**成都信息工程学院计算机学院**

**课**

**程**

**实**

**验**

**报**

**告**

|  |  |
| --- | --- |
| **实验课程：** | **网络游戏编程** |
| **实验项目：** | **网络游戏编程最终项目** |
| **指导教师：** | **陈睿** |
| **学生姓名:** | **唐亚婷** |
| **学生学号：** | **2019053019** |
| **班 级：** | **数字媒体技术191** |
| **实验地点：** | **B215** |
| **实验时间：** | **2022年5月20日14点~16点** |
| **实验成绩：** |  |

## 一【上机实验内容】

1. 注册登录

添加注册登录系统场景

向服务器发送注册申请、用户名空格检查

已存在用户名的显示

用户已登录的显示（包括服务器端代码）

密码错误的显示（包括服务器端代码）

用户名不存在的显示（包括服务器端代码）

1. 房间系统

能接收服务器发来的房间列表

能用某种方式显示房间列表

能用某种方式选中某房间，并发送进入房间协议

能接收服务器发来的开始游戏协议，并进入游戏

1. 同步系统

能初始化两个hero的站位

能同步位置

能同步hero朝向

能同步开火

Rocket击中hero时能减少血量

hero2的血条跟随

能同步血量

在血量为0时能结束游戏，发送结果给服务器记录（包括服务器端代码，及协议编写）

把房间系统中显示的ip地址换为用户名（包括服务器端代码、数据结构与逻辑）。

## 二【上机实验步骤】（重点）

协议列表：

1. 注册登录

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 协议名 | 参数 | 示例 | 方向 | 含义 | 备注 |
| 1 | Regist | 2 userName,  passWord | Regist tyt101 tyt101 | 客户端->服务器端 | 客户端申请注册 | 客户端用户名不可有空格，对密码加密 |
| 2 | RegistFail | 1 | RegistFail 1 | 服务器->客户端 | 注册失败 | 用户名已存在 |
| 3 | RegistSuccess | 无 | RegistSuccess | 服务器->客户端 | 注册成功 |  |
| 4 | Login | 2  username  passWorld | Login tyt101 tyt101 | 客户端->服务器端 | 客户端申请登录 |  |
| 5 | LoginFail | 1(值为1，2，3) | LoginFail 1 | 服务器->客户端 | 注册失败 | 根据值分析失败原因 |
| 6 | LoginSuccess | 无 | LoginSuccess | 服务器->客户端 | 登录成功 |  |

1. 房间系统

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 协议名 | 参数 | 示例 | 方向 | 含义 | 备注 |
| 1 | roomList | 1+n个，第一个为房间数，后面n个房间地址 | roomList 3 111 222  333 | 服务器->客户端 | 房间列表 | 服务器分发，2s一次 |
| 2 | createRoom | 无 | createRoom | 客户端->服务器端 | 创建房间 | 客户端控制能吃创建一次 |
| 3 | enterRoom | 1个，加入的房间地址 | enterRoom 111 | 客户端->服务器端 | 加入房间 |  |
| 4 | beginGame | 1个(0,1).区分进入游戏的两个客户端 | Begin 0 | 服务器->客户端 | 进入游戏 |  |

1. 同步系统

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 协议名 | 参数 | 示例 | 方向 | 含义 | 备注 |
| 1 | Position | 3个，第一个hero的横坐标，第二个hero纵坐标，第三个血量 | Position 10 10 80 | 客户端->服务器端->客户端 | 同步位置+血量 |  |
| 2 | Flip | 无 | Flip | 客户端->服务器端->客户端 | 转身 |  |
| 3 | Fire | 无 | Fire | 客户端->服务器端->客户端 | 开火 |  |

协议实现、

1. 注册登录，房间系统的界面设计(附界面截图)

注册：

图形用户界面, 应用程序

描述已自动生成

登录：

图形用户界面

描述已自动生成

房间系统：

图形用户界面

描述已自动生成

图形用户界面

描述已自动生成

1. 每条协议实现的具体步骤与代码(附游戏界面截图)

# 注册

public void RegistBtnClicked()

{

userName = GameObject.Find("Canvas/RegistPanel/usrPasbox/userInputField/Text").GetComponent<Text>().text;

password = GameObject.Find("Canvas/RegistPanel/usrPasbox/passInputField/Text").GetComponent<Text>().text;

var spaceIndex = userName.IndexOf(" ");

if (spaceIndex != -1)

