中国科学 : 信息科学

SCIENTIA SINICA Informationis

论文

Developing a Java Game from Scratch

沈童菲[1]

1. 南京大学计算机科学与技术系, 江苏南京210023

E-mail: 191220091.smail.nju.edu.cn

摘要 本文介绍了我在高级Java程序设计这门课程中使用Java语言完成一个小游戏开发的情况，包括开发目标中的游戏介绍、设计想法，类的设计理念，还有在开发过程中遇到的技术问题、工程问题以及解决问题使用到的工具和方法.

关键词 高级Java程序设计, 开发目标, 设计理念, 技术问题, 工程问题

1 开发目标

1.1 游戏简介

简单小游戏，分为玩家、怪物两方，玩家通过躲避怪物攻击，收集爱心通关晋级，可以通过放置墙壁阻挡并杀死怪物，收集其他物品帮助增强玩家发现爱心的能力。

1.2 具体玩法介绍

玩家通过键盘的上下左右键在屏幕上移动，通过WSAD四个键在自己的上下前后放置墙壁；玩家每轮的目标是活着收集齐屏幕上的所有爱心，游戏等级越高需要收集的爱心数量越多。

怪物每隔一段时间随机出现，每轮游戏同时出现在屏幕上的怪物数量有上限，游戏等级越高怪物数量越多，生命力和攻击能力越强。对于怪物攻击，玩家可以逃跑躲避或者放置墙壁阻挡。

除了爱心，玩家还可以吃掉其他物品增加能力，比如：药品，玩家吃掉后可以恢复一定的HP值；放大镜，玩家吃掉后一定时间内可视范围增加一倍。

1.3 玩法设计过程

我没有游戏设计的经验，在我的概念里，一个简单小游戏要有比较高的可玩性的重点是玩家赢或输的条件达到一个动态平衡，即在游戏进行的过程中，玩家基本时刻都有赢和输的可能性而且两种可能性相当，这样不至于游戏太难或者太简单导致没什么意思。

设计游戏玩法时，我先设定了玩家游戏胜利和失败的标志：收集齐所有爱心即为通关，尚未集齐时因遭到怪物攻击导致HP值降为0就失败。怪物的设定就是瞄准玩家，向玩家的方向行进，到达玩家旁边时对玩家发起攻击，直到“打死”玩家。玩家有一定的可视范围，只有当爱心在这个范围里玩家才能看见，所以玩家需要通过四处走动寻找并收集爱心。

后来由于玩家面对怪物只能逃跑没有意思，就增加了玩家可以通过向自己四周设置墙壁阻挡怪物的功能，怪物撞到墙壁会“死亡”，同时墙壁消失。于是怪物的设计变成源源不断地产生，每隔一定时间就会产生一个新的怪物试图追踪攻击玩家，但无止境地产生怪物会使游戏难度过大，于是限制屏幕中的怪物数量。将游戏改为晋级式，每获得一次胜利就进入到下一个等级，同时难度增加，即增加要收集的爱心数量，屏幕中怪物数量的上限也增加，怪物的种类和攻击能力也增加，撞到墙壁可能不死亡，玩家如果撞到自己放置的墙壁也会减少一定的HP值。添加其他可收集的物品，药品和放大镜。

2 设计理念

2.1 类的设计

2.1.1 world

package world下的类介绍如下：

World：总体游戏世界，包含所有的Creature（玩家、怪物和物品等）以及它们所处位置Tile；通过二维Tile数组将界面划分为一格一格的可供Creature站立的位置，一个Creature的List储存所有的生物；World操控着Creature的存在与否。

WorldBuilder：可以按要求生成一个World，包含初始的Tile等。

Tile：枚举类，主要分为Floor和Wall，存在于World中供Creature占位，Creature被设定为只能站在Floor上。

Creature：生物的总类型，通过CreatureType和CreatureAI区分其中具体的类型，可以在World中移动，做出攻击/被攻击等行为。

CreatureType：枚举类型，将Creature具体分为PLAYER，MONSTER，FUNGUS，MEDICINE，AMPLIFIER五类。

CreatureAI：控制不同种类的Creature做出不同行为，具体有五个子类PLAYERAI，MONSTERAI，FUNGUSAI，M

CreatureFactory：根据要求创造不同类型的Creature并添加进相应的world.

类图如1所示.

