现代操作系统应用开发实验报告

学号： 15331132　　　　**班级　：** 大二（3）

姓名： 金子力　　　　**实验名称：** hw10

一．参考资料

http://blog.csdn.net/ganpengjin1/article/details/17299165

http://www.cnblogs.com/skysand/p/4247823.html

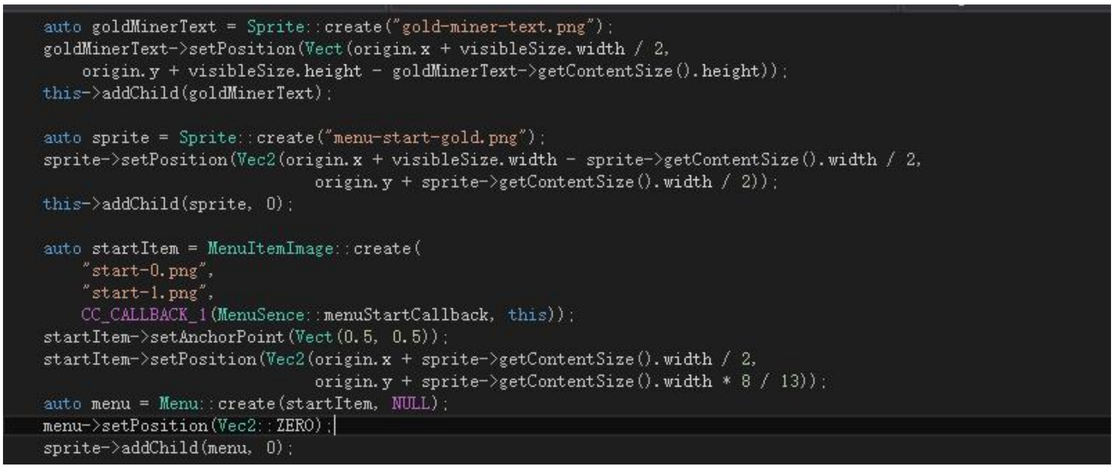
http://www.jellythink.com/archives/737

http://blog.csdn.net/a102111/article/details/38657323

http://blog.csdn.net/nanshan_hzq/article/details/40184251

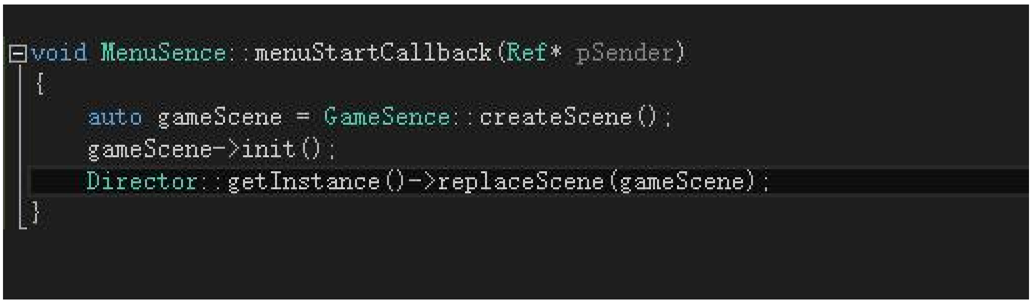
二．实验步骤

设置主界面的各个图标



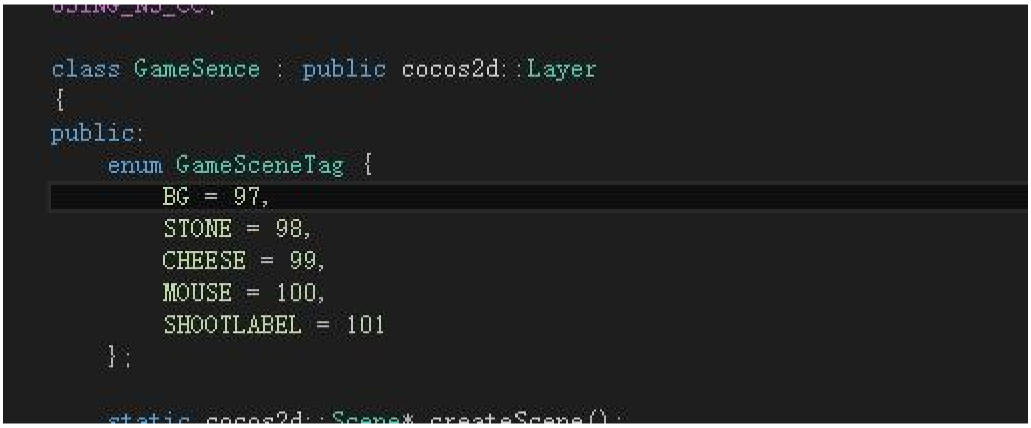
```
auto goldMinerText = Sprite::create("gold-miner-text.png");
goldMinerText->setPosition(Vect(origin.x + visibleSize.width / 2,
    origin.y + visibleSize.height - goldMinerText->getContentSize().height));
this->addChild(goldMinerText);

auto sprite = Sprite::create("menu-start-gold.png");
sprite->setPosition(Vec2(origin.x + visibleSize.width - sprite->getContentSize().width / 2,
