Multiplayer Networking mit Unity5

Marcel Mayer

18.04.2016

Inhalt

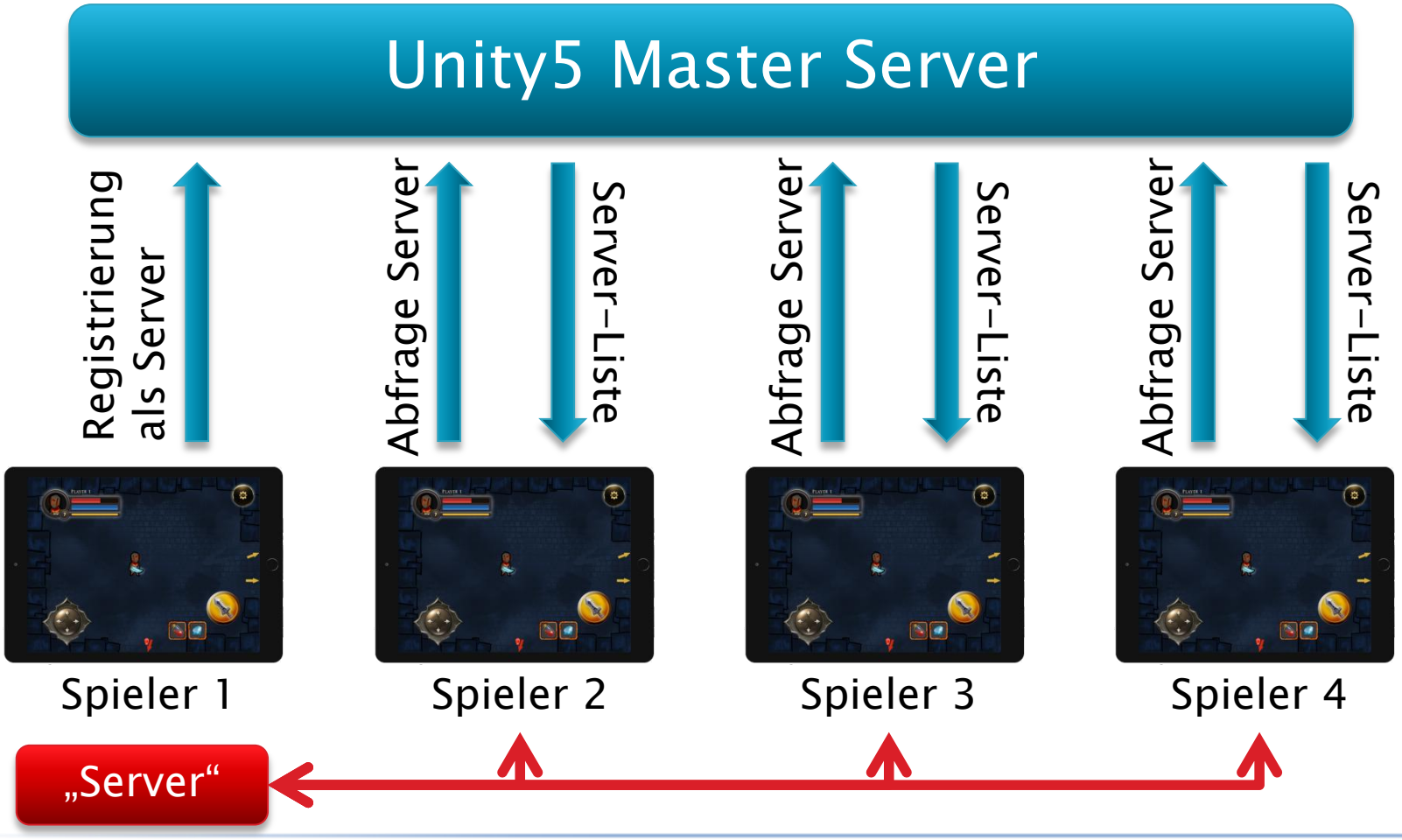
- Was kann Unity5 Networking?
- Unity5 Master Server
- Wichtige Components für das Networking
- Klassen netzwerkfähig machen
- Live-Demo

Was kann Unity5 Networking?

