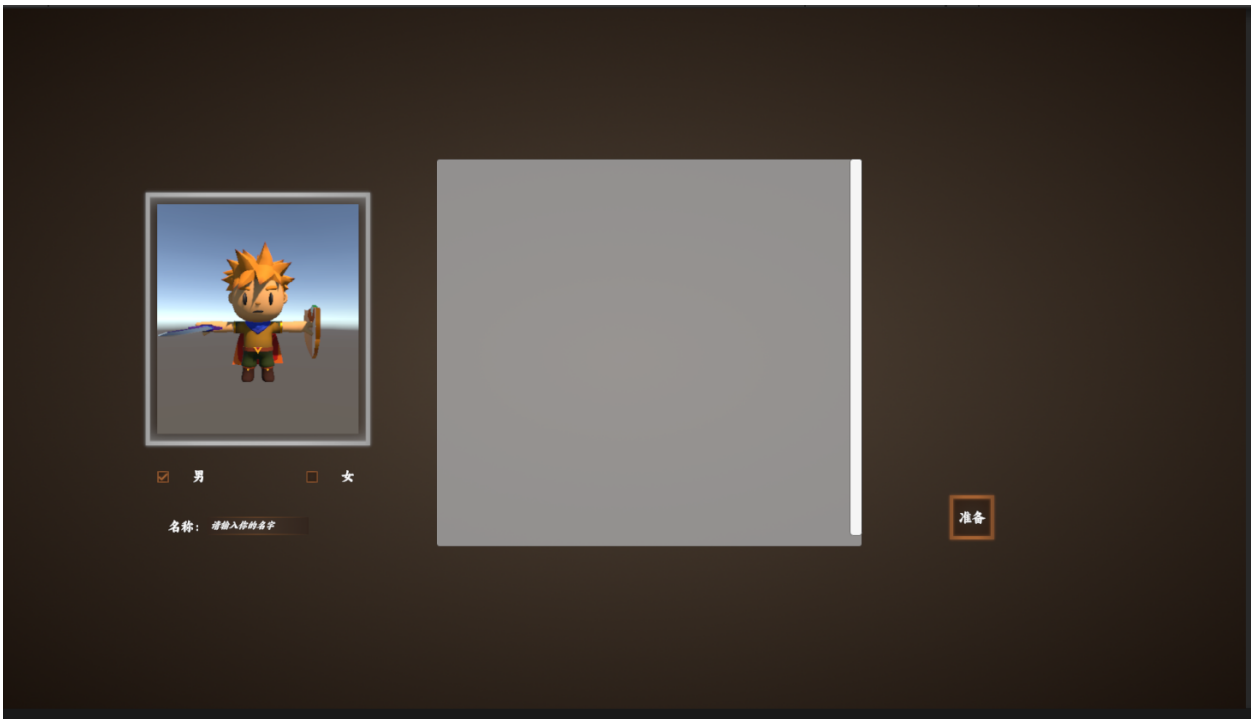
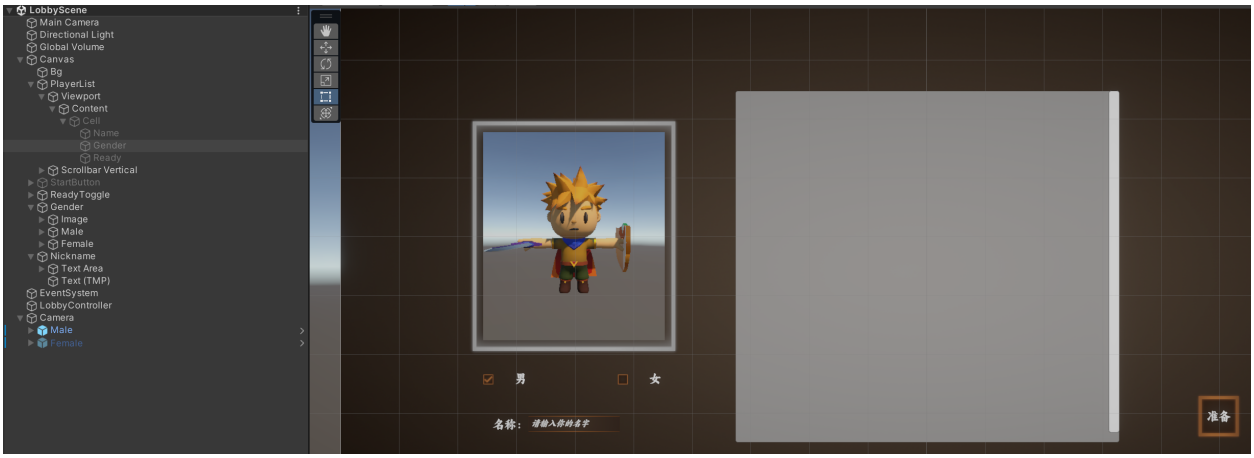
大厅场景