                         origin.y + sprite->getContentSize().width / 2));
this->addChild(sprite, 0);

auto startItem = MenuItemImage::create(
    "start-0.png",
    "start-1.png",
    CC_CALLBACK_1(MenuSence::menuStartCallback, this));
startItem->setAnchorPoint(Vect(0.5, 0.5));
startItem->setPosition(Vec2(origin.x + sprite->getContentSize().width / 2,
                         origin.y + sprite->getContentSize().width * 8 / 13));
auto menu = Menu::create(startItem, NULL);
menu->setPosition(Vec2::ZERO);
sprite->addChild(menu, 0);
```

start 按钮的场景切换回调函数



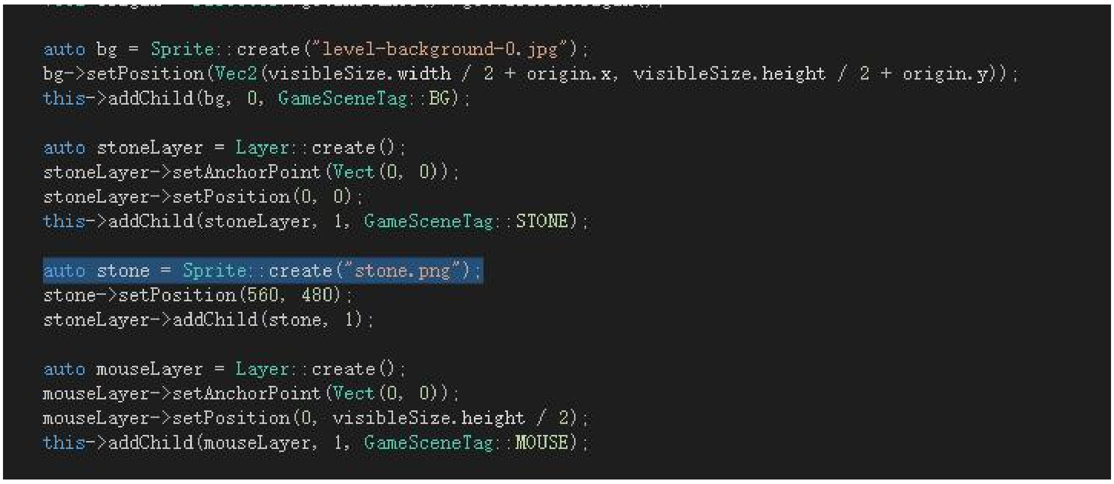
```
void MenuSence::menuStartCallback(Ref* pSender)
{
    auto gameScene = GameSence::createScene();
    gameScene->init();
    Director::getInstance()->replaceScene(gameScene);
}
```

