中国科学: 信息科学

SCIENTIA SINICA Informationis

论文

Developing a Java Game from Scratch

孙立帆[1]

1. 南京大学软件学院, 南京 210023
E-mail: 201250181@smail.nju.edu.cn

摘要 本文是南京大学计算机科学与技术系 2021 秋季课程《Java 高级程序设计》的大作业总结论文，主要对大作业的开发目标、设计理念、技术问题、工程问题以及课程感言做一个总结。本课程的大作业是使用 Java 开发出一款能支持存档、网络对战等功能的 Roguelike 游戏。本游戏主要使用了 Java swing 来开发图形用户界面，采用序列化来进行游戏存档，采用 nio 的 selector 来进行网络通信，采用 maven 来进行项目构建和依赖管理。

关键词 Java, swing, nio, socket, maven, serialization, object-oriented

1 概述

《Java 高级程序设计》是一门内容十分丰富的课程，从面向对象思想到 Java 的一些高级特性，均有涉及。在这门课上，我学到了很多在大一的 Java 编程学习中都未曾听过的 Java 特性，比如序列化、自动构建等。同时，也对一些以前学过的知识有了更深的理解，主要包括以下:

1. Java 的并发程序设计中管程的概念
2. 如何编写多线程的程序，之前主要是编写一个实现 Runnable 接口的类，然后直接 start 一个新线程，在本课程的学习中学习到了一些更加高级的方法，比如使用线程池来管理线程，避免了创建过多的线程所带来的开销，也简化了管理线程的逻辑。同时还学会了结合利用 lambda 表达式，用更简单的语法来表达同样的事情。
3. Java 网络通信，主要是 NIO 和 reactor 设计模式, 之前都是使用 blocking IO
4. 面向对象的设计方法，之前对于面向对象方法主要还是停留在认知方面，实践的机会比较少。

以上是概述部分。
接下来的部分我将就这次大作业的开发目标、设计理念、技术问题、工程问题和课程感言进行逐一展开。

2 开发目标

《Escape this Dungeon》是一款采用 maven 进行项目构建, 采用 Java 语言进行开发的 2D roguelike game, 游戏有两个模式: 单机模式-PVE , 联网模式-PVP, 玩家初始会出生在一个地牢, 在单机模式中, 玩家的目标就是躲避怪物的伤害, 找到钥匙并从地图中唯一的门逃离地牢, 而在联网模式中, 游戏目标是击败所有其他玩家, 并同时躲避怪物的伤害, 找到地图中的钥匙并逃离地牢。整个游戏过程中, 定时刷新 bat, rat, ghost 三种怪物, 各自拥有不同的属性值, 在地图中某些地方会有抽屉, 打开可以获得金币, 击杀怪物或者其他玩家也可以获得金币, 金币可用于打开地图中的箱子, 打开箱子会获得最大生命提升, 生命恢复, 获得护甲, 攻击提升其中之一种增益效果。游戏的灵感来源于 YouTube 上一个 up 主的回合制的 roguelike 游戏, 本游戏的 gui 以及人物的图片大部分也来自于此。

本来后续想要加入但是由于时间不太够所以暂时没有加入的功能:

1. 商店: 玩家可以使用金币在地图的固定地点购买一些道具, 获得某些特定增益。
2. 在随机空闲位置上生成新玩家: 目前只能支持比较有限数量的玩家, 根据进入游戏的顺序分别生成在不同特定位置, 后续可能会加入在随机位置生成新玩家的机制, 允许更多玩家同时进行游戏。
3. 有限视野: 玩家只能看到自己周围有限的区域, 其他区域将渲染成黑色, 只有进入自己视野范围的物体/生物可以被当前客户端的玩家看见。
4. 在随机位置生成钥匙: 目前的实现钥匙生成在一个固定的地方, 多次进入游戏都是在同一个地方, 这造成了可玩性的下降, 但是目前还没有解决, 后续将实现在随机位置或者让玩家以更加困难的方式来获得钥匙。
5. PVP 模式中怪物的生成, 由于通信问题的原因还没做好。

3 设计理念

游戏采用了面向对象的编程范式, 尽量通过对象之间的交互来模拟游戏的过程。游戏主要分为以下模块:

1. 游戏中的实体。主要是表示生物、子弹的类和表示地图的类。游戏的逻辑主要依靠这几个类的相互通信来完成。
2. 游戏控制模块。包括了控制游戏初始化, 游戏进度, 网络通信, 进度保存的模块等。
3. 图形用户界面。这里用 Java 的 swing 库实现了一个比较简陋的图形用户界面。主要包括了窗口、渲染等。