说明：大厅场景主要有玩家选择角色性别、设置角色名、准备按钮以及当前房间内玩家信息，其中主机玩家还有开始游戏按钮





核心脚本：

1. LobbyController.cs：控制大厅场景
2. PlayerSelectController.cs：用于控制玩家选择性别时切换模型
3. PlayerListCellController.cs：房间内玩家信息组件控制器
4. PlayerInfo.cs：玩家数据

大厅场景控制器 LobbyController.cs

重要属性：

```
[SerializeField] private Transform canvers;                                  // 大厅场景画布
private Transform content;                                                   // 中间显示房间玩家信息content
private GameObject cell;                                                     // 中间显示房间单条玩家信息
private Button startButton;                                                  // 开始游戏按钮
private Toggle readyToggle;                                                  // 准备按钮
private TMP_InputField nickname;                                             // 玩家名
private Dictionary<ulong, PlayerListCellController> playerListCellDict;  // 房间玩家信息各种组件
private Dictionary<ulong, PlayerInfoData> playerInfoDataDict;            // 房间玩家信息数据
```

OnNetworkSpawn

1. 查看当前是否是Server，如果是的话则向NetworkManager.OnClientConectedCallback中添加一个事件即<***新客户端连接方法***>，主要目的是其他客户端连接到房间内后需要执行一些操作，比如通知其他客户端更新玩家信息等
2. 初始化中间数据，例如playerInfoDataDict、playerListCellDict等
3. 获取当前场景中需要点击的组件并绑定事件
    a. content/cell：用于显示房间内玩家信息的UI
    b. startButton：开始按钮，从canvas中找到StartBotton游戏对象并获取Button组件，然后绑定<***开始按钮点击事件***>
    c. readyToggle：准备开关，从canvas中找到StartBotton游戏对象并获取Toggle组件，然后添加onValueChange事件<***准备开关点击事件***>
    d. nickName：玩家昵称，从canvas中找到NickName游戏对象并获取TMP_InputField组件，然后添加onEndEdit事件<***当玩家修改昵称方法***>
4. 将本地玩家添加到网络中<***添加一名玩家***>
5. 添加监听玩家改变性别的事件，即玩家点击男性女性后模型会对应改变
    a. maleToggle：选择男性开关，从canvas中找到Male游戏对象并获取Toggle组件，然后添加onValueChange事件<***当玩家选择男性方法***>

b. femaleToggle：选择女性开关，从canvas中找到Female游戏对象并获取Toggle组件，然后添加onValueChange事件<***当玩家选择女性方法***>

添加一名玩家

1. 实例化出一个房间内玩家信息cell并激活该组件，设置cell的parent为content
2. 从cell上得到`PlayerListCellController`组件并使用传入的playerInfo进行初始化设置UI内容
3. 将传入的玩家数据放入playerInfoDataDict进行缓存
4. 将生成的cell放入playerListCellDict进行缓存

新客户端连接方法

1. 根据传入的playerId创建一个playerInfoData
2. 根据创建数据<***添加一名玩家***>
3. <***服务端通知其他客户端更新玩家信息***>

服务端通知其他客户端更新玩家信息

1. 遍历当前playerInfoDataDict，对于每个玩家数据都调用<***客户端更新玩家信息***>，玩家信息更新后查看是否所有玩家都进入了准备状态
2. 如果所有玩家都进入了准备状态则在主机房间页面上显示开始游戏按钮，否则则不显示开始游戏按钮

客户端更新玩家信息 `UpdatePlayerInfoClientRpc`

1. 查看当前是否为服务端，如果是服务端则返回，避免出现重复添加角色的情况
2. 查看当前客户段中playerInfoDataDict中是否包含传入的新玩家数据，如果包含则更新数据否则<***添加一名玩家***>，将其添加到数据字典中并在大厅列表中显示对应玩家
3. <***更新大厅房间内所有玩家显示UI***>