定义一个宏，以便于后面用 tag 寻找 child

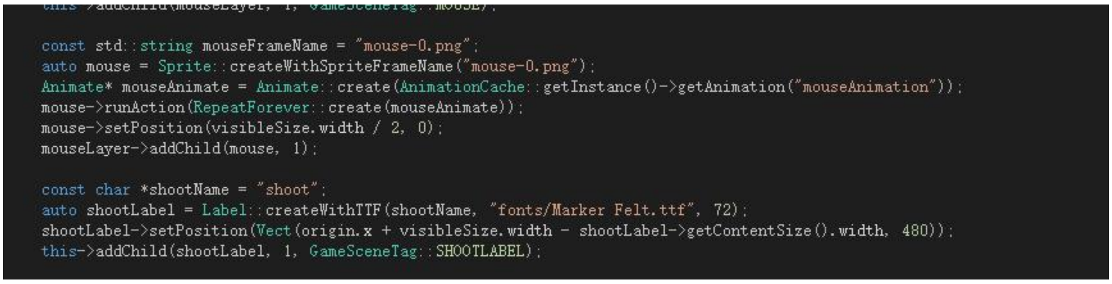


```
USING_NS_CC;

class GameSence : public cocos2d::Layer
{
public:
    enum GameSceneTag {
        BG = 97,
        STONE = 98,
        CHEESE = 99,
        MOUSE = 100,
        SHOOTLABEL = 101
    };

    static cocos2d::Scene* createScene();
```

放置背景。放置老鼠，石块和他们的 layer。再放置 shoot 按钮。

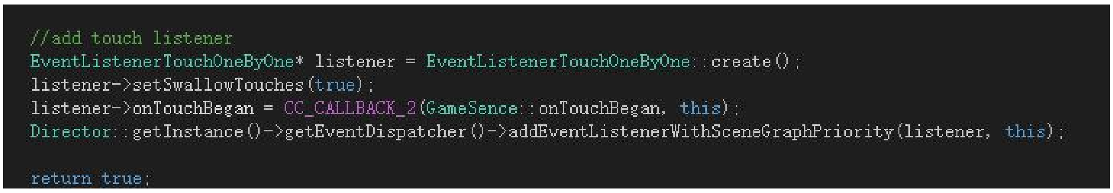


```
auto bg = Sprite::create("level-background-0.jpg");
bg->setPosition(Vec2(visibleSize.width / 2 + origin.x, visibleSize.height / 2 + origin.y));
this->addChild(bg, 0, GameSceneTag::BG);

auto stoneLayer = Layer::create();
stoneLayer->setAnchorPoint(Vect(0, 0));
stoneLayer->setPosition(0, 0);
this->addChild(stoneLayer, 1, GameSceneTag::STONE);

auto stone = Sprite::create("stone.png");
stone->setPosition(560, 480);
stoneLayer->addChild(stone, 1);

auto mouseLayer = Layer::create();
mouseLayer->setAnchorPoint(Vect(0, 0));
mouseLayer->setPosition(0, visibleSize.height / 2);
this->addChild(mouseLayer, 1, GameSceneTag::MOUSE);
```



