中国科学: 信息科学

SCIENTIA SINICA Informationis

论文

Developing a Java Game from Scratch

赵一霖[1]

1. 南京大学, 南京, 210000

E-mail: 191220174@smail.nju.edu.cn

收稿日期: 2021-12-30; 接受日期: 2021-12-30; 网络出版日期: 2021-12-30

摘要 通过一学期在高级 java 程序设计课上的学习，完成的课程项目的开发目标，设计理念等，以及自己的心路历程。

关键词 Java, 坦克大战

引用格式: 赵一霖. Developing a Java Game from Scratch. 中国科学: 信息科学, 在审文章
yilier. Title for citation (in Chinese). Sci Sin Inform, for review

1 开发目标

我写的这个游戏，灵感来源于坦克大战。一写完那个迷宫作业以后，立马我就感觉能整个坦克大战出来。但地图不使用坦克大战的那个默认地图，也不需要保护基地，地图就一直沿用随机生成出的迷宫，将某些特定地方的墙去掉，将不同的葫芦娃作为多人联机的玩家而小妖精作为敌人，两者之间互相发射子弹攻击，一个不那么经典的坦克大战游戏的雏形就这么出来了。

2 设计理念

代码的大体框架一直没怎么变过，直接用 asciiPanel 中的内容在终端显示图片，直接在 world，worldscreen 等类中修改添加各种交互属性。

2.1 游戏主场景设计

World 中存有 tiles 数组，这个 tiles 可以是墙壁（可破坏），空地，子弹，敌人，玩家等各个单位。每个单位都有自己的图标，颜色等。通过之前走迷宫的方法相应按键控制玩家行走与攻击。不可移动到墙壁上，但可以主动移动到子弹上并扣血。

相关代码见图1, 图2

```
for (int i = 0; i < WIDTH; i++) {
    for (int j = 0; j < HEIGHT; j++) {
        if(i%5==0 || j%5==0){
            if(i!=0&&j!=0)
            maze[i][j] = 1;
        }
        if(i>=12&&i<=18&&j>=12&&j<=18){
            maze[i][j]=1;
        }
        tiles[i][j] = new Tile<>(i, j);
        if(maze[i][j]==0){
            tiles[i][j].setThing(new Wall(this));
        }
        else tiles[i][j].setThing(new Floor(this));
    }
    hpthread hhh = new hpthread();
    hhh.start();
}
```

图 1 从迷宫数组中读取地图信息并生成 Tiles 数组

2.2 敌人设计

敌人则通过一个线程控制，这个线程在敌人的生命值大于 0 时运行，通过 worldscreen 获得玩家的位置，并坚定不移地朝玩家移动。移动只会往空地上走，也就是说敌人不会去撞子弹，这样还赋予了敌人和你隔了一面墙 peek 走位的能力。原本的设计可能会被墙角卡住（因为不是正经 DFS 等寻路算法），之后简单调整了一下被卡住时的行为，被卡住时会破坏墙壁，这样就可以确保不会被卡住。敌人的移动目的地是与玩家处于同一水平或垂直线上，并且没有障碍物（墙壁，敌人）阻挡，这样就会朝玩家发射子弹。这样写可以保证敌人不会攻击到自己的友军。

相关代码见图3, 图4

2.3 子弹设计

子弹一旦打出，不分阵营，无论打到谁都会扣血。子弹是一个 runnable 类，在创造时设定一个方向，子弹就会在 world 中朝这个方向改变自己的索引值。如果目标地点是空地就接着移动，如果是墙壁、玩家或敌人就扣除他们的生命值。

相关代码见图5

3 技术问题

3.1 并发与 IO

并发编程时没遇到什么大问题，写好 runnable 类后直接 run 然后就不管了。在实现保存读取进度时使用到了 IO 部分的功能。因为 C++ 的 IO 用的也比较多了，在编写过程中也没有遇到太大问题，就是打开文件 File 类后，又打开了一个字符读流 Reader，还得继续打开一个带缓冲区的 Bufferreader 才可以一次一行地读数据，几个层次没太套清楚最后关的时候还得从内到外按顺序关。

```
public void run(){
    while(true){
        try{
            Thread.sleep(100);
            if(score == 3){
                if_win = 1;
            }
            for(int i=0;i<30;i++){
                for(int j=0;j<30;j++){
                    if(tiles[i][j].getThing().getGlyph()!=(char)32){
                        if(tiles[i][j].getThing().hp<=0){
                            if(tiles[i][j].getThing().getGlyph()==(char)177){
                                put(setFloor(),i, j);
                            }
                            else if(tiles[i][j].getThing().is_dead == 0){

                                tiles[i][j].getThing().is_dead = 1;
                                if(tiles[i][j].getThing().getGlyph() == (char)1){

                                    put(setFloor(),i, j);
                                }
                                else if(tiles[i][j].getThing().getGlyph() == (char)2){
                                    score++;
                                }
                                put(setFloor(),i, j);
                            }
```

