现代操作系统应用开发实验报告

学号： 14970011　　**班级：** 2015 级教务 2 班

姓名： 宋思亭　　**实验名称：** homework12

一．参考资料

作业要求文档, 课件 PPT，

官网：http://www.cocos.com/

Github：https://github.com/cocos2d/cocos2d-x

用户手册：http://www.cocos2d-x.org/wiki/Cocos2d-x

商店：http://store.cocos.com/

API: http://api.cocos.com/

cocos2d 无法打开包含文件：http://blog.csdn.net/cdamber/article/details/44700817

二．实验步骤

1. 阅读作业需求和课件 PPT，了解阅读课件内容以及作业要求，了解 Cocos2dx 的数据结构，内存管理，本地数据存储瓦片地图与 tilemap 使用。下载 tilemap。了解 demo 中的 monster 容器以及实现相关函数的原理。
2. 补充 Monster 中的函数，注意使用迭代器循环遍历 Vector 容器的方式，特别是删除元素时，erase 函数返回下一个迭代器，无需+1

```
void Factory::removeMonster(Sprite* sp) {
    Vector<Sprite*>::iterator it = monster.begin();
    for (; it != monster.end();) {
        if (sp == (*it)) {
            it = monster.erase(it);
        }
        else {
            it++;
        }
    }
}
void Factory::moveMonster(Vec2 playerPos,float time){
    Vector<Sprite*>::iterator it = monster.begin();
    for (; it != monster.end(); it++) {
        Vec2 mp = (*it)->getPosition();
        Vec2 direaction = playerPos - mp;
        direaction.normalize();
        (*it)->runAction(MoveBy::create(time, direaction * 30));
    }
}

Sprite* Factory::collider(Rect rect) {
    for (Vector<Sprite*>::iterator it = monster.begin(); it != monster.end(); it++) {
        if (rect.containsPoint((*it)->getPosition())) {
            return *it;
        }
    }
    return NULL;
}
```

3. 写游戏类.h 文件

```
class HelloWorld : public cocos2d::Layer
{
public:
    static cocos2d::Scene* createScene();
    virtual bool init();
    void moveEvent(Ref*, char);
    void actionEvent(Ref*, char);
    void stopAc();
    void updateKill(float dt);
    void creatMonster(float dt);
    void moveMonster(float dt);
    void hitByMonster(float dt);
    bool attackmonster();

    CREATE_FUNC(HelloWorld);

private:
    cocos2d::Sprite* player;
    cocos2d::ProgressTimer* timer;
    cocos2d::Vector<SpriteFrame*> attack;
    cocos2d::Vector<SpriteFrame*> dead;
    cocos2d::Vector<SpriteFrame*> run;
    cocos2d::Vector<SpriteFrame*> idle;
    cocos2d::Size visibleSize;
    cocos2d::Vec2 origin;
    cocos2d::Label* killNum;
    int num;
    int hp = 100;
    bool actionOn = true;
    bool rotate = false;
    bool isEnd = false;
};
```

4. 写 creatMonster 函数，实现随机产生怪物

moveMonster 函数，实现所有怪物向人物移动

```
void HelloWorld::creatMonster(float dt) {
    auto fac = Factory::getInstance();
    auto newMonster = fac->createMonster();
    newMonster->setPosition(random(origin.x, visibleSize.width), random(origin.y, visibleSize.height));
    addChild(newMonster, 2);
}

void HelloWorld::moveMonster(float dt) {
    auto position = player->getPosition();
    Factory::getInstance()->moveMonster(position, 4.0f);
}
```

