Cocos2dx数据结构，本地存储和tilemap

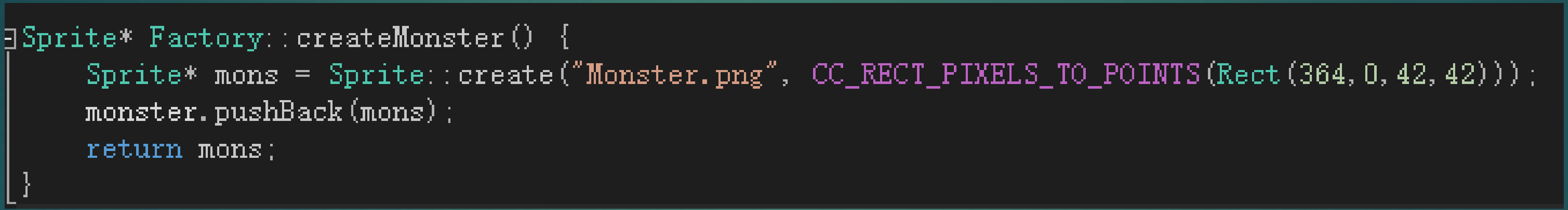
雷志强

作业

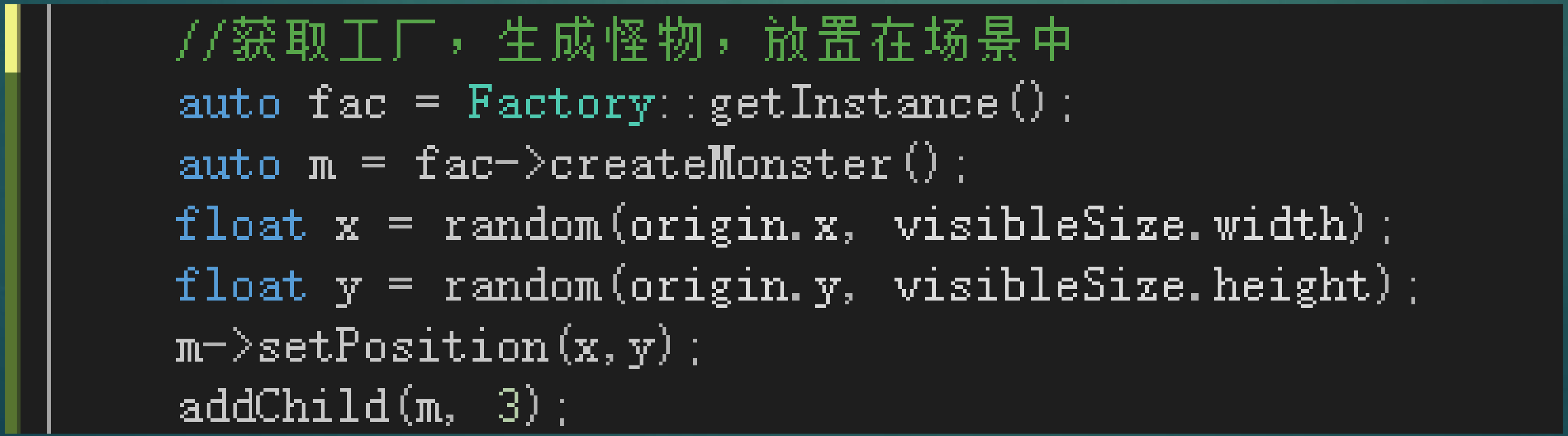
- 在上次横版游戏的作业基础上，加上怪物，需要实现以下功能：
- 1.随机产生怪物
- 2.怪物碰到角色后，角色掉血
- 3.角色可以攻击怪物
- 4.使用tilemap创建地图
- 5.加分项：使用本地数据存储，记录打到的怪物数量，并将倒计时改为显示打倒数量。

