中国科学：信息科学

SCIENTIA SINICA Informationis

论文

Developing a Java Game from Scratch

张津榕 191220158

1. 南京大学
E-mail: 1159695291@qq.com

收稿日期: ; 接受日期:

java-2021 课程群

摘要 本文是一个普通大学生在学习 java 半年后从零开发一个 java 游戏的感想与报告，主要包括游戏的设计理念以及代码细节，和自己学习 java 后对 java 的感想

关键词 开发目标，设计理念，技术问题，工程问题，课程感言

1 开发目标

我的开发目标是完成一个类似于魔塔的 rougelike 游戏，玩家通过四处走动撞击怪物来击杀怪物，每一次攻击造成的伤害 = 攻击力-防御力，一旦撞到怪物，则玩家和怪物之间必须分生死。怪物随时间随机生成，并且越来越强大。因此图鉴是必不可少的，在玩家攻击前可以通过按 F4 查看图鉴来查看附近怪物的攻击与防御力，判断击杀这个怪物合不合算。同时当然不能少了经典的宝石，拾取宝石会增加攻击与防御力，起初我是想将宝石放在墙中，留一道门，通过钥匙开门拾取宝石，但是对于一个 rougelike 而言，要么宝石固定在某几个位置，要么在地图上生成墙 + 门（这样游戏后期玩家完全没地方走了，因为全地图都是墙），因此我设计成宝石附近有怪物守护，想拿宝石必须杀死怪物，这也是一种取舍。游戏中有商店，模仿魔塔的设计，每次购买后下次购买的花费会增加。

图 1 游戏截图

2 设计理念

java 是一门面向对象的语言，因此游戏的所有内容都是由对象实现的，代码的总体设计以 jw04 的 rougelike 分支为模板，ascii 文件夹下的文件负责 ui 的构建;

以 Creature 类作为所有生物的父类，构建了 diamond,monster,player 这三个子类，以 Tile 类的 enum 对象构建地板与墙壁，将 Creature 与 Tile 打包在 World 文件夹下，

World 类构建地图同时利用 CreatureFactory 生成生物，保存生物的信息，这样可以很方便地创建新生物以及调整生物的数据，获取生物对象的信息也很方便，因为都是储存在 world 中的，直接调用 world 中函数即可;

screen 类负责屏幕输出与对用户按键的响应，游戏内容的展示在 playscreen 下完成，将屏幕输出单独划分为一个类可以很方便的输出游戏内容，以便于将重心放在整个游戏的构建下

jw06 添加了 test 文件对工程进行测试，并以 maven 自动构建项目工程，同时 jw07 添加了 netTool 实现网络数据传输与上传，玩家上传得分并获取自己得分排名。

