Java大作业报告

计63 刘家硕 2016011286

一、整体思路与架构

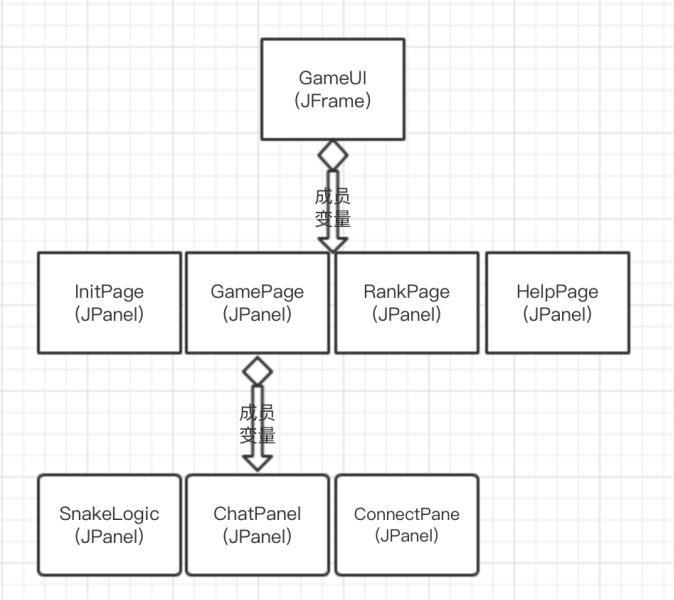
本项目主要分为三个部分，即UI界面、游戏逻辑与网络传输。

UI界面部分，主要是为了实现不同界面之间的切换。包括了注册登陆窗口（loginFrame）与游戏主窗口（GameUI）；游戏主窗口中含有游戏初始界面(InitPage)、游戏界面(GamePage)、排名界面(RankPage)、游戏帮助（规则）界面(HelpPage)；为了实现聊天等功能，游戏界面中包含了三个部分(SnakeLogic, ConnectPanel,ChatPage),对应于贪吃蛇游戏画布、服务区客户端拨号连接、聊天界面，这部分设计灵感源于QQ游戏的窗口，即大部分为游戏界面，游戏的旁边可以聊天。关系图如图一所示：

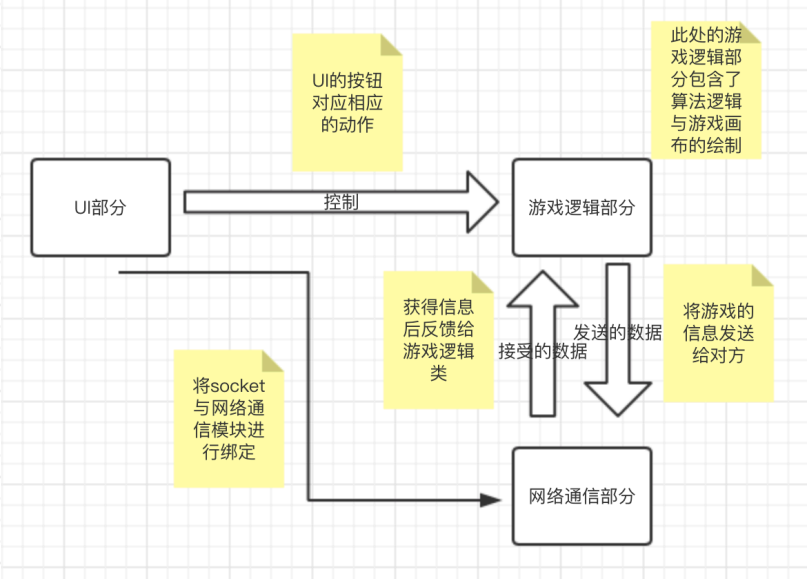
游戏逻辑部分，主要负责贪吃蛇游戏的运行、判定与游戏画布的绘制，并且尽可能地将游戏逻辑与图像绘制分离开来，其中含有socket通信类的实例作为成员变量来负责网络通信，具体见第二部分。

网络通信部分，主要负责游戏中双方信息的同步、聊天内容的同步、游戏开始时的同步。作为一个独立的模块，在游戏主逻辑类、连接界面类、聊天类中均含有其实例作为成员变量来负责信息的传输与接收。具体见第二部分。

三个主要部分的调用关系为图二所示



图一：**UI界面关系示意图**



图二：三个部分关系图

二、模块的设计与实现思路

1、UI模块：贪吃蛇游戏本身只需要一个界面即可，但是考虑到排行榜、聊天等功能，UI界面就要复杂一些。首先排行榜需要用户名等信息，于是我设计了登陆注册界面；之后进入游戏主窗口之后，首先呈现游戏初始界面，可以选择开始游戏、查看排行榜、游戏规则帮助、退出。对应着图一中的四个界面（JPanel）。

