动手做联机五子棋对弈游戏

本篇概要：

我们通过现有单机版五子棋的基础上，继续开发联机游戏版本。本篇并不重点介绍如何联机，着重建设服务器端游戏结构。

开发环境：前端HTML5 服务器端 Java

单机版五子棋GitHub仓库地址：

<https://github.com/liuxinyumocn/LittleGame>

本项目GitHub仓库地址：

<https://github.com/liuxinyumocn/GobangOnline>

文章首发地址：

<http://liuxinyumo.cn/>

上期地址：

<http://liuxinyumo.cn/index.php?s=/Home/Index/content/id/7.html>

服务器端采用Java语言设计，游戏通讯采用Socket通讯协议，但本篇暂时并不介绍如何联机，而是着重建设服务器端游戏结构。

打开Eclipse，并新建一个Java Project，我将工程目录放置本项目仓库目录下Server文件夹内，请小伙伴下载后直接使用Eclipse打开即可。

开发服务器端我们就需要一些新的设计思想。为保证游戏公平性，实际上，游戏开发中应在服务器建设玩家角色，而客户端仅仅向服务器端发送玩家角色的行为动作，游戏逻辑处理虽然在客户端有执行，而服务器仍然要执行一次。客户端的执行是为了给玩家更好的体验，而服务器端的再执行是为了保证游戏公平不被恶意篡改。

为了更好的理解我打一个形象的比喻：

Socket链接我们比作阿凡达电影中的神经链接，人类即为真实玩家，利用Socket链接来远程控制阿凡达（即服务器端中的Player），神经链接是双向通信的，所控制的阿凡达会不断将自己的感知画面传回人类（即游戏客户端画面会不断受服务器响应而变化），如若Socket链接断开，则意味神经链接断开，服务器端则应处于断线等待链接状态。

因此按照该思路，实际上我们的服务器端每一次对局都将是一个三方交互的情景，玩家A，玩家B，裁判（服务器）。依靠我们习惯性的面向对象思想，所以我们优先以裁判视角来先设计。

所谓“裁判”，其实由一个房间而生，从玩家进入房间、双方准备、开始对局。因此这里我们设计“裁判”类，实际上是“房间”（Room）：

注意：虽然我们称“房间”是一个视角活动，但我们设计时为了减少服务器开销，实际上“房间”的所有活动均应该是被动产生。

考虑一下房间的行为与属性：

告诉“裁判”我（Player）进来了，

告诉“裁判”我准备好了（如果双方均准备好则开始游戏），

如果开始游戏：

告诉裁判我下哪里。

裁判告诉我胜利/失败。

属性：

房间号。