通过采用面向对象的设计理念，面向问题空间进行求解，通过封装、继承、多态等机制来进行抽象，使得处理一些复杂问题的思路更加清晰，编写代码时的难度降低。

4 技术问题

由于是第一次进行一个相比之前规模稍大的项目开发，对于各种 Java 的高级特性也不甚熟悉，在开发的过程中遇到了一些问题。主要有以下:

1. 并发中的 race condition 问题。一个例子是原来我在游戏控制模块中保存了目前在游戏中当前的所有子弹，存在一个 list 中，在 gui 渲染的时候通过访问这个 list 来看哪些子弹还活跃，活跃的就渲染，不活跃的就从 list 里面删除，由于某些不明的同步问题，在 gui 渲染子弹的时候常常出现问题。最后将 list 换成了同步版本的 vector，虽然牺牲了一些效率但是降低了编程的复杂性。而其他的 race condition 带来的问题我主要通过 synchronize 的使用进行处理，如两个生物不能同时踏上同一个砖块，多个子弹攻击同一个生物应该先后发生等等。

2. 类型设计仍然不是很合理。在编写测试代码的时候发现要测试一个类常常与其他类交叉在一起。这要通过更合理的类型设计来解决，在开发过程中我进行了一些尝试重构了一些代码，比最初的版本稍好但是仍然效果不算理想。在编写类型时应该弄清楚各个类型的职责，及其协作关系，之后尽量编写单一职责的类，虽然道理大概清楚，但是实际上操作起来还是有些难度。

3. 网络通信。要实现同步各个客户端同步画面，不能使用太慢的 IO，在网络通信中我使用了 NIO，以避免使用传统的 blocking IO 所带来的效率问题，同时使用 selector 使用一个线程来管理服务器和多个客户端之间的通信，解决了传统的多线程模型管理网络通信的资源占用问题。同时我设计了一个 Message 接口，通过编写不同的类型 message 类来实现这个接口，利用多态方便地进行消息的解码。

以上遇到的问题主要还是因为对相关方面的知识理解不够深入，相关的库使用不够熟练，虽然通过一些改进能够使得程序正常运行，但是仍然存在许多缺陷。同时由于一些知识，比如计算机网络、操作系统等还没有学过，对有些内容的理解比较粗浅，因此采取的改进措施虽然还算有效，但是从可维护性和可扩展性上看可能不是很好。

后续还可以继续优化的方向:

1. 并发相关的代码：由于当时并发程序设计的知识掌握的不是太好，对于某些并发程序设计中的问题采用了一些比较简单粗暴的方式进行处理，牺牲了一些效率，并且可能埋下了某些新的隐患。

2. 类型设计：类之间的耦合度较高，而且类中的某些方法的设计不是很符合面向对象的思想，造成了在编写后续代码的时候有点别扭。

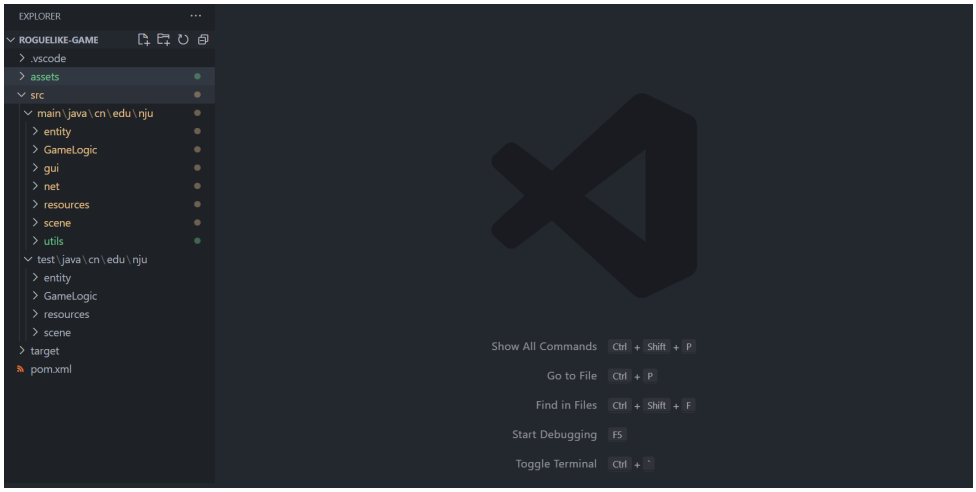

图 1　代码大致情况

Figure 1　Caption

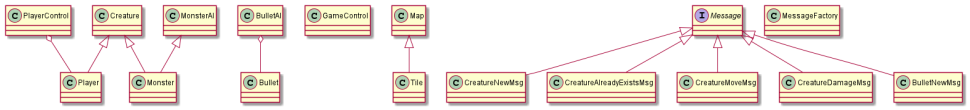

图 2　使用 PlantUML 绘制简略类图

Figure 2　Caption

3. 网络通信：由于对于计算机网络知识的不了解，本次的网络通信部分其实大部分基于课上给出的链接中的 NIO 示例代码进行修改得到，可能有些修改的地方不是很好，但目前不知道有什么问题以及如何解决。