初始界面（InitPage）负责用户的界面选择与切换，根据不同的按钮选择跳转不同的界面。

排行榜（RankPage）负责加载排行榜、新增高分排名，显示前10位的得分，其中包含了从本地文件（GreedSnake/data/rank.txt）中读取排名并显示、向该文件写入排名等功能，在游戏结束时会更新排名。

游戏界面（GamePage）负责整个游戏主体，包含了三块JPanel:SnakeLogic, ConnectLogic, ChatLogic。其中SnakeLogic为贪吃蛇游戏的显示画布，具体设计会在游戏逻辑类部分介绍。ConnectLogic为连接部分，负责建立服务器或与服务器连接，当连接完成后会将建立的Socket与网络传输模块绑定起来，从而完成游戏中的通信部分。ChatLogic为聊天室部分，负责双方的聊天，通过绑定后的网络传输模块来完成。

游戏规则帮助界面（HelpPage）比较简单，仅为显示游戏规则。

以上四个界面均位于游戏主窗口（GameUI）中，实现了四个窗口之间的自由切换。

2、游戏主逻辑模块：SnakeLogics

贪吃蛇游戏包括两个部分：算法方面的逻辑与游戏画布的绘制。

算法方面：

（1）SnakeLogic类中包含有Snake类、Entity类的实例作为成员变量，为了尽可能的体现面向对象的原则，我将涉及到蛇的功能均在Snake类的内部实现，例如蛇的移动、死亡判定、复活、进洞出洞等，所需要的地图信息由函数的参数传入，符合“开闭”原则。Snake类中还含有蛇的生命数、所得分数等信息。

（2）SnakeLogic类中含有Food、Obstacle、Hole类，这三个类均继承自Entity类，其中封装了物体之间的碰撞检测。Snake类中表示蛇身体使用的是SnakeEntity类的对象数组，该类同样继承自Entity类，碰撞检测交由这几个底层的类来完成，尽可能地体现了面向对象的思想。

（3）SnakeLogic类中含有包含地图信息的数组GridBlocked与GridState,分别表示障碍物信息与物体信息，在碰撞检测时作为参数传递。

（4）SnakeLogic类中包含了Wall类来表示墙壁，其中含有不同形状的墙壁作为障碍物，通过维护GridBlocked数组来判定蛇的碰撞。

画布绘制方面：

（1）由于贪吃蛇属于即时类游戏，对于渲染效率要求相对较高，而画面中的物体占少数，整个游戏棋盘类似于一个“稀疏矩阵”，为了尽可能的提高渲染效率，我没有采取遍历画面中的格子逐个渲染，而是预先加载背景图片（草地），之后渲染时只将地图中的物体（鸡蛋、石头、墙壁、洞）所在位置分别渲染其对应的图片（同样已经预先处理为了Image对象），以及两条蛇的身体进行渲染。我在Food,Obstacle,Hole,SnakeEntity类中均重写了基类Entity中的PaintObject(Graphics g)方法，在其对应位置画对应的图片，在SnakeLogic类的Paint()中只需分别调用地图中物体各自的绘画函数即可完成绘制，这样做可扩展性很强，加入新的物体只需要写好碰撞函数与绘制函数即可。

（2）由于从文件中读取图片很慢，为了提高效率，我在初始化的时候预先将用到的图片加载到了bufferedImages类中作为其成员变量，在绘图的时候根据不同类型的物体选择对应的图片drawImage即可。

3、网络通信模块：SocketHandle

由于是网络对战版贪吃蛇，对网络通信的要求比较高，我先后尝试了两种架构的通信：

（1）双方本地计算的方式：即为了最大化即时性，尽可能地减少数据传输量，双方只同步“上下左右”的操作信息，游戏逻辑均在服务器与客户端的本地进行，只要传输得到的操作信息一致，在初始地图一致的情况下，双方游戏内容必定是一致的。这样我个人认为相当于“双核”运转。但在完成之后实际测试时，发现这种通信方式的容错率几乎为零，一旦哪里传输出了问题，哪怕是一个信息没有到位，之后的游戏信息就完全不一样了。在我看来，维护游戏的即时性与游戏双方的一致性本就是一个tradeoff,不可能都做的很好，但一定不能有一方很差。考虑到贪吃蛇游戏比较简单，数据