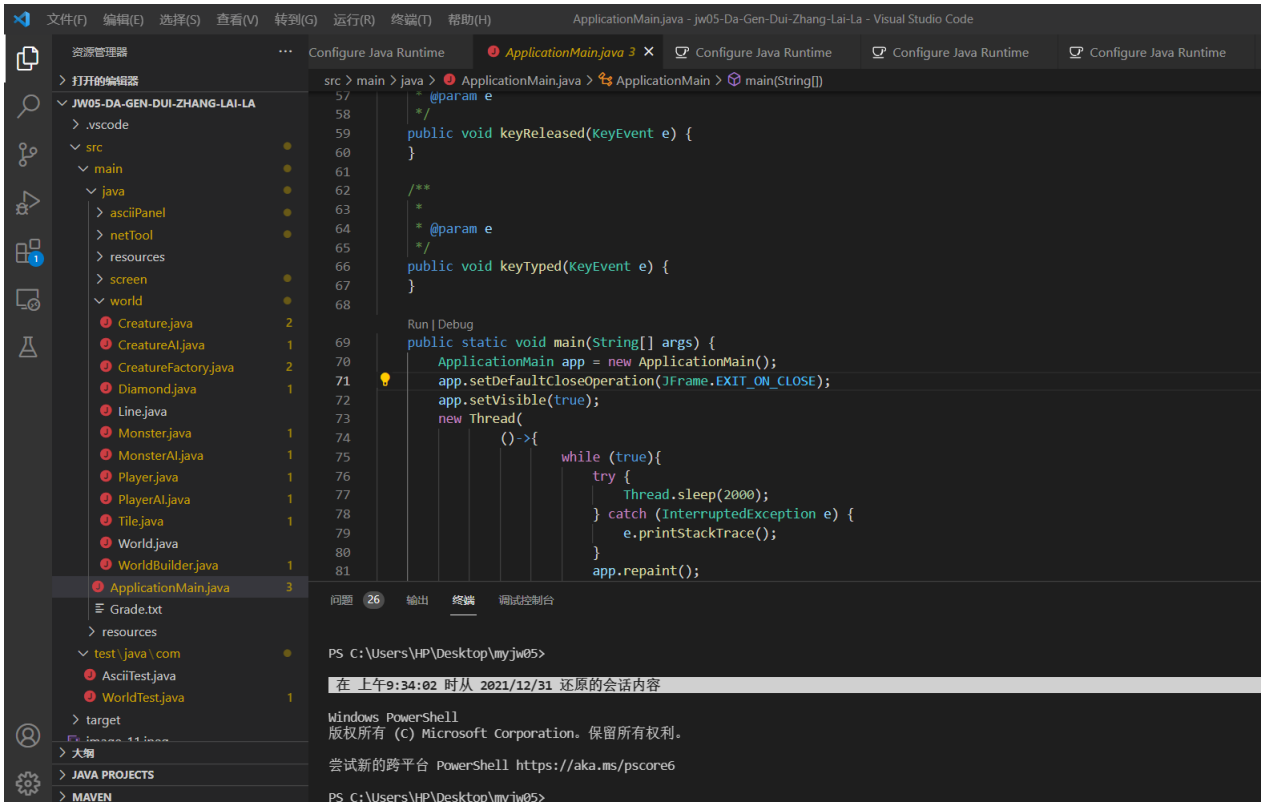

图 2 项目工程框架

3 技术问题

在主函数下调用一个 thread 作为总刷新

```
Run | Debug
public static void main(String[] args) {
    ApplicationMain app = new ApplicationMain();
    app.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    app.setVisible(true);
    new Thread(
            ()->{
                while (true){
                    try {
                        Thread.sleep(2000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    app.repaint();
                }
            }
    ).start();
}
}
```

图 3 主函数线程

同时在 palyscreen 下利用线程实现怪物和宝石的自动创建

```
new Thread(     //每隔一段时间生成宝石
()->{
        while (true){
            try {
                Thread.sleep(40000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            creatureFactory.newDiamond(this.messages);
                }
}
).start();
}
```

图 4 宝石线程

```
new Thread(     //循环生成怪物，时间越长怪物越多越强
()->{
        while (true){
            try {
                Thread.sleep(10000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
                countTime++;
                for(int i=0;i<2;i++){
                Random rand = new Random();
                int level=rand.nextInt(30)+countTime;
                if(level<=10)
                creatureFactory.newShrem(this.messages);
                else if(level<=20)
                creatureFactory.newBat(this.messages);
                else if(level<=30)
                creatureFactory.newSkeleton(this.messages);
                else if(level<=40)
                creatureFactory.newBull(this.messages);
                else
                creatureFactory.newDragon(this.messages);
                }
            }
```

图 5 怪物线程

关于通信部分，在 restartscreen 里，按下 F1 启动 netServer

```
public Screen respondToUserInput(KeyEvent key) {
    switch (key.getKeyCode()) {
        case KeyEvent.VK_ENTER:
            return new PlayScreen();
        case KeyEvent.VK_F1:{
            try {
                NetServer netserver=new NetServer();
                 netserver.serverStart();
            } catch (IOException e) {
                e.printStackTrace();
            }
            return new ThankScreen();
    }
        default:
            return this;
    }
}
```

图 6 开启 server

在游戏失败后回到 losescreen，在里面开启 netClient

```
private void getRank() {
    try {
        NetClient netClient = new NetClient(this.finalGrade);
        this.rank= netClient.getRank();
    } catch (IOException e) {
        e.printStackTrace();
        return;
    }
}
```