5　工程问题

虽然这并不是一个规模算大的项目，但是在开发过程中仍然会遇到一些工程上的问题，比如相较于以前的课程作业，这次的大作业中要编写的类较多，关系较为复杂，如果不边写边进行测试，后期可能难以发现错误，又比如整个项目要如何进行管理，如何运用一些设计模式和抽象手段提高代码的简洁度和可读性等等，为了在这些方面做出一些改进，我主要采用了以下手段：

1. 项目采用 Maven 进行依赖管理和自动构建，在项目构建，依赖管理，测试等方面都取得了很多的便捷之处，使得整个项目更加符合工程上的规范。

2. 使用 JUnit 编写单元测试，后期采取了一种测试驱动的开发模式，在编写类型之前，先编写好接口以及其测试，再完成其实现，，使得编写方法的思路更加清晰，一边进行任务代码的编写，同时进行测试，也有利于及早发现一些 bug，避免在后期的代码编写以及测试中由于前面的问题而出现一些难以定位的 bug。然而由于编写测试的经验不足，在使用单元测试这方面也还存在许多问题。

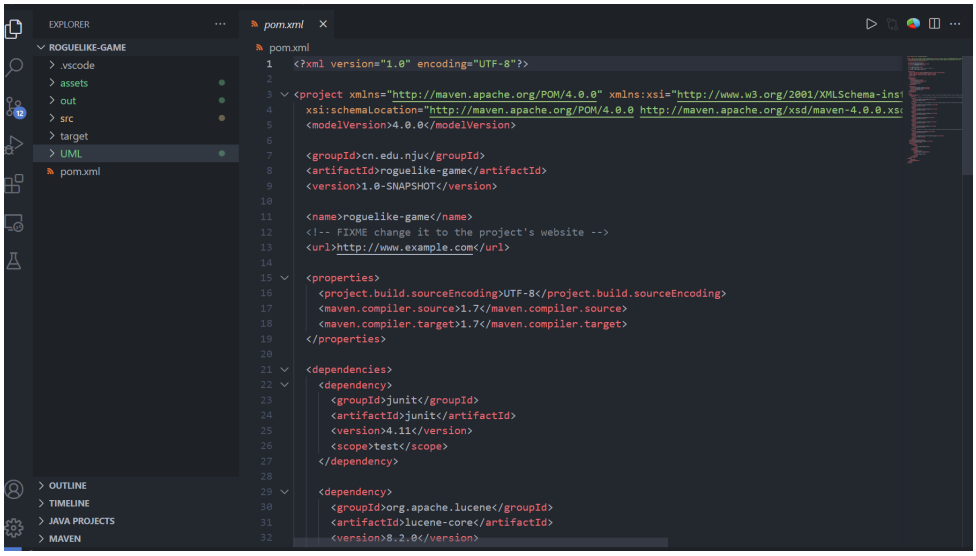

图 3　使用 Maven 进行依赖管理和自动构建

Figure 3　Caption

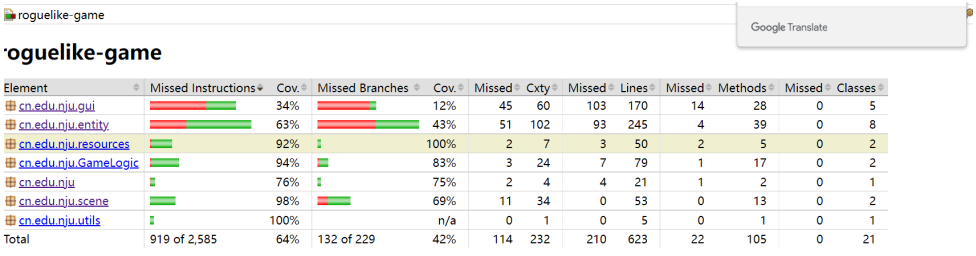

图 4　使用单元测试

Figure 4　Caption

3. 在代码编写过程中也尝试使用了一些设计模式，比如使用工厂模式创建 Message 类型的对象，通过统一的接口，用服务器传过来的字节流消息，创建不同类型的 Message 对象，使得代码编写的逻辑变得更加简单清晰。