- Matchmaking
  - Wie finden die Spieler zueinander?
  - Wer ist der „Server“?
- Kommunikation zwischen den Spielern während des Spiels
  - Jeder darf nur seinen eigenen Spieler bewegen
  - Übertragung der Bewegung auf die anderen Clients

Unity5 Master Server

- Verwaltet die Spiele und vermittelt die Spieler

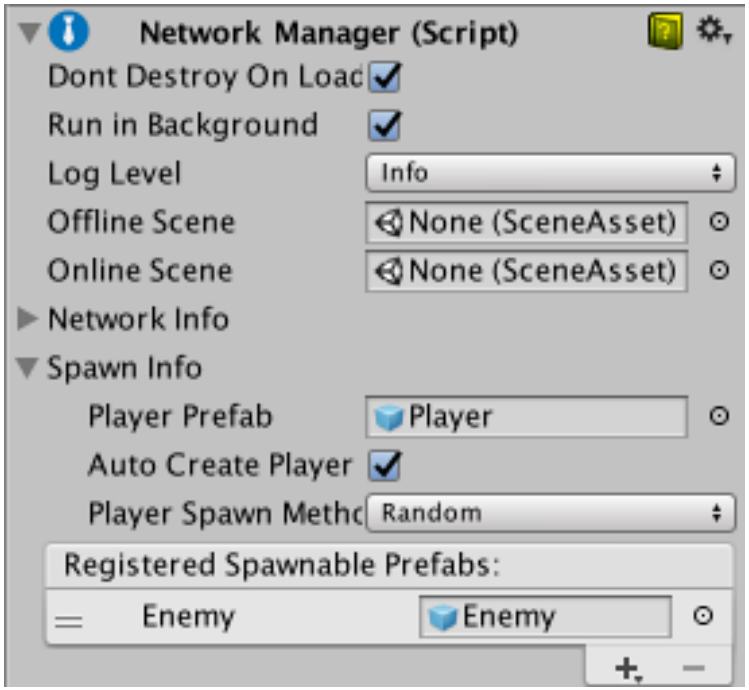


Wichtige Components

- Network Manager
- Network Manager HUD
- Network Identity
- Network Transform

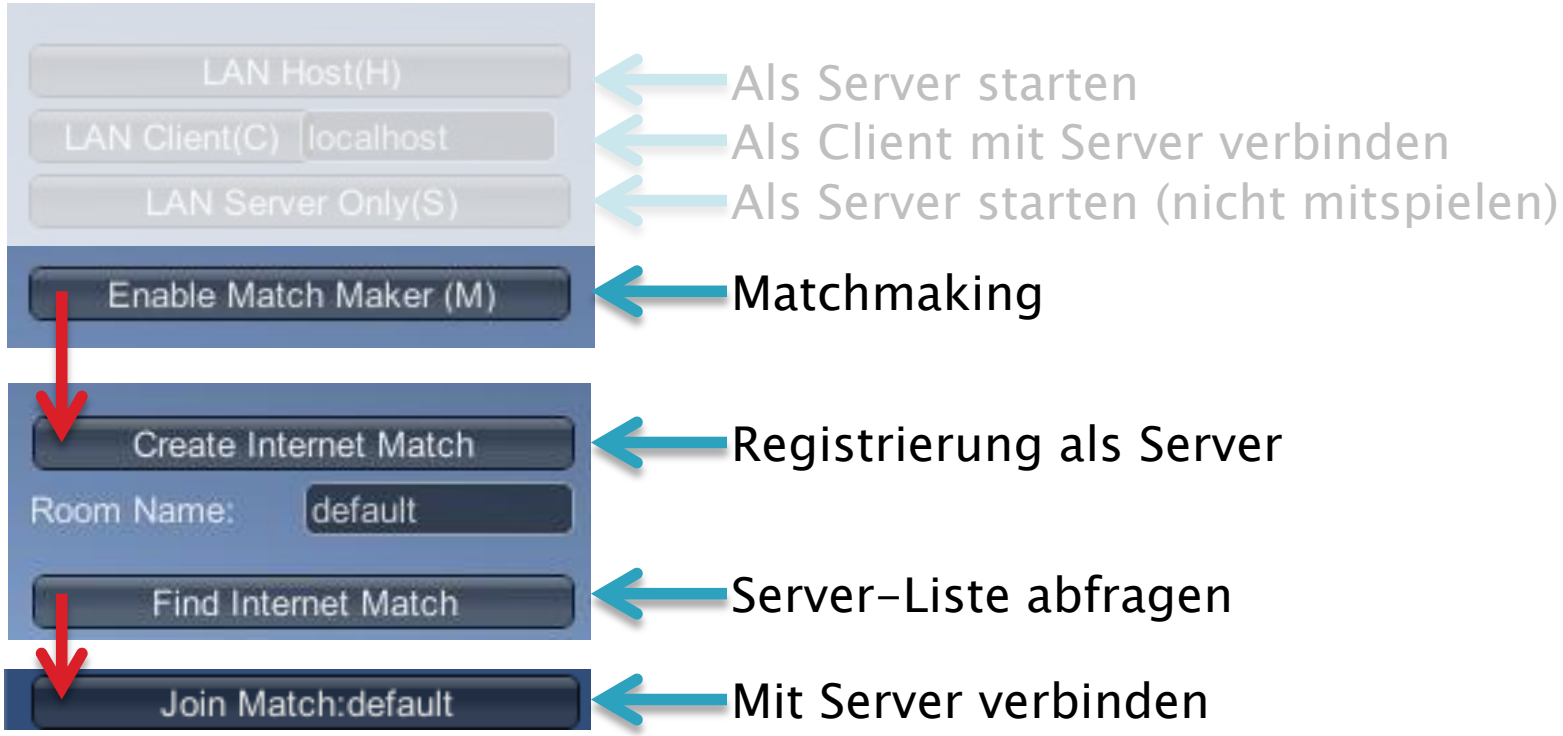
Network Manager

- Kümmert sich um Networking
- Spawnen der Spieler und weiterer Game Objects
- Verwalten der Scenes



Network Manager HUD

- Einfaches Interface für das Matchmaking



- Kann durch eigenes Interface ersetzt werden

Network Manager HUD