2.1.2 screen

package screen下的类介绍如下：

Screen：接口类，包含显示屏幕的displayOutput和对键盘按键事件作出反应的respondToUserInput两个抽象方法。

PlayScreen：实现了Screen接口，是游戏进行中的界面，包含游戏的world和实时显示的游戏等级、进度等，每个玩家和怪物都是其中开启的一个线程。

RestartScreen：实现了Screen接口，是游戏间隔的界面，按下回车键可以开启一个PlayScreen，分为StartScreen, WinScreen和LoseScreen.

StartScreen：游戏最开始的界面，显示信息通知用户游戏将开始，检查是否存在上一轮被保存的游戏，询问用户是否要继续上一次的游戏。

WinScreen：游戏胜利的界面，显示相应提示信息，按下回车键进入下一等级的PlayScreen.

LoseScreen：游戏失败的界面，显示相应提示信息，按下回车键重新开启刚刚失败等级的PlayScreen.

类图如2所示.

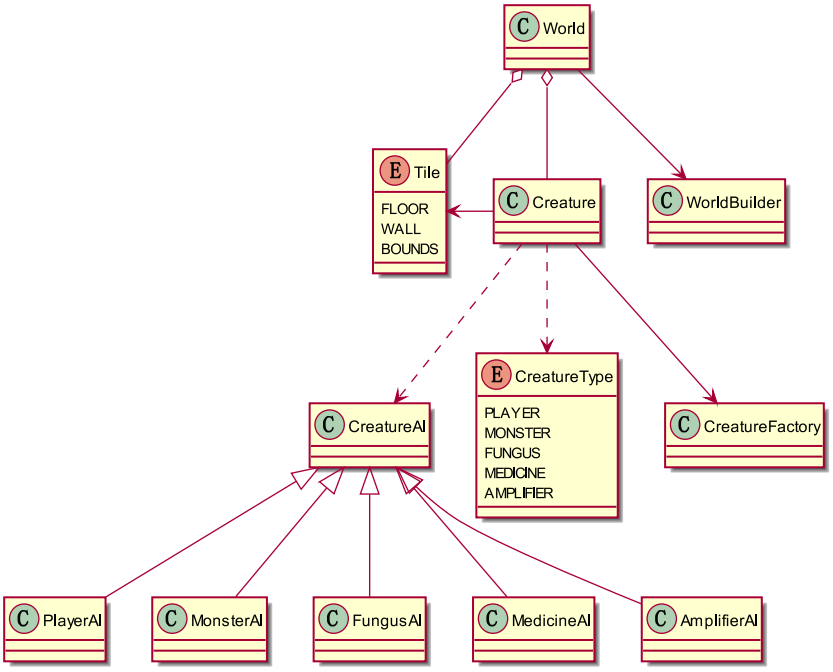

图 1　world下类的UML图

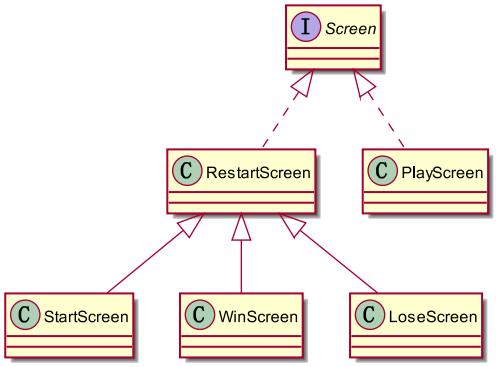

图 2　screen下类的UML图

2.1.3　ApplicationMain

ApplicationMain，SelectorServer，Client：JFrame的子类，实现KeyListener接口，有main函数，是程序的入口，包含Screen和要显示的AsciiPanel，监听键盘事件并交由Screen做出响应。

2.2 设计想法

面向对象的类的设计通过模拟游戏中的每个事物完成设计，通过对整个游戏功能的划分，将其分成一个一个有不同功能的类，彼此之间通过发送接收消息来进行信息传递。所有类的划分设计很符合现实世界的具体概念，以便设计者更好理解代码，故设计者在编码时只是更需要注意行为、状态与概念间关系的合理性，可以适当忽略非对象的复杂过程。

以对象为设计中心，迫使设计者在关注程序所需实现功能的同时不至于忘记通过抽象去塑造概念，以便用对象表达之。由于抽象获得的对象有助于隐藏复杂度，这在一定程度上简化了通过对象表达和理解软件虚拟世界的难度。也由于对象的存在，使得设计更加的生动和具有更强的自我解释能力。