{

GameObject.Find("Canvas/RegistPanel/usrPasbox/userInputField").GetComponent<InputField>().text = "用户名不能

有空格";

return;

}

//密码加密

MD5 md5 = new MD5CryptoServiceProvider();

byte[] passWordMD5Byte = md5.ComputeHash(System.Text.Encoding.Default.GetBytes(password)); //MD5 散列

passwordMD5 = BitConverter.ToString(passWordMD5Byte).Replace("-", ""); //转换为字符串

sendBuff = System.Text.Encoding.UTF8.GetBytes("Regist " + userName + " " + passwordMD5);

clientSocket.Send(sendBuff);

}

if ("RegistFail" == recvStrs[0])

{

Debug.Log("用户名已存在");

GameObject.Find("Canvas/RegistPanel/usrPasbox/userInputField").GetComponent<InputField>().text = "用户名已存在";

}

if ("RegistSuccess" == recvStrs[0])

{

Debug.Log("注册成功");

SceneManager.LoadScene("Room");

}

注册：图形用户界面

描述已自动生成用户名已存在：图形用户界面, 应用程序

描述已自动生成

注册成功

用户名已存在表格

描述已自动生成

用户名不能含空格：图形用户界面

描述已自动生成含空格显示图形用户界面, 应用程序

描述已自动生成

# 登录：

public void LoginBtnClicked()

{

userName = GameObject.Find("Canvas/LoginPanel/usrPasbox/userInputField/Text").GetComponent<Text>().text;

password = GameObject.Find("Canvas/LoginPanel/usrPasbox/passInputField/Text").GetComponent<Text>().text;

var spaceIndex = userName.IndexOf(" ");

if (spaceIndex != -1)

{

GameObject.Find("Canvas/LoginPanel/usrPasbox/userInputField").GetComponent<InputField>().text = "用户名不能有空格";

return;

}

//密码加密

MD5 md5 = new MD5CryptoServiceProvider();

byte[] passWordMD5Byte = md5.ComputeHash(System.Text.Encoding.Default.GetBytes(password)); //MD5 散列

passwordMD5 = BitConverter.ToString(passWordMD5Byte).Replace("-", ""); //转换为字符串

sendBuff = System.Text.Encoding.UTF8.GetBytes("Login " + userName + " " + passwordMD5);

clientSocket.Send(sendBuff);

}

## 客户端

if(recvStrs[0] == "LoginSuccess")

{

Debug.Log("登录成功");

SceneManager.LoadScene("Room");

}

if (recvStrs[0] == "LoginFail")

{

if (recvStrs[1] == "3")

{

GameObject.Find("Canvas/LoginPanel/usrPasbox/userInputField").GetComponent<InputField>().text = "该用户已

登录";

Debug.Log("用户名已登录");

}

else if (recvStrs[1] == "2")

{

GameObject.Find("Canvas/LoginPanel/usrPasbox/passInputField").GetComponent<InputField>().text = "密码错误";

Debug.Log("密码错误");

}

else if(recvStrs[1] == "1")

{

GameObject.Find("Canvas/LoginPanel/usrPasbox/userInputField").GetComponent<InputField>().text = "用户名

不存在";

}

}

## 服务器端：

void LoginHandle(string[] args,StateObject stateo1)

{

string userName = args[1];

string passWordMD5 = args[2];

//查询是否有同名记录

string dbPath = "C:\\Users\\tyt\\Desktop\\Server01\\Server01\\User1.mdb";

OleDbConnection conn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" + dbPath);

conn.Open(); //打开数据库连接

OleDbCommand cmd = conn.CreateCommand();//建立SQL查询

cmd.CommandText = "SELECT \* FROM table1 WHERE username='" + userName + "'";

OleDbDataReader odrReader = cmd.ExecuteReader(); //执行Command命令

if (odrReader.HasRows) //有同名记录

{

while(odrReader.Read())

{

/\*Console.WriteLine(odrReader[2].ToString());\*/

if (odrReader[3].ToString() == "true")//用户已登录

{

stateo1.sock.Send(System.Text.Encoding.ASCII.GetBytes("LoginFail 3 "));