5. 写 hitByMonster 类，实现怪物碰到角色后，角色掉血

若角色掉血到 0 或以下，显示游戏结束画面，并将击杀怪物计数重置为 0

```
void HelloWorld::hitByMonster(float dt) {
    auto fac = Factory::getInstance();
    Sprite* collision = fac->collider(player->getBoundingBox());
    if (collision != NULL) {
        actionOn = true;
        FiniteTimeAction *deadAction = Repeat::create(Animate::create(Animation::createWithSpriteFrames(dead, 0.1f)), 1);
        FiniteTimeAction *idleAction = Repeat::create(Animate::create(Animation::createWithSpriteFrames(idle, 0.1f)), 1);
        auto stopAction = CallFunc::create(CC_CALLBACK_0(HelloWorld::stopAc, this));
        if (hp > 0) {
            hp = hp - 20 <= 0 ? 0 : hp - 20;
        }
        if (hp <= 0) {
            timer->runAction(Sequence::create(CCProgressTo::create(2, 0), CallFunc::create([this]() {
                player->runAction(Sequence::create(ScaleTo::create(2.0, 1.0), FadeOut::create(1.0), nullptr)); // 人物消失
                auto over = Sprite::create("over.png");
                float winw = visibleSize.width;  // 获取屏幕宽度
                float winh = visibleSize.height; // 获取屏幕高度
                over->setPosition(Vec2(winw / 2 + origin.x, winh / 2 + origin.y));
                float spx = over->getTextureRect().getMaxX();
                over->setScaleX(winw / spx);//  背景缩放
                over->setScaleY(winw / spx);
                this->addChild(over, 2);
                isEnd = true;
                num = 0;
                database->setIntegerForKey("killNum", num);
            }), nullptr));
            return;
        }
        fac->removeMonster(collision);
        CCProgressTo* progress = CCProgressTo::create(2, hp);
        timer->runAction(progress);
        player->runAction(Sequence::create(deadAction, idleAction, stopAction, NULL));
        removeChild(collision);
    }
}
```

