包

四个主要的包：

serverUI： 服务器

clientUI： 客户端

snakedotwar： 蛇点大作战

snakeswar： 蛇蛇大作战

其他包：

resource: 图片和音乐资源

frame: 一些公用的常量和类

biochemistry: 生化模式（未实现）

运行方法

首先把GreedySnake\_V2.6这个文件夹放到workspace目录，然后从eclipse新建一个projects，名字为GreedySnake\_V2.6，再直接点finish即可。

服务器： Server.class类

客户端： Client.class类

游戏规则

多人对战版贪吃蛇

清真蛇点大作战

1、一条蛇，多个点，由不同玩家控制。

2、随着时间的增长，蛇的长度会变长，同时难度增大，吃掉所有点，蛇获胜。

3、点拖到蛇死亡，点获胜。

4、四周是否有墙，待判定。

5、点可以穿过蛇的身体。

6、是否有时间限制，是否刷点，待测试。

清真蛇蛇大作战

1、多条蛇，多名玩家控制。

2、蛇的速度一致，互相吃，蛇可以从另一条蛇的中间攻击，中点位置到尾巴，变成固定不动的点，可被其他蛇吃掉。

3、胜负判定：蛇斩断其他蛇+3分，吃到死亡掉落的点+1分，死亡不扣分，积分系统排出名次。

霸道生化模式

1、一条蛇，四个点，分别由不同玩家控制。

2、蛇吃到点，点不能动，变成蛇。（蛇的长度短于母蛇）

3、当最后只剩一个点，地图上出现地道，亮点进入地道，亮点获胜。

4、蛇吃光所有点，蛇获胜。有时间限定，时间到，点存活，点获胜，或蛇均死亡。

5、蛇与蛇之间有无碰撞？

蛇点大作战实现方法

游戏规则：

1、一条蛇，多个点，由不同玩家控制。

2、随着时间的增长，蛇的长度会变长，同时难度增大，吃掉所有点，蛇获胜。

3、点拖到蛇死亡，点获胜。

4、四周是否有墙，待判定。

5、点可以穿过蛇的身体。

6、是否有时间限制，是否刷点，待测试。

需要的类：

AllConstValue {

包含所使用到的所有常量

}

Node implements Serializable{

所有的蛇和点的组成单元，一条蛇可以是多个Node，一个点只能是一个Node

成员变量: x, y, flag // flag用于标记该Node是否是蛇头

}

Snake implements Serializable {

成员变量: snake(node[]), curDir, cnt

方法: getNode(), getLength(), getDir(), isDead() , move(), changeDir(), add() // 不需要判断是否吃到了点，点碰到蛇就自动消失

}

Dot implements Serializable{

成员变量:dot( node), curDir

方法:isDead(), move(), changeDir()

难点：isDead()，当点碰到蛇头就会死亡，所以需要判断点是否和蛇头相遇，而服务器会总是向所有玩家报告蛇和点的位置信息，并且所有玩家都需要记录，这样所有点玩家都可以得到蛇头的位置。

}

SnakeWin {

使用蛇的玩家使用此类

用途：让蛇运动(新建一个线程让蛇跑起来，监听键盘用于改变蛇跑动的方向)；每隔一定时间就要调用add()函数，让蛇变长，修改线程睡眠时间让蛇的速度增大；在终端显示所有玩家的位置信息。

}

DotWin {

使用点的玩家使用此类

用途：让点运动(新建一个线程让点跑起来，监听键盘用于改变点跑动的方向)；在终端显示所有玩家的位置信息。

}

Server {

服务器使用

用途：连接所有玩家，通知玩家开始游戏，每个玩家时刻报告自己的位置给Server，

Server再将这些信息汇总起来，发送给所有玩家

}

Client {

玩家使用

用途：玩家通过Client向服务器报告自己的位置，并接收由服务器发送过来的所有玩家位置信息，并转发给Show用于在玩家终端上显示

}

难点：

1. 关于socket发送序列化对象，由于各客户端的都是独立运行，不能相互通信，所有各客户端发送的信息是没有顺序的，服务器端怎么判断发送过来的是什么？也就是说服务器端不能判断接收到信息是点还是蛇，因为收到的都是字节。

解决办法：服务器端给与相应客户端通信的线程编号，规定玩蛇的编号为0，其余依次增加。当服务器的某个线程端收到信息时，先检查该线程的编号，在决定对哪个对象进行操作。

1. 服务器需要向所有客户端报告当前所有蛇和点的位置信息，以便在客户端能够显示出来，但报告之前需要接收到所有客户端发送过来的信息，并进行汇总，问题在于服务器如何知道所有客户端都已经发送了，也就是说服务器端不知道什么时候向所有客户端报告位置信息。

解决办法：设置一个状态数组vis[]，对于每一轮接受信息，每有一个客户端发送过来信息，vis中相应的位置为true，当检查到vis中所有位都为true的时候，就可以向所有客户端报告信息。

待解决问题

1. 多线程网络编程的资源同步还没解决，所以目前只能支持两个玩家
2. 客户端卡顿问题未解决