现代操作系统应用开发实验报告

姓名：_____金汇丰_____

学号：_____16340097_____

实验名称：__HelloWorld界面（9）、
_____黄金矿工（10）、
_____横版游戏（11）_____

一.参考资料

https://gfzheng.github.io/MOSAD/
Week9-Week11相应课件

二.实验步骤

1.Week9

（1）cocos安装：

官方网站下载相应的安装包

配置python环境

终端使用python运行安装程序

（2）在相应文件夹新建一个项目

cocos new HelloWorld -l cpp

（3）界面设计

添加一个label显示学号姓名

```
auto label = Label::createWithTTF("16340097金汇丰", "fonts/xingshu.ttf", 24);
label->setPosition(Vec2(origin.x + visibleSize.width/2,
origin.y + visibleSize.height - label->getContentSize().height));
this->addChild(label, 1);
```

用图片新建一个精灵

```
auto sprite = Sprite::create("pig.jpg");
sprite->setName("pic");
sprite->setPosition(Vec2(visibleSize.width/2 + origin.x, visibleSize.height/2 + origin.y));
this->addChild(sprite, 0);
```

（bonus）设计一个换图片的点击事件，直接用了提供的图片设置了一个点击按钮，类型是MenuItemImage

```
auto changeButton = MenuItemImage::create(
                        "CloseNormal.png",
                        "CloseSelected.png",
                        CC_CALLBACK_1(HelloWorld::change, this));
changeButton->setPosition(Vec2(15,15));
auto menu = Menu::create(closeItem,changeButton,NULL);
menu->setPosition(Vec2::ZERO);
this->addChild(menu, 1);
```

点击触发换图片的回调函数

```
void HelloWorld::change(Ref* pSender){
auto visibleSize = Director::getInstance()->getVisibleSize();
Vec2 origin = Director::getInstance()->getVisibleOrigin();
this->removeChildByName("pic");
auto sprite = Sprite::create("pig2.jpg");
sprite->setPosition(Vec2(visibleSize.width/2 + origin.x, visibleSize.height/2 + origin.y));
  this->addChild(sprite, 0);
}
```

注：提前将设置图片资源、字体资源放入Resources中，更换图片前要把给之前的精灵设置名字，并且回调函数函数中将其删除，添加的函数首先在header中声明。

2.Week10

（1）主页面添加Start的menuitem，使用所给的图片，并且触发事件是切换到另一个场景。

```
auto start = MenuItemImage::create(
                                    "start-0.png",
                                    "start-1.png",
                                    CC_CALLBACK_1(MenuScene::changeScene, this));
    auto menu = Menu::create(start, NULL);
    menu->setPosition(Vec2(750 + origin.x, 145));
    this->addChild(menu, 1);
```

切换场景回调函数

```
    void MenuScene::changeScene(cocos2d::Ref* pSender){
   auto scene = GameSence::createScene();
   Director::getInstance()->
 replaceScene(TransitionFade::create(0.5, scene, Color3B(0,     255, 255)));
   }
```

（2）添加背景图片、石头层、老鼠层

```
    //背景图片
   auto bg = Sprite::create("level-background-0.jpg");
  bg->setPosition(Vec2(visibleSize.width / 2 + origin.x, visibleSize.height / 2 +
origin.y ));
    this->addChild(bg, 0);
    //石头层，在所要求的位置添加石头图片
    stoneLayer = Layer::create();
    this->addChild(stoneLayer);
    stoneLayer->setPosition(0, 0);
    stone = Sprite::create("stone.png");
    stoneLayer->addChild(stone);
    stone->setPosition(560, 480);
    //添加老鼠层
    mouseLayer = Layer::create();
    this->addChild(mouseLayer);
    mouseLayer->setPosition(0, visibleSize.height/2);
```

