中国科学：信息科学

SCIENTIA SINICA Informationis

论文

Developing a Java Game from Scratch

唐家昕[1]

1. 181860086

E-mail: 181860086@smail.nju.edu.cn

摘要 本学期的 Java 课程的大作业内容为完成一个大型的多线程网络对战游戏。游戏需要使用图形界面进行绘制，使用多线程对游戏中的人物进行控制，使用键盘进行实时交互，通过序列化技术支持存档及读档，同时需要使用 Socket 编程与非阻塞 IO 技术实现多人联机对战。本文主要包含本次 Java 大作业的开发目标、设计理念、技术问题、工程问题等内容。

关键词 Java, 游戏, 面向对象程序设计，Socket 编程

1 开发目标

在设想多种思路后最终决定，本次作业的开发目标是实现一个吃豆子小游戏。这一游戏的灵感来源为曾经常玩的、兼具惊险与趣味性的吃豆子游戏。在这一游戏中，每个玩家使用键盘操作各自的小人在棋盘中进行移动，在遇到豆子时可以将其吃掉。玩家在吃豆子时需要小心会伤人的怪物，如果碰到怪物则会损失体力值。当体力值降为 0 时，玩家死亡。根据难度由低到高，怪物们会由随机在棋盘中移动到向玩家追来。当棋盘中存在多个玩家时，怪物会优先追踪距离最近的玩家。棋盘中有两种药剂：生命药剂，吃掉这瓶药剂可以恢复 1 点生命值；无敌药剂，吃掉这瓶药剂可以进入无敌状态 10 秒。在无敌状态下玩家碰到怪物不会损失体力值。当棋盘中的豆子全部被玩家吃掉时，玩家获胜；当所有玩家均死亡时，玩家失败。

游戏支持存档和读档功能，在游戏中按下 S 键可以将当前的游戏状态存入文件，在初始界面按下 C 键可以从文件中读取之前存储的游戏信息并恢复该游戏状态。

游戏支持通过网络进行连接及对战功能。游戏分为客户端和服务器两部分，客户端会通过网络向服务器请求当前的游戏场景，服务器会管理当前的场景并将其反馈给客户端。对于每一个玩家，都会建立一个客户端，有一块屏幕显示当前的游戏场景，玩家可以在该屏幕上进行操作。所有玩家看到的游戏场景是完全相同的。

2 设计理念

本次项目的代码主要分为几个模块：asciiPanel，calabashbros，mazeGenerator，screen，server 和 client。其中，asciiPanel 模块主要负责在屏幕上显示图片和字符，calabashbros 模块为游戏中的各

种主要元素，mazeGenerator 模块用于产生游戏地图，screen 模块用于管理当前的游戏场景，server 和 client 分别为服务器和客户端。

2.1 server 和 client

首先，一位玩家会拥有一个 client 类的实例。后者会拥有一块 Jframe 屏幕，用于显示当前的游戏界面。client 通过 SocketChannel 与 server 进行数据交互。具体的，client 会监听键盘事件，当键盘被按下时会将这一消息发送至 server；client 会定期向 server 发送刷新屏幕请求。服务器则只有一个 server 类实例，后者会拥有一个 screen 对象，可以读取并修改当前的游戏场景。当 server 收到 client 发送来的消息时，会解析这一消息，并调用 screen 的方法，在地图上进行相应的操作。当它收到刷新屏幕请求时，会读取当前的游戏界面，转换成字符串发送给 client，后者会解析字符串，调用 asciiPanel 中的方法将这一界面显示在自己的屏幕上。在 screen 对象被构造时，会调用 mazeGenerator 模块中的方法来创建一张包含各种元素的地图数组，接着会根据这一数组创建 calabashbros 模块中的各种元素，从而生成地图。

server 和 client 之间交互的示意图如图1所示.