一些实现思路

- 1.随机产生怪物，根据游戏边界，随机选取游戏内一个坐标生成一个怪物。



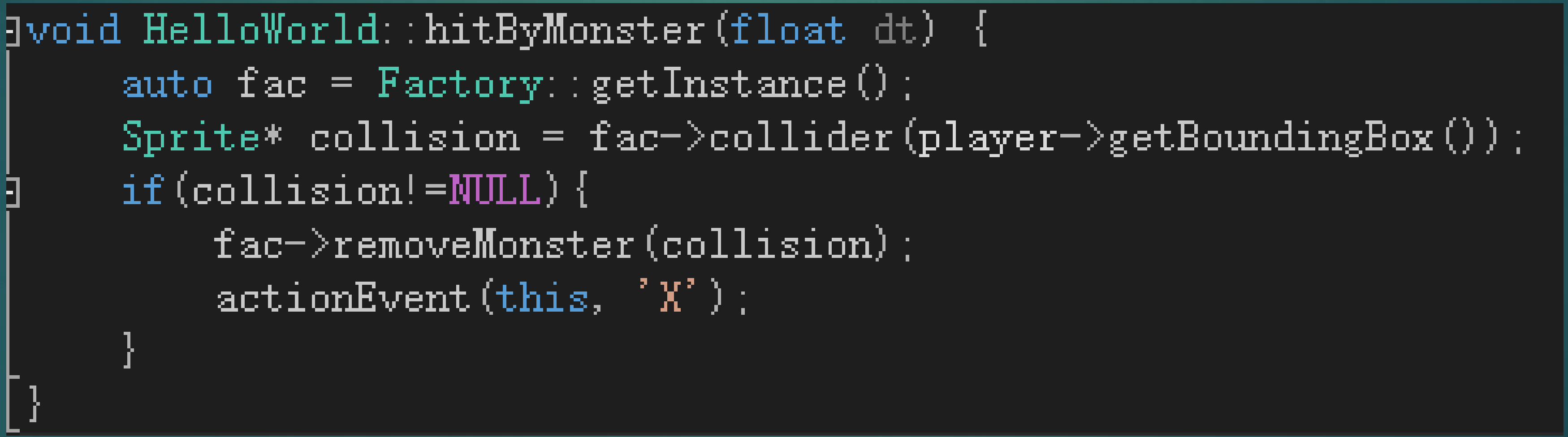
```
Sprite* Factory::createMonster() {
    Sprite* mons = Sprite::create("Monster.png", CC_RECT_PIXELS_TO_POINTS(Rect(364,0,42,42)));
    monster.pushBack(mons);
    return mons;
}
```



```
    //获取工厂，生成怪物，放置在场景中
    auto fac = Factory::getInstance();
    auto m = fac->createMonster();
    float x = random(origin.x, visibleSize.width);
    float y = random(origin.y, visibleSize.height);
    m->setPosition(x,y);
    addChild(m, 3);
```

一些实现思路：

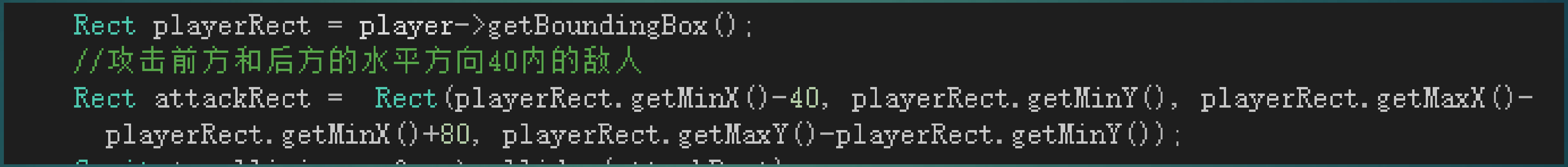
- 2.怪物碰到角色后，角色掉血，使用Rect类中的containsPoint做简单的碰撞检测（例如怪物的坐标在角色的Rect中时，发生碰撞）移除碰撞的怪物，并让角色掉血