（3）在老鼠层添加老鼠动画，并且添加生成奶酪的事件（出现奶酪图片并渐变消失，老鼠向奶酪移动）

```
//打开存放动画帧的plist
SpriteFrameCache::getInstance()->
addSpriteFramesWithFile("level-sheet.plist");
int totalFrames = 8;  //一共有8帧
char frameName[20];
Animation* mouseAnimation = Animation::create();
//创建动画
for (int i = 0; i < totalFrames; i++)
{
    sprintf(frameName, "gem-mouse-%d.png", i);
        mouseAnimation->addSpriteFrame(SpriteFrameCache::getInstance()->getSpriteFrameByName(frameName));
}
mouseAnimation->setDelayPerUnit(0.1);
//在AnimationCache中添加一个老鼠动画
AnimationCache::getInstance()->addAnimation(mouseAnimation, "mouseAnimation");
//用第一帧创建一个精灵
mouse = Sprite::createWithSpriteFrameName(“gem-mouse-0.png");
//获取老鼠动画
Animate* mouseAnimate = Animate::create(AnimationCache::getInstance()->getAnimation(“mouseAnimation"));
 //老鼠执行动画效果
mouse->runAction(RepeatForever::create(mouseAnimate));
mouseLayer->addChild(mouse);
mouse->setPosition(visibleSize.width/2, 0);



//点击事件回调函数
bool GameSence::onTouchBegan(Touch *touch, Event *unused_event) {

auto location = touch->getLocation(); // 获取点击位置
auto cheese = Sprite::create(“cheese.png”);// 在老鼠层在点击的位置用奶酪创
                                          //建一个精灵

auto loc = mouseLayer->convertToNodeSpace(location);
```

```
mouseLayer->addChild(cheese);
cheese->setPosition(loc); //老鼠层的节点位置添加奶酪
auto moveTo = MoveTo::create(2, loc);
mouse->runAction(moveTo);  // 把老鼠移动到该位置
ActionInterval *forwardOut = FadeOut::create(4.0f); // 奶酪渐变消失
cheese->runAction(forwardOut);


return true;
}
```

（4）添加shoot事件

```
 //首先添加一个MenuItemLabel，内容为“Shoot”，并且绑定一个回调函数
auto shoot = Label::createWithSystemFont("Shoot","Marker Felt",32);
auto labelItem = MenuItemLabel::create(shoot,
CC_CALLBACK_1(GameSence::shootFunc, this));
auto menu = Menu::create(labelItem, NULL);
menu->setPosition(Vec2(700,500));
this->addChild(menu, 1);


//shoot点击的时候将会触发这个函数
//包括动作：石头飞向老鼠的位置并渐变消失，
                老鼠移动到一个随机位置并留下一颗钻石
                （bonus）留下钻石后，钻石将会以动画效果显示

void GameSence::shootFunc(Ref* pSender){

//获取老鼠移动前位置
auto originalMouseLoc = mouse->getPosition();

 //创建一个钻石动画，在老鼠原来的位置添加一个钻石精灵，并执行这个动画
SpriteFrameCache::getInstance()->
addSpriteFramesWithFile("level-sheet.plist");
int totalFrames = 7;
char frameName[20];
Animation* diamondAnimation = Animation::create();
for (int i = 0; i < totalFrames; i++)
{
    sprintf(frameName, "diamond-%d.png", i);
    diamondAnimation->addSpriteFrame(SpriteFrameCache::getInstance()
    ->getSpriteFrameByName(frameName));
```

```
    }
    diamondAnimation->setDelayPerUnit(0.1);
    AnimationCache::getInstance()->
    addAnimation(diamondAnimation, "diamondAnimation");
    auto diamond = Sprite::createWithSpriteFrameName("diamond-0.png");
    Animate* diamondAnimate = Animate::create(AnimationCache::getInstance()
    ->getAnimation("diamondAnimation"));
    diamond->runAction(RepeatForever::create(diamondAnimate));
    mouseLayer->addChild(diamond);
    diamond->setPosition(originalMouseLoc)；

    //用石头图片在石头层首先创建一个精灵，添加到初始石头精灵的位置
    auto shootStone = Sprite::create("stone.png");
    stoneLayer->addChild(shootStone);
    shootStone->setPosition(560,480);

    //老鼠层的老鼠位置转换到石头层的对应位置
    auto worldLoc = mouseLayer->convertToWorldSpace(mouse->getPosition());
    auto newLoc = stoneLayer->convertToNodeSpace(worldLoc);
    auto moveTo = MoveTo::create(1, newLoc);//石头移动到老鼠原来的位置
    shootStone->runAction(moveTo);
    ActionInterval *forwardOut = FadeOut::create(8.0f);//石头渐变消失
    shootStone->runAction(forwardOut);

    //老鼠移动到一个随机位置
    auto visibleSize = Director::getInstance()->getVisibleSize();
    float randomX = rand()/double(RAND_MAX);
    float randomY = rand()/double(RAND_MAX);
    float loc_x = randomX * visibleSize.width;
    float loc_y = randomY * visibleSize.height;
    auto mouseMove = mouseLayer->convertToNodeSpace(Vec2(loc_x,loc_y));
    auto MouseMoveTo = MoveTo::create(2, mouseMove);
    mouse->runAction(MouseMoveTo);

}
```