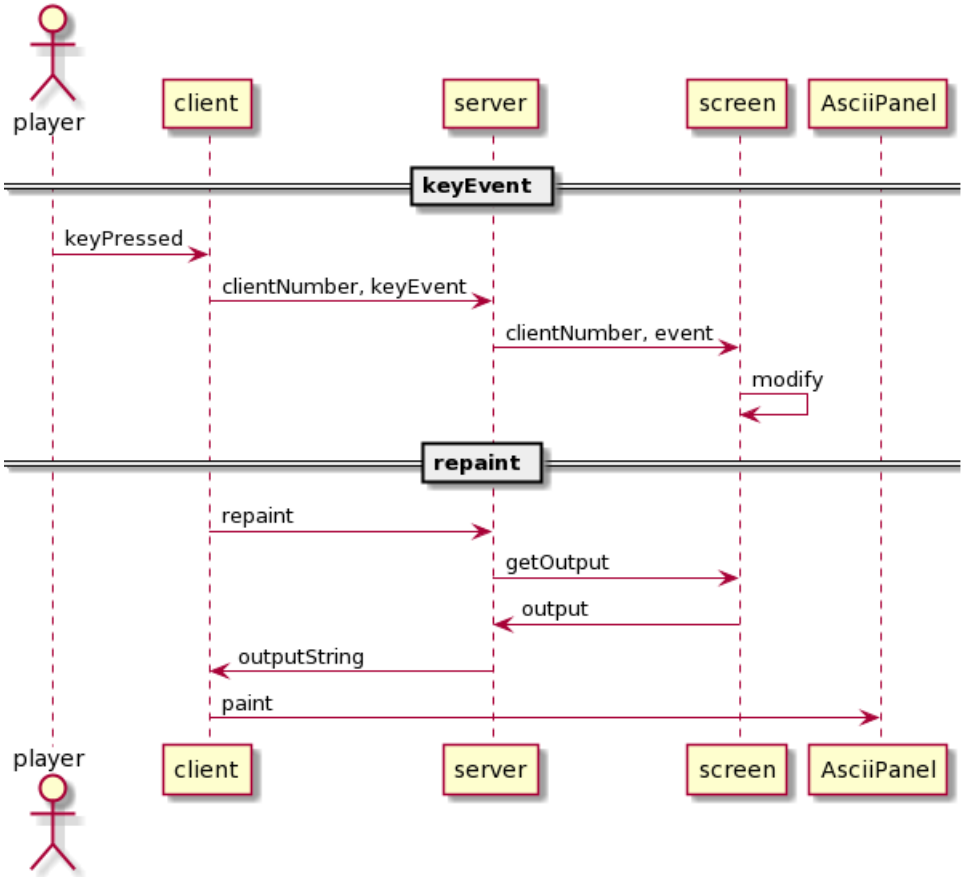

图 1　server 和 client 之间的交互

Figure 1　Interaction between server and client

2.2 calabashbros

在 calabashbros 模块中有一系列界面中的元素，包括玩家、怪物、砖块等。Thing 是所有元素的基类，可以被放在地图上；Moveable 是可移动的元素的基类，可以进行移动，可以被停止；Creature 是具有生命的元素的基类，可以被杀死。每一个 Moveable 类都实现了 Runnable 接口，从而可以被 Thread 装载，作为线程来运行。游戏界面中的每个元素都是 Thing，墙壁和药水是普通的 Thing，玩家射出的箭是 Moveable 的，玩家和怪物都是 Creature。

