中国科学 : 信息科学

SCIENTIA SINICA Informationis

论文

# Developing a Java Game from Scratch

陈杨[1, *]

1. 南京大学, 南京 210033

E-mail: 2471933872@qq.com

**摘要** ：如今快节奏与高强度的生活、工作环境使得游戏成为人们放松身心的不错选择，2D 像素风 RPG 游戏在二三十年前曾经红极一时，凭借自身的特点，在今天也得到了很多人的喜爱。文中设计开发了一款基于 Java 的 Roguelike 游戏, 该游戏以 2D 为视角, 采用 JFrame 和 AsciiPanel 进行显示，用了多线程实行对象控制，用 nio 实现了网络通信，实现了多人游戏的模式选择、人物怪物交互、获得宝物等多项功能。该游戏的设计作为高级 JAVA 程序设计的大作业。

**关键词** JAVA, 游戏, Roguelike, 多线程, 网络通信

## 1 引言

Java 是当前最流行的、应用最广泛的编程语言之一，因其具有良好的跨平台性和可移植性，可以适用于大多数环境，在游戏开发中具有非常重要的地位。随着现代社会生活节奏不断加快、生活压力越来越大游戏作为大众生活必不可少的一项娱乐方式，近年来发展势头良好。本文着眼于 JAVA 的应用，开发一款 Roguelike 多方网络游戏。Roguelike 游戏是一款策略类的 RPG 游戏，具有很高的游戏性、趣味性，并拥有一定的随机性，给人带来持续不断的新颖的体验。本文基于 JAVA 平台对 Roguelike 游戏进行设计和开发。整个游戏在 Microsoft Windows 10 64 位操作系统下，利用 Visual Studio Code 开发工具进行开发。葫芦娃的移动使用方向键上、下、左、右以及 W、A、S、D，攻击使用 1 以及 J，技能使用 2 以及 K。

## 2 开发目标

### 2.1 游戏类型

**定义1** (Roguelike) Roguelike 是欧美国家对一类游戏的统称，是角色扮演游戏（RPG）的一个子类（Roguelike-RPG）。指二十世纪八十年代初，由 Michael Toy 和 Glenn Wichman 两位软件工程师共同在 UNIX 系统上开发，在大型机上运行的游戏。其类型始祖游戏 Rogue 在 2009 年被游戏权威杂志“PC WORLD”评为“史上最伟大的十个游戏之一”。

我希望开发出一款能够实现网络多人联机的 Roguelike 游戏。它应该具有以下特点：


**图 1　Roguelike**
**Figure 1**　Roguelike

1、生成随机性。每一次新开局游戏都会随机生成游戏场景，敌人，宝物等不同事物。因此而玩家的每一次冒险历程也自然是是独一无二，不可复制的。

2、进程单向性。当你在玩一款 Roguelike 游戏时，不应该设置存档点，保持 Roguelike 游戏特有的“一命通关”的特性，这种存档机制确保玩家无法利用”S/L 大法” 来降低游戏难度。

3、不可挽回性。在大多数 Roguelike 游戏中，玩家作出的决定应是不可挽回的，因此玩家应该慎重考略每一次决策，这是游戏中策略性的所在之处。同时，玩家控制的角色只应该有一次生命，不应该存在复活机制。

4、画面朴素性。Roguelike 类游戏会直接使用 ASCII 字符来表示游戏画面，这简化了美术设计而让游戏更专注于玩法和游戏性。

5、联机无差性。要求通信过程全局状态一致，所有玩家看到的游戏过程完全一样。

## 2.2　设计灵感

**定义2** (魔塔)　魔塔游戏是一种解谜类电子角色扮演游戏的通称，起源于日本的一款 PC-9801 的 MS-DOS 游戏《魔塔》。目前演化出各种不同的版本。大多数魔塔游戏以一位勇士解救被困在魔塔中的公主作为背景，玩家需要操控勇士在魔塔内行走，打败怪物、积累宝物，最终救出公主。

灵感来源于一款 00 年代的 RPG ——《魔塔》，我在少年时期对魔塔了长时间的游玩，因此当我拿到需求时，我第一个想到了这个游戏。同时，魔塔的美术要求比较低，仅通过简单的贴图就完成美术方面的需求。但《魔塔》是一款数值固定的 RPG，因此为了满足本次作业的需求，我融入了 Roguelike 元素，整个游戏流程类似于逃出地牢，葫芦娃被困在怪物巢穴中，需要得到钥匙，逃出巢穴，在这个过程中，葫芦娃每次逃亡都会遇到随机的地形、随机的敌人、随机的宝物，葫芦娃可以通过积累数值的方式打败敌人。这样的设计更增添了游戏性。


**图 2　Roguelike**

**Figure 2**　Roguelike

## 3　设计理念

Roguelike 需要实现以下功能:

(1) 游戏模式选择：当玩家运行.class 文件与服务器进行连接后，可以选择模式（现在只完成了 New Game 模式）

(2) 随机化生成：玩家进入游戏后, 地形、怪物、宝物应随机化初始化。

(3) 怪物逻辑：怪物应实现 Runable 接口，每个怪物占据一个线程运行

(4) 玩家操作：玩家通过按键对葫芦娃进行一系列控制，包括行走、普通攻击、获取宝物、释放火球

(5) 服务器与客户端的网络通信：客户端应发送按键信息给服务器，服务器应对客户端信息作出反应，发送计算后的信息给客户端

(6) 游戏结束: 当葫芦娃生命清零后，游戏失败。当葫芦娃逃出巢穴后，游戏胜利。

### 3.1　界面设计

使用 Java 的 swing 来建立 GUI 界面，创建了 JFrame 类型，通过 JFrame 可以利用 Java 代码实现窗体功能。本游戏的界面简洁明了，板式布局简单清晰。通过 AsciiPanel 类型完成 Ascii 字符显示图像的功能

### 3.2　葫芦娃设计

葫芦娃有四种行为，walk，get，attack，bomb，对每种行为进行设计　　walk 的核心时 tryMove，对其进行 synchronized 关键词修饰，保证资源不会同时占用。

```
if (conditions)
    world.tryMove(this, x, y, targetx, targety);
```

tryattack 是 attack 的核心，也用 synchronized 关键词修饰，保证攻击次序按顺序发生

```
if (conditions)
    Thing th = world.get(targetx, targety); * 对眼前的生物体伤害 *
    th.decreaseHealth(damage);
```

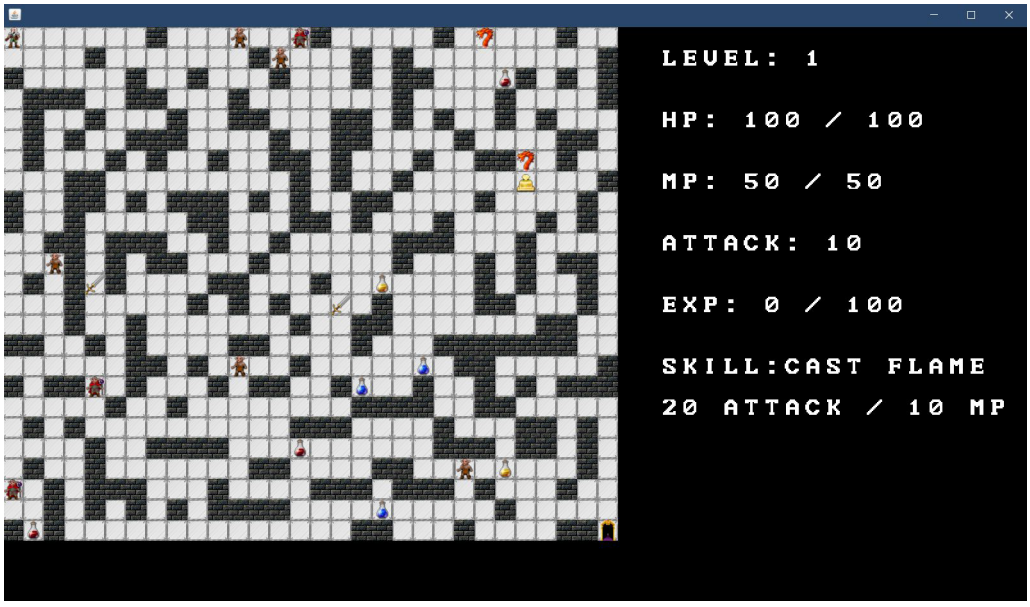


**图 3　游戏界面**

**Figure 3**　WolrdScreen

bomb 产生一个火球，需要拉起新的线程，Flame 可以朝一个方向冲击，造成伤害或者击毁墙壁

if (conditions)

Flame flame = new Flame(world, this, moveX, moveY);

Thread thread = new Thread(flame);

thread.start();

### 3.3　怪物设计

每个怪物被一个线程控制，有 CommonMonster、MagicMonster、EliteMonster 三类，其中 MagicMonster 可以释放火球

怪物应在出生地一定范围内移动, 会随机选一个方向直到不能 move

while (tryWalk(choose))

如果有敌人出现在视线内

protected int inSight(int d)

会尝试追赶，如果相邻，会尝试攻击

### 3.4　客户端与服务器

客户端应发送按键信息给服务器，服务器应对客户端信息作出反应，发送计算后的信息给客户端

在 client 监听键盘信息时应将按键信息传给服务器

public void keyPressed(KeyEvent e)

client.write(sBuffer);

服务器对信息作出反应

app.response(input, client);