4. 在类型设计时，使用 PlantUML 来简单绘制类图，在编写代码之前提前对类型设计有一些认识，也让类型关系变得更加清楚，在编写代码的时候能够对类之间的关系有一个基本的把握。

5. 使用 git 来进行版本控制管理，及时保存进度，在出问题时也方便退回。

6. 为游戏编写了一个简单的文档，提前对要实现什么功能有一个清楚的认识，也方便定位自己目前的进度，使项目开发更加顺畅。

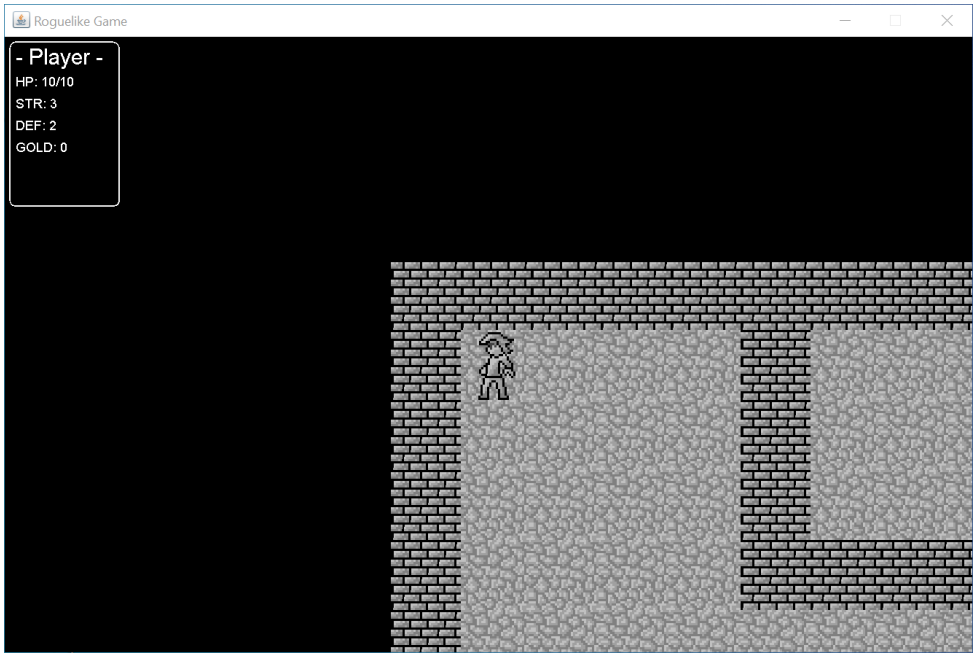

图 5　游戏效果图

Figure 5　Caption

6　课程感言

对于我来说，Java 高级程序设计这门课是一门内容十分丰富，形式也很多样的课程。老曹上课很认真，很有激情，课程内容也很有意思，通过应用学习的 Java 技术，可以实现很多想不到的事情，比如排序可视化，学习类的加载那次的一个隐写术的作业，以及最后的这个小游戏，虽然做的也并不是很大的东西，但是真切地让我感受到 Java 在实际生活中的应用，而不仅仅是对着控制台做输入输出，让人很有成就感。

在这门课上除了学习到很多之前没见过的 Java 特性之外，另外一个最大的收获是养成了看书、看文档和博客学习的习惯，通过课程的学习，我认识到这种学习的方法是很重要的，而且这也是我在之前的 Java 学习中欠缺的。

虽然这学期由于某些时间管理上的失误，课程的内容可能掌握的并不算好，最后的作业完成的也有点吃力，但是通过这门课的学习，我深刻感受到了 Java 技术的广泛，希望日后能在 Java 的学习上更进一步。

致谢　致谢.

最后，感谢曹老师这门 Java 高程带我走上了 Java 学习的道路。

参考文献

1　Chun Cao. Java Advanced Programming. Slides, 2021: all of pages

Developing a Java Game from Scratch

Lifan Sun[1]

1. *Nanjing University Software Institute, Nanjing* 210023*, China*
E-mail: 201250181@smail.nju.edu.cn

Abstract This article mainly reviews the developing goals, design ideas, technical problems, engineering problems and reflections on this course, Advanced Java Programming, 2021 Fall, the Department of Computer Science and Technology, Nanjing University. The final programming assignment is to develop a Roguelike game that support progress saving and network communicating. This game's graphical user interface is mainly developed using Java swing. Progress saving is implemented using Java Serialization, and network communicating is implemented using Java nio selector. Finally, Maven is used to build the project and manage the dependency.

Keywords Java, swing, nio, socket, maven, serialization, object-oriented