6. 写 attackMonster 类，实现角色可以攻击怪物

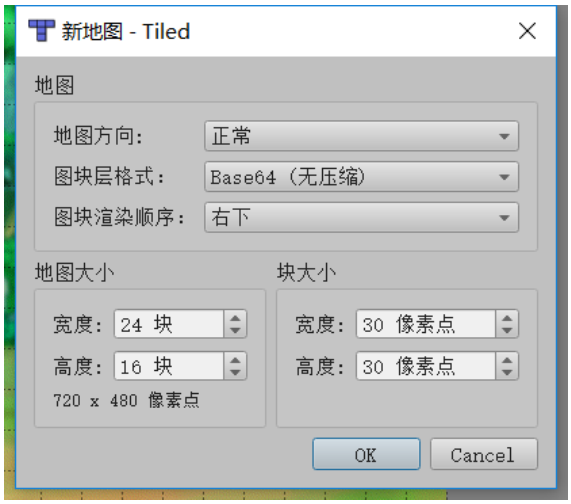
在游戏中点击 Y 按钮，人物播放攻击动画，若人物前后一定范围内存在怪物，

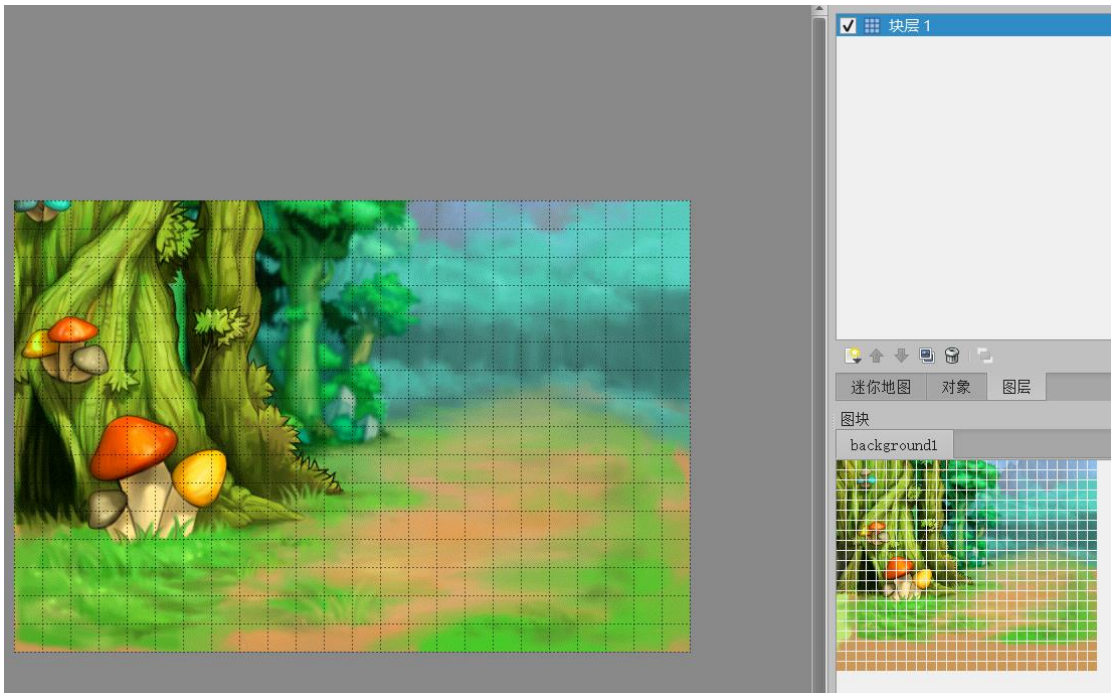
attackMonster 返回 true，删除该怪物并恢复角色的一定血量

```
void HelloWorld::actionEvent(Ref*, char cid) {
    if (actionOn && !isEnd) {
        FiniteTimeAction *deadAction = Repeat::create(Animate::create(Animation::createWithSpriteFrames(dead, 0.1f)), 1);
        FiniteTimeAction *attackAction = Repeat::create(Animate::create(Animation::createWithSpriteFrames(attack, 0.1f)), 1);
        FiniteTimeAction *idleAction = Repeat::create(Animate::create(Animation::createWithSpriteFrames(idle, 0.1f)), 1);
        auto stopAction = CallFunc::create(CC_CALLBACK_0(HelloWorld::stopAc, this));
        if (actionOn) {
            actionOn = false;
            player->runAction(Sequence::create(attackAction, idleAction, stopAction, NULL));
            if (attackmonster()) {
                if (hp < 100) {
                    hp = hp + 20 >= 100 ? 100 : hp + 20;
                }
                num++;
                database->setIntegerForKey("killNum", num);
            }
        }
        CCProgressTo* progress = CCProgressTo::create(2, hp);
        timer->runAction(progress);
    }
}

bool HelloWorld::attackmonster() {
    Rect girlRect = player->getBoundingBox();
    Rect attackRect = Rect(girlRect.getMinX() - 40, girlRect.getMinY(),
        girlRect.getMaxX() - girlRect.getMinX() + 80, girlRect.getMaxY() - girlRect.getMinY());
    Sprite* collision = Factory::getInstance()->collider(attackRect);
    if (collision != NULL) {
        removeChild(collision);
        Factory::getInstance()->removeMonster(collision);
    }
    return collision != NULL;
}
```

7. 使用 tilemap 创建地图





在 init 文件中创建地图

```
//创建地图
TMXTiledMap* tmx = TMXTiledMap::create("background.tmx");
tmx->setPosition(winw / 2, winh / 2);
tmx->setAnchorPoint(Vec2(0.5, 0.5));
tmx->setScale(Director::getInstance()->getContentScaleFactor());
this->addChild(tmx, 0);
```

8. 使用本地数据存储，记录打到的怪物数量，并将倒计时改为显示打倒数量

定义 database

```
#define database UserDefault::getInstance()
```

在 init 函数中初始化

```
num = database->getIntegerForKey("killNum");
```

将倒计时改为显示打倒数量

```
killNum = Label::createWithTTF("0", "fonts/Marker Felt.ttf", 40);
```