}

else

if(odrReader[2].ToString() == passWordMD5)//登录成功

{

stateo1.sock.Send(System.Text.Encoding.ASCII.GetBytes("LoginSuccess "));

}

else//密码错误

{

stateo1.sock.Send(System.Text.Encoding.ASCII.GetBytes("LoginFail 2 "));

}

}

}

else

{

stateo1.sock.Send(System.Text.Encoding.ASCII.GetBytes("LoginFail 1 "));

}

conn.Close(); //关闭数据库连接

}

输入错误的用户名电子设备的屏幕

描述已自动生成用户名不存在电脑屏幕的照片

描述已自动生成

输入错误的密码电脑屏幕的照片

描述已自动生成密码错误电脑屏幕的照片

描述已自动生成

# 房间系统

## RoomList服务器

public void SendRoomList(List<RoomObject> rooms)

{

String str = "roomList ";

str += (rooms.Count).ToString();

foreach (RoomObject room in rooms)

{

if (!room.playing)

{

str += " ";

str += ((IPEndPoint)(room.client0.sock.RemoteEndPoint)).ToString();

}

}

byte[] sendBuff = new byte[1024];

sendBuff = System.Text.Encoding.ASCII.GetBytes(str);

for (int i = 0; i < index; i++)

{

try

{

stateOs[i].sock.Send(sendBuff);

textBox1.Text = "发送:" + str + "To:" + stateOs[i].sock.RemoteEndPoint.ToString();

}

catch (System.Exception ex)

{

textBox1.Text = ex.ToString();

}

}

}

## createRoom

public void BtnCreateRoomClicked()

{

//点击创建房间，创建后(创建按钮)失效

tempSocket.Send(System.Text.Encoding.Default.GetBytes("createRoom "));

GameObject.Find("Canvas/btnCreate").GetComponent<Button>().enabled = false;

}

## enterRoom

public void onClick(GameObject sender)

{

//发送进入房间信息给服务器

int btnIndex = sender.GetComponent<EnterRoomBtnNum>().getNum();

string sendStr = "enterRoom " + strRooms[btnIndex + 2];

tempSocket.Send(System.Text.Encoding.Default.GetBytes(sendStr));

sender.GetComponent<Button>().enabled = false;

}

## 4. beginGame服务器

public void EnterRoom(String strClient0, StateObject obj1)

{

foreach (RoomObject room in roomList)

{

if (room.client0.sock.RemoteEndPoint.ToString() == strClient0)

{

room.EnterRoom(obj1);

room.client0.sock.Send(System.Text.Encoding.ASCII.GetBytes("beginGame 0 "));

room.client1.sock.Send(System.Text.Encoding.ASCII.GetBytes("beginGame 1 "));

}

}

}

## 5．beginGame客户端

if ("beginGame" == recvStrs[0])