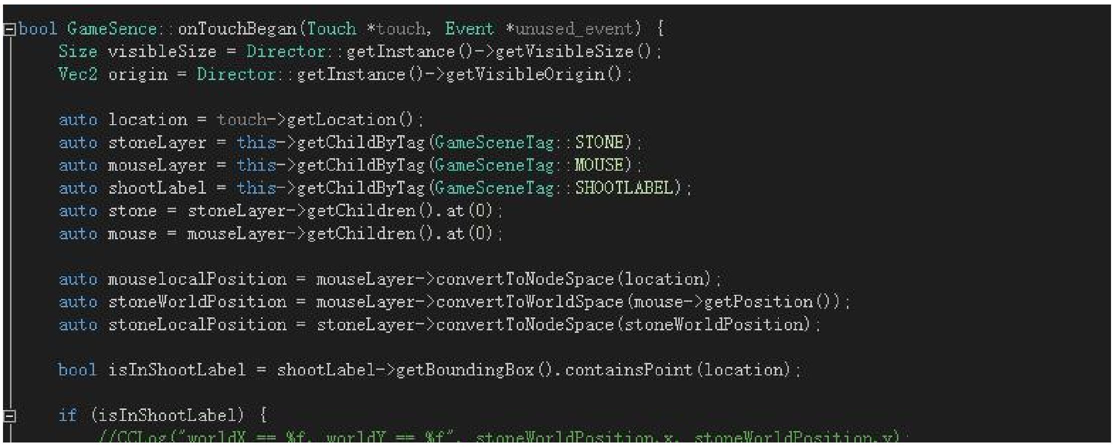
```
this->addChild(mouseLayer, 1, GameSceneTag::MOUSE);

const std::string mouseFrameName = "mouse-0.png";
auto mouse = Sprite::createWithSpriteFrameName("mouse-0.png");
Animate* mouseAnimate = Animate::create(AnimationCache::getInstance()->getAnimation("mouseAnimation"));
mouse->runAction(RepeatForever::create(mouseAnimate));
mouse->setPosition(visibleSize.width / 2, 0);
mouseLayer->addChild(mouse, 1);

const char *shootName = "shoot";
auto shootLabel = Label::createWithTTF(shootName, "fonts/Marker Felt.ttf", 72);
shootLabel->setPosition(Vect(origin.x + visibleSize.width - shootLabel->getContentSize().width, 480));
this->addChild(shootLabel, 1, GameSceneTag::SHOOTLABEL);
```

监听触摸事件



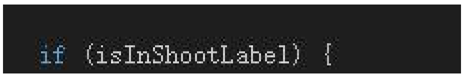
```
//add touch listener
EventListenerTouchOneByOne* listener = EventListenerTouchOneByOne::create();
listener->setSwallowTouches(true);
listener->onTouchBegan = CC_CALLBACK_2(GameSence::onTouchBegan, this);
Director::getInstance()->getEventDispatcher()->addEventListenerWithSceneGraphPriority(listener, this);

return true;
```

触摸的回调函数



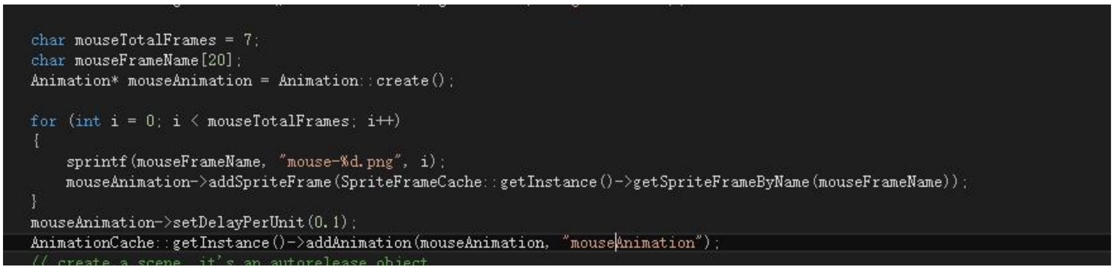
```
bool GameSence::onTouchBegan(Touch *touch, Event *unused_event) {
    Size visibleSize = Director::getInstance()->getVisibleSize();
    Vec2 origin = Director::getInstance()->getVisibleOrigin();

    auto location = touch->getLocation();
    auto stoneLayer = this->getChildByTag(GameSceneTag::STONE);
    auto mouseLayer = this->getChildByTag(GameSceneTag::MOUSE);
    auto shootLabel = this->getChildByTag(GameSceneTag::SHOOTLABEL);
    auto stone = stoneLayer->getChildren().at(0);
    auto mouse = mouseLayer->getChildren().at(0);

    auto mouselocalPosition = mouseLayer->convertToNodeSpace(location);
    auto stoneWorldPosition = mouseLayer->convertToWorldSpace(mouse->getPosition());
    auto stoneLocalPosition = stoneLayer->convertToNodeSpace(stoneWorldPosition);

    bool isInShootLabel = shootLabel->getBoundingBox().containsPoint(location);

    if (isInShootLabel) {
        //CCLog("worldX == %f, worldY == %f", stoneWorldPosition.x, stoneWorldPosition.y);
```