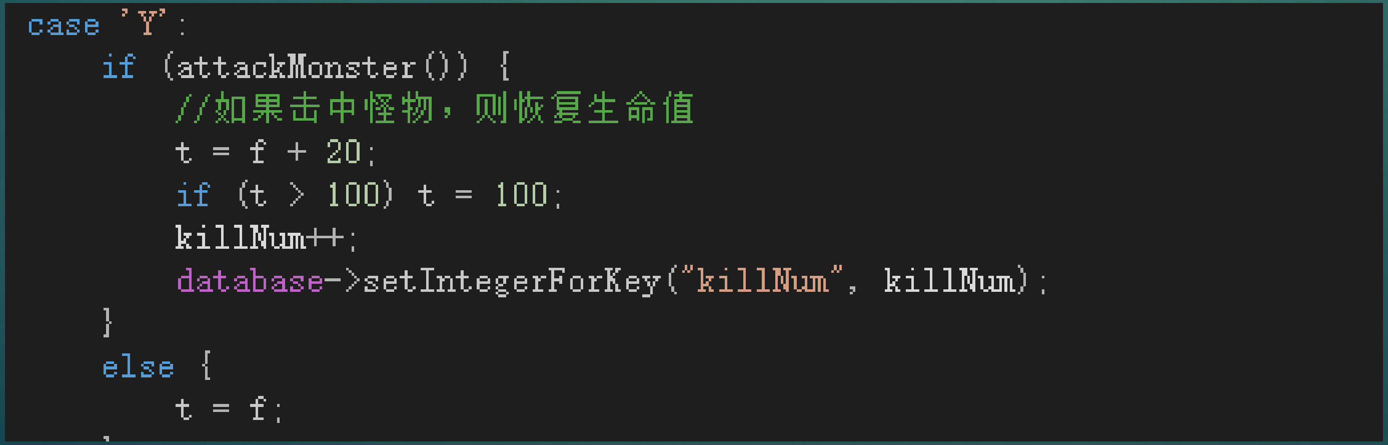
```cpp
void HelloWorld::hitByMonster(float dt) {
    auto fac = Factory::getInstance();
    Sprite* collision = fac->collider(player->getBoundingBox());
    if(collision!=NULL){
        fac->removeMonster(collision);
        actionEvent(this, 'X');
    }
}
```

一些实现思路：

- 3.角色可以攻击怪物，打到怪物后回血，使用2中的碰撞检测，将检测的Rect改成我们设置的攻击范围（Rect），再判断Rect中是否有怪物，如果有怪物，消灭一只怪物，并回血。



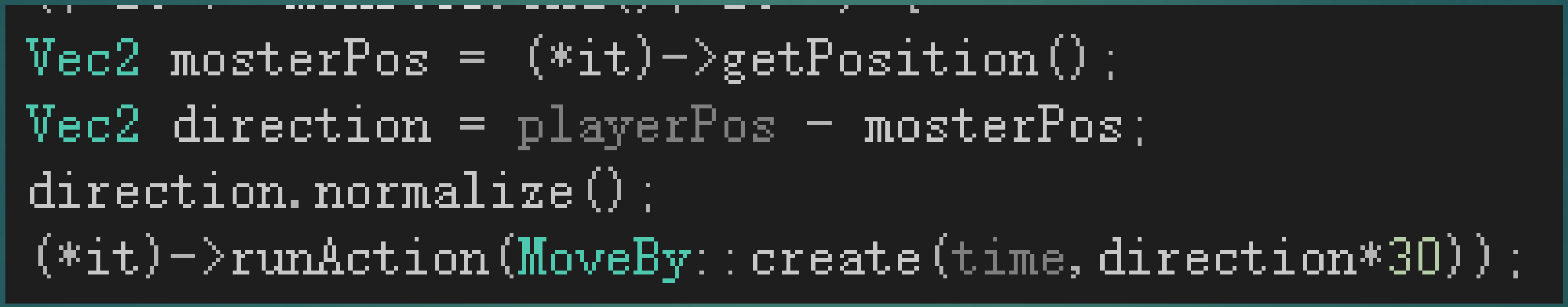
```
Rect playerRect = player->getBoundingBox();
//攻击前方和后方的水平方向40内的敌人
Rect attackRect =  Rect(playerRect.getMinX()-40, playerRect.getMinY(), playerRect.getMaxX()-
  playerRect.getMinX()+80, playerRect.getMaxY()-playerRect.getMinY());
```



```
case 'Y':
    if (attackMonster()) {
        //如果击中怪物，则恢复生命值
        t = f + 20;
        if (t > 100) t = 100;
        killNum++;
        database->setIntegerForKey("killNum", killNum);
    }
    else {
        t = f;
```