传输量本身不大，过于减少数据的传输反而极大地损害了游戏双方的一致性，有些得不偿失，于是我换用了下 main第二种方法。

（2）服务器本地计算，客户端只负责向服务器发送操作信息、接受服务器的信息并根据信息完成绘图。实现的核心主要是信息传输的方式，这种架构下服务器需要向客户端传输足够多的信息来使客户端完成绘图。但由于我的SnakeLogic写的比较OOP，其实需要传的消息并不多。首先地图上的墙壁、障碍物、洞是不会改变的，只需要游戏开始时同步一次即可，游戏中需要发送给客户端的只有蛇的位置、属性信息与随机生成的蛋的信息。而查阅资料发现Java中支持传输对象，通过序列化与反序列化的方式，问题便迎刃而解。相比于人工编码为字符串再人工解码，这样的方式会快一些。只要有了蛇的信息与蛋的信息，由于绘图都是交由最底层完成的，客户端便可以很方便的完成绘图。服务器方面，每隔一定时间向客户端发送一个ConnectCheck信号，客户端收到信号后开始repaint()更新。可以保证客户端绘制的图片与服务器上当前的状态绝大多数情况是一致的（自己测试没有出现过不一致），极小的概率不一致（但总有一些很特殊的情况，应该很难避免）。

4、UI与数据交互方面：

（1）界面方面，界面只负责按钮的显示与事件响应，具体功能由内部的游戏逻辑类来完成。

（2）游戏界面方面，界面的绘制交由最底层的物体类来完成。物体类作为最基础的单元，具有位置信息，可以绘制物体图片。

三、特别的设计思路

1、图片渲染：考虑到IO操作费时，于是在初始化的时候将需要用到的图片都读了进来作为Image对象，之后在渲染的时候只需要drawImage即可，提高了渲染速度与效率。

2、蛇的移动：将蛇的移动分为两步，尾巴去掉与头变长两个函数，这样设计对于蛇吃食物的变长（头增尾不减），蛇进洞（头不增尾减）都有很好的扩展性，简化了代码。

3、网络通信：服务器向客户端传输的是对象，通过Java自己的序列化与反序列化来实现，代码简单效率很高，相比于人工编码字符串再人工解码要高效、方便，更加保证网路通信的即时性。

四、实现的一些附加功能及说明

1、聊天功能：在游戏主界面中服务器与客户端连接成功之后，可以在编辑框中输入内容，点击发送按钮即可发送。在建立连接之前是锁死的。

2、播放音乐：游戏主界面中有“音符”按钮，点击即可播放本地音乐，点击两次为暂停播放。（音乐为wav格式的音频）。

3、排行榜功能：在游戏正常结束后，可以进入Rank页面中查看排名，排名文件位于GreedSnake/data/rank.txt中。

4、登陆注册功能：双击打开游戏之后会首先弹出登陆注册界面，写入用户名与密码进行注册，注册后使用相应用户名密码进行登录，注册时用户名已存在会提示，登陆时错误也会提示，信息保存在了GreedSnake/data/name.txt与pass.txt中。

5、地图生成：地图的生成考虑到了墙的均匀性以及多样性，内置了直线形、直角形、圆弧形的墙壁（圆弧形使用了中点画圆法，但由于格子数比较小，效果一般，目前没有展示，代码中可见该部分），在生成墙壁的时候，在地图外围会随机生成四堵直角形的墙壁，并且方向根据位置进行调整使得开口朝向地图内部。之后在地图内部随机生成一些直线形的墙壁。有的时候会不均匀，但大致是比较均匀的。

6、多UI界面切换：实现了两个窗口，4个界面的切换，并且切换方便不会出bug。

五、基本功能的说明

1、窗口自适应大小：最开始的登陆界面窗口大小设置为了不可变，之后的游戏界面可以自由拖动大小，组件会根据窗口大小来自适应调整。

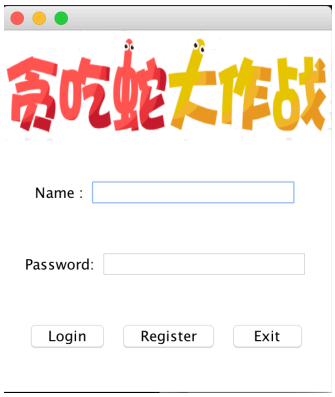
2、服务器与客户端的连接：“server”为建立服务器，“client”为客户端连接（输入对应ip与端口）,连接成功会有倒计时，成功后会解除对聊天界面的锁死。同步完毕之后游戏会开始。连接不成功会有提示，需要重新连接。

3、游戏部分：自己方的蛇蛇头为红色，对方的蛇蛇头为灰色已便于区分。地图中的墙壁、石头、蛇的身体均视为障碍物，撞到即死。蛇最开始的初始化位置固定为（20，20）与（40，40），之后的复活位置为随机一个洞口，但考虑到游戏体验，自己和对方的蛇都是从两个洞中随机生成的。另外出洞时蛇的方向进行了处理，不会出现一出洞就死掉的情况。游戏暂停对双方均有效（空格键暂停），游戏开始也对双方有效，会在地图的顶端进行提示。游戏分数以及各自的生命数、蛇的状态也会在顶端进行提示。蛇的加速为右侧”Speed”栏，拖动可改变蛇的移动速度。死亡之后会弹出提示，之后回到游戏初始界面而不是退出程序，还可以继续进行连接与游戏。