点击在 shoot 图标内外会有不同行为



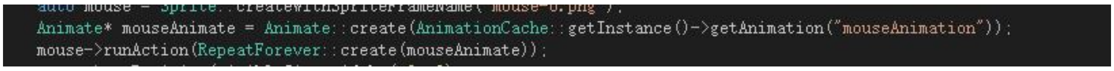
```
if (isInShootLabel) {
```

缓存动画



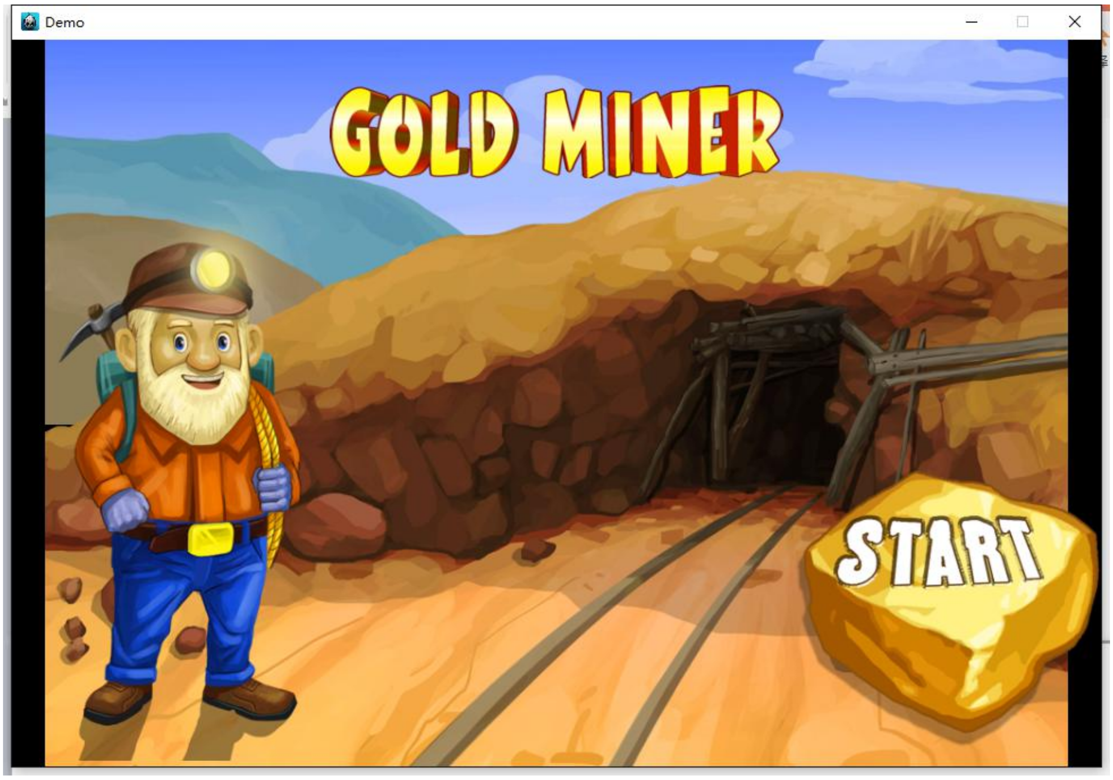
```
char mouseTotalFrames = 7;
char mouseFrameName[20];
Animation* mouseAnimation = Animation::create();

for (int i = 0; i < mouseTotalFrames; i++)
{
    sprintf(mouseFrameName, "mouse-%d.png", i);
    mouseAnimation->addSpriteFrame(SpriteFrameCache::getInstance()->getSpriteFrameByName(mouseFrameName));
}
mouseAnimation->setDelayPerUnit(0.1);
AnimationCache::getInstance()->addAnimation(mouseAnimation, "mouseAnimation");
// create a scene. it's an autorelease object
```