图 2 监控各个单位生命值的线程

```
for(int i=0;i<4;i++){
    int t = Math.abs(world.screen.bros[i].getX()-this.getX())+Math.abs(world.screen.bros[i].getY()-this.getY());
    if(t < m && world.screen.bros[i].hp > 0){
        tar = i;
        m = t;
    }
}
tx = world.screen.bros[tar].getX();
ty = world.screen.bros[tar].getY();
```

图 3 获取离自己最近的玩家作为目标

不过最后还是顺利解决了

3.2 联机实现

目前遇到最大的技术问题是写联机功能的时候。感觉 NIO 过于抽象（尤其是当时还没好好听课），最后在 B 站，廖雪峰 java 教程等地方一天速成 NIO 并利用教程的聊天室范例代码，修改而成了自己的网络通信功能。服务器接受的信息是客户端对应的玩家以及操作的键码，在自己维持的 world 上运算，并生成此时的地图信息，而服务器发送给每个玩家的信息就是实时的地图信息，玩家收到地图信息后解码出地图并显示在终端上。服务器发送给所有玩家的信息一致，以保证所有玩家看到的画面都是相同的。客户端接受到长信息时判断为地图，并将它拆开放入数组中，repaint 则打印出数组中的内容。

相关代码见图6, 图7.

```
int readyattack = 0;
if(tx == this.getX()||ty == this.getY()){
    readyattack = 1;
    if(tx == this.getX()){
        if(ty>this.getY()){
            for(int i=this.getY()+1;i!=ty;i++){
                if(world.get(tx, i).getGlyph()!=(char)32 && world.get(tx, i).getGlyph()!=(char)250){
                    readyattack = 0;
                    break;
                }
            }
            if(readyattack == 1){
                if (ty==this.getY()+1){
                    world.screen.bros[tar].hp--;
                }
                else{
                Thread t = new Thread (new Bullet(world,4,this.getX(),this.getY()+1));
                t.start();
                }
            }
        }
        else{
            for(int i=this.getY()-1;i!=ty;i--){
```

图 4 每次循环要么攻击要么移动

3.3 面向对象的优势

面向对象的设计方法让我知道了，我的什么操作该在什么类中实现，什么类可以干什么、保存什么。这让我对我自己代码的结构的熟悉和掌握程度大大提高，如果哪一步的表现超出预期，我不费吹灰之力就可以定位到可能出错的方法并改正它。并且在复用性上，面向对象显然有着极大的优势。就比如说在从单机转网络对战这一部分，worldscreen 的内容有极大差别而子弹、敌人等类又不需要有任何变动。如果面向过程的编码修改时可能会伤筋动骨，而面向对象时，我只需要修改一下 worldscreen 的初始化方法使其生成对战专用 world，然后将回应按键的方法内容搬到一个加了对应玩家编号的新方法里就行了。我们面向对象真是太棒辣！

4 工程问题

说来惭愧，我在编码过程中并没有主动使用什么能提高代码质量的设计模式和方法等，都是脑子里一想这里应该有什么功能，就直接现加一个方法，发现这个地方需要持续监控某个条件或属性，就直接加一个专门的线程。

5 课程感言

非常优秀的课程，非常有趣的老师，我愿称为本学期（我学的课）最强

6 心路历程

这学期选的高程和 java 都是能做游戏的大课，任务都相当繁重但是结束后的成就感和收获却能让我感到极大满足。作为一个并不优秀的学生，我坚持了下来并成功做出了一个符合要求的游戏，相比半路感觉难度大或任务重就退课的同学们来说（两门课都到学期末少了将近一半人），我感觉我还挺强的，当然主要是能自己做个游戏出来真的很吸引我。然而学期结束的时候看到大佬们的作品确

```
try{
    Thread.sleep(300);
    lastx = posx;
    lasty = posy;
    if(dir == 1)
    posx--;
    else if(dir == 2)
    posy--;
    else if(dir == 3)
    posx++;
    else posy++;
    if(posx<0||posx>=30||posy<0||posy>=30){
        dir = -1;
        world.put(new Floor(world), lastx, lasty);
        continue;
    }
    if(world.get(posx,posy).getGlyph() == (char)32){
        world.put(this, posx, posy);
        world.put(new Floor(world), lastx, lasty);
    }
    else if(world.get(posx,posy).getGlyph() == (char)1 || world.get(posx,posy).getGlyph() == (char)2 || world.get(posx,posy).getGlyph(
        world.get(posx,posy).hp --;
        System.out.println("mingzhong");
        world.put(new Floor(world), lastx, lasty);
        dir = -1;
    }
    else {
```