游戏界面中的各种元素类之间的关系示意图如图2所示。

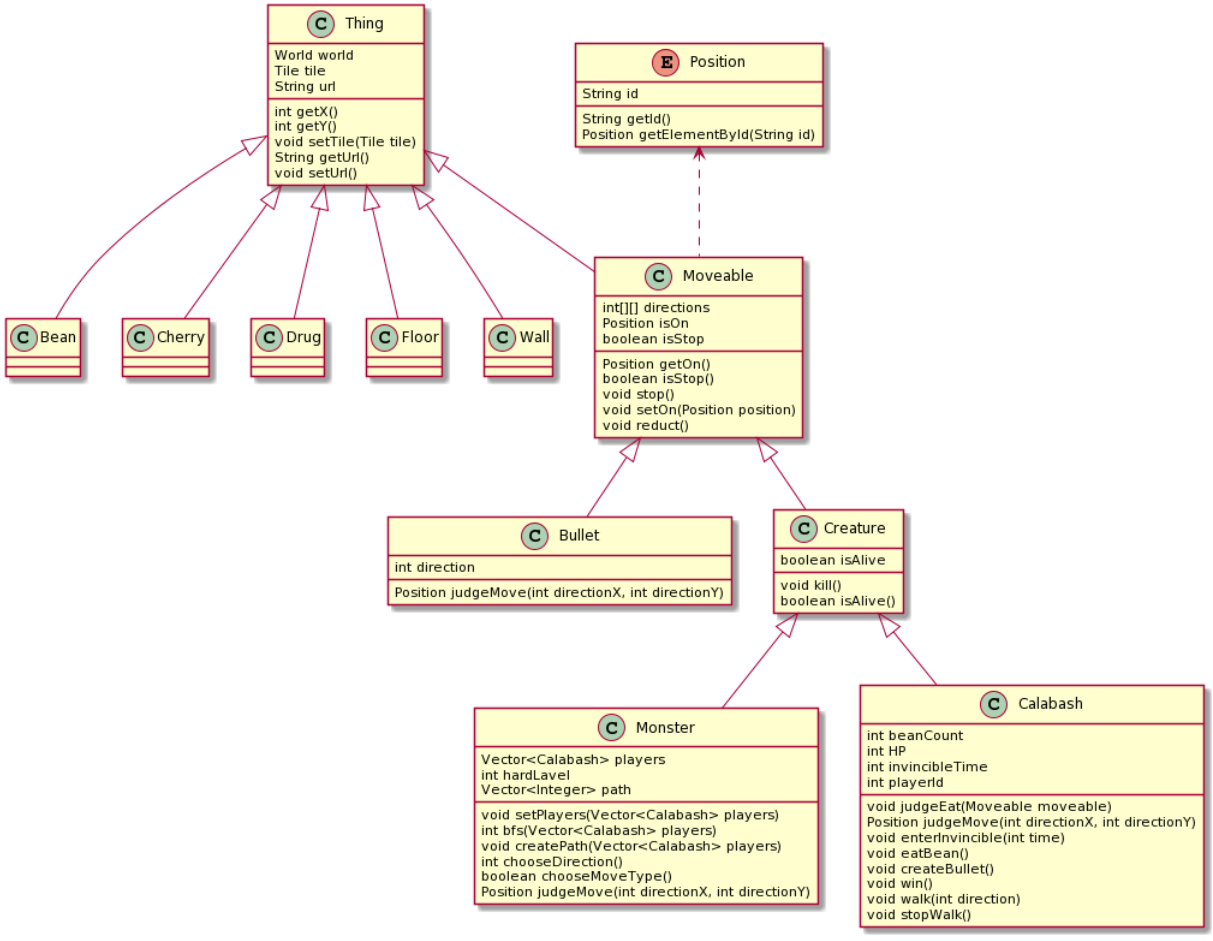

图 2　游戏中的各种元素

Figure 2　elements classes in this game

2.3 screen

screen 是一个在 server 和 calabashbros 模块之间进行交互，并对在棋盘中的所有元素进行管理的模块。这一模块会负责创建地图并根据 server 发送的指令对地图进行读写。其主要功能包括：调

用 MazeGenerator 创建地图数组、根据地图数组创建地图、创建所有元素并放置在地图上、响应用户的键盘事件并修改地图、读取地图并转换为字符串数组、以序列化的方式存储及读取游戏场景。

2.4 mazeGenerator

mazeGenerator 是一个用于产生地图的模块。其主要功能为创建地图并保持其连通性（以确保玩家可以吃到所有豆子）。为此，可以从地图的左上角出发进行随机广度优先遍历，在遍历过程中随机将当前位置的邻接坐标加入队列，并标记该坐标为可达。由于这一方式生成的地图过于狭窄，需要随机将一些不可达的坐标标记为可达以增加游戏空间，并在标记完成后执行一遍广度优先遍历以确保地图的连通性。接着便可以在空地上随机创建豆子和两种药水，并放置玩家。