一些实现思路：

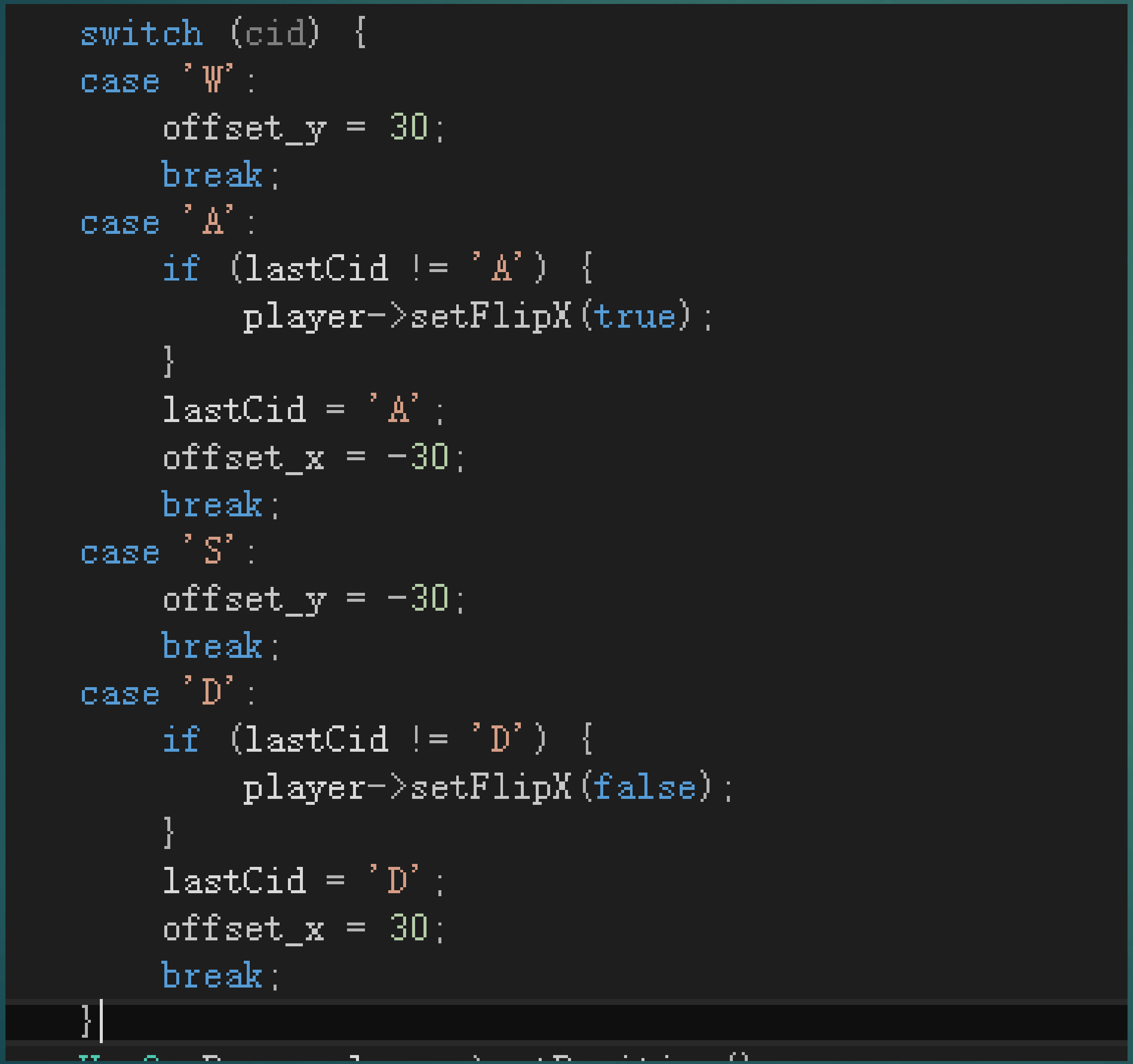
- 4.怪物移动，怪物的坐标和角色的坐标都是Vec2（向量），作减法就可以获得从怪物到角色的方向



```
Vec2 mosterPos = (*it)->getPosition();
Vec2 direction = playerPos - mosterPos;
direction.normalize();
(*it)->runAction(MoveBy::create(time,direction*30));
```

