if (key == "level"){

HUD\_manger.\_instance.setLevel(this.level)

tip\_str = "lv up!:Lv." + this.level, COLOR.RGB.GREEN

}

if (key == "HP"){

if (this.HP <= 0){

this.ondead()

}

var max\_hp = role\_config[this.level].HP

if (this.HP > role\_config[this.level].HP){

this.HP = max\_hp

}

HUD\_manger.\_instance.setHpbarNum(this.HP / max\_hp)

}

//提示

window.SysTip.fireFlowTip(tip\_str, value < 0?COLOR.RGB.RED:COLOR.RGB.GREEN, this.node)

}

},

ondead: function () {

OnFileData.\_instance.SetOnFileList(1, {level: this.level, part:window.AI\_director.\_curr\_part})

this.node.removeFromParent()

//初始化死亡结算面板 计算数值改动

HUD\_manger.\_instance.openLoseView()

//存档

},

onBeginContact: function (contact, selfCollider, otherCollider) {

},

// 只在两个碰撞体结束接触时被调用一次

onEndContact: function (contact, selfCollider, otherCollider) {

},

// 每次将要处理碰撞体接触逻辑时被调用

onPreSolve: function (contact, selfCollider, otherCollider) {

},

// 每次处理完碰撞体接触逻辑时被调用

onPostSolve: function (contact, selfCollider, otherCollider) {

},

onCollisionEnter: function (other, self) {

if (other.tag == 100){

this.UpdateInfo({HP:-50});

}

},

// changeAnimate: function (str) {

// if (this.\_animate\_state ~= str){

// var anim = this.getComponent(cc.Animation);

// log("player\_" + str);

// anim.play("player\_" + str);

// };

// },

// setJumpAction: function () {

// // 跳跃上升

// var jumpUp = cc.moveBy(this.jumpDuration, cc.v2(0, this.jumpHeight)).easing(cc.easeCubicActionOut());

// // 下落

// var jumpDown = cc.moveBy(this.jumpDuration, cc.v2(0, -this.jumpHeight)).easing(cc.easeCubicActionIn());

// // 不断重复

// return cc.repeatForever(cc.sequence(jumpUp, jumpDown));

// },

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var SkillManger = cc.Class({

extends: cc.Component,

properties: {

skill\_layer: {

default: null,

type: cc.Node,

displayName: '技能层',

},

skill\_1: cc.Prefab, //预制件

},

statics: {

\_instance: null

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

SkillManger.\_instance = this;

},

start () {

},

// update (dt) {},

fire: function(skill\_id, sign){

// log("skill fire" + skill\_id)

let node = cc.instantiate(this["skill\_" + skill\_id]);

node.x = sign.x

node.y = sign.y

node.parent = this.skill\_layer;

this.scheduleOnce(function() {

// 这里的 this 指向 component

node.removeFromParent()

// log("remove skill fire")

}, 0.4);

},

});

// SkillManger.\_instance = new SkillManger();

module.exports = SkillManger;// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var COLOR = require('color\_def');

var UIinstallTool = cc.Class({

extends: cc.Component,

properties: {

light: cc.Prefab, //预制件

shadow: cc.Prefab, //预制件

item\_shadow: cc.Prefab, //预制件

lbl\_pre: cc.Prefab, //预制件

weapon: cc.Prefab, //预制件

\_shadow\_offest\_y: 0,

},

// LIFE-CYCLE CALLBACKS:

statics: {

\_instance: null

},

onLoad () {

UIinstallTool.\_instance = this;

this.\_shadow\_offest\_y = -5;

},

start () {

},

// update (dt) {},

AddPopText: function(parent, data){

//获取预制文本

var node = cc.instantiate(this.lbl\_pre);