2.5 asciiPanel

asciiPanel 是一个用于在界面上绘制图像的模块。在这一项目中，有两种元素会被绘制在屏幕上：图片和字符。其中每张图片的大小都是相同的。由于屏幕刷新速度非常快，若每次都要绘制整个屏幕的内容必然会导致闪烁。因此采用局部刷新技术，将屏幕划分为一个个与图片等大的方格，以字符串（实际为图片的 url）来代表图片，维护一个与屏幕方格数组尺寸相同的二维字符串数组。在每次刷新屏幕时，若某块方格中的图片未发生变化，则无需重新绘制该方格中的内容。对于字符，将其作为一个长度为 1 的字符串存入数组即可。

为了提高加载速度，可以无需在每次绘制图片时都通过 url 加载图片并进行绘制，而是在模块初始化时就预先加载所有所需的图片，放入 HashMap 以实现根据 url 来迅速读取。

2.6 模块化的优势

使用这样的模块化设计的最大好处是各个模块分工清晰，代码简洁且耦合度低。server 和 client 位于最前端，client 通过屏幕显示和键盘事件直接与玩家进行交互，server 处理玩家发送的所有请求，对 client 屏蔽 screen 的具体细节而只发送游戏的界面信息。这样，client 便可以不用考虑 screen 中复杂的元素和线程，而只需关注当前的界面是什么样的。server 也可以不必考虑界面的显示以及与玩家的交互，只需要对 screen 进行读写即可。MazeGenerator 只需负责生成地图数组，asciiPanel 则专注于将当前的场景绘制在屏幕上，screen 作为桥梁在 calabashbros 和 server 之间进行信息传递。每个模块各司其职，通过设计好的接口进行相互调用，使得代码整体十分清晰易懂，易于理解测试。

3 技术问题

本次实验的技术问题主要涉及以下几点：server 和 client 间的信息传递、游戏显示、线程交互、面向对象程序设计。

3.1 server 和 client 间的信息传递

由于棋盘大小非常大（50*50），故如何将棋盘内容从 server 发送给 client 是一个需要解决的问题，并且这一数据传输的效率将直接影响到整个游戏的响应速率。由于棋盘中的图片种类很多，故使用图片的 url 来代表图片。

首先考虑将所有 url 存入字符串数组，将整个数组在 server 处以对象的形式打包发送，在 client 处直接获取这一数组。但是数组对象传递的效率低下，导致屏幕刷新速度过慢。接着改为按行将棋盘上的图片的 url 传递给 client，url 之间用‘|’分割。但是这样会导致一次刷新需要传递 50 次数据，很容易造成一次请求的数据还未传输完成下一次请求就到来的情况，导致多次传输的数据混杂在一起。在多次尝试修改后，采用了如下优化方案：首先，由于屏幕的高度只够显示下 50*25 的棋盘，故无需将整个棋盘上的数据都传输给 client，而只需传输一半的数据。其次，可以将这 50 次数据传输合并为一次，使用 while 循环确保数据可以完全被传输。最后，如果数据仍然难以传输完成，可以为棋盘上的每种图片的 url 创建一个序号，用两位的序号代替十余位的 url 可以大幅降低需要传递的数据的长度。实验结果证明前两种优化已经可以保证游戏流畅可玩。

3.2 游戏的显示

如何将游戏界面显示在屏幕上也是本次项目遇到的一个主要问题。在实现了屏幕的刷新显示后，发现游戏过程中常常出现有一些位置的图片消失不见的情况。在调整屏幕刷新速率无效的情况下，通过在终端输出显示在屏幕上的字符的方式发现问题在于每次在输出游戏元素之间都会进行的 clear 操作。这一操作会在屏幕上输出空格字符，从而使得原本显示在上面的图片消失。而由于本项目采用的是局部刷新技术，每一张图片的修改都能够及时被更新到屏幕上，故这一 clear 操作是不必要的。将这一操作取消之后，问题得到了解决。