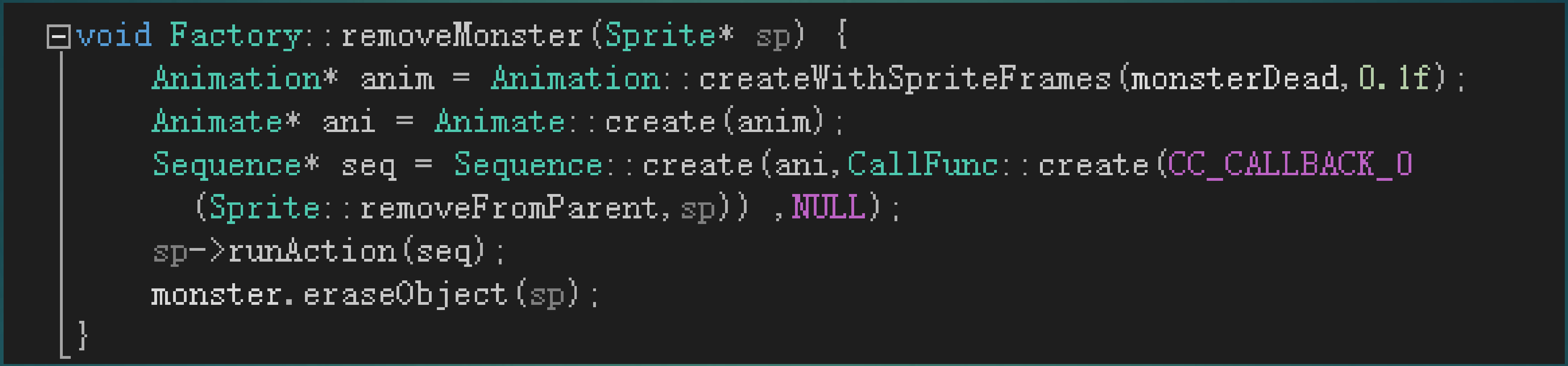
- 实现的思路仅供参考，在碰撞判断，攻击范围的设定上有着更好的实现，不必拘泥于思路中的实现方法，能达到相应的效果即可