// node.getComponent("cc.Label").spriteFrame.getTexture().setFilters(cc.Texture2D.Filter.NEAREST, cc.Texture2D.Filter.NEAREST)

node.getComponent("cc.Label").string = data.str

null == data.color?node.color = COLOR.RGB.WITE:node.color = data.color

if (null != data.anchor\_x){

node.anchorX = data.anchor\_x

}

if (null != data.x || null != data.y){

node.x = data.x

node.y = data.y

}

if (null != data.sign\_node){

node.x = data.sign\_node.x

node.y = data.sign\_node.y + 5

}

if (null != data.font\_size){

node.getComponent("cc.Label").fontSize = data.font\_size

}

node.parent = parent

return node

},

AddShadow: function(){

let shadow = cc.instantiate(this.shadow);

shadow.parent = cc.director.getScene().getChildByName('Canvas').getChildByName('root\_node').getChildByName('shadow\_layer');

return {

shadow: shadow,

DeleteMe: function () {

shadow.removeFromParent();

},

Update: function (bind\_parent\_node) {

shadow.x = bind\_parent\_node.x;

shadow.y = bind\_parent\_node.y - 6;

},

}

},

CreatAndBindWeapon: function(weapon\_id){

let node = cc.instantiate(this.weapon);

node.parent = cc.director.getScene().getChildByName('Canvas').getChildByName('root\_node').getChildByName('weapon\_layer');

return {

node: node,

DeleteMe: function () {

node.removeFromParent();

},

Change: function (weapon\_id) {

},

Update: function (bind\_parent\_node) {

node.x = bind\_parent\_node.x;

node.y = bind\_parent\_node.y;

},

}

},

});

module.exports = UIinstallTool;

module.exports = {

RGB: {

GREEN: cc.color(130,238,168,255),

WITE: cc.color(255,255,255,255),

RED: cc.color(229,110,110,255),

},

};

var NodeTools = cc.Class({

// 成员变量

name : "",

age : 0,

ctor () {

},

statics: {

\_instance: null

},

flashNode: function(node, in\_time, out\_time){

var action1= cc.fadeOut(in\_time)

var action2= cc.fadeIn(out\_time)

var sequence = cc.sequence(action1, action2)

// var action = cc.repeatForever(sequence)

node.runAction(sequence.repeatForever())

},

});

NodeTools.\_instance = new NodeTools();

module.exports = NodeTools;module.exports = {

//物理碰撞 碰撞使用同一枚举

COLLID: {

player: 1,

moster: 100,

skill: 1000,

item: 1200,

},

};

//建筑物配置

//不同建筑可能有不同的属性 需根据属性刷新相关逻辑

//特殊属性可触发特定事件

module.exports = {

//生成点

//1 某些怪物生成点 可提供怪物加成(value \*= config)

broth: {

1:{},

2:{HP: 1.5, SPEED: 1.2},

},

//建筑结构

structure: {

1:{},

},

}module.exports = {

1 : {name:"1级回血包", attr\_list:{HP: 100,}, dis\_time: 10},

2 : {name:"2级回血包", attr\_list:{HP: 200,}, dis\_time: 10},

3 : {name:"3级回血包", attr\_list:{HP: 200},},

4 : {name:"4级回血包", attr\_list:{HP: 200},},

5 : {name:"4级回血包", attr\_list:{HP: 200},},

6 : {name:"4级回血包", attr\_list:{HP: 200},},

7 : {name:"4级回血包", attr\_list:{HP: 200},},

8 : {name:"4级回血包", attr\_list:{HP: 200},},

9 : {name:"4级回血包", attr\_list:{HP: 200},},

10 : {name:"4级回血包", attr\_list:{HP: 200},},

11 : {name:"1级回蓝包", attr\_list:{MP: 10,}, dis\_time: 20},

20 : {name:"1级空投包", attr\_list:{MP: 10, HP: 10, EXP: 10,}, dis\_time: 20},

};//关卡配置

// 每关怪物数量固定 道具数量有上限 频率由ai导演控制

module.exports = {

// 怪物数量 空投包数量 回血包数量 回蓝包数量

1 : {mos\_num: 5, item\_list:{ 20: {max\_num:10}, 1: {max\_num:10}, 11:{max\_num:1}}},

