|  |
| --- |
| 成绩 |
| 批阅 |
| 日期 |

**台州学院**

**电子与信息工程学院实验报告**

班级 16计应2 学号 1629210089 姓名 蔡大航

同组 黄睿博、李武

实验课程： 游戏开发

实验项目：  **综合项目设计**

实验日期： 2019 年 6 月 11 日

主要内容（参考）

预习简要（可选） 四、实验内容、方法、步骤

一、实验目的和任务 五、实验数据记录与处理

二、实验原理 六、实验结果分析、思考、心得

三、实验器材 七、原始数据

实验内容

制作一款扁平风格休闲类小游戏

游戏名：六角消除

游戏规则：随机生成21种不同形状的不同颜色的六角小方块组合，每次出现三种形状的方块组合，通过拖动放置在绘制的棋盘上，当小方块在棋盘上连成一行或者一列的时候进行消除，消除后的位置仍可以继续放置符合形状的小方块，每次放置小方块都会加分，消除时双倍加分，分数会一直累加，并且会记录最高分，初始最高分为0，当棋盘无法继续放置当前形状的小方块时游戏结束，结束之后可以选择再次游戏挑战自己的最高分或者退出游戏。



图1游戏界面

这是一款二维的小游戏，所需的技术主要有

1、游戏地图、场景创建；

2、添加玩家（就是用户登录，也可以不登录以游客进入）；

3、单点触摸屏幕，控制方块运动；

4、判断游戏规则的逻辑；

5、随机生成与循环；

6、界面美化；

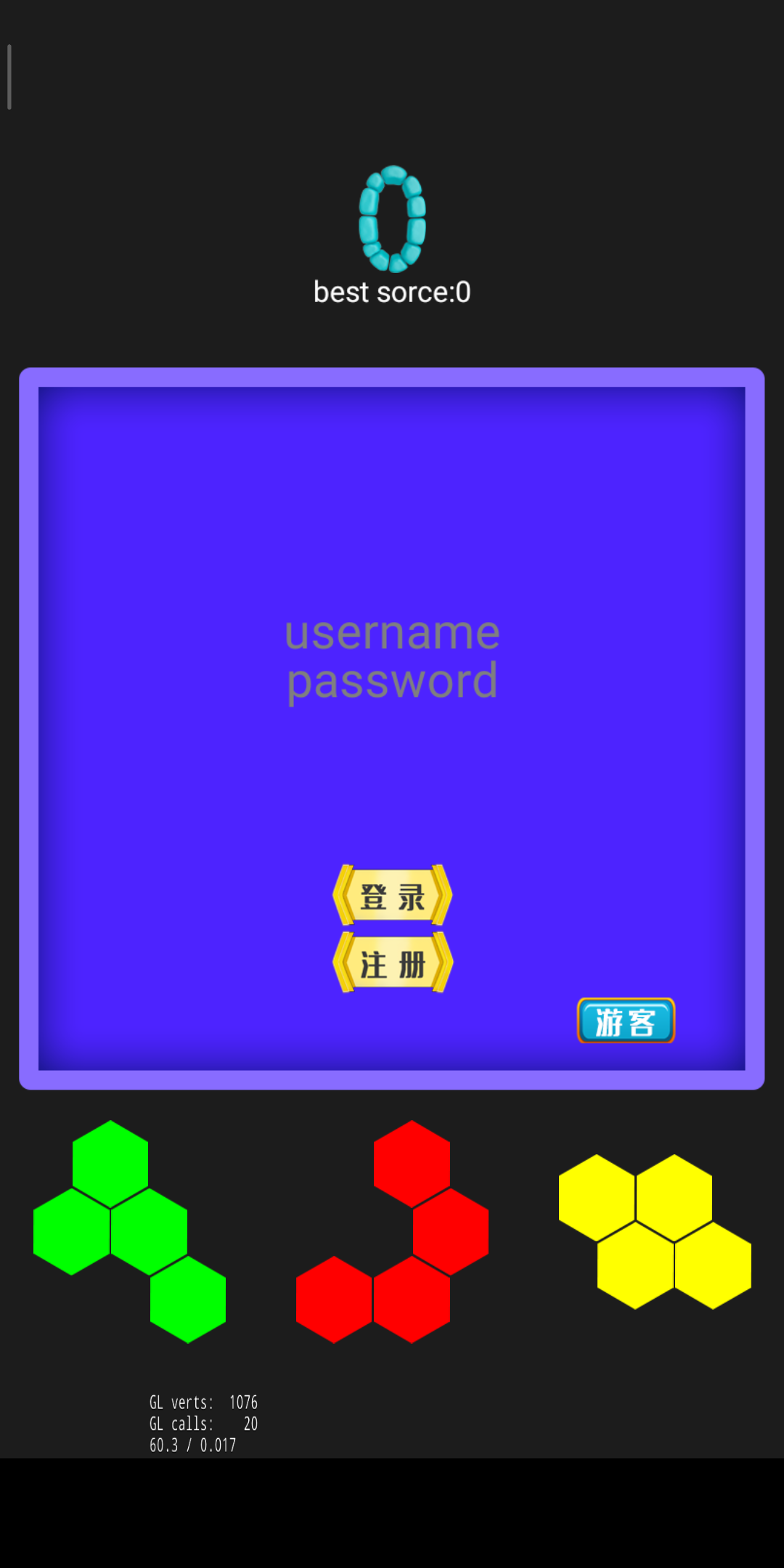
7、粒子系统；

8、音频的制作和处理，音频在游戏中的交互；

9、进行网络连接注册账户，信息以json形式传递

数据记录

登录界面 开始游戏



游戏结束





1、创建游戏场景，直接加入了全局背景音乐

Scene\* GameScene::createScene()

{

auto scene = Scene::create();

auto layer = GameScene::create();

scene->addChild(layer);

return scene;

}

auto audio = SimpleAudioEngine::getInstance();

audio->playBackgroundMusic("sound/bgmusictwo.mp3", true);

2、绘制棋盘背景，这里只用了一张小方块的图片来拼合出整个棋盘背景

m\_color = Color3B(79, 79, 79);

for (int i = 0; i < 4; ++i)

{

m\_shadowS[i] = Sprite::create("shadow.png");

m\_shadowS[i]->retain();

m\_shadowV[i] = 0;

}

for (int i = 0; i < gMapLineM; ++i)

{

int lineLength = gMapLineM - abs(4 - i);

int j = 0;

if (i > 4){

j = i - 4;

}

int rowMaxI = j + lineLength;

for (int k = 0; k < j; ++k)

m\_Map[i][k] = MAP\_NOUSE;

for ( ; j < rowMaxI; ++j)

m\_Map[i][j] = MAP\_EMPTY;

for ( ; j < gMapLineM; ++j)