图 7 开启 client 向 server 发送信息

netServer 与 netclient 的细节是：
Client 获取得分后向 Server 发送得分，获得排名后返回给 screen

```
//Client与Server互相发消息，Clinet将得分送给Server，获得排名后返回
public class NetClient {
    private String rank;

    public NetClient(int grade) throws IOException {
        String playGrade=String.valueOf(grade);
        InetAddress add = InetAddress.getByName("localhost");
        int port = 8080;
        DatagramPacket sendPacket = new DatagramPacket(playGrade.getBytes(),playGrade.getBytes().length , add
        DatagramSocket localSocket = new DatagramSocket();
        localSocket.send(sendPacket);
        byte[] data2 = new byte[10];
        DatagramPacket receivePacket = new DatagramPacket(data2, data2.length);
        System.out.println("Sending Grade To local host");
        localSocket.receive(receivePacket );
        String arr = new String(receivePacket.getData());
        this.rank=arr;
        System.out.println("Finish");
        localSocket.close();
    }

    public String getRank() {
        return this.rank;
    }
}
```

图 8 client 获取并发送信息

Server 收到 client 发送的得分后读取 Grade 文件，计算排名后发送给 client

```java
//Client与Server互相发消息，Server将得分纪录到排名文件中，返回给Client排名
public class NetServer {
    private final static int Port = 8080;
    private static String[] fileContentArr;
    private static byte[] fileContent;
    public NetServer() {

        try {
            FileInputStream in = new FileInputStream("rougegame/Grade.txt");
           fileContent=new byte[in.available()];
            in.read(fileContent);
            in.close();
            fileContentArr = new String(fileContent).split("\r\n");//按回车分割string
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
```

图 9 server 的构造函数开启文件并读取信息

```java
public  void serverStart ()throws IOException {
    NetServer newServer=new NetServer();
    DatagramSocket localServer = new DatagramSocket(Port);
    byte[] newscore = new byte[1];
    DatagramPacket packet = new DatagramPacket(newscore, newscore.length);
    System.out.println("Server Start");
    localServer.receive(packet);
    String grade = new String(packet.getData());
    int rank=fileContentArr.length+1;
    for(int i=0;i<fileContentArr.length;i++){//获取排名
        if(grade.compareTo(fileContentArr[i])>=0) {
            rank--;
        }
    }
    String fin=new String();
    for(int i=0;i<fileContentArr.length;i++){
        fin=fin+fileContentArr[i]+"\r\n";
    }
    fin+=grade;
    FileOutputStream fos = new FileOutputStream("rougegame/Grade.txt");//写入文件
        fos.write(fin.getBytes());
        fos.flush();
        fos.close();
    InetAddress add = packet.getAddress();
    int cport = packet.getPort();
    String rankArr=String.valueOf(rank);
    byte[] data=rankArr.getBytes();
    DatagramPacket sendPacket = new DatagramPacket(data, data.length, add, cport);
    localServer.send(sendPacket);
    System.out.println("Server Close");
    localServer.close();
}
```

图 10 计算排名后将数据写入 grade.txt，并发送排名给 client

在学习 java 之前，我从未想过代码可以完全使用类来构建整个工程，诚然在之前我也学习过 C++，但是 C++ 的面向对象和 java 又有不同，例如在学习 java 之前我 C++ 的 io 都是在函数中实现的，从未想过构建两个类，一个作为 server 一个作为 client，互相发送信息，这样的编程思路让我受益匪浅，这简直就是对现实世界的完全映射。

4 工程问题

我觉得最实用的设计方法就是《技术问题》中提到的，完全面向对象实现工程，屏幕输出、io 通信，乃至于地图、生物、UI，都是面向对象设计的，这样写出来的工程非常易读，因为每个对象类都是对现实的映射，即使没有完全弄明白部分实现细节，也可以轻松进行修改，例如修改 asciiPanel 完成 UI 的设计，并不需要我完全弄懂 asciiPanel 的实现细节，我只需要根据函数名弄明白每个函数是做什么的，然后根据需要修改函数内容即可。

5 课程感言

在期末考试前几天复习阶段，我每天看文档复习，越看越后悔，要是早点看就好了，期末考试挤一起，好多东西没时间细琢磨，java 这门语言真的很实用，也很好用，同时我也理解了老师为什么要我们从零开始写一个 java 游戏，java 的很多东西靠讲靠看文档是无法体会到的，所谓” 纸上得来终觉浅，绝知此事要躬行” 正是这个道理，在写代码的时候我常常思考，如果这些代码换成别的语言来实现，该怎么写？在语言的互相比较与实际操作之后，我对 java，更准确的说是对面向对象编程这一编程范式，有了更深的理解。

关于课程设计，我觉得这样的设计很好，大作业既有趣，又能学到很多真东西，这些靠讲是没法真正体会的。