图 5 子弹线程，控制它该继续移动还是击中目标并扣血

实震惊到了，并感叹人和人的差距有时候真的比人和狗的差距还大，当然挫败感几乎没有毕竟我对自己的水平还是有清醒的认识的（说白了就是不思进取）。java 考试是真的很牛，除了字节码我感觉每个题我都能写出来东西但都不敢保证是对的，但有着老曹“认真完成作业就能及格”的保证，我还是不太担心成绩的，毕竟每个作业我都尽力完成按时提交了哎嘿。

学期之初，这门课和高程讲的几乎重合，像是面向对象啊，继承啊，namuda 表达式啊，泛型啊什么的，学起来相对轻松，作业也偏向简单，大部分框架都有基本上只需要几十分钟就能搞定内容。从并发编程开始，突然操作系统的东西出来了，噩梦被唤醒，之后的网络通信更是两眼一抹黑，作业也是越来越有难度，Java 这座山突然就高大了起来。我清晰的记得，当我还在完成作业 5 的时候，作业 6 和 7 的要求放了出来，看到那一条条要求，尤其是作业 7 的“实现网络对战功能”，我直接震惊了，想的就是“这真的是我能写出来的东西吗”，这种心态有点像程设实验之类的刚开始写大项目的时候，总想着“这东西真不是给人写的”，最终写完了之后感叹一句“我真牛逼”。

存取进度的作业还好。刚写完高程游戏的自定义地图功能，直接一套，存上当前所有格子，然后把所有线程再一 run，这进度就走下去了。自动构建和单元测试这一块的时候，搜各种教程安装 maven，配环境，再 vscode 里新建一个 maven 规定的项目格式，一编译给抱一千七百多个格式错误，又整了半天看如何跳过 check-style 检测，输了能查到的所有似乎有用的指令都不靠谱，最后还是在 pom 文件里面改了属性才解决。然后才开始一个个写测试样例，结果发现测试不会模拟按键导致我占大部分的放在 responsetoKeypressed 中的代码测试不进去，而且 run 线程的时候好像也只能等 run 完了才会往下执行，导致覆盖度上不去。（这个覆盖度也是，watch 了半天显示是 0，最后得输 jacoco 的指令才能显示出来）。

用 NIO 写网络联机的时候真的是，完全不知道咋写，跟着教程写完的东西就像是一个黑盒，直接把 socket 接口当 IO 写，跟存取地图时一样把 stream 往里一摆，哎好像还真可行！考虑到需要所有玩家的画面同步，干脆客户端不负责计算，只负责显示，所有的指令交给服务端并由服务端计

```
readchannel.register(selector, SelectionKey.OP_READ);

if(code.length()==1){
    System.out.println("receive number");
    System.out.println(code);
    client.number = Integer.valueOf(code);
}
else if(code.length()==3){
    String []arr = code.split(" ");

    client.paintWinner(Integer.valueOf(arr[1]));

}
else if(code.length()>10){
    String []arr = code.split(" ");

    for(int i=0;i<4;i++){
        client.hp[i]=Integer.valueOf(arr[i]);
    }
    for(int i=4;i<904;i++){
        int j = i - 4;
        client.map[j/30][j%30] = Integer.valueOf(arr[i]);
    }
    client.repaint();
}
```

图 6 客户端接受到长信息时判断为地图，并将它拆开放入数组中，repaint 则打印出数组中的内容

算，将计算结果发送给所有的客户端，这样就能保持一致。

最终，写完这篇报告，总算是完成了所有作业。虽然这个小游戏比较简陋，但真的让我有慢慢的成就感。一想到放假可以在七大姑八大姨面前展示这个小巧而不失精美的作品，感受他们震撼的眼光，我就激动的浑身颤抖。我也可以自豪说出“我也是拿 java 写过游戏的人了”。完结撒花。

```
}
if(selectionKey.isReadable()){
    SocketChannel readchannel = (SocketChannel)selectionKey.channel();
    ByteBuffer bf = ByteBuffer.allocate(1024);

    int readlength = readchannel.read(bf);
    String code = "";
    if(readlength>0){
        bf.flip();

        code += Charset.forName("UTF-8").decode(bf);
    }
    String []arr = code.split(" ");
    screen.movePlayer(Integer.valueOf(arr[0])-1, Integer.valueOf(arr[1]));


    readchannel.register(selector, SelectionKey.OP_READ);
    //广播给其他客户端
    if(code.length()>0){
        System.out.println("receive");
        System.out.println(code);
        castOther(selector);
    }
}
```

图 7 服务端接收到客户端的信息，使用 movePlayer 方法移动玩家，并用 castOther 方法广播给其他所有玩家此时的地图信息

Title

yilier[1]

1. *NanJing University, NanJing* 210000, *China*
E-mail: 191220174@smail.nju.edu.cn