m\_Map[i][j] = MAP\_NOUSE;

}

float disY = gSpriteH -(gSpriteW \* sqrt(3) / 6)

+ gSpace / sqrt(3) \* 2;

float disOneLineX = gSpriteW + gSpace;

float disRowLineX = disOneLineX / 2;

float flagPosY = winSize.height - (winSize.height - 8 \* disY) / 2;

float flagPosX = (winSize.width - 8 \* disOneLineX) / 2;

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

float firstBlockX;

if (i < 4)

firstBlockX = flagPosX + (4 - i) \* disRowLineX;

else{

firstBlockX = flagPosX + (i - 4) \* (disRowLineX - disOneLineX);

}

for (int j = 0; j < gMapRowM; ++j){

if (MAP\_NOUSE != m\_Map[i][j])

{

auto block = Block::create();

block->setLRIndex(i, j);

block->setSpriteColor(m\_color);

block->setPosition(Point(firstBlockX + j \* disOneLineX, flagPosY - i \* disY));

addChild(block);

m\_blocks[i][j] = block;

}

else

{

m\_blocks[i][j] = NULL;

}

}

}

return true;

}

3、随机出21种不同组合的小方块并且设置它们的出现位置

for (int i = 0; i < 21; ++i)m\_currentShapes[i] = new BlockShape();

float difH = gSpriteW \* sqrt(3) / 6;

float disY = gSpriteH - difH + gSpace;

float disOneLineX = gSpriteW + gSpace;

float ySpace = (winSize.height - 8 \* disY) / 2;

m\_currentShapes[1]->setOriPos(winSize.width / 2, ySpace / 2, true);

m\_currentShapes[0]->setOriPos((winSize.width / 2 - gSpriteH \* 3), ySpace / 2, true);

m\_currentShapes[2]->setOriPos((winSize.width / 2 + gSpriteH \* 3), ySpace / 2, true);

addChild(m\_currentShapes[0]);

addChild(m\_currentShapes[1]);

addChild(m\_currentShapes[2]);