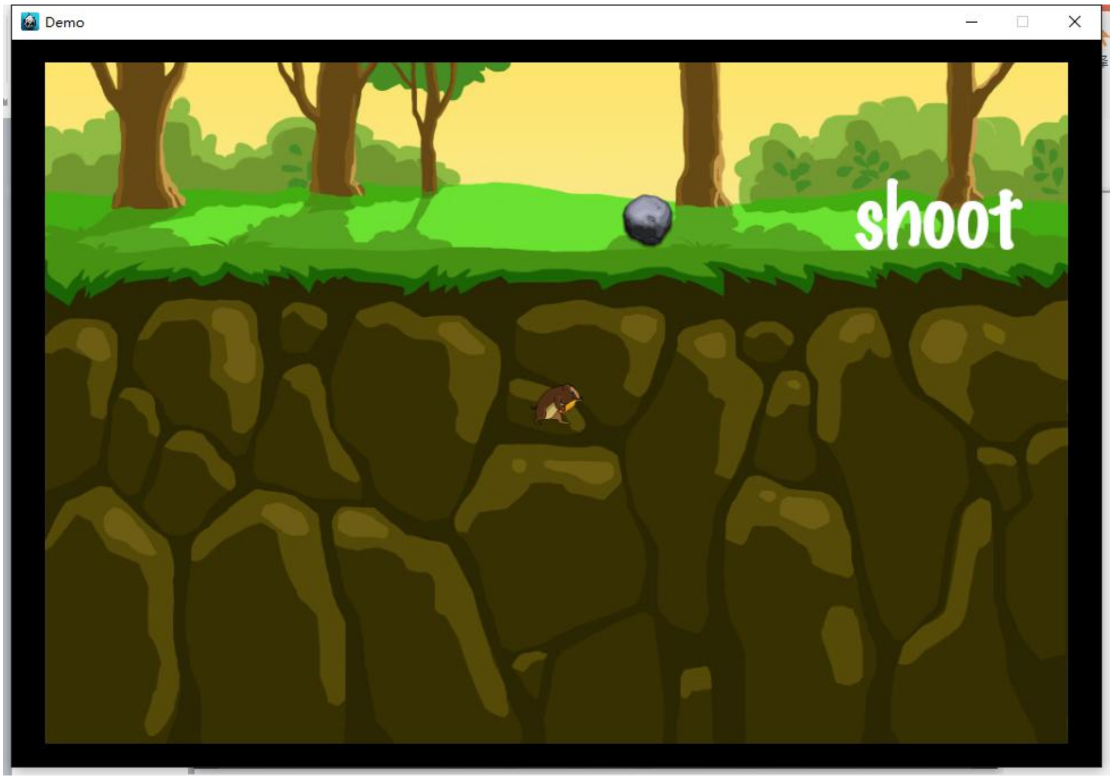
给老鼠施加动画



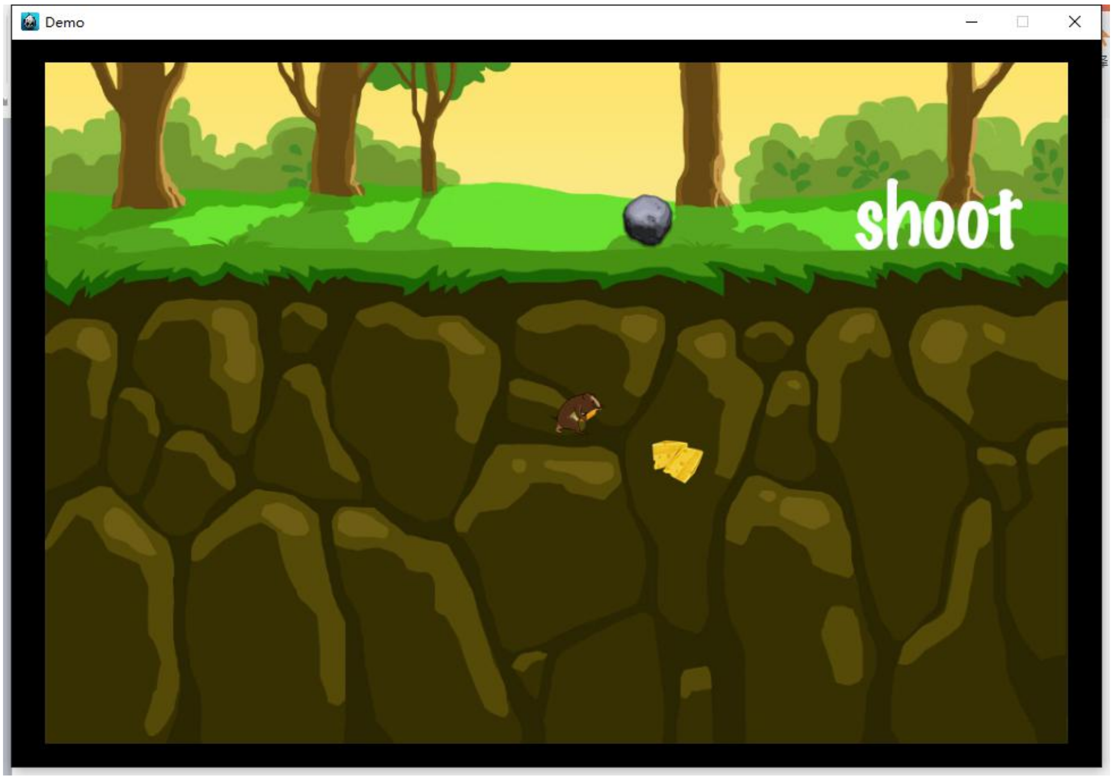
```
auto mouse = Sprite::createWithSpriteFrameName("mouse-0.png");
Animate* mouseAnimate = Animate::create(AnimationCache::getInstance()->getAnimation("mouseAnimation"));
mouse->runAction(RepeatForever::create(mouseAnimate));
```