3 技术问题

3.1 并发控制

首先遇到的问题是并发控制，关于怪物和玩家在同一个world中移动，为确保它们不会踏上同一个格子，需要对Creature的move方法上锁；为保证怪物只能攻击玩家而不能攻击其他生物需要对怪物的行动路线进行控制，每次有可走的路才向玩家方向迈进一步，还要能实时检查要攻击的玩家当前位置；玩家防止墙壁也只能在没有生物的地方放，放之前也需要自动检查；还有为保证屏幕中的怪物数量上限，每次要新发送一个怪物时需要检查当前world中存在的怪物数量，如果在PlayScreen中检查world中的Creature，会产生ConcurrentModificationException，于是给world中遍历Creature的方法上锁，PlayScreen只能通过world浏览Creature并且不会改变Creature，消除了两个线程可能同时对Creature进行修改的问题。

3.2 通信

多人游戏中，服务端通过NIO Selector管理客户端，在进行所有客户端同步时，一个客户端接收键盘消息准备移动玩家时，将玩家移动的消息发给服务器，服务器再把该消息发给所有客户端，接收到服务器消息的客户端做出相应玩家的移动，包括原来发送消息给服务器的客户端，以此实现同步。在实现服务器接收消息并向所有客户端转发时，由于selector每次在某个通道就绪时会自动将其放入已选择键集中，故必须在处理完通道时手动移除，如果一接收到消息，通道尚未处理完就立即重新向所有就绪通道发送消息就会产生iterator 在遍历时被修改的异常，于是需要服务器现储存该消息，下一次从头遍历时再向所有可写入的通道写入该消息。

3.3 输入输出

输入输出主要的问题是将序列化的对象保存至文件，再从文件中读出该对象并反序列化；还有服务器用Selector和客户端通过Channel通信需要用到的Buffer等，Java中各种输入输出流弄得我眼花缭乱，通过各种网上搜索学习才分清各种流的用法。

在实现保存游戏进度的功能时，我用了很久才完成对象的序列化，试图序列化整个world时其中的每个Creature也有成员变量是其所在的world，还有一些动态的变量需要在程序运行时才能确定，这些问题导致我序列化和反序列化一直不能成功。后来使用Junit 单元测试对最简单的world对象进行序列化和反序列化的尝试，在不断调试中找到了问题所在并成功解决。

4 工程问题

用Java进行面向对象的设计开发，强调以对象为中心，因而具备更强的封装能力。更强的封装能力除了意味着更具信息隐藏能力外，还使得封装的边界既明显又更不易被突破，这有助于在游戏的维护过程中维持“形”，而且面向对象设计中的继承和多态技术除了进一步提高通过软件模拟现实世界的能力外，还能让设计更灵活、易变更和方便复用。

根据要求采用maven管理项目的开发，结合VScode，在单元测试、生成JAR文件的功能上更加方便；使用VScode中的插件TabNine 进行代码补全，TabNine 通过智能联想自动补全正在编写的代码字段，使得写代码时对于以前输入过的长字段不用重新打一遍或者复制粘贴，减少了重复劳动，敲代码的效率变得更高。

5 课程感言

本学期我学习了高级Java程序设计课程，学习使用Java语言进行面向对象的程序设计，通过听上课讲和完成各种有意思的作业，我对Java有了初步的理解和掌握，除了像以前学习过的语言一样通过编程实现简单的算法，还学习了Java中的对象序列化、网络通信等技术，对面相对象编程的高内聚、低耦合有了更深刻的认识，同时我也感到目前只学习了Java的一点皮毛，学习更多只是为了完成作业，没有真正深入理解这门语言，也并没有用它完成过大型的工作，希望以后可以继续关于Java和面向对象的学习和探索。

在磕磕绊绊中完成了这一学期的课程，我却并没有实现很“高级”的Java程序设计，这当然都是我的问题，和老师没有关系。曹春老师是我这学期遇到最好的老师，将整个学期的课程安排得井井有条，讲课十分清晰，自己对Java的知识烂熟于心的情况下还能清楚明白地讲给同学们听，用各种尽量简洁的例子使我们理解其中的原理，在作业难度递增有挑战性的同时给出的框架也很便于理解，不会让人望而却步，是少见的科研能力和教学能力都很强的老师。

对于课程的建议就是希望老师可以多录几节Java的课程传到B站供同学们学习，特别是关于网络通信的章节。