翻转的实现思路：



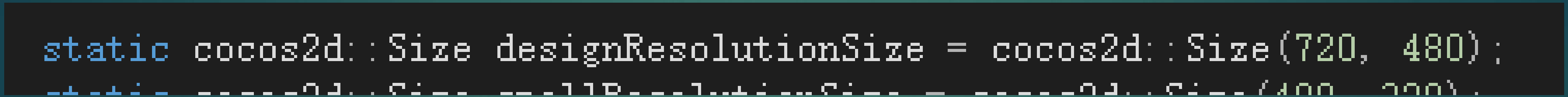
```
switch (cid) {
case 'W':
    offset_y = 30;
    break;
case 'A':
    if (lastCid != 'A') {
        player->setFlipX(true);
    }
    lastCid = 'A';
    offset_x = -30;
    break;
case 'S':
    offset_y = -30;
    break;
case 'D':
    if (lastCid != 'D') {
        player->setFlipX(false);
    }
    lastCid = 'D';
    offset_x = 30;
    break;
}
```

移除怪物的实现思路：

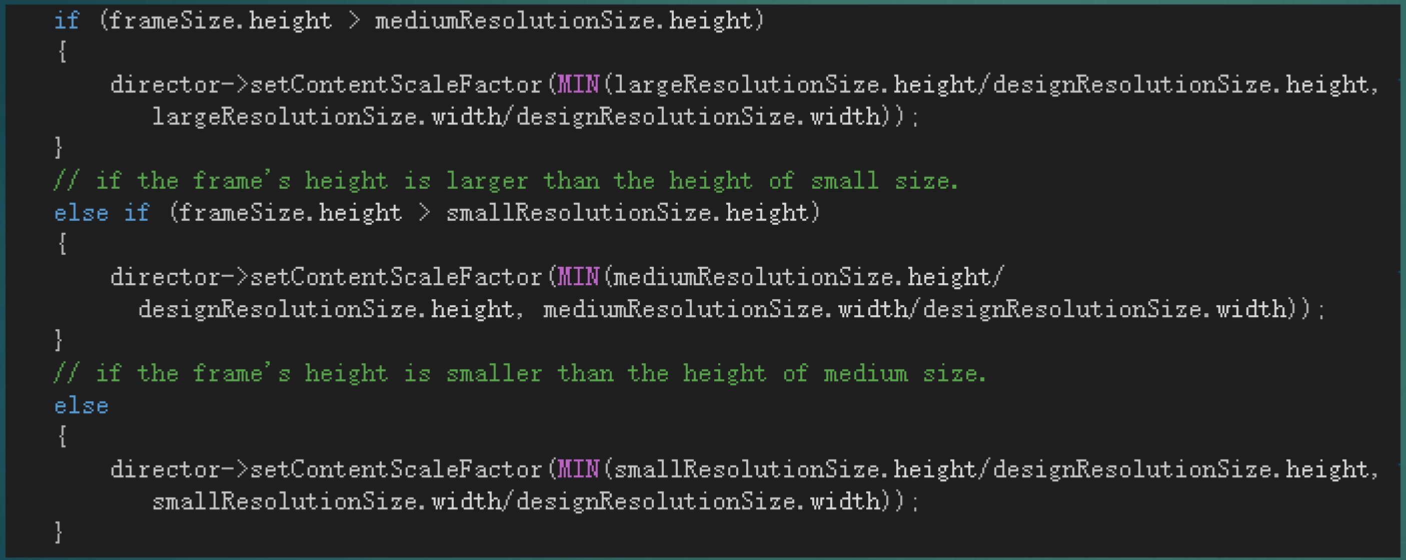


```
void Factory::removeMonster(Sprite* sp) {
    Animation* anim = Animation::createWithSpriteFrames(monsterDead,0.1f);
    Animate* ani = Animate::create(anim);
    Sequence* seq = Sequence::create(ani,CallFunc::create(CC_CALLBACK_0
      (Sprite::removeFromParent,sp)) ,NULL);
    sp->runAction(seq);
    monster.eraseObject(sp);
}
```