3.3 线程的交互

游戏中的每个 Moveable 的元素都是一个线程。如何控制使得玩家、怪物、豆子等元素之间的交互高效而可靠，是一个值得思考的问题。此处需要特别注意的是，由于地图的每个位置都只能有一个元素，故当怪物移动到豆子所在的位置时，豆子需要暂时从屏幕上消失，当怪物移动开时豆子才会再次出现在屏幕上。另外，若玩家碰到了与豆子处在同一个位置的怪物，则会在杀死怪物的同时吃到该位置上的豆子。

为此，可以为每一个 Moveable 的对象均创建一个 Position 类（一个枚举类）的对象，后者用于标识当前对象正处于何种类型的对象之上。当对象需要移动时，在移动前根据其所处的 position 恢复其下方的对象，并根据其所希望前往的位置更新其 position 的值。为了解决元素间交互的问题，当元素的移动方向确定后，会先获取并判断其希望移动到的格子内的元素的类型。若当前元素与目标格内元素会产生交互（例如玩家和豆子、怪物和玩家），则会先结算对目标格内元素的影响（例如玩家扣血、怪物被杀）。接着将元素类型以 Position 的形式返回，根据类型结算对当前元素的影响。

3.4 面向对象程序设计的优势

本次项目整体采用面向对象的程序设计方法。这一设计方法的最大优势是可以使得代码看上去更加贴近现实，易于分析及思考，易于使用流程图表示对象间的数据传递关系。整个游戏中涉及到的元素数目众多，如果可以清晰地表示出它们之间的关系，则写代码的过程就如同玩自己写的游戏的过程，可以使得代码非常易于开发及维护。例如，怪物的移动过程就像是一位在丛林中探险的旅行者，它先确定自己要移动的方向，接着使用小刀劈开自己前进的路上的障碍（这一操作会对目标位置上的物体造成破坏），最后移动到目标位置上。服务器和客户端之间的交互则像是实现了某种网络协议，两者互相从对方处读取数据并解析，接着进行相应的处理。这样写出的代码十分生动形象，通俗易懂。

4 工程问题

在本次项目中，采用面向对象的设计方法和模块化的设计方法均能够大幅度提升开发效率及代码质量。在开始着手写代码之前，有必要理清楚各个类之间的关系，从而为每个类设计出形象的、参数和返回值含义清晰的、易于使用的方法。这样的设计方法使得项目的开发易于建模，能够按部就班地、一个一个模块、一个一个功能地完成，避免了在面对很复杂的工程时无从下手的状况，也能够大幅度降低出错的可能性。例如，在设计客户端与服务器的交互过程时，无需考虑复杂的功能需求，而只需考虑两者需要向对方传递什么样的数据，以及这样的数据从何而来即可。接着便可以围绕着数据的传输，逐步构建两者的代码。另外，这样的设计的可扩展性也非常强。例如，在棋盘中添加一种新的元素时，将无需为该元素设置多种多样的属性，而只需根据其是否是生物、是否会移动来继承 Creature 或 Moveable 类即可。这样可以使得代码能够得到很好的复用，大幅节省了项目的开发时间。

5 课程感言

这是一个大型的、多模块的项目，但若使用合适的开发方式，循序渐进地进行开发，可以使得开发难度大大降低。因此，本次项目作业的设计十分合理，既没有给人太重的负担，又让人感觉到收获满满。另外，项目并未给出具体要求，一切都由我们自由发挥，这使得我们可以综合自身的兴趣想法、时间安排等因素去考虑自己要设计出一款怎样的游戏。总体来说，这是一次十分有意义的大型项目。

参考文献

1 Author A, Author B, Author C. Reference title. Journal, Year, Vol: Number or pages

Jiaxin Tang[1]

1. *181860086*
E-mail: 181860086@smail.nju.edu.cn