服务端更新玩家信息 `UpdatePlayerInfoServerRpc`

注意：RequireOwnership=false

1. 更新当前机器上的playerInfoDataDict字典中的数据

2. 根据当前playerId更新大厅玩家信息<***更新玩家UI信息***>
3. <***服务端通知其他客户端更新玩家信息***>

触发服务端/客户端更新玩家信息

1. 如果当前是服务端则调用<***服务端通知其他客户端更新玩家信息***>
2. 如果当前是客户端则调用<***服务端更新玩家信息***>，相当于客户端告知服务端先更新本地数据然后再通知其他客户端更新玩家信息

更新玩家信息

1. 根据传入的playerId、玩家名、是否准备、性别来创建一个playerInfoData对象并将其存入字典中
2. 根据新创建的数据对象<***更新大厅房间内单个玩家显示UI***>

更新大厅房间内单个玩家显示UI

1. 调用PlayerListCellController中<***更新玩家UI信息***>

更新大厅房间内所有玩家显示UI

1. 遍历当前playerInfoDataDict，调用<***更新大厅房间内单个玩家显示UI***>方法更新所有玩家UI

准备开关点击事件

1. 使用`NetworkManager.LocalClientId`获取当前客户端的id
2. 设置当前玩家数据playerInfoData的isReady为传入bool值
3. <***更新玩家信息***>后<***触发服务端/客户端更新玩家信息***>

开始按钮点击事件

1. 调用GameManager<***开始游戏***>方法，主要是服务端将玩家数据同步给客户端
2. 加载游戏场景<***加载场景***>

当玩家选择男性方法

1. 使用`NetworkManager.LocalClientId`获取当前客户端的id
2. 如果当前传入玩家性别为男性，则需要调用PlayerSelectController<***切换性别***>方法将当前玩家切换成男性并<***更新玩家信息***>，如果传入玩家性别为女性也执行类似操作
3. <***触发服务端/客户端更新玩家信息***>

当玩家选择女性方法

1. 使用`NetworkManager.LocalClientId`获取当前客户端的id
2. 如果当前传入玩家性别为女性，则需要调用PlayerSelectController<***切换性别***>方法将当前玩家切换成女性并<***更新玩家信息***>，如果传入玩家性别为男性也执行类似操作
3. <***触发服务端/客户端更新玩家信息***>

当玩家修改昵称方法

1. 使用`NetworkManager.LocalClientId`获取当前客户端的id
2. 根据id获取对应playerInfoData，将传入的name设置为玩家名称<***更新玩家信息***>
3. <***触发服务端/客户端更新玩家信息***>

玩家模型切换控制器 PlayerSelectController.cs

说明：该脚本是一个简单单例脚本，没有复杂的成员

Start

1. 设置单例Instance

切换性别

1. 根据传入的玩家性别控制当前Camera组件下的Male、Female组件的显示

```
transform.GetChild((int)gender).gameObject.SetActive(true);
transform.GetChild(1 - (int)gender).gameObject.SetActive(false);
```

玩家数据 PlayerInfo.cs

说明：玩家数据继承自`INetworkSerializable`该接口为自定义可序列化类型结果，所有在Network中传递的数据均需要继承自该类

重要属性：

```
public ulong playerId;          // 玩家id
public string playerName;       // 玩家名: 默认是"玩家+id"
public bool isReady;            // 是否准备
public GENDER gender;           // 玩家性别
```

初始化方法

1. 根据传入的玩家id设置默认玩家数据，例如默认性别为男性、未准备、玩家名为”玩家+id”

自定义序列化方法 INetworkSerializable.NetworkSerialize

1. 调用`BufferSerializer.SerializeValue`序列化玩家id、姓名、性别、准备状态，注意需要使用`ref`传入他们的引用

房间内玩家信息组件控制器 PlayerListCellController.cs

注意：当前控制器是挂载到PlayerList/Viewport/Content/Cell组件上的

重要属性：

```
private TMP_Text playerName;      // 控制玩家姓名UI
private TMP_Text ready;           // 控制玩家状态UI
private TMP_Text gender;          // 控制玩家性别UI
```

初始化方法

1. 从当前Cell下获得Name、Ready、Gender游戏对象并得到他们的TMP_Text组件
2. 根据传入的playerInfoData<***设置玩家名***>、<***设置玩家准备状态***>、<***设置玩家性别***>

更新玩家UI信息

1. 根据传入的playerInfoData<***设置玩家名***>、<***设置玩家准备状态***>、<***设置玩家性别***>

设置玩家名

1. 根据传入的玩家名设置playerName.text

设置玩家准备状态

1. 根据传入的准备状态设置ready.text

设置玩家性别

1. 根据传入的玩家性别设置gender.text