随机函数和颜色

BlockShape::BlockShape()

{

int shape =rand()%21;

int colorid =rand()%7;

m\_colors[0] = Color3B(255, 0, 0);

m\_colors[1] = Color3B(255, 128, 0);

m\_colors[2] = Color3B(255, 255, 0);

m\_colors[3] = Color3B(0, 255, 0);

m\_colors[4] = Color3B(255, 52, 179);

m\_colors[5] = Color3B(0, 0, 255);

m\_colors[6] = Color3B(128, 0, 255);

m\_shape = shape;

m\_color = m\_colors[colorid];

for (int i = 0; i < 4; ++i)

{

m\_blocks[i] = (Block\*)Block::create();

m\_blocks[i]->retain();

}

m\_space = m\_dspace;

setAnchorPoint(Point(0.5, 0.5));

if (shape < 0 || shape > 20)

{

shape = 0;

}

if (shape == 0){

addChild(m\_blocks[0]);

}

else{

for (int i = 0; i < 4; ++i)

addChild(m\_blocks[i]);

}

generateShape();

runScaleAction();

addTouchEvent();

}

4、触摸事件，将小方块放入棋盘

void BlockShape::addTouchEvent()

{

auto listener = EventListenerTouchOneByOne::create();

listener->setSwallowTouches(true);

listener->onTouchBegan = [&](Touch\* touch, Event\* event){

auto target = event->getCurrentTarget();

sTarget = (BlockShape\*)target;

auto pos = target->convertToNodeSpace(touch->getLocation());

auto size = target->getContentSize();

Rect rect = Rect(0, 0, size.width, size.height);

if (rect.containsPoint(pos))

{

BlockShape::sTouchBegan = true;

BlockShape::sTouchEnd = false;

auto audio = SimpleAudioEngine::getInstance();

audio->playEffect("sound/fangru.wav", false, 1.0f, 0.0f, 1.0f);

plusSpace(5);

auto scale = ScaleTo::create(0, 0.9);

scale->setTag(2);

target->stopActionByTag(2);

target->runAction(scale);

target->setAnchorPoint(Point(0.4, 0.3));

return true;

}

return false;

};

listener->onTouchMoved = [](Touch\* touch, Event\* event){

auto target = event->getCurrentTarget();

auto delta = touch->getDelta();

auto pos = target->getPosition();

target->setPosition(Point(pos.x + delta.x, pos.y + delta.y));

};

listener->onTouchEnded = [&](Touch\* touch, Event\* event){

auto target = event->getCurrentTarget();

BlockShape::sTouchEnd = true;

BlockShape::sTouchBegan = false;

target->setAnchorPoint(Point(0.5, 0.5));

auto moveback = MoveTo::create(0.3, Point(m\_ori PosX, m\_oriPosY));

auto scaleback = ScaleTo::create(0.3, 0.7);

auto actions = Spawn::create(moveback, scaleback, NULL);

actions->setTag(1);

target->stopActionByTag(1);

target->runAction(actions);

plusSpace(0);

};

\_eventDispatcher->addEventListenerWithSceneGraphPriority(listener, this);

}

5、满足条件时消除

void GameScene::fullLineClean()

{

int fullLine = 0;

int linef[9] = { 0 };

int rowf[9] = { 0 };

int rowff[9] = { 0 };

auto mapv = m\_mapLayer->m\_Map;

auto mapbs = m\_mapLayer->m\_blocks;

for(int i = 0; i <= 8; ++i)

{

if (0 == linef[i]) continue;

for (int j = 0; j <= 8; ++j)

{

if (MAP\_NOUSE == mapv[i][j]) continue;

mapbs[i][j]->setSpriteColor(Color3B(255, 255, 255));

auto delayTF = 0.001;

auto actionTime = 0.1;

auto fadeOutA = FadeOut::create(0.5);

auto delayT = DelayTime::create(actionTime \* j + delayTF \* (8 - j));

auto mapColor = m\_mapLayer->m\_color;

auto cleanAction = Sequence::create(delayT, fadeOutA, CallFunc::create([=](){

mapbs[i][j]->setSpriteColor(mapColor);