2 : {mos\_num: 10, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 20}, 11:{max\_num: 1}}},

3 : {mos\_num: 15, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 30}, 11:{max\_num: 1}}},

4 : {mos\_num: 20, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 40}, 11:{max\_num: 1}}},

5 : {mos\_num: 30, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

6 : {mos\_num: 35, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

7 : {mos\_num: 40, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

8 : {mos\_num: 45, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

9 : {mos\_num: 50, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

10 : {mos\_num: 55, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

11 : {mos\_num: 60, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

12 : {mos\_num: 65, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

13 : {mos\_num: 70, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

14 : {mos\_num: 80, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

15 : {mos\_num: 90, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

16 : {mos\_num: 100, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

17 : {mos\_num: 110, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

18 : {mos\_num: 130, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

19 : {mos\_num: 150, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

20 : {mos\_num: 170, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

21 : {mos\_num: 200, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

22 : {mos\_num: 230, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

23 : {mos\_num: 260, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

24 : {mos\_num: 290, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

25 : {mos\_num: 320, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

26 : {mos\_num: 350, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

27 : {mos\_num: 380, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

28 : {mos\_num: 410, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

29 : {mos\_num: 440, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

30 : {mos\_num: 500, item\_list:{ 20: {max\_num:10}, 1:{max\_num: 50}, 11:{max\_num: 1}}},

};module.exports = {

1 : {HP: 100, MP: 10, EXP: 10},

2 : {HP: 200, MP: 10, EXP: 20},

3 : {HP: 300, MP: 10, EXP: 40},

4 : {HP: 400, MP: 10, EXP: 70},

5 : {HP: 500, MP: 10, EXP: 100},

6 : {HP: 600, MP: 10, EXP: 150},

};var OnFileData = cc.Class({

//存档数据列表

properties: {

onflie\_list:[],

},

ctor () {

if (null != cc.sys.localStorage.getItem('onfileList')){

// log("onfileList" + cc.sys.localStorage.getItem('onfileList'))

this.onflie\_list = JSON.parse(cc.sys.localStorage.getItem('onfileList'));

}

},

statics: {

\_instance: null

},

GetOnFileList: function(data, name){

return this.onflie\_list

},

SetOnFileList: function(id, data){

// log("SetOnFileList" + id + data.level)

this.onflie\_list[id] = data

},

SaveOnFile: function(){

// log("SaveOnFile" + JSON.stringify(this.onflie\_list))

cc.sys.localStorage.setItem('onfileList', JSON.stringify(this.onflie\_list));

},

});

OnFileData.\_instance = new OnFileData();

module.exports = OnFileData;// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

cc.Class({

extends: cc.Component,

properties: {

id:0,

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

getConfig: function () {

if (null == bulid\_config.broth[this.id]){

// log("not find bulid-broth config id: " + this.id)

}

return bulid\_config.broth[this.id]

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var bulid\_config = require('bulid\_config');

cc.Class({

extends: cc.Component,

properties: {

id:0,

player: {

default: null,

type: cc.Node,

displayName: '玩家',

},

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

getConfig: function () {

if (null == bulid\_config.structure[this.id]){

// log("not find bulid-structure config id: " + this.id)

}

return bulid\_config.structure[this.id]

},

onBeginContact: function (contact, selfCollider, otherCollider) {

// log("onBeginContact---------------")

// this.node.opacity = 100;

},

// 只在两个碰撞体结束接触时被调用一次

onEndContact: function (contact, selfCollider, otherCollider) {

// this.node.opacity = 255;

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var HUD\_manger = require('HUD\_manger');

cc.Class({

extends: cc.Component,

properties: {

// foo: {

// // ATTRIBUTES:

// default: null, // The default value will be used only when the component attaching

// // to a node for the first time

// type: cc.SpriteFrame, // optional, default is typeof default

// serializable: true, // optional, default is true

// },

// bar: {

// get () {

// return this.\_bar;

// },

// set (value) {

// this.\_bar = value;