3.Week11

（1）Label点击事件，本次横版游戏共有6个虚拟按键

控制移动方向：WASD

攻击和死亡动画：XY

```
//创建6个按钮的Label添加到界面，并且绑定对应的回调函数
auto Wbtn = Label::createWithSystemFont("W","arial.ttf",36);
auto Abtn = Label::createWithSystemFont("A","arial.ttf",36);
auto Sbtn = Label::createWithSystemFont("S","arial.ttf",36);
auto Dbtn = Label::createWithSystemFont("D","arial.ttf",36);
auto Xbtn = Label::createWithSystemFont("X","arial.ttf",36);
auto Ybtn = Label::createWithSystemFont("Y","arial.ttf",36);
auto WLabel = MenuItemLabel::create(Wbtn,
                    CC_CALLBACK_1(HelloWorld::wFunc, this));
auto ALabel = MenuItemLabel::create(Abtn,
                    CC_CALLBACK_1(HelloWorld::aFunc, this));
auto SLabel = MenuItemLabel::create(Sbtn,
                    CC_CALLBACK_1(HelloWorld::sFunc, this));
auto DLabel = MenuItemLabel::create(Dbtn,
                    CC_CALLBACK_1(HelloWorld::dFunc, this));
auto XLabel = MenuItemLabel::create(Xbtn,
                    CC_CALLBACK_1(HelloWorld::xFunc, this));
auto YLabel = MenuItemLabel::create(Ybtn,
                    CC_CALLBACK_1(HelloWorld::yFunc, this));
WLabel->setPosition(Vec2(50,80));
ALabel->setPosition(Vec2(20,50));
SLabel->setPosition(Vec2(50,50));
DLabel->setPosition(Vec2(80,50));
XLabel->setPosition(Vec2(700,80));
YLabel->setPosition(Vec2(670,50));
auto menu = Menu::create(WLabel,ALabel,SLabel,
                         DLabel,XLabel,YLabel,NULL);
menu->setPosition(Vec2(0,0));
this->addChild(menu, 1);
```

```
//首先创建两个动画的每帧添加到对应的vector里：
//移动动画、死亡动画，因为虚拟按键在触发时要表现对应动作
//仿照攻击动作

//死亡动作
auto texture2 = Director::getInstance()->
getTextureCache()->addImage("$lucia_dead.png");
for (int i = 0; i < 22; i++) {
    auto frame = SpriteFrame::createWithTexture(texture2,
    CC_RECT_PIXELS_TO_POINTS(Rect(79 * i, 0, 79, 90)));
    dead.pushBack(frame);
}

//移动动作
auto texture3 = Director::getInstance()->
getTextureCache()->addImage("$lucia_forward.png");
for (int i = 0; i < 8; i++) {
    auto frame = SpriteFrame::createWithTexture(texture3,
    CC_RECT_PIXELS_TO_POINTS(Rect(68 * i, 0, 68, 101)));
    run.pushBack(frame);
}



//因移动函数类似，这里仅举一例
void HelloWorld::wFunc(Ref* pSender){
//用run的vector里的每一帧创建一个动画
auto run_animation = Animation::createWithSpriteFrames(run, 0.1f);
run_animation->setRestoreOriginalFrame(true);
AnimationCache::getInstance()->
            addAnimation(run_animation, “runAnimation");
//当人物要越出边界，移动到边界
if (player->getPosition().y + 20 > visibleSize.height - 20) {
        auto moveTo = MoveTo::create(0.3f,
        Vec2(player->getPosition().x,visibleSize.height - 20));
        //执行移动动画
        player->runAction(Animate::create(AnimationCache::getInstance()->
                getAnimation(“runAnimation")));
        //移动到对应位置
        player->runAction(moveTo);
}else{
        //人物不将要越界时，向对应方向移动20个点位
        auto moveBy = MoveBy::create(0.3f, Vec2(0,20));
```

```
        player->runAction(Animate::create(AnimationCache::getInstance()
        ->getAnimation("runAnimation")));
        player->runAction(moveBy);
    }
}
```