}), NULL);

cleanAction->setTag(1);

mapbs[i][j]->stopActionByTag(1);

mapbs[i][j]->runAction(cleanAction);

mapv[i][j] = MAP\_EMPTY;

test(mapbs[i][j]->getPosition());

}

}

// 消除列 上三角

for (int i = 4; i >= 0; --i)

{

if (0 == rowf[i]) continue;

for (int j = i, k = 0; j <= 8, k <= 8 - i; ++j, ++k)

{

mapbs[j][k]->setSpriteColor(Color3B(255, 255, 255));

auto delayTF = 0.001;

auto actionTime = 0.1;

auto fadeOutA = FadeOut::create(0.5);

auto delayT = DelayTime::create(actionTime \* (8 - i) + delayTF \* k);

auto mapColor = m\_mapLayer->m\_color;

auto cleanAction = Sequence::create(delayT, fadeOutA, CallFunc::create([=](){

mapbs[j][k]->setSpriteColor(mapColor);

}), NULL);

cleanAction->setTag(1);

mapbs[j][k]->stopActionByTag(1);

mapbs[j][k]->runAction(cleanAction);

mapv[j][k] = MAP\_EMPTY;

test(mapbs[j][k]->getPosition());

}

}

for (int i = 1; i <= 4; ++i)

{

if (0 == rowf[4 + i]) continue;

for (int j = i, k = 0; j <= 8, k <= 8 - i; ++j, ++k)

{

mapbs[k][j]->setSpriteColor(Color3B(255, 255, 255));

//消除动作

auto delayTF = 0.001;

auto actionTime = 0.1;

auto fadeOutA = FadeOut::create(0.5);

auto delayT = DelayTime::create(actionTime \* (8 - i) + delayTF \* k);

auto mapColor = m\_mapLayer->m\_color;

auto cleanAction = Sequence::create(delayT, fadeOutA, CallFunc::create([=](){

mapbs[k][j]->setSpriteColor(mapColor);

}), NULL);

cleanAction->setTag(1);

mapbs[k][j]->stopActionByTag(1);

mapbs[k][j]->runAction(cleanAction);

mapv[k][j] = MAP\_EMPTY;

test(mapbs[k][j]->getPosition());

}

}

for (int i = 0; i <= 8; ++i)

{

if (0 == rowff[i]) continue;

int j = 0;

int condition = 4 + i;

if (i > 4)

{

j = i - 4;

condition = 8;

}

for (j; j <= condition; ++j)

{

// 统一设置其颜色，延迟执行消除动作

mapbs[j][i]->setSpriteColor(Color3B(255, 255, 255));

//消除动作

auto delayTF = 0.001;

auto actionTime = 0.1;

auto fadeOutA = FadeOut::create(0.5);

//动作的延迟时间

auto delayT = DelayTime::create(actionTime \* 8 + delayTF \* j);

auto mapColor = m\_mapLayer->m\_color;

auto cleanAction = Sequence::create(delayT, fadeOutA, CallFunc::create([=](){

mapbs[j][i]->setSpriteColor(mapColor);

}), NULL);

cleanAction->setTag(1);

mapbs[j][i]->stopActionByTag(1);

mapbs[j][i]->runAction(cleanAction);

mapv[j][i] = MAP\_EMPTY;

test(mapbs[j][i]->getPosition());

}

}

EventCustom event(UPDATESCOR\_EVENT);

ScoreInfo info;

info.cleanCountOnce = fullLine;

info.score = fullLine \* 40 + 40;

event.setUserData(&info);

\_eventDispatcher->dispatchEvent(&event);

}

消除时使用粒子系统在消除的位置出现特效

void GameScene::test(Vec2 x)

{

auto winSize = Director::getInstance()->getWinSize();

CCParticleSystem\*m\_emitter = new CCParticleSystem();

m\_emitter = CCParticleMeteor::create();

m\_emitter->retain();

m\_emitter->setEmissionRate(800);

m\_emitter->setDuration(0.001);

m\_emitter->setSpeed(50);

m\_emitter->setEndSize(20);

m\_emitter->setTexture(CCTextureCache::sharedTextureCache()->addImage("block.png"));

m\_emitter->setPosition(x);

this->addChild(m\_emitter, 10);

m\_emitter->setAutoRemoveOnFinish(true);