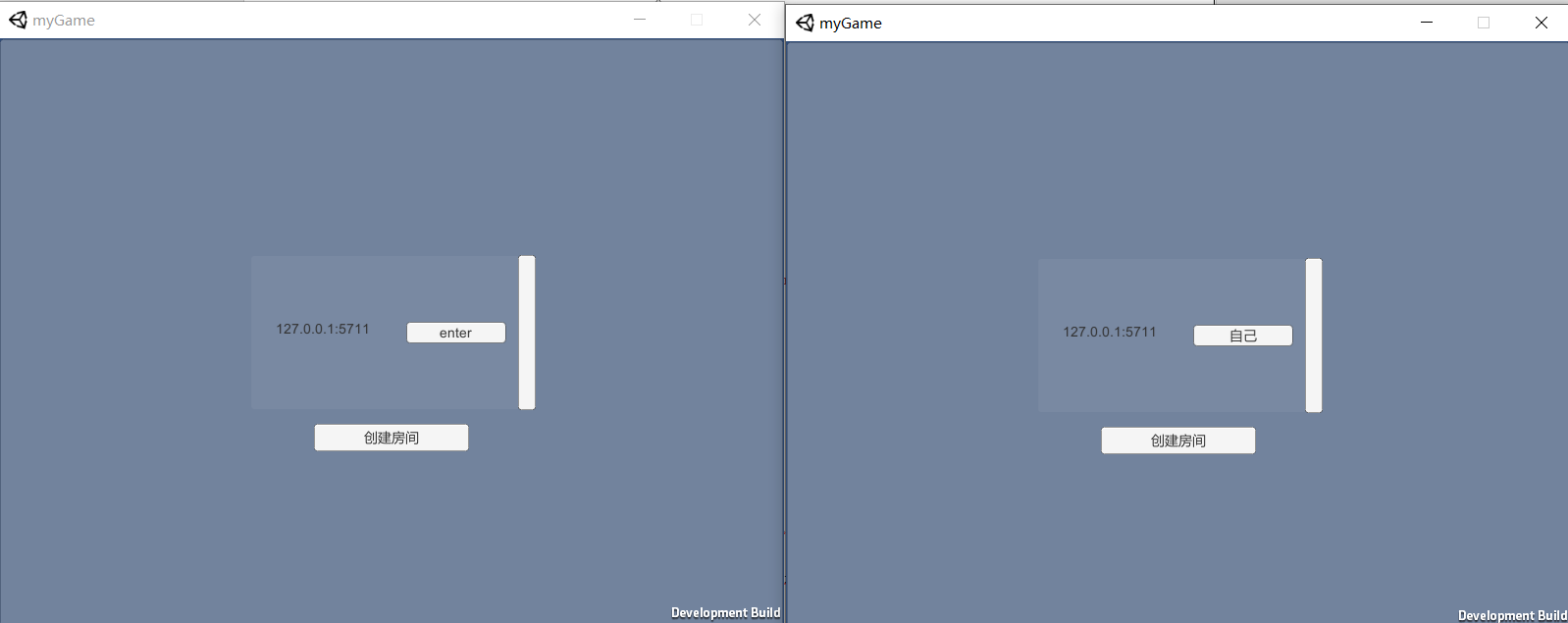
{

whoAmI = recvStrs[1];

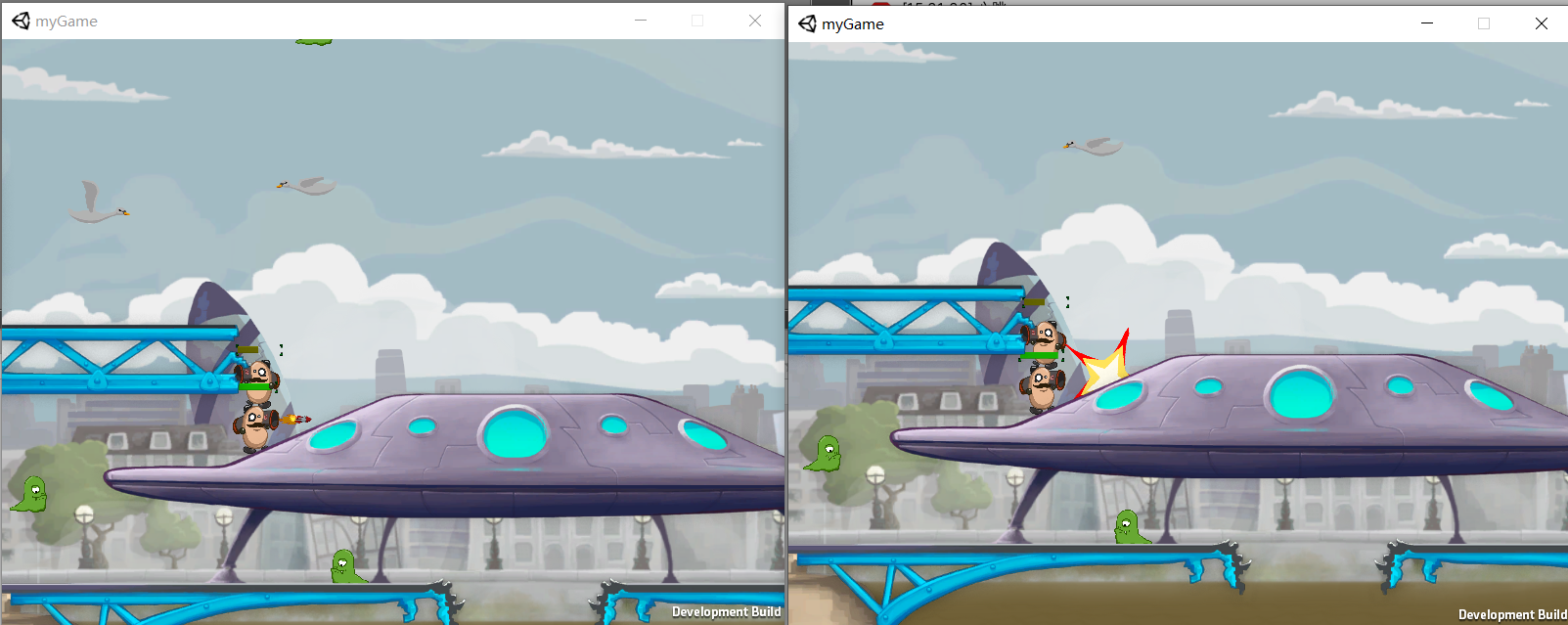
SceneManager.LoadScene("Level");