（2）死亡和攻击动画，二者不能同时执行

（bonus）血条变化

```
//x,y实现方法相同，这里用x的对应动作举例
void HelloWorld::xFunc(Ref* pSender){

//当一个动作执行时（x或y），设定一个tag用于检测另一个动作是否执行中
//本次实验均设置tag为666，当另一个动作执行时，返回
    if (player->getActionByTag(666) != NULL) {
      return;
    }
//执行动画
auto die_animation = Animation::createWithSpriteFrames(dead, 0.1f);
die_animation->setRestoreOriginalFrame(true);
AnimationCache::getInstance()->
          addAnimation(die_animation,      "dieAnimation");
auto action = Animate::create(AnimationCache::getInstance()->
          getAnimation("dieAnimation"));
//给这个动作设置tag
action->setTag(666);
player->runAction(action);

//血条变化
//为了让血条渐变，这里要用到调度器
if(pT->getPercentage() <= 0){
    pT->setPercentage(0);
}else{
  //自定义调度器，调度20次减血参数，每次间隔0.05s；
  schedule(schedule_selector(HelloWorld::subHp), 0.05f, 20, 0);
}
}

//每次减1%的血量，y函数和对应的加血函数和这些实现方法相同
void HelloWorld::subHp(float dt){
  pT->setPercentage(pT->getPercentage() - 1);
}
```

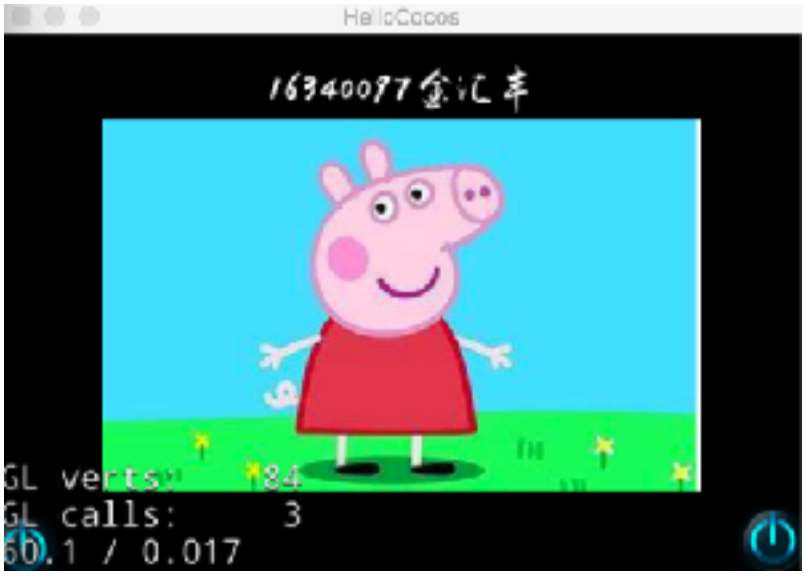
(3) 计时器

```
  //创建倒计时的Label
time = Label::createWithSystemFont("180", "arial.ttf", 36);
time->setPosition(Vec2(visibleSize.width/2,
                visibleSize.height -15));
addChild(time);
//将时间总量初始为180
dtime = 180;
//每隔一秒调度一次减时间的函数
schedule(schedule_selector(HelloWorld::subtTime),
             1.0f, 180, 0);

//时间递减的函数
void HelloWorld::subtTime(float dt){
  //剩余时间-1
  dtime—;
  //更改Label的内容
  string temp = to_string(dtime);
  time->setString(temp);
}
```