}

6、设置分数与最高分

这里使用了一种提取图片中数字的方法来设置分数的样式，图片需要自己设计

m\_score = 0;

auto winSize = Director::getInstance()->getWinSize();

m\_scoreLabel = LabelAtlas::create();

m\_scoreLabel->initWithString("0", "num.png", 79, 111, 48);

m\_scoreLabel->setPosition(winSize.width / 2, winSize.height \*0.85);

m\_scoreLabel->setAnchorPoint(Vec2(0.5, 0.5));

addChild(m\_scoreLabel);



设置最高分，初始为0，判断这次的分数是否超过上一次的分数

int score = m\_statusLayer->getScore();

if (score > Global::maxscore)

Global::maxscore = score;

std::string s = "best sorce:";

std::stringstream ss;

ss << Global::maxscore;

s = s + ss.str();

CCLabelTTF \*label = CCLabelTTF::create(s.c\_str(), "arial", 30);

label->setPosition(winSize.width / 2, winSize.height \*0.8);

addChild(label, 0, "maxscore");

7、登录界面

设置登录背景

auto winSize = Director::getInstance()->getWinSize();

auto bg = Scale9Sprite::create("dialog2.png");

bg->setCapInsets(Rect(12, 12, 81, 78));

bg->setPosition(winSize.width / 2, winSize.height / 2);

addChild(bg);

设置游客登录

auto backButton = Button::create("youke1.png");

backButton->addTouchEventListener([&](Ref\* sender, Widget::TouchEventType type){

auto audio = CocosDenshion::SimpleAudioEngine::getInstance();

audio->playBackgroundMusic("sound/bgmusictwo.mp3", true);

GameScene\* parent = static\_cast<GameScene\*>(this->getParent());

if (this->getParent())

{

parent->replayGame(1);

parent->removeChild(this);

}

});

if (backButton) {

auto visibleSize = Director::getInstance()->getVisibleSize();

Vec2 origin = Director::getInstance()->getVisibleOrigin();

float x = origin.x + visibleSize.width / 2 + 240.f;

float y = origin.y + visibleSize.height / 2 - 300.f;

backButton->setPosition(Vec2(x, y));

}

this->addChild(backButton);

auto visibleSize = Director::getInstance()->getVisibleSize();

Vec2 origin = Director::getInstance()->getVisibleOrigin();

设置服务器登陆

auto loginButton = Button::create("login.png");

if (loginButton) {

float x = origin.x + visibleSize.width / 2;

float y = origin.y + visibleSize.height / 2 - 170.f;

loginButton->setPosition(Vec2(x, y));

}

loginButton->addTouchEventListener([&](Ref\* sender, Widget::TouchEventType type){

// Your code here

//对应着服务端的username和password，传入json格式的数据

std::string username = usernameInput->getStringValue();

std::string password = passwordInput->getStringValue();

std::string postData = "{\"username\":\"" + username + "\"," + "\"password\":\"" + password + "\"}";

//用post方式传输,根据ppt里面的提示，用auth

HttpRequest\* request = new HttpRequest();

request->setUrl("http://bobhuang.xyz:5001/auth");

request->setRequestType(HttpRequest::Type::POST);

request->setResponseCallback(CC\_CALLBACK\_2(LoginRegisterScene::onHttpRequestCompleted, this));

request->setRequestData(postData.c\_str(), strlen(postData.c\_str()));

//这里就是使用Cookies的地方，只要加这一句就好，可以在报头看到

//在登录的时候加上cookies就可以，这样服务端就可以记录正在使用的客户

//作为切换用户的标识

cocos2d::network::HttpClient::getInstance()->enableCookies(NULL);

cocos2d::network::HttpClient::getInstance()->send(request);

request->release();

});

this->addChild(loginButton);

auto registerButton = Button::create("register.png");

if (registerButton) {

float x = origin.x + visibleSize.width / 2;

float y = origin.y + visibleSize.height / 2 - 240.f;

registerButton->setPosition(Vec2(x, y));

}

registerButton->addTouchEventListener([&](Ref\* sender, Widget::TouchEventType type){

// Your code here

//和登录函数差不多

std::string username = usernameInput->getStringValue();

std::string password = passwordInput->getStringValue();

std::string postData = "{\"username\":\"" + username + "\"," + "\"password\":\"" + password + "\"}";