// }

// },

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

onClickOk: function (data) {

var self = this;

this.node.opacity = 0

self.node.removeFromParent()

HUD\_manger.\_instance.\_achive\_view = null;

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var OnFileData = require('onfile\_data');

cc.Class({

extends: cc.Component,

properties: {

// foo: {

// // ATTRIBUTES:

// default: null, // The default value will be used only when the component attaching

// // to a node for the first time

// type: cc.SpriteFrame, // optional, default is typeof default

// serializable: true, // optional, default is true

// },

// bar: {

// get () {

// return this.\_bar;

// },

// set (value) {

// this.\_bar = value;

// }

// },

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

//

onGameOver: function () {

// cc.audioEngine.playEffect(self.audio, false, 0.1);

OnFileData.\_instance.SaveOnFile(1)

cc.director.loadScene('start');

}

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var HUD\_manger = require('HUD\_manger');

cc.Class({

extends: cc.Component,

properties: {

// foo: {

// // ATTRIBUTES:

// default: null, // The default value will be used only when the component attaching

// // to a node for the first time

// type: cc.SpriteFrame, // optional, default is typeof default

// serializable: true, // optional, default is true

// },

// bar: {

// get () {

// return this.\_bar;

// },

// set (value) {

// this.\_bar = value;

// }

// },

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

},

// update (dt) {},

onClickOk: function (data) {

var self = this;

this.node.opacity = 0

self.node.removeFromParent()

HUD\_manger.\_instance.\_rank\_view = null;

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var HUD\_manger = require('HUD\_manger');

var OnFileData = require('onfile\_data');

cc.Class({

extends: cc.Component,

properties: {

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

},

start () {

},

// update (dt) {},

onBgmSoundSlider: function (slider, data) {

},

onEffSoundSlider: function (slider, data) {

},

onClickOk: function (data) {

var self = this;

this.node.opacity = 0

self.node.removeFromParent()

HUD\_manger.\_instance.\_seting\_view = null;

},

onClickMainMenu: function (data) {

OnFileData.\_instance.SaveOnFile(1)

cc.director.loadScene('start');

},

onClickHelp: function (data) {

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var COLOR = require('color\_def');

var UIinstallTool = require('UI\_install\_tool');

cc.Class({

extends: cc.Component,

properties: {

pop\_hegiht: 20,

pop\_time: 1,

pop\_time\_limt: 0.1,

\_cache\_txts\_: [],

\_is\_act: false,

\_next\_fire\_time: 0,

},

// LIFE-CYCLE CALLBACKS:

// onLoad () {},

start () {

window.SysTip = this

},

fireTip: function(str, color, x, y){

//延时调用 在重载场景时 会引发错误 成员变量 可能会被清空

this.\_cache\_txts\_.push({str:str, color:color, x: x, y:y})

},

fireFlowTip: function(str, color, sign\_node){

//延时调用 在重载场景时 会引发错误 成员变量 可能会被清空

this.\_cache\_txts\_.push({str:str, color:color, sign\_node: sign\_node, font\_size: 8, show\_style:"boom"})

},

fireSystimTip: function(str, color){

//延时调用 在重载场景时 会引发错误 成员变量 可能会被清空

this.\_cache\_txts\_.push({str:str, color:color, anchor\_x: 0,x: -105, y:-10})

},

update (dt) {

if (window.NowTime >= this.\_next\_fire\_time && this.\_cache\_txts\_.length > 0){

this.\_next\_fire\_time = window.NowTime + this.pop\_time\_limt

var data = this.\_cache\_txts\_.shift()

var node = UIinstallTool.\_instance.AddPopText(this.node, data)

//动作

var callback = cc.callFunc(function () {

node.removeFromParent()

});

var comp\_action = null

if(null == data.show\_style){

var actionTo = cc.moveTo(this.pop\_time, cc.v2(node.x, node.y + this.pop\_hegiht));

var fiadein = cc.fadeOut(this.pop\_time)

comp\_action = cc.sequence(cc.spawn(actionTo, fiadein), callback)