然后将信息给客户端

```
sBuffer.put(gson.toJson((WorldScreen) app.getScreen(temp)).getBytes());
```

## 4 技术问题

本次游戏开发我使用了许多相关技术解决相关问题，比如用 JFrame 类以及 AsciiPanel 完成 gui 的设计；用 Runnable 接口进行多线程的开发；使用 ObjectInputStream 和 ObjectOutputStream 完成输入输出保证持久化；使用 nio 和 setector 完成客户端和服务器的交互。

### 4.1 多线程并发控制

Java 给多线程编程提供了内置的支持。一条线程指的是进程中一个单一顺序的控制流，一个进程中可以并发多个线程，每条线程并行执行不同的任务。多线程是多任务的一种特别的形式，但多线程使用了更小的资源开销。多线程能满足程序员编写高效率的程序来达到充分利用 CPU 的目的。

在面向对象编程中，创建和销毁对象是很费时间的，因为创建一个对象要获取内存资源或者其它更多资源。需要尽可能减少创建和销毁对象的次数来提高服务程序效率，特别是一些很耗资源的对象创建和销毁。

我采用了线程池的设计，创建一个固定线程数的线程池，在任何时候最多只有 n 个线程被创建。如果在所有线程都处于活动状态时，有其他任务提交，他们将等待队列中直到线程可用。如果任何线程由于执行过程中的故障而终止，将会有一个新线程取代这个线程执行后续任务。

一般情况下，只要涉及到多线程编程，程序的复杂性就会显著上升，性能显著下降，多线程编程本意是将一段程序并行运行，提升数据处理能力，但是由于大部分情况下都涉及到共有资源的竞争，所以修改资源对象时必须加以控制。资源控制有很多种方法，比如互斥量 mutex 和 lock 加锁的方法，但考虑到 lock 的资源消耗比较大，我还是选择了关键字 synchronized：java 为放置资源冲突提供了内置的支持关键字 synchronized，当代码执行到被 synchronized 保护的代码片段的时候，它会检查锁是否可用，然后获取锁，释放锁。当然这也需要比较多的资源，因此我选择了非常小的封锁粒度。

### 4.2 输入输出

对象输入/输出流是用来存储/恢复之前序列化存储的对象，我为了将 world 进行存储完成保存、加载的功能，为 world 和其中的成员对象实现了 Serializable 接口，对象输入/输出流让从流中读取的对象匹配 Java 虚拟机中已经存在的类，根据需求使用标准机制加载类。

### 4.3 NIO

IO 的各种流是阻塞的。当一个线程调用 read() 或 write() 时，该线程被阻塞，直到有一些数据被读取，或数据完全写入。NIO 的非阻塞模式，使一个线程从某通道发送请求读取数据，但是它仅能得到目前可用的数据，如果目前没有数据可用时，就什么都不会获取。而不是保持线程阻塞，所以直至数据变得可以读取之前，该线程可以继续做其他的事情。

NIO 主要有三大核心部分：Channel(通道)，Buffer(缓冲区), Selector。传统 IO 基于字节流和字符流进行操作，而 NIO 基于 Channel 和 Buffer(缓冲区) 进行操作，数据总是从通道读取到缓冲区中，或者从缓冲区写入到通道中。Selector(选择区) 用于监听多个通道的事件（比如：连接打开，数据到达）。因此，单个线程可以监听多个数据通道。本次游戏的多人网络通信就是使用 nio 进行的服务器与客户端的交互, 使得服务器可以同时监听多个客户端的信息并作出反应。

Selector 运行单线程处理多个 Channel，如果你的应用打开了多个通道，调用它的 select() 方法。这个方法会一直阻塞到某个注册的通道有事件就绪。一旦这个方法返回，线程就可以处理这些事件，事件的例子有如新的连接进来、数据接收等。

### 4.4 json

在 nio 中由于写入的是字节，我选择了将状态转换成 json 的方式，这里我用了外部的依赖——Gson 来解决类的转化问题，同时要注意的是这并不是直接的转化，因为 screen 的成员存在环形引用，于是我又用了 Expose 注解和 GsonBuilder 来解决。

### 4.5 面向对象设计方法

面对对象程序设计尽可能模拟人类的思维，将客观世界中的抽象问题转化为具体的问题对象。面向对象设计方法首先要分析问题所涉及的基本对象和他们间的相互关系，然后将这些现实对象映射到计算机中，实现计算机对现实问题的模拟，将“问题空间”直接映射到“解空间”。能够提高重用性，先开发一个比较小、比较简单的来，作为开发比较大、比较复杂的类的基础。能够提高可扩充性，即在原有基础上有更多的创新，开发新的功能模块。