在设置 num 之后注意保存

```
database->setIntegerForKey("killNum", num);
```

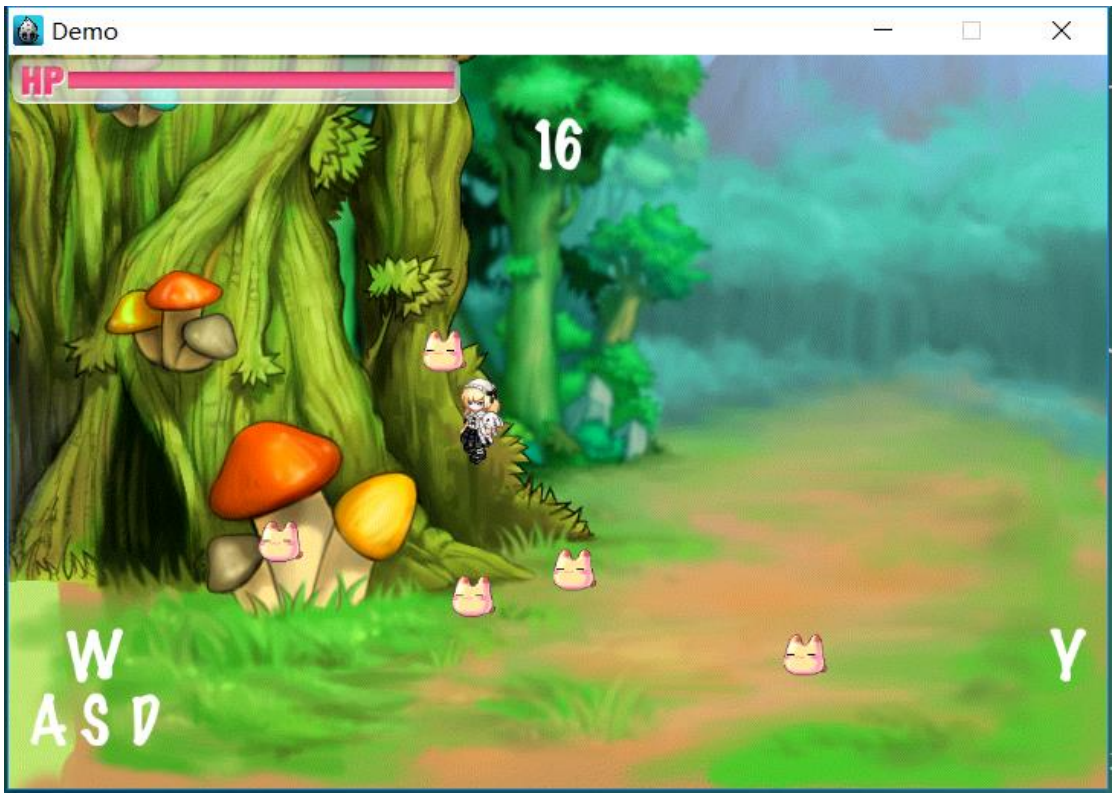
9. 设置周期性调度

```
//周期性调用调度器
schedule(schedule_selector(HelloWorld::updateKill), 0.1f, kRepeatForever, 0);
schedule(schedule_selector(HelloWorld::creatMonster), 5.0f, kRepeatForever, 0);
schedule(schedule_selector(HelloWorld::moveMonster), 5.0f, kRepeatForever, 0);
schedule(schedule_selector(HelloWorld::hitByMonster), 0.5f, kRepeatForever, 0);
```

