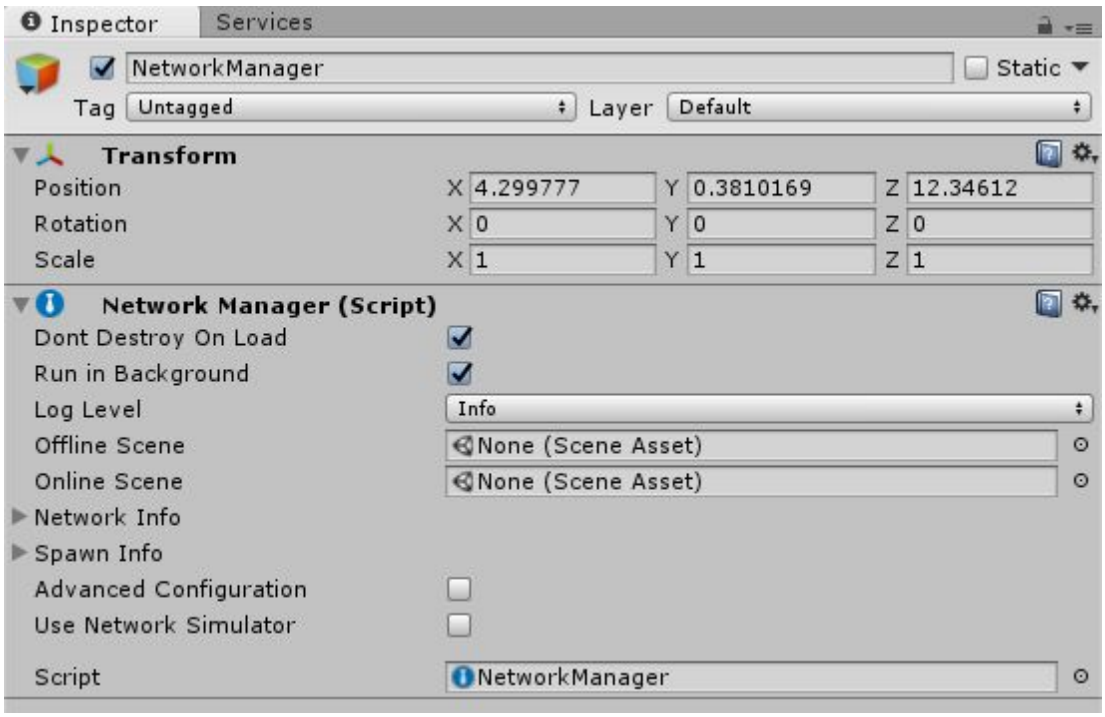
Documentation réseau pour Unity

Création de la partie réseau en partant de zéro :

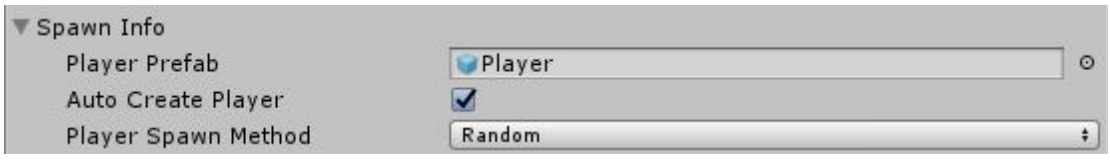
1- Ajoutez un GameObject vide et insérez le composant “NetworkManager”. Ce composant permet de configurer les paramètres du réseau et permet son lancement et son arrêt.



https://docs.unity3d.com/ScriptReference/Networking.NetworkManager.html

2 - Ajout du composant “NetworkIdentity” et “NetworkTransform” pour le joueur. Tous les objets pouvant être générés dans le jeu doivent avoir un composant “NetworkIdentity” pour établir son identité et “NetworkTransform” pour la synchronisation avec le serveur.

3 - Mettre à jour le PlayerPrefab dans le “NetworkManager” en ajoutant le prefab du joueur. Permet de gérer le spawn du joueur.



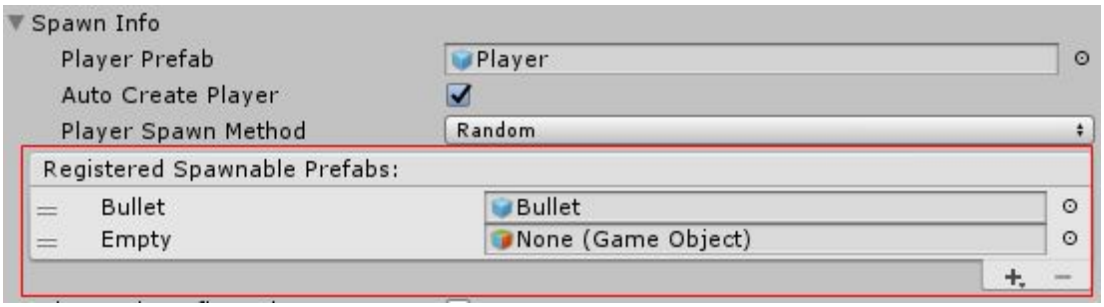
4 - Modification des scripts liés au joueur

- Dans la déclaration des classes, changez “MonoBehaviour” en “NetworkBehaviour”
- Ajoutez dans la fonction update

  if (!isLocalPlayer)

  return;

  Pour indiquer que seulement le joueur local peut effectuer l’action en question
- Cochez la case “LocalPlayerAuthority” dans le NetworkIdentity du prefab joueur pour lui donner les droits de pouvoir avoir le contrôle de son personnage.
- Ajoutez le composant “NetworkTransform” dans le prefab du joueur pour activer la synchronisation des mouvements. (Si pas déjà fait dans l’étape 2)
- Ajoutez les objets pouvant être générés dans le “NetworkManager” dans l’onglet “RegisteredSpawnablePrefabs”



- Ajoutez [Command] avant la fonction du tir pour que l’action s'exécute sur le serveur. (Il faut ajouter Cmd au début du nom de la fonction)

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

https://docs.unity3d.com/ScriptReference/Networking.CommandAttribute.html

Et ajout de la commande permettant de faire spawn la balle/sort.

NetworkServer.Spawn(bullet);

- Pour synchroniser des valeurs comme la vie du joueur ajoutez [SyncVar] avant la déclaration de la variable.

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

https://docs.unity3d.com/Manual/UNetStateSync.html

- Synchronisation de l'affichage sur tous les clients
  Pour synchroniser le déplacement du joueur / mouvement de la caméra / les changements d'armes, il faut tout d'abord que le Client lance sa fonction sur le serveur avec [Command] avant la fonction.
  (*Synchronisation entre Client local <-> Serveur*)
  A l'intérieur de cette fonction lancer une fonction Rpc. La déclaration de la fonction Rpc est similaire à la fonction Cmd avec [ClientRpc] avant la fonction et qui effectue la même action.
  (*Synchronisation entre Serveur <-> tous les Clients*)

  Exemple avec le mouvement de la caméra

```
/**
 * UpdateCameraXY
 * Update the X and Y axis input
 * @param X_val : X axis input
 * @param Y_val : Y axis input
 **/
[Command]
public void CmdUpdateCameraXY (float X_val, float Y_val) {
    _XInput = X_val;
    _YInput = Y_val;
    RpcUpdateCameraXY (X_val, Y_val);
}

[ClientRpc]
void RpcUpdateCameraXY (float X_val, float Y_val) {
    _XInput = X_val;
    _YInput = Y_val;
}
```

  https://docs.unity3d.com/Manual/UNetActions.html