}

else if(data.show\_style == "boom"){

var actionTo = cc.moveTo(this.pop\_time, cc.v2(node.x, node.y + this.pop\_hegiht));

var fiadein = cc.fadeOut(this.pop\_time)

var scaleTo = cc.scaleTo(0.5, 0.8)

comp\_action = cc.sequence(cc.spawn(actionTo, fiadein, scaleTo), callback)

}

node.runAction(comp\_action)

}

},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var NodeTools = require('node\_tools');

cc.Class({

extends: cc.Component,

properties: {

audio: {

default: null,

type: cc.AudioClip,

},

title\_1: {

default: null,

type: cc.Node,

},

title\_2: {

default: null,

type: cc.Node,

},

title\_3: {

default: null,

type: cc.Node,

},

save\_create\_veiw: {

default: null,

type: cc.Node,

},

tip: {

default: null,

type: cc.Label,

},

eff\_1: cc.Prefab, //预制件

eff\_2: cc.Prefab, //预制件

eff\_3: cc.Prefab, //预制件

can\_ignore\_animate:true,

},

onLoad () {

cc.view.enableAntiAlias(false);

// 存档新建界面

this.save\_create\_veiw.opacity = 0

var self = this;

var touch\_callback = function () {

// can\_ignore\_animate 是否可忽略动画

// log("touch\_callback=========")

if (self.can\_ignore\_animate || self.show\_end){

self.node.off(cc.Node.EventType.TOUCH\_START, self.on\_touch\_moved ,self)

self.node.runAction(cc.fadeOut(1))

self.scheduleOnce(function () {

self.save\_create\_veiw.opacity = 255

}, 1)

// self.save\_create\_veiw.runAction(cc.fadeIn(3))

}

}

self.node.on(cc.Node.EventType.TOUCH\_START, touch\_callback);

cc.director.preloadScene("game", function () {

// cc.log("Next scene preloaded");

});

cc.audioEngine.play(self.audio, true, 1);

},

palyeLoadAnimate: function () {

this.title\_1.opacity = 0

this.title\_2.opacity = 0

this.title\_3.opacity = 0

this.title\_1.runAction(cc.fadeIn(1))

this.scheduleOnce(function () {

var node = cc.instantiate(this.eff\_1)

node.parent = this.title\_1

var anim = node.getComponent(cc.Animation);

var animState = anim.play("1")

animState.repeatCount = 1;

var self = this

anim.on('stop',function () {

node.removeFromParent()

var eff\_2 = cc.instantiate(self.eff\_2)

eff\_2.x = -50

eff\_2.y = 20

eff\_2.parent = self.title\_1

var anim = eff\_2.getComponent(cc.Animation);

var animState = anim.play("2")

var eff\_3 = cc.instantiate(self.eff\_3)

eff\_3.x = 50

eff\_3.y = -20

eff\_3.parent = self.title\_1

var anim = eff\_3.getComponent(cc.Animation);

var animState = anim.play("3")

});

this.title\_2.runAction(cc.fadeIn(1))

}, 1)

this.scheduleOnce(function () {

this.title\_3.runAction(cc.fadeIn(1))

}, 2)

this.scheduleOnce(function () {

this.show\_end = true

}, 3)

},

start () {

this.palyeLoadAnimate()

NodeTools.\_instance.flashNode(this.tip.node, 1, 1)

},

// update (dt) {},

});

// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

var OnFileData = require('onfile\_data');

cc.Class({

extends: cc.Component,

properties: {

btn\_go: {

default: null,

type: cc.Node,

},

audio: {

default: null,

type: cc.AudioClip,

},

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

},

start () {

// log("OnFileData============" + OnFileData.\_instance.GetOnFileList())

},

// update (dt) {},

//

//

onGameStart: function () {

cc.audioEngine.playEffect(this.audio, false, 0.1);