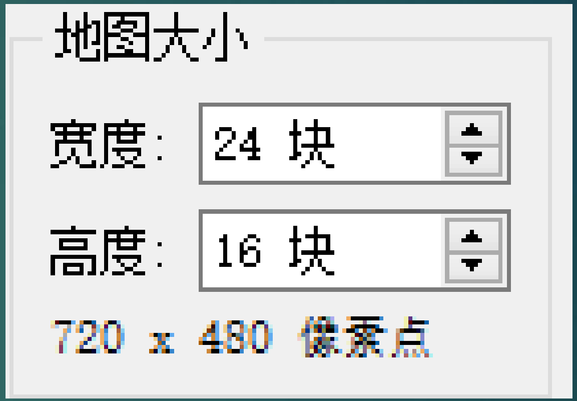
Tmx文件和屏幕的适配：根据放大因子大小来设置



```cpp
static cocos2d::Size designResolutionSize = cocos2d::Size(720, 480);
static cocos2d::Size smallResolutionSize = cocos2d::Size(480, 320);
```



```cpp
if (frameSize.height > mediumResolutionSize.height)
{
    director->setContentScaleFactor(MIN(largeResolutionSize.height/designResolutionSize.height,
       largeResolutionSize.width/designResolutionSize.width));
}
// if the frame's height is larger than the height of small size.
else if (frameSize.height > smallResolutionSize.height)
{
    director->setContentScaleFactor(MIN(mediumResolutionSize.height/
      designResolutionSize.height, mediumResolutionSize.width/designResolutionSize.width));
}
// if the frame's height is smaller than the height of medium size.
else
{
    director->setContentScaleFactor(MIN(smallResolutionSize.height/designResolutionSize.height,
       smallResolutionSize.width/designResolutionSize.width));
}
```