}

创建房间



进入房间，开始游戏（初始站位在两端，此截图为我移动到中间的截图）



# 同步系统

## 1同步位置和血量

#### 同步

if (recvStrs[0] == "Position")

{

//位置同步

float x = float.Parse(recvStrs[1]);

float y = float.Parse(recvStrs[2]);

Vector2 newPostion = new Vector2(x, y);

GameObject.Find("hero2").transform.position = newPostion;

float h = float.Parse(recvStrs[3]);

//血量同步

GameObject.Find("hero2").GetComponent<PlayerHealth2>().health = h;

GameObject.Find("hero2").GetComponent<PlayerHealth2>().UpdateHealthBar();

PlayerControl2 pl2 = GameObject.Find("hero2").GetComponent<PlayerControl2>();

if (h <= 0 && !pl2.isDie)

{

byte[] sendBuff = new byte[1024];

sendBuff = System.Text.Encoding.Default.GetBytes("Die ");

pl2.anim.SetTrigger("Die");

pl2.isDie = true;

}

else if(h > 0)

{

pl2.isDie = false;

}

}

#### 初始化位置

void setPosition(short woAmI)

{

//初始化玩家位置

hero1 = GameObject.Find("hero");

hero2 = GameObject.Find("hero2");

float x1, x2, y1, y2;

y1 = y2 = hero1.transform.position.y;

if (woAmI == 0)

{

x1 = 17;

x2 = -17;

}

else

{

x1 = -17;

x2 = 17;

}

Vector2 position1 = new Vector2(x1, y1);

Vector2 position2 = new Vector2(x2, y2);

hero1.transform.position = position1;

hero2.transform.position = position2;

}

#### 发送同步协议

if (add1PerFrame % 5 == 0)

{

string sendStr = "Position ";

if (hero1 == null)

{

hero1 = GameObject.Find("hero1");

}

if (hero1 != null)

{

sendStr += hero1.transform.position.x.ToString() + ' '; //!!!!!!!!!

sendStr += hero1.transform.position.y.ToString() + ' ';

//血量同步

float h = hero1.GetComponent<PlayerHealth>().health;

sendStr += h.ToString() + ' ';

sendBuff = System.Text.Encoding.UTF8.GetBytes(sendStr); //!!!!!!!!!!!!!

tempSocket.Send(sendBuff);

}

}

## 2.同步开火

#### 客户端发送协议

if(Input.GetButtonDown("Fire1"))

{

byte[] sendBuff = new byte[1024];

#### //客户端发送fire协议

#### sendBuff = System.Text.Encoding.Default.GetBytes("Fire ");

#### clientSocket.Send(sendBuff);

anim.SetTrigger("Shoot");

GetComponent<AudioSource>().Play();

if(playerCtrl.facingRight)

{

Rigidbody2D bulletInstance = Instantiate(rocket, transform.position, Quaternion.Euler(new Vector3(0,0,0))) as Rigidbody2D;

bulletInstance.velocity = new Vector2(speed, 0);

}

else

{

Rigidbody2D bulletInstance = Instantiate(rocket, transform.position, Quaternion.Euler(new Vector3(0,0,180f))) as Rigidbody2D;

bulletInstance.velocity = new Vector2(-speed, 0);

}

}

#### 服务器端接收

if ("Position" == args[0]||"Flip" == args[0]||"Fire"==args[0])

foreach (RoomObject room in roomList)

{

if (client == room.client0.sock)

room.client1.sock.Send(stateo1.buffer);

lse if (client == room.client1.sock)

room.client0.sock.Send(stateo1.buffer);

}

#### 客户端接收

//开火同步

if (recvStrs[0] == "Fire")