- Einfaches Interface für das Matchmaking


```
NetworkManager.singelton.StartHost();
NetworkManager.singelton.StartClient();
NetworkManager.singelton.StartServer();
NetworkManager.singelton.StartMatchmaker();
NetworkMatch.CreateMatch();
NetworkMatch.ListMatches();
NetworkMatch.JoinMatch();
```

- Kann durch eigenes Interface ersetzt werden

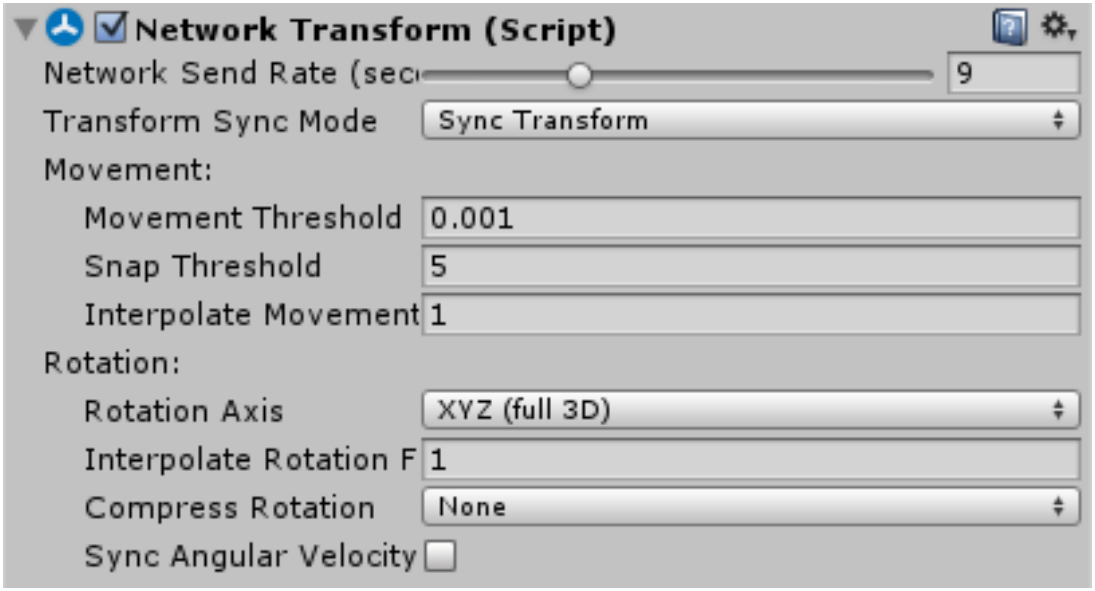
Network Identity

- Macht das Game Object im Netzwerk bekannt

- Server Only:
  Game Object nur auf dem Server erzeugen
- Local Player Authority:
  Jeder Client hat seine eigene Instanz und verwaltet diese selbst

Network Transform

- Synchronisiert die Bewegungen eines Game Objects



- Network Send Rate:
  Wie oft pro Sekunde sollen Informationen übertragen werden