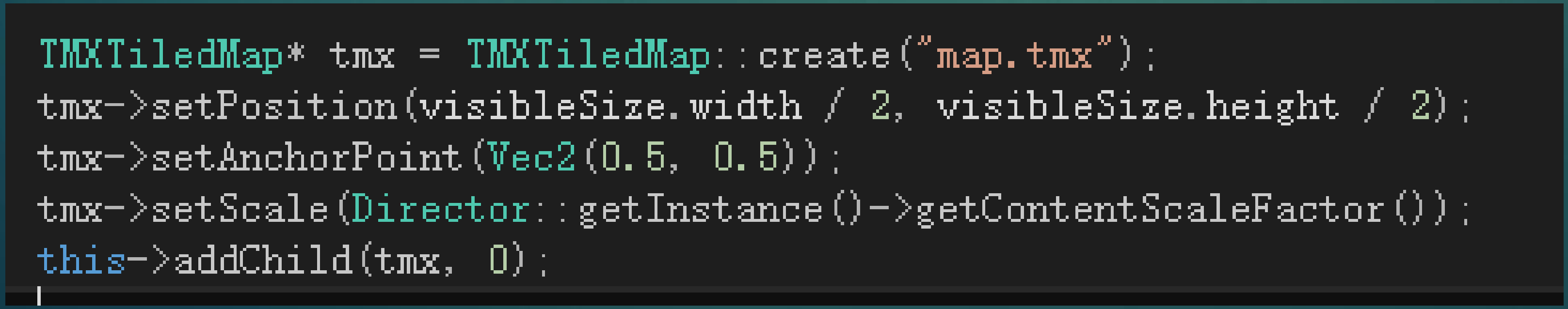


地图大小

宽度：24 块

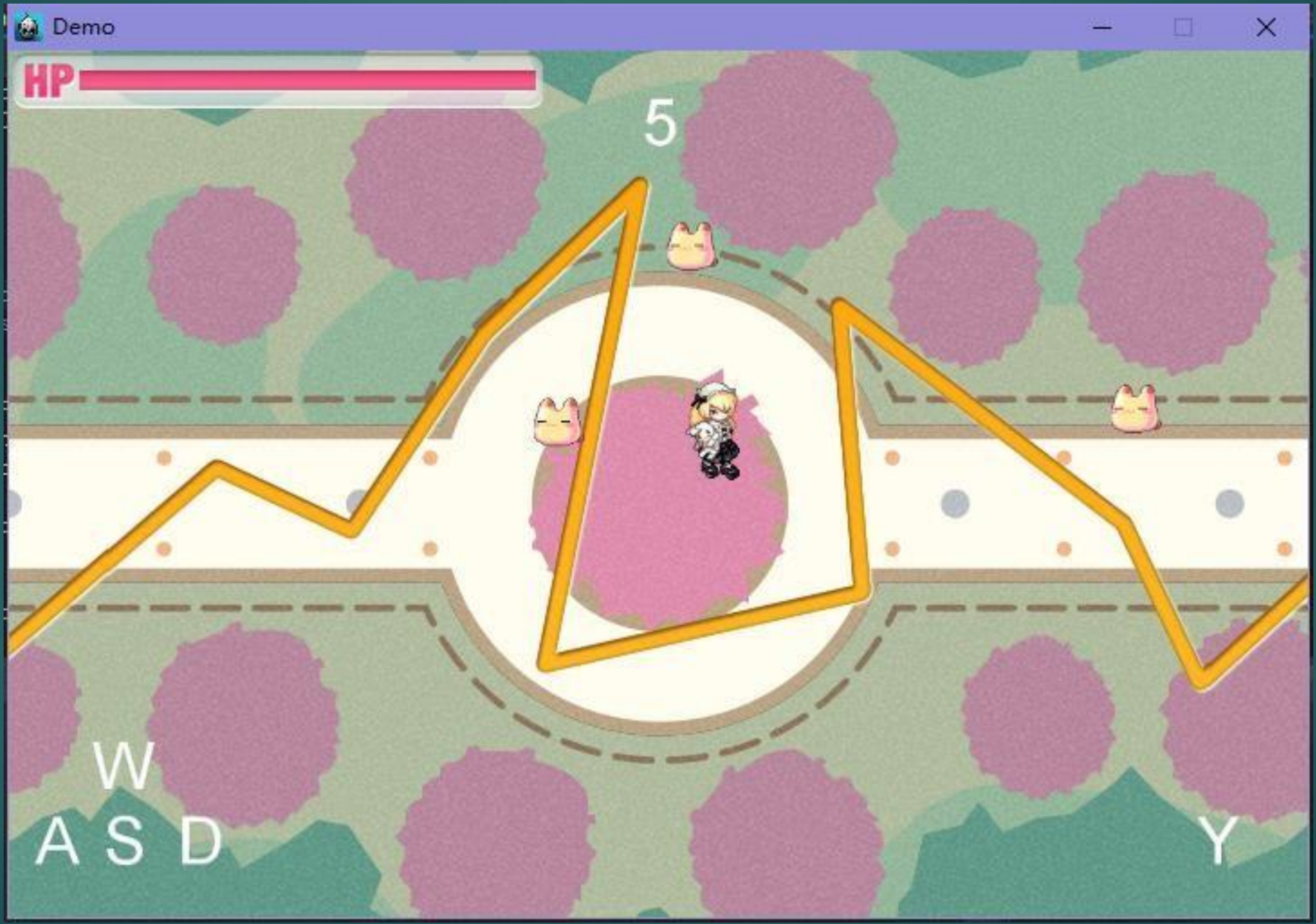
高度：16 块

720 x 480 像素点



```cpp
TMXTiledMap* tmx = TMXTiledMap::create("map.tmx");
tmx->setPosition(visibleSize.width / 2, visibleSize.height / 2);
tmx->setAnchorPoint(Vec2(0.5, 0.5));
tmx->setScale(Director::getInstance()->getContentScaleFactor());
this->addChild(tmx, 0);
```

Demo展示



THE END

THANKS FOR WATCHING