{

GameObject.Find("hero2/Gun").GetComponent<Gun2>().Fire();

}

### 同步转身

#### 客户端发送协议

void Flip ()

{

facingRight = !facingRight;

Vector3 theScale = transform.localScale;

theScale.x \*= -1;

transform.localScale = theScale;

#### byte[] sendBuff = new byte[1024];

#### sendBuff = System.Text.Encoding.Default.GetBytes("Flip ");

#### clientSocket.Send(sendBuff);

}

#### 服务器端接收

if ("Position" == args[0]||"Flip" == args[0]||"Fire"==args[0])

foreach (RoomObject room in roomList)

{

if (client == room.client0.sock)

room.client1.sock.Send(stateo1.buffer);

lse if (client == room.client1.sock)

room.client0.sock.Send(stateo1.buffer);

}

#### 客户端接收

//转身同步

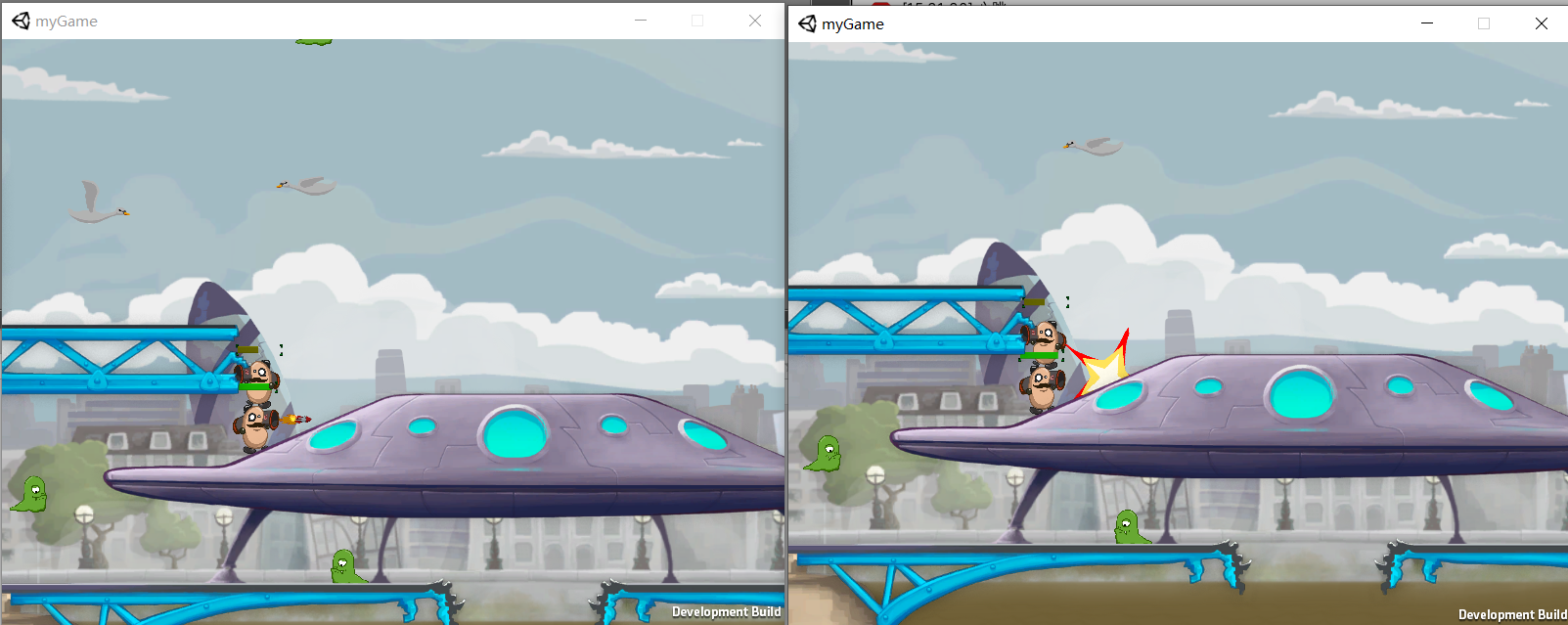
if (recvStrs[0] == "Flip")