## 5 工程问题

### 5.1 工程构建

**定义3** (maven) Maven 项目对象模型 (POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的项目管理工具软件。Maven 除了以程序构建能力为特色之外，还提供高级项目管理工具。由于 Maven 的缺省构建规则有较高的可重用性，所以常常用两三行 Maven 构建脚本就可以构建简单的项目。

Maven 能够很方便的帮你管理项目报告，生成站点，管理 JAR 文件，本游戏就使用了 Maven 进行项目管理。

### 5.2 设计模式

本游戏未能有设计模式的应用，因此存在一种设计模式上的优化，可以向 Reactor 模式靠拢。对于高并发系统，用其代替常用的多线程处理方式，可以节省系统的资源，提高系统的吞吐量。

Reactor pattern 是一种为处理并发服务请求，并将请求提交到一个或者多个服务处理程序的事件设计模式。当客户端请求抵达后，服务处理程序使用多路分配策略，由一个非阻塞的线程来接收所有的请求，然后派发这些请求至相关的工作线程进行处理。有一个或多个并发输入源，有一个 Service Handler，有多个 Request Handlers；这个 Service Handler 会同步的将输入的请求（Event）多路复用的分发给相应的 Request Handler。每当一个 Event 输入到 Service Handler 之后，该 Service Handler 会主动的根据不同的 Event 类型将其分发给对应的 Request Handler 来处理。

## 6 课程感言

选课之前也询问了多方意见，学长都说这是一门能学到东西的课程，对曹春老师也是“不识庐山真面目”，还是希望能够学点东西，毅然选了老师的两门课程。不知不觉到了学期末，经过一个学期学

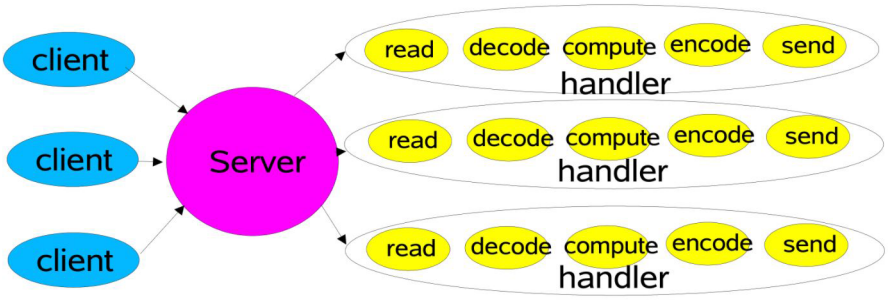


**图 4 Reactor Pattern**

**Figure 4** Reactor Pattern

习，我觉得受益匪浅。这门课程更偏重于实践，犹记当是一节课不到就讲完了 JAVA 的语法。课程很好的锻炼了我们每一个人的程序能力，这门课不仅仅是从上课中学到的知识，更多的是从网络资料、从 JAVA 的文档进行课程学习，充分要求我们的自学能力。

这门课不仅仅讲 JAVA，在作业的要求中，也涉及许许多多的工具使用 UML、wsl、asciinema、隐写术、maven、latex 等等。培养我们综合运用所学知识，发现、提出、分析和解决实际问题，锻炼实践能力的重要环节。整个学习过程是痛苦的，虽然我现在也忘了前面作业用到的工具，但是自我学习的过程就是对自己的一次提升，走出舒适区，突破了自己。当然，和同学相比，我有着许许多多的不足，每次在群里瞻仰同学的作业都让我深刻地认识到与同学们之间能力的差距，包括现在我还没完善大作业加载游戏的过程，以后有时间再完善吧。

学习 JAVA，应该尽量多动手，很多时候，听讲、在脑子中思考并不能认识到问题，就比如网络通信，样例十分简单，事实上实际应用有太多的问题要考虑。这一点是我学习一学期感触最深的一点。在 JAVA 的学习中，我认识到自身的一个问题，我经常面向 Google 面向百度编程，虽然查文档是好事，但是这也是 JAVA 用的不熟练的问题，还是需要多写代码。

最后对课程有一点拙见。大作业中比较困难的部分讲的比较迟，比如网络通信，我觉得应该提前一点讲，我认为多线程和网络通信是本次课程中最困难的部分，应该早做安排。

**致谢** 曹春

# Developing a Java Game from Scratch

Yang Chen[1, *]

1. *Nanjing University, Nanjing* 210033, *China*
E-mail: 2471933872@qq.com

**Abstract** Today’ s fast-paced and high-intensity life and work environment make games a good choice for people to relax. A Java-based Roguelike game was designed and developed in this article. The game uses JFrame and AsciiPanel as the perspective to display, uses multi-threading to implement object control, uses nio to achieve network communication, and realizes the mode selection of multiplayer games. Multiple functions such as interaction between characters and monsters, obtaining treasures, etc. The design of the game is a major task of high-level JAVA programming.

**Keywords** JAVA, Game, Roguelike, Multithreading, Telecommunication