三．实验结果截图

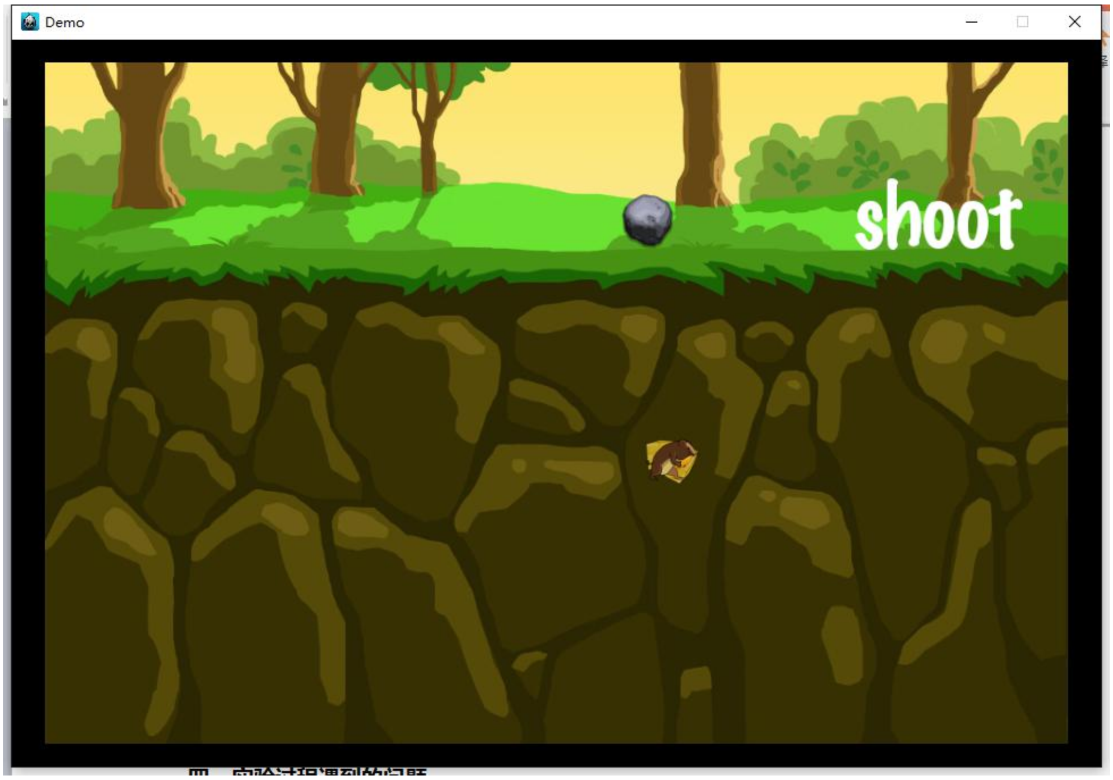




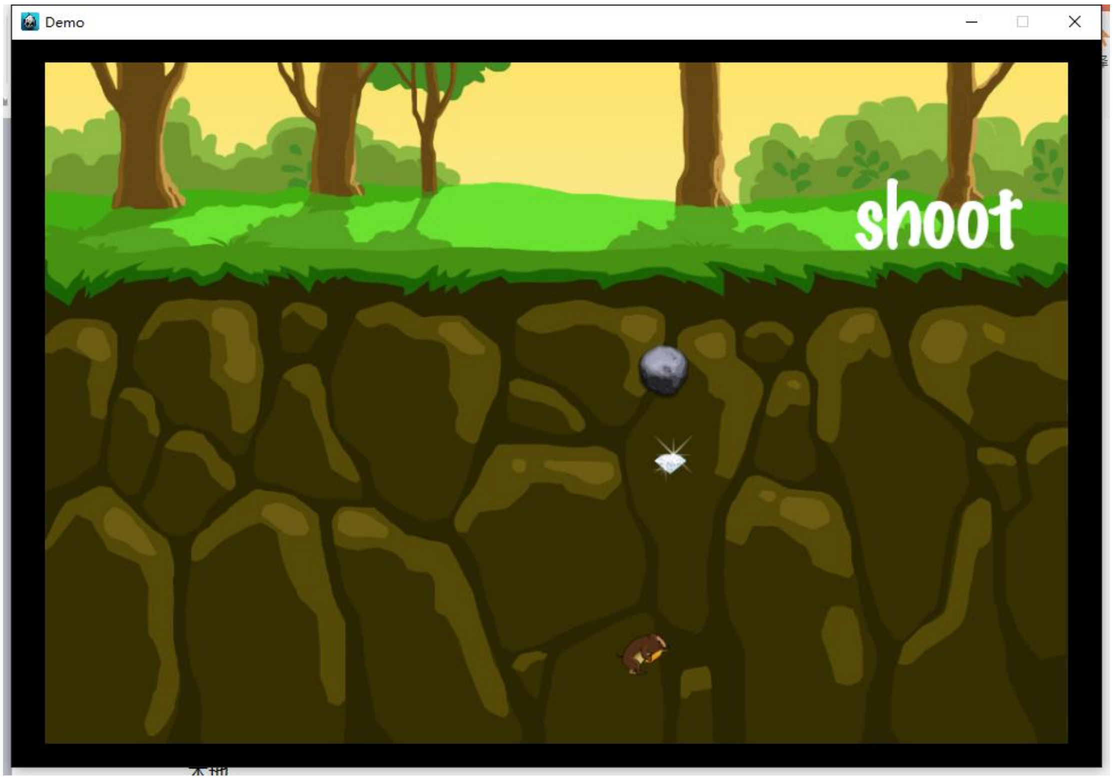
点击任意一点



吃掉奶酪



Shoot 时，石头砸向老鼠，老鼠逃跑，留下钻石



四．实验过程遇到的问题

本地坐标系与世界坐标系的转换

lambda 捕捉 this

查找特定的 child

五．思考与总结

要更加精确地描述自己搜索的问题。

在实验时，曾经遇到一个问题：无法从 frame 缓存中读取出图片，卡了 1 个小时，才知道是自己根本没把 level-sheet.plist 读入缓存（demo 自带读入了 general-sheet.plist 但没有帮我们读入 level-sheet.plist）。这是个低级错误，以后应当在做作业时保持清醒，才能避免无用功

1. 实验报告提交格式为 pdf。

2. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。