{

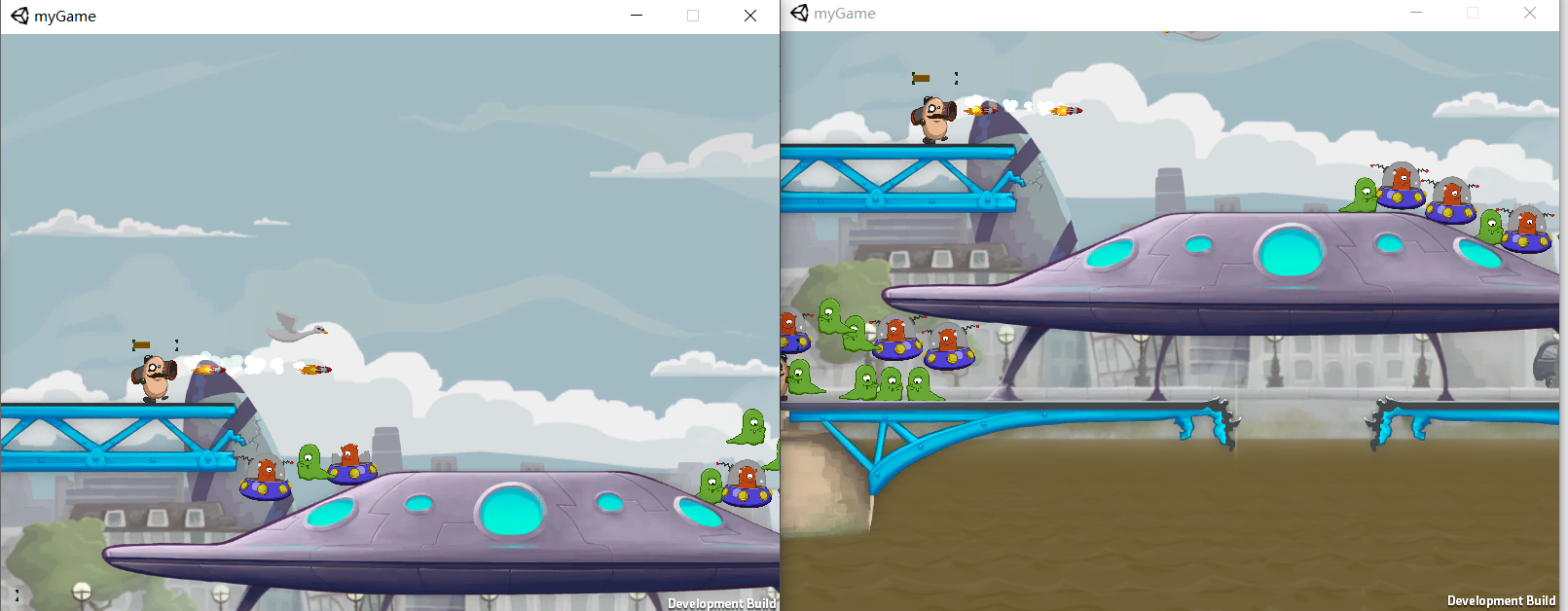
GameObject.Find("hero2").GetComponent<PlayerControl2>().Flip();