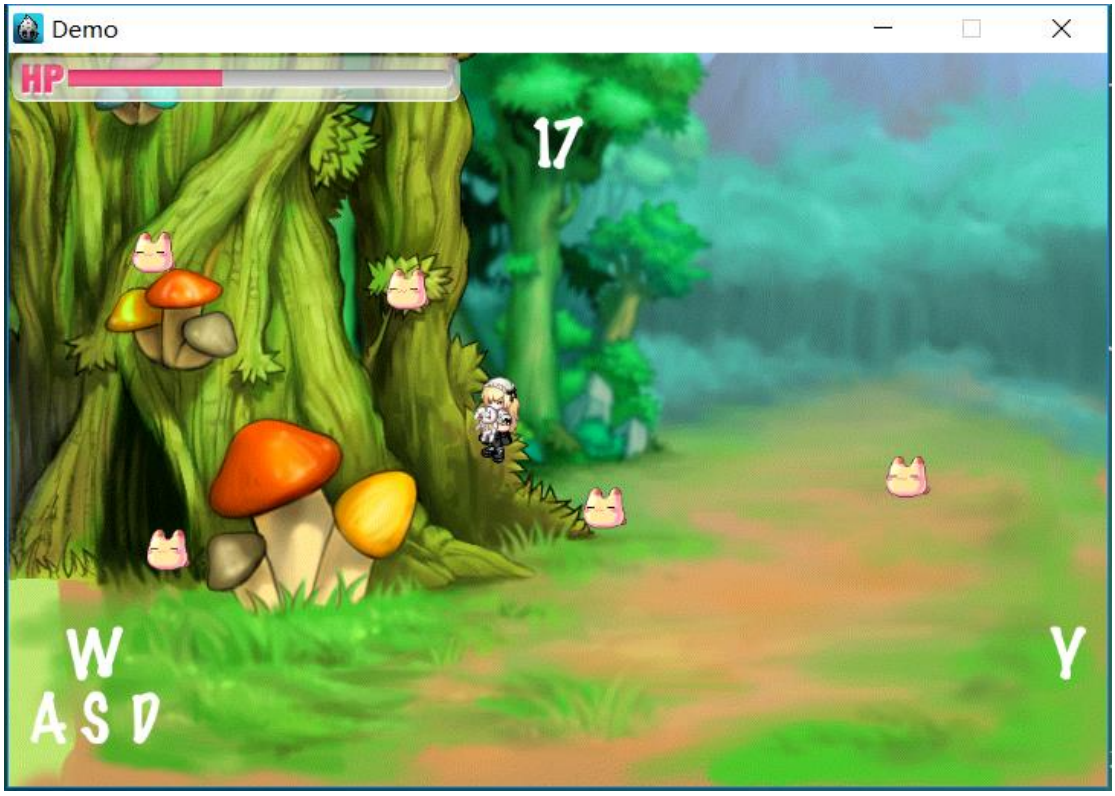
10. 调试项目

三．实验结果截图

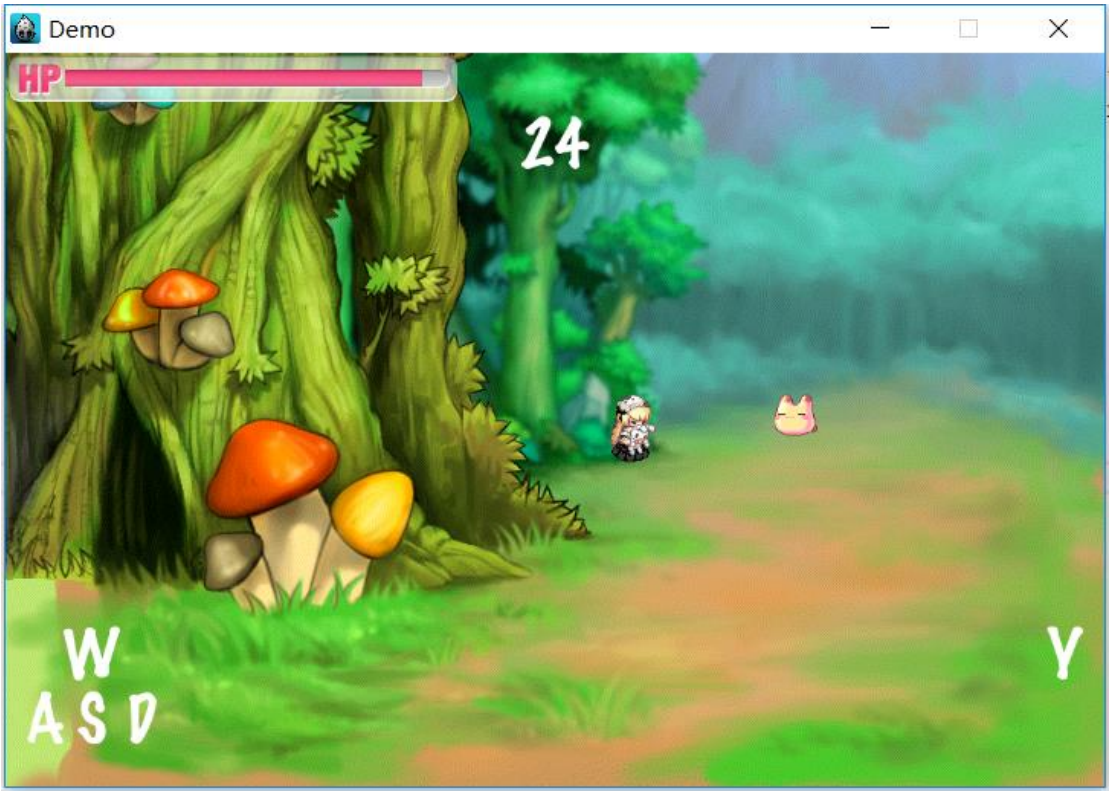
1. 随机产生怪物



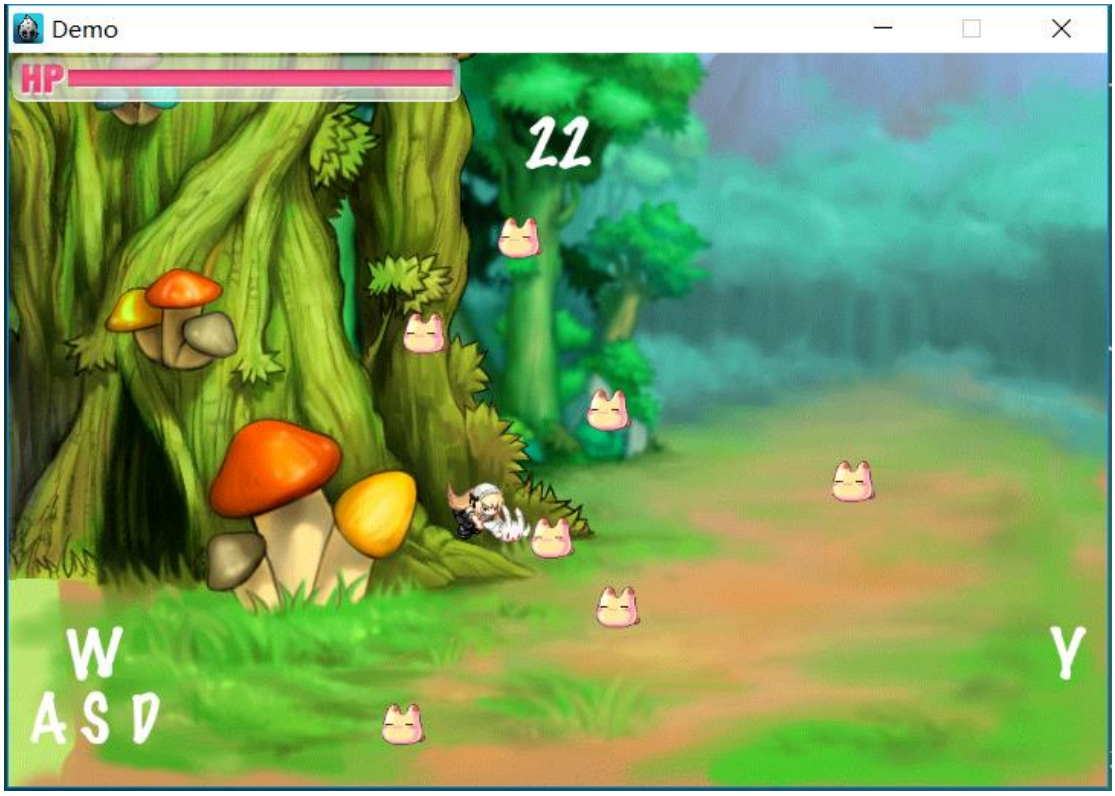
向人物方向移动



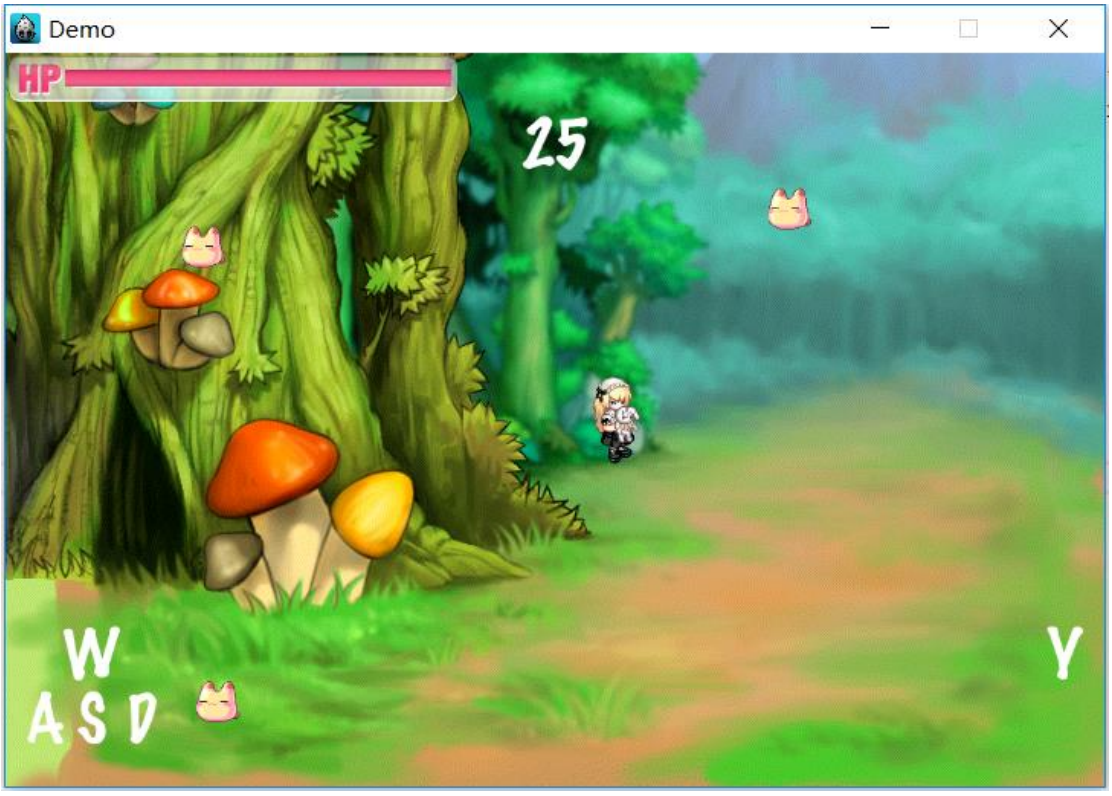
2. 怪物碰到角色后，角色掉血