Klasse netzwerkfähig machen

- `using UnityEngine.Networking;`
- `public class PlayerCtrl : NetworkBehaviour`

  statt

  `public class PlayerCtrl : MonoBehaviour`

Wichtige Eigenschaften und Methoden

- Eigenschaft `isLocalPlayer`
  `true`, wenn das Game Object von diesem Client verwaltet wird
- Eigenschaft `isServer`
  `true`, wenn Code auf dem Server läuft
- `public override void OnStartClient() { }`
  Wird aufgerufen, sobald der Client verbunden ist
- `public override void OnStartServer() { }`
  Wird aufgerufen, sobald der Server registriert wurde

Methode auf Server bzw. Client starten

- Client ruft Methode auf,
  die auf dem Server laufen soll

```
[Command]
void CmdFire() {   }
```

- Server ruft Methode auf,
  die auf allen Clients laufen soll

```
[ClientRpc]
void RpcRespawn() {   }
```

Variablen synchronisieren & Spawning

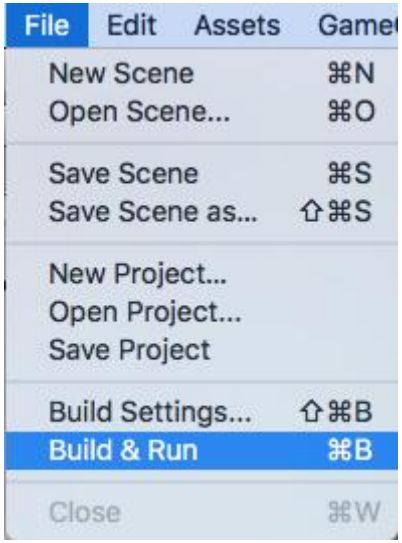
- Zustand einer Variablen soll vom Server auf alle Clients synchronisiert werden

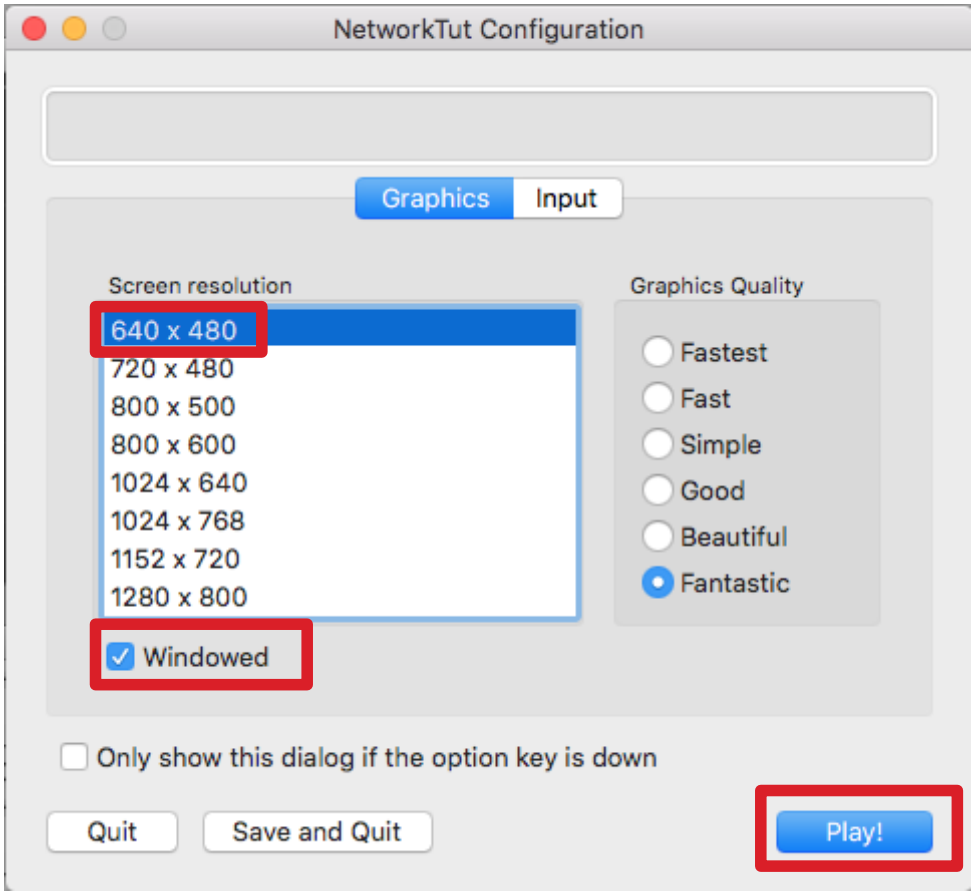
```
[SyncVar]
public int health = 100;
```