}

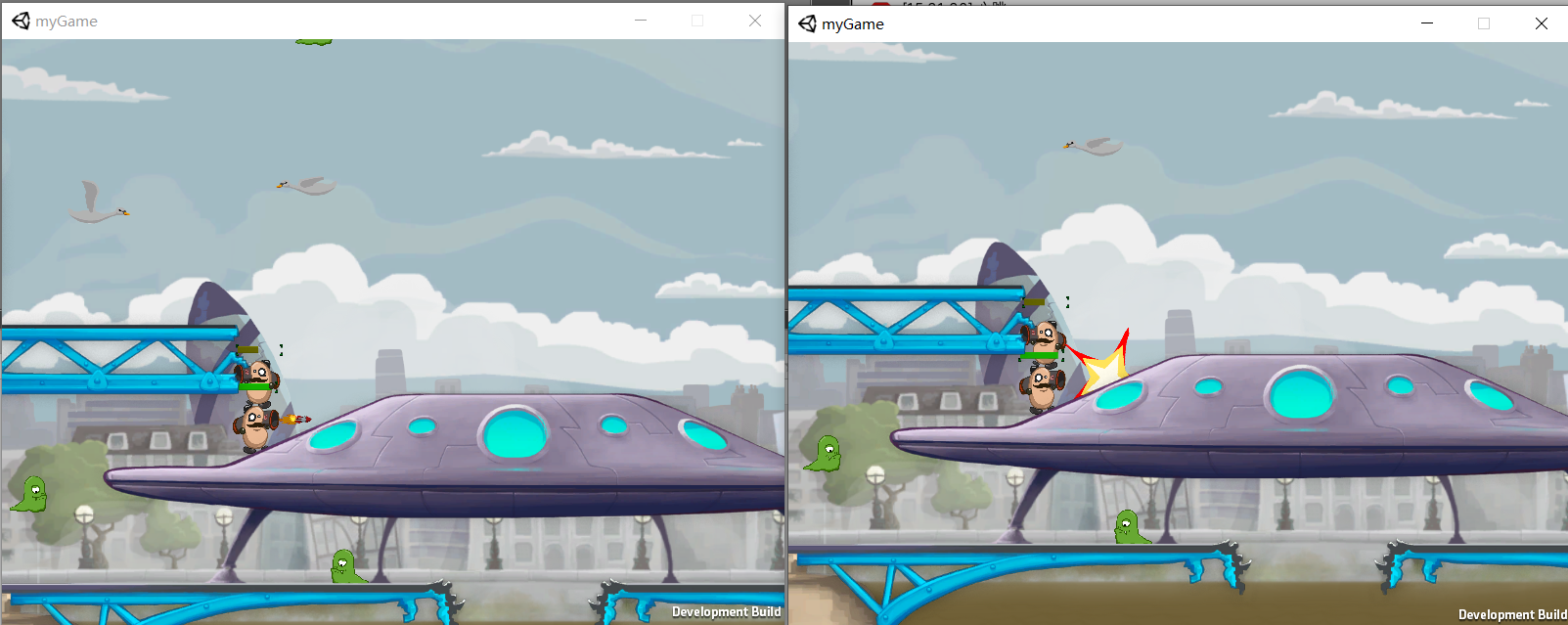
位置血条同步:



开火同步



转身同步



## 三【上机实验心得】

#### 遇到的主要问题与解决方案

做此次网络编程游戏，最大的问题是对网络编程不了解不熟悉。所以主要问题如下：

1. 实验难度大，pdf有些逻辑不是特别清晰，需要自己理清逻辑加查阅资料
2. 对网络协议不熟悉，服务器用的老师的服务器，所以需要先读懂服务器端代码和看懂协议层
3. 房间系统处遇到了数据收发中断的情况
4. 同步系统在同时开启两个房间时候会导致同步一段时间后出现数据中断的情况。
5. 服务器端对查询账号密码数据库的编写（登录服务器端），没有接触过写服务器代码，需要自行查询资料加理解

解决问题：

1. pdf不清楚的部分通过询问那些问过老师的同学和逻辑不懂的地方通过百度查询结合询问会unity的学长学姐
2. 通过反复阅读老师给的协议层和调试服务器代码，大致明白了服务器逻辑
3. Update中断，可能是unity引擎出了问题，重新安装unity再将项目重新打包到新项目，问题就解决了。
4. 未解决此问题，感觉是服务器端存在问题不能同时开启两个房间进行同步，碍于不会修改服务器，所以以一个房间作为实例来同步
5. 百度查询到了相关登录系统的服务器端代码，然后自己添加了对应1,2,3的值，就解决了。

#### 项目心得体会

此次项目，对于我来说整体难度较大，每周要花很多时间再它上面才能勉强完成作业，可能还是源于对网络编程的不熟悉以及计算机网络学的不扎实，但这次项目也让我收获了许多，比如说加深了对TCP的了解和明白了TCP协议的用处，以前只知道这个协议，但不知道具体怎么使用。再比如说，对于如何做游戏同步有了一定的思路，能够通过自主学习加查询资料独立完成一些简单的同步操作了。这次项目也让我对游戏性能优化有了一定的思考，比如制作游戏要考虑延迟问题，帧速问题等。虽然写代码没有提升多少能力，但是对于制作游戏的思想有了一定的提升，从一个一个遇到的问题中，让我在逐渐在思考问题的时候，相对会考虑的更加全面一些，但这些能力都还需要长时间的努力和实践才能真正锻炼出来，所以希望通过这次项目之后，有更多耐心去做这种有一定难度的东西。