一，测试说明：

Unity基于其特性，难以手动编写“单元测试”的方法进行测试。

难点在于Unity的程序结构，GameObject管理者是UnityEngine，代码只是GameObject的一个Component，继承自Unity基础类MonoBehavior，不允许使用new的方式初始化。

另外Unity本身的特点在于可以快速地建立场景，直接执行代码可以立刻查看运行结果，所以我们主要的测试方法为黑盒测试：即每写1-2个功能，便立刻通过执行程序的方法直观地观察程序运行情况。

二，测试中出现过的典型bug：

1.网络数据不通，一收数据就死机。一开始以为是C#网络类的错误，后来发现是C#的Stream类公有的错误，StreamReader.Readline不能读取StreamWriter.Writeline的数据，只能读取Write(info+”\r\n”)的数据。这个问题应该是属于微软的问题了，不属于我们自己的问题。

2.控件失灵，所有控件都不响应OnClick命令。这个错误的原因是因为我们在底板上放了个大的鼠标响应器，但是因为Unity是三维引擎，而新的响应器的大小比物件实际体积大，覆盖了盒子上方的所有控件。

3.经常会在git同步以后丢失各个控件的prefab消息。这个是unity和git之间不兼容导致的。我们惟一能做的就是merge后经常重新输入prefab数据。

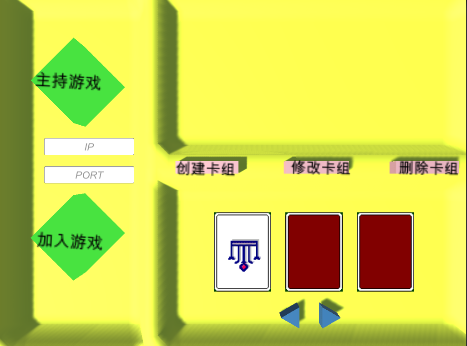
其它bug：

没有清理测试信息导致的bug。

随机数种子使用顺序不一致或其它没有同步的数据导致的server和client数据不一致的bug。

三，程序测试流程

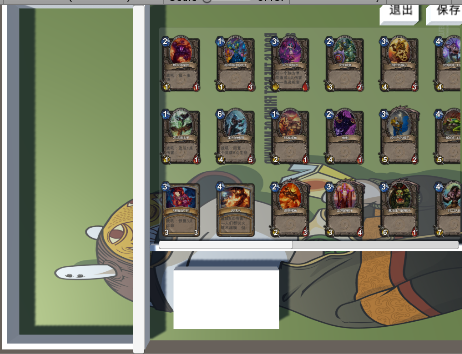
1.打开程序，能看到主界面。



2.点击添加卡组，能够看到卡组的总数增加；

3.选择一个卡组之后点击删除卡组，能够看到卡组总数减少；

4.选择一个卡组选择修改卡组，能够进入卡组修改界面；



5.在卡组修改界面，能够通过拖动滑条看到所有可以使用的卡片和放在卡组里的卡片；

6.通过鼠标拖入和拖出，可以修改卡组里的卡片；

7.点击保存再点击退出，可以保存你的卡组；不点击保存直接点击退出，则可以取消这次修改的记录；

8.再次选择修改过的卡组并且从主界面进入卡组修改界面，可以看到你的卡组已经更新。

——单机测试部分结束——

9.准备两台电脑连入Internet并且同时执行程序，选好准备使用的卡组。

10.一台机器作为主机，输入准备使用的端口号并且点击主持游戏，进入等待状态。

11.另一台机器作为参与者，输入对方的IP地址和端口号进入游戏。此时双方应该都进入游戏界面。



12.双方开始，随机选定一个玩家作为先手。

13.在玩家的回合内，如果将随从卡拖上场，那么随从将被召唤上场。

14.拥有战吼的随从在上场之后会发动战吼效果；如果需要选定目标，则在上场后点击使用。

15.刚上场的随从不能攻击；上一回合上场的随从，在之后的每一回合都可以攻击一次。

16.可以将法术卡拖向指定目标，就会自动执行法术卡。

17.点击英雄技能图标后点击单位，就会发动英雄技能。英雄技能每回合只能发动一次。

18.使用任何卡片都要消耗法力。法力不足时不能发动。在玩家的回合开始时，玩家的最大法力上限增加1（不会超过10），并且回复法力值到其上限。

19.点击回合结束按钮，就会进入对方的回合。在对方的回合，你不能做任何操作，但是可以看到对手的所有操作。

20.如果某一方英雄生命力降为0，则游戏结束。

按照这样的次序进行测试，若没有问题，则游戏测试正常。

另外，因为卡片效果的多样化，所以要测试每个卡片是否正常使用，就需要在游戏状态下分别使用这些卡片。