//用Post传输方式，根据PPT里面的提示用users

HttpRequest\* request = new HttpRequest();

request->setUrl("http://bobhuang.xyz:5001/users");

request->setRequestType(HttpRequest::Type::POST);

request->setResponseCallback(CC\_CALLBACK\_2(LoginRegisterScene::onHttpRequestCompleted1, this));

request->setRequestData(postData.c\_str(), strlen(postData.c\_str()));

cocos2d::network::HttpClient::getInstance()->send(request);

request->release();

});

this->addChild(registerButton);

用户名和密码

usernameInput = TextField::create("username", "arial", 50);

if (usernameInput) {

float x = origin.x + visibleSize.width / 2;

float y = origin.y + visibleSize.height /2+100.f;

usernameInput->setPosition(Vec2(x, y));

this->addChild(usernameInput, 1);

}

passwordInput = TextField::create("password", "arial", 50);

if (passwordInput) {

float x = origin.x + visibleSize.width / 2;

float y = origin.y + visibleSize.height /2+50.f;

passwordInput->setPosition(Vec2(x, y));

this->addChild(passwordInput, 1);

}

messageBox = Label::create("", "arial", 30);

if (messageBox) {

float x = origin.x + visibleSize.width / 2;

float y = origin.y + visibleSize.height/2 - 100.0f;

messageBox->setPosition(Vec2(x, y));

this->addChild(messageBox, 1);

}

8、游戏结束界面

设置游戏结束背景

auto winSize = Director::getInstance()->getWinSize();

auto bg = Scale9Sprite::create("dialog.png");

bg->setCapInsets(Rect(12, 12, 81, 78));

bg->setContentSize(Size(400, 300));

bg->setPosition(winSize.width / 2, winSize.height / 2);

addChild(bg);

设置菜单

MenuItemFont::setFontName("fonts/Marker Felt.ttf");

MenuItemFont::setFontSize(50);

auto item1 = MenuItemFont::create("Try again", CC\_CALLBACK\_0(GameOverLayer::retryCb, this, 1));

auto item2 = MenuItemFont::create("Exit", CC\_CALLBACK\_0(GameOverLayer::exitGameCb, this));

item1->setColor(Color3B::GREEN);

item2->setColor(Color3B::RED);

auto menu = Menu::create(item1, item2, nullptr);

menu->alignItemsVerticallyWithPadding(30);

menu->setPosition(Vec2(winSize.width / 2, winSize.height / 2));

addChild(menu);

9、交叉编译到安卓

这里要安装在自己电脑安装cocos ，必须用到JDK、NDK、SDK、把环境都部署成功后，按照老师给的文档修改android里的文件，最后设置一下竖屏，图标和安装包名称，使用cocos compile -p android --ap android-24进行编译，因为我安装的sdk是android24版本，跟老师不大一样，不过问题不大，但是sdk跟ndk版本必须一样，编译成功后发送到手机或者模拟器进行测试。软件图标是直接自己制作了尺寸一样的图标在文件夹中进行了替换。

//修改android.mk代码

FILE\_LIST := hellocpp/main.cpp

FILE\_LIST += $(wildcard $(LOCAL\_PATH)/../../Classes/\*.cpp)

LOCAL\_SRC\_FILES := $(FILE\_LIST:$(LOCAL\_PATH)/%=%)

//修改string.xml文件

<string name="app\_name">六角消除</string>

10、实验总结：

本次期末设计还是比较有趣的，做的时间也很久，组员有mac系统，所以有些东西管理起来不太方便。最终我们get到了ndk和sdk以及jdk是有版本要求的，完成了安卓编译，这个过程还是很曲折的。生成乐一个有class包的apk，所以apk本质就是一个打包文件，存了应用的信息，被有序的放在了一起，编译相当于平台移植，所以编译非常耗CPU。代码方面还是感觉到了游戏开发是一门非常有趣的课，所以我们也花了很多时间在里面，无聊时也会玩自己做的游戏，非常感动。

11、组内分工：

|  |  |  |
| --- | --- | --- |
| 姓名 | 主要工作 | 占比 |
| 蔡大航 | 界面美化及相关代码、文档、安卓编译 | 40% |
| 黄睿博 | 代码逻辑设计、服务器设置、修改BUG | 40% |
| 李武 | 素材收集、测试 | 20% |

教师签名：