- Game Object soll vom Server auf alle Clients gespawned werden

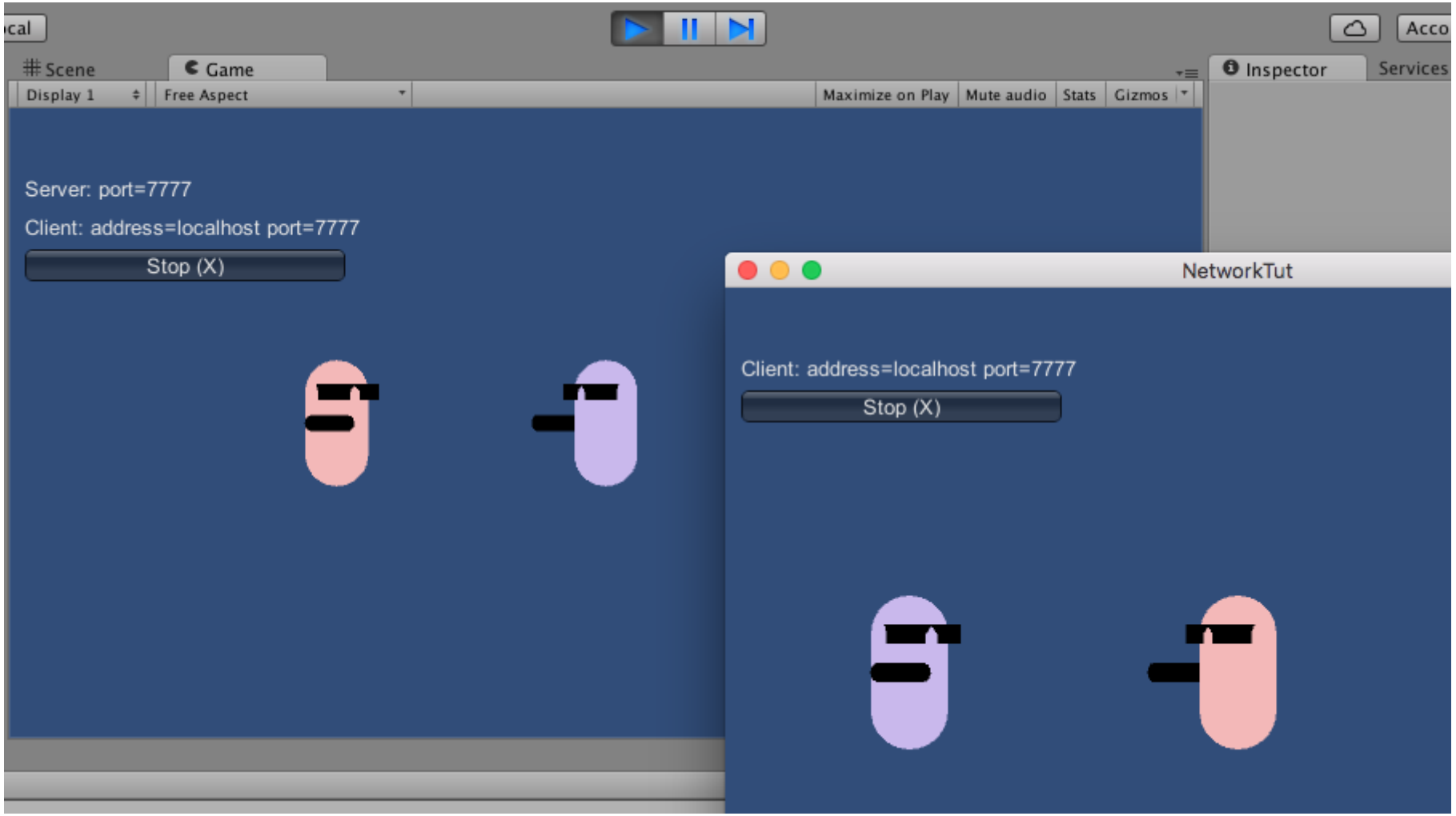
```
var bullet = (GameObject)Instantiate(
                bulletPrefab,
                bulletSpawn.position,
                bulletSpawn.rotation);

NetworkServer.Spawn(bullet);
```

Testen mit 2 Spiel-Instanzen





Testen mit 2 Spiel-Instanzen



Nützliche Links

- Unity5 Multiplayer Networking Tutorial: https://unity3d.com/learn/tutorials/topics/multiplayer-networking
- Unity5 Script Reference: http://docs.unity3d.com/ScriptReference/Networking.NetworkManager.html

  http://docs.unity3d.com/ScriptReference/Networking.Match.NetworkMatch.html

Live-Demo