（备注：在快速按下两个方向键时可能会导致蛇的死亡，因为方向改变是即时生效的，但刷新是有时间间隔的。）

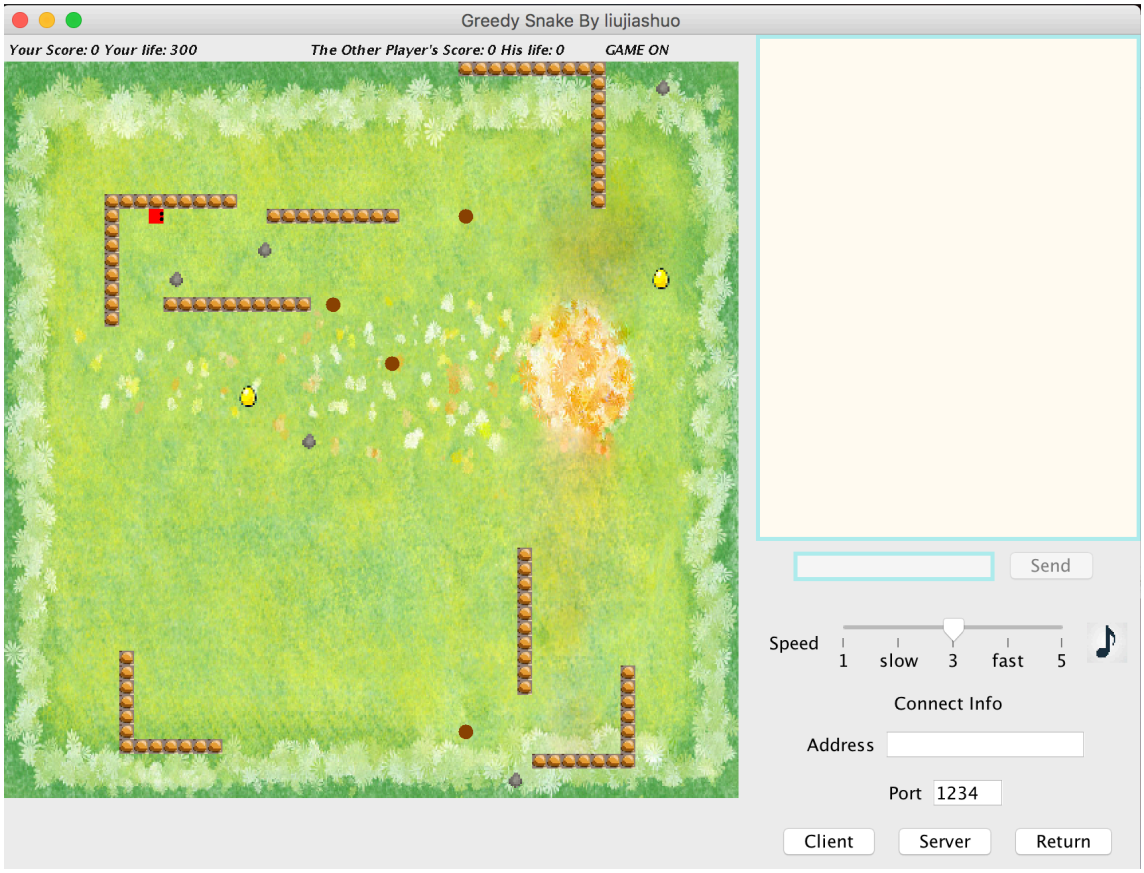
4、按键的锁死：在服务器与客户端的连接成功之前会锁死聊天界面，以及速度设定的控制栏。

5、网络通信异常的处理：当游戏进行中，服务器突然关闭时，客户端用户在点击“上下左右”控制方向之后会弹出提示；客户端突然退出时，服务器端也会弹出提示，之后会回到游戏初始界面。



图三：游戏登陆界面

图四：游戏初始界面



图五：游戏主界面

Greedy Snake By liujiashuo

Rank	Name	Score
1	liujiashuo	8
2	qwe	4
3	liu	2
4		N/A
5		N/A
6		N/A
7		N/A
8		N/A
9		N/A
10		N/A

图六：排行榜界面

六、一点点说明

由于条件限制，我的代码一直是在本机进行测试，并且操作系统为OSX，在两台电脑以及其他系统没有测试过，若出现问题烦请助教联系我（13015155336），实在抱歉。（大作业写了很久投入很多，不希望因为这个前功尽弃，望助教理解）