三.关键步骤截图

Week9



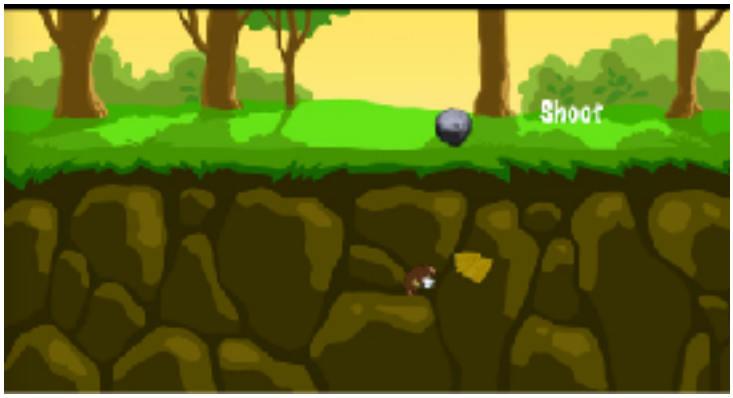
变换图片后（bouns，简单点击时间）

Week10

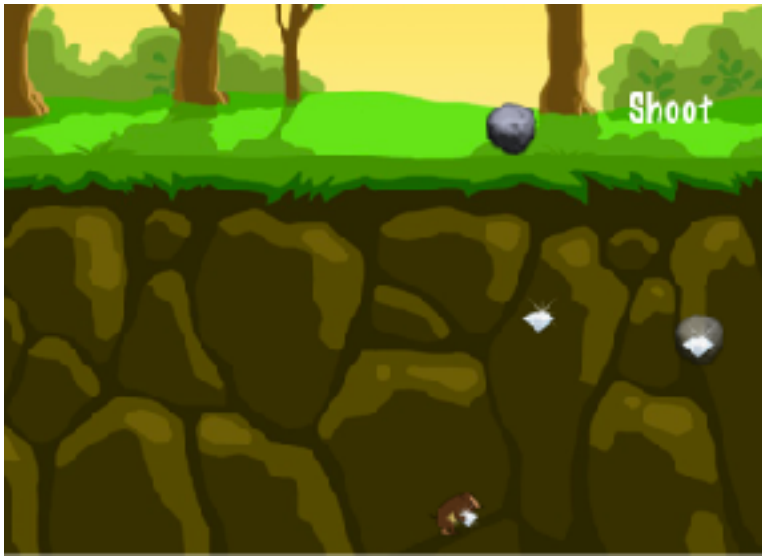
主页面

游戏界面

老鼠向奶酪移动



石头射向老鼠，老鼠留下钻石（动画效果）



Week11

四.亮点与改进（可选）

Helloworld界面：加入新的字体，完成了简单的点击触发事件
黄金矿工：增加了留下钻石的动画
横版游戏：用调度器完成了血条的变化

注：实验效果TA课堂上已检查过，具体代码上文已给出

五.遇到的问题

1.页面显示中文的问题
参考了网上的博客，添加下面的代码即可解决
#pragma execution_character_set(“utf-8")
2.黄金矿工所给的demo里Gamescene的拼写错误有点坑。。。
3.世界坐标和本地坐标的理解很重要，每个对象的设置坐标实际上是相对于父节点的本地坐标，所以要注意绝对坐标和本地坐标的转化，这个问题当初理解错了所以导致老鼠移动

的位置不对。同时，在使用转换坐标函数的时候，要理解好锚点的影响，是否要考虑，本次实验需要考虑。
4.横版游戏中，血条的变化一开始没用调度器，就直接变化了一次，效果不好，用了调度器之后，血条的变化更加连贯
5.获取某一个对象的时候，可以设定对应的tag或者name

六.思考与总结

这是我第一次接触cocos，感觉对于开发者来说，这个框架还是很好用的，因为本学期我也选修了3d游戏的开发，真的是比unity3d好写多了，开发者只要专注于代码部分就可以了。同时，cocos开发主要用C++，这让我写的也比较容易，可能比用C#用的更加习惯吧。

cocos作为一个跨平台的框架，这几周我都是用macOs的Xcode做的，也是比较方便，不用开windows虚拟机了，个人感觉Xcode在项目第一次编译比Vs快了不少。而且一开始在windows做总会有一些未知名的系统bug，后来转战macOs就不会再出现了那些麻烦的事情。

本门课程里能够学习到cocos我觉得是对今后发展很有用的，可以学习一些制作游戏的知识，比如导演、场景、布局之类的，这几周作业也想对比容易了不少，希望以后能用cocos写出更有趣的游戏。