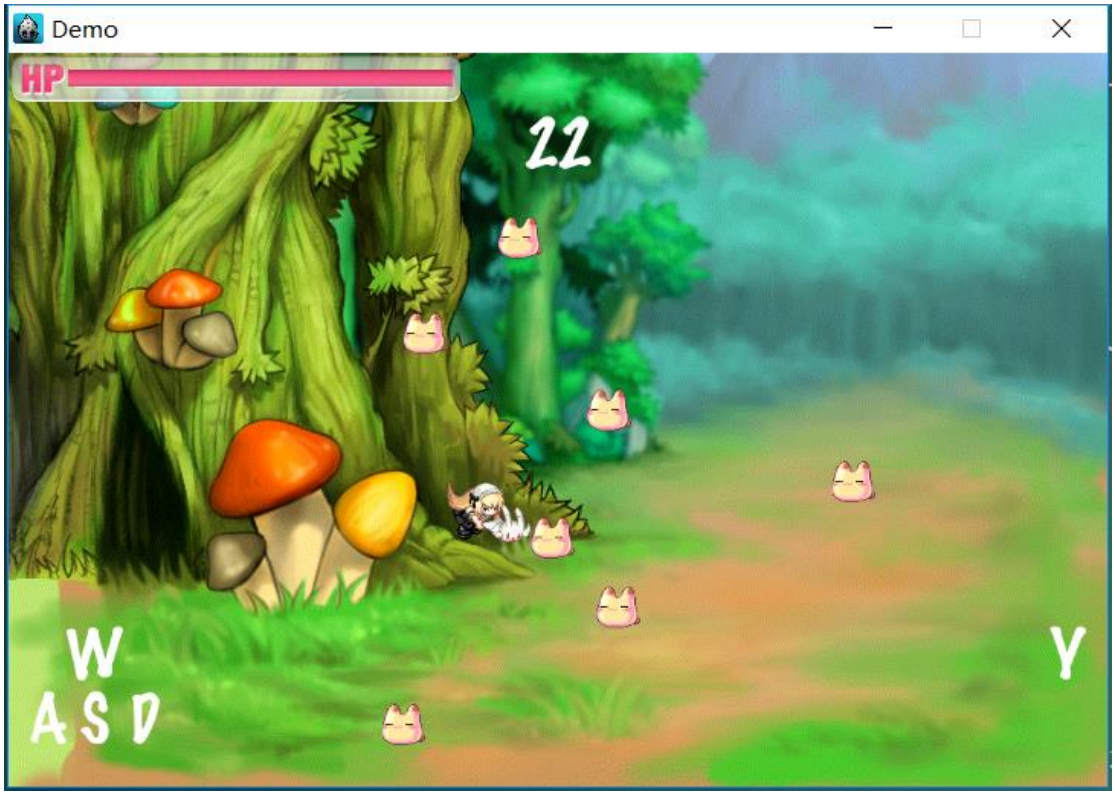
3. 角色可以攻击怪物



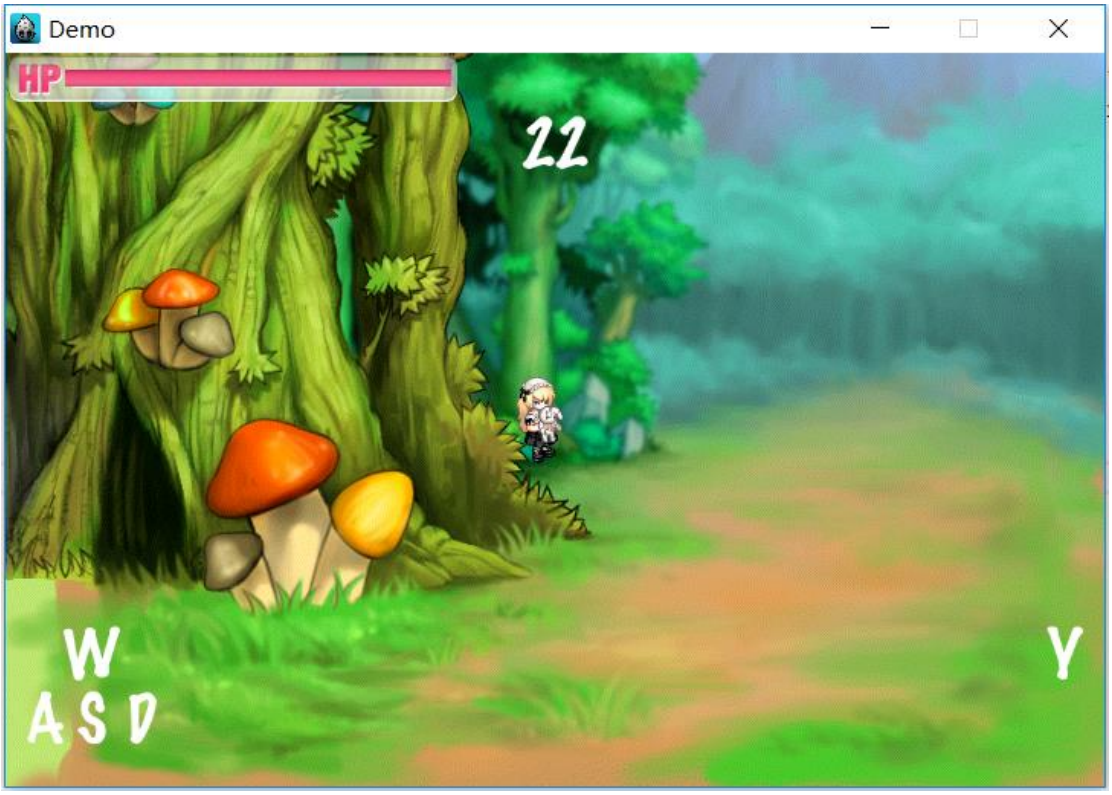
攻击后恢复血量



4. 使用本地数据存储，记录打到的怪物数量



关闭游戏后重新打开，仍然记录怪物数量



四．实验过程遇到的问题

1. 报错：无法解析的外部符号 public: static class cocos2d::Scene * __cdecl M，

enuSence。

根据博客中的方法：在你自己的头文件中加入#include “extensions/cocos-ext.h"，使用命名空间 USING_NS_CC_EXT;，选中工程右键“属性” ->"配置属性 “->"c/c++"->"常规” ->"附加包含目录 “中添加$(EngineRoot)、$(EngineRoot)cocos\editor-support、$(EngineRoot)cocos解决。

2. 导入地图的时候出现异常 layerInfo->_tiles 是 0x1110112，用 tilemap 新建地图的时候 TileLayerfomat 应选择为 Base64(无压缩)解决。
3. 地图黑屏，用 resource 文件夹中的图片创建 tmx 文件解决

五．思考与总结

1. 真看似简单的一个小游戏，实现起来还是遇到很多问题，需要通过网络搜索，经过这次作业，对帧动画和调度器有了更深的了解。
2. 把程序分解成一个个小的部分，分而治之，更有效率而且更容易排错。
3. 重用代码时需要细心，复制后需要更改对应的变量。