//不知为啥 要在切场景前 设置为true

//然后新场景重新 调用cc.view.enableAntiAlias(false) 才能全部设置为不抗锯齿

//否则有部分资源为抗锯齿(动态加载？？)

cc.view.enableAntiAlias(true);

cc.director.loadScene('game');

}

});

//获取游戏状态 调整游戏难度

var ItemManger = require('item\_manger');

var part\_config = require('part\_config');

var Lfunction = require('Lfunction');

var OnFileData = require('onfile\_data');

cc.Class({

extends: cc.Component,

properties: {

//当前关卡数

\_curr\_part:1,

//玩家击杀怪物数

\_kill\_monster\_num:0,

//场景道具信息

\_item\_info:null,

\_item\_arry:[],

time\_limit: 1,

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

//事件监听

var self = this;

cc.director.GlobalEvent.on('mos\_dead', function (data) {

self.onMosDead(data)

}, this.node);

self.\_item\_info = new Map();

this.schedule(function() {

if (part\_config[self.\_curr\_part]){

for(var item\_id in part\_config[self.\_curr\_part].item\_list){

if (null == self.\_item\_info[item\_id]){

self.\_item\_info[item\_id] = 0

}

var data = part\_config[self.\_curr\_part].item\_list[item\_id]

if (data.max\_num > self.\_item\_info[item\_id]){

var item = ItemManger.\_instance.fire(item\_id)

item.x = Lfunction.\_instance.randomNum(-70, 70)

item.y = Lfunction.\_instance.randomNum(-50, -150)

self.\_item\_info[item\_id] += 1

}

}

}

}, 3);

},

start () {

var onfile = OnFileData.\_instance.GetOnFileList()[1]

if (null != onfile && null != onfile.part){

this.\_curr\_part = OnFileData.\_instance.GetOnFileList()[1].part

cc.director.GlobalEvent.emit('part\_change', {curr\_part: this.\_curr\_part});

}

window.AI\_director = this

},

//ai导演逻辑 统一管理更新 manger不参与更新

update (dt) {

//怪物生成

this.time\_limit = this.time\_limit + dt

if (this.time\_limit > 1){

var node = window.MosterManger.loadMoster();

var random\_place = Lfunction.\_instance.randomNum(1, 3)

node.x = window.MapManger.getBulidLayer().getChildByName("broth\_" + random\_place).x

node.y = window.MapManger.getBulidLayer().getChildByName("broth\_" + random\_place).y

this.time\_limit = 0

}

},

onMosDead: function (value) {

this.\_kill\_monster\_num += 1;

// console.log(this.\_curr\_part)

if (part\_config[this.\_curr\_part] && this.\_kill\_monster\_num >= part\_config[this.\_curr\_part].mos\_num){

this.\_curr\_part += 1;

cc.director.GlobalEvent.emit('part\_change', {curr\_part: this.\_curr\_part});

}

},

puaseGame: function(){

this.game\_state = "puase"

},

});// Learn cc.Class:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/class.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/class.html

// Learn Attribute:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/reference/attributes.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/reference/attributes.html

// Learn life-cycle callbacks:

// - [Chinese] http://docs.cocos.com/creator/manual/zh/scripting/life-cycle-callbacks.html

// - [English] http://www.cocos2d-x.org/docs/creator/en/scripting/life-cycle-callbacks.html

cc.Class({

extends: cc.Component,

properties: {

map\_bg: {

default: null,

type: cc.Node,

},

map\_1:cc.Prefab,

\_curr\_map\_id: 1,

\_curr\_map\_node: null,

},

// LIFE-CYCLE CALLBACKS:

onLoad () {

window.MapManger = this

// this.map\_bg.getComponent(cc.Sprite).spriteFrame.getTexture().setFilters(cc.Texture2D.Filter.NEAREST, cc.Texture2D.Filter.NEAREST)

},

start () {

this.\_curr\_map\_node = cc.instantiate(this.map\_1);

this.\_curr\_map\_node.parent = this.node

},

// update (dt) {},

getBulidLayer:function () {

return this.\_curr\_map\_node

}

});