**Knowledge Transfer Document: VR Multiplayer Experience**

**Purpose**

This document provides an overview of the VR Multiplayer Experience codebase, setup instructions, and dependencies to enable a new developer to understand, configure, and contribute to the project. It covers the project’s architecture, key components, and troubleshooting tips, ensuring a smooth onboarding process.

**Project Overview**

The VR Multiplayer Experience is a Unity-based virtual reality game built with Normcore for real-time multiplayer networking and XR Interaction Toolkit for VR interactions. Players can customize their avatar (name and color), create or join rooms, switch between four scenes (CommonRoom, Showroom, Clinic, Nature), and interact with networked objects (e.g., spheres, cubes). The project uses XR Device Simulator for testing without VR hardware, making it accessible for development on non-VR setups.

**Key Features**

* **Pre-Game Customization**: UI for setting player name, avatar color, and room name.
* **Multiplayer Rooms**: Real-time avatar synchronization using Normcore.
* **Scene Switching**: Room creator can switch scenes, with all clients syncing.
* **Networked Objects**: Grabbable objects synchronized across clients.
* **VR Support**: Compatible with XR Device Simulator and VR headsets (e.g., Oculus Quest via OpenXR).
* **Avatar Management**: Single avatar per player, linked to XR Rig, with cleanup on scene switch.

**Codebase Overview**

The codebase is organized under Assets/ with a focus on modularity and multiplayer synchronization. Below are the key components and their roles.

**Directory Structure**

MyProject/

├── Assets/

│ ├── Scenes/Flow

│ │ ├── 0\_PreGame.unity # Customization UI

│ │ ├── 1\_CommonRoom.unity # Main multiplayer room

│ │ ├── 2\_Showroom.unity # Showroom environment

│ │ ├── 3\_Clinic.unity # Clinic environment

│ │ └── 4\_Nature.unity # Nature environment

│ ├── Scripts/

│ │ ├── PreGameCustomization.cs # Handles pre-game UI

│ │ ├── CustomizationData.cs # Stores player data

│ │ ├── NetworkedAvatar.cs # Manages avatar sync and XR Riglinking

│ │ ├── RoomManager.cs # Controls room, scene switching,obj

│ │ ├── SceneSwitchPanel.cs # Creator-only scene-switch UI

│ │ └── SyncGrab.cs # Networked object grabbing

│ ├── Prefabs/

│ │ ├── AvatarPrefab.prefab # Player avatar

│ │ ├── SpherePrefab.prefab # Networked sphere

│ │ └── CubePrefab.prefab # Networked cube

│ ├── Art/ # Models, textures, sprites

│ ├── Audio/ # Sound effects & music

│ ├── Materials/

│ │ ├── Red.mat # Red avatar material

│ │ ├── Blue.mat # Blue avatar material

│ │ ├── Green.mat # Green avatar material

│ │ ├── Yellow.mat # Yellow avatar material

│ │ └── Cyan.mat # Cyan avatar material

│ └── Normcore/ # Normcore plugin files

├── Builds/ # Compiled builds

├── Demo.mp4 # Demo video

├── RND.docx # R&D notes

├── KnowledgeTransfer.docx # This document

├── Packages/ # Unity package dependencies

├── ProjectSettings/ # Unity configurations

├── .gitignore # Git ignore rules

└── README.md # Project overview

**Key Scripts**

* **PreGameCustomization.cs**:
  + Location: Assets/Scripts/PreGameCustomization.cs
  + Purpose: Manages the pre-game UI (name input, color dropdown, room input, create/join buttons).
  + Key Methods: CreateRoom(), JoinRoom().
  + Dependencies: TextMeshPro, CustomizationData.
* **CustomizationData.cs**:
  + Location: Assets/Scripts/CustomizationData.cs
  + Purpose: Stores player data (name, color index, room name) via PlayerPrefs.
  + Usage: Passed between scenes for avatar setup.
* **NetworkedAvatar.cs**:
  + Location: Assets/Scripts/NetworkedAvatar.cs
  + Purpose: Synchronizes avatar position, name, and color; links avatar to XR Rig.
  + Key Methods: LinkXRRig() (aligns avatar head to Main Camera).
  + Dependencies: Normcore (RealtimeView, RealtimeTransform), TextMeshPro.
* **RoomManager.cs**:
  + Location: Assets/Scripts/RoomManager.cs
  + Purpose: Manages Normcore room connection, scene switching, object spawning, and avatar cleanup.
  + Key Methods: SwitchScene(), CleanupLocalAvatar(), CheckAvatarSpawn().
  + Dependencies: Normcore (Realtime, RealtimeAvatarManager), TextMeshPro.
* **SceneSwitchPanel.cs**:
  + Location: Assets/Scripts/SceneSwitchPanel.cs
  + Purpose: Provides UI for the room creator to switch scenes.
  + Dependencies: RoomManager, TextMeshPro.
* **SyncGrab.cs**:
  + Location: Assets/Scripts/SyncGrab.cs
  + Purpose: Synchronizes networked object grabbing across clients.
  + Dependencies: Normcore, XR Interaction Toolkit (XRGrabInteractable).

**Scene Structure**

* **0\_PreGame**:
  + Contains: Customization UI (Canvas with TMP\_InputField, TMP\_Dropdown, Buttons), PreGameManager (PreGameCustomization), XR Setup.
  + Purpose: Player configures name, color, and room before joining.
* **1\_CommonRoom, 2\_Showroom, 3\_Clinic, 4\_Nature**:
  + Contains: XR Setup, AvatarManager (RealtimeAvatarManager), RoomManager, SceneSwitchCanvas (SceneSwitchPanel), table for objects, scene-specific assets.
  + Purpose: Multiplayer environments with synced avatars and objects.

**XR Setup**

* Hierarchy:
* XR Setup
* |- XR Device Simulator
* |- XR Origin (XR Rig) [Tag: XRRig]
* └── Camera Offset
* ├── Main Camera
* ├── Left Controller
* └── Right Controller
* Purpose: Provides VR input via XR Device Simulator or VR headset. NetworkedAvatar links to Main Camera for head tracking.

**Dependencies**

* **Unity**: Version 2021.3 LTS or later.
* **Packages**:
  + **XR Interaction Toolkit**: For VR interactions (grabbing, UI).
  + **TextMeshPro**: For UI text (name tags, popups).
  + **Normcore**: For multiplayer networking (avatars, objects).
* **External**:
  + Normcore Account: Sign up at [normcore.io](https://normcore.io/) for an App Key.
  + Internet: Required for Normcore connectivity.
* **Optional Hardware**:
  + VR Headset: Oculus Quest or OpenXR-compatible (not required; XR Device Simulator suffices).

**Setup Steps**

1. **Clone the Repository**
2. **Open in Unity**:
   * In Unity Hub, click **Add**, select MyProject.
   * Use Unity 2021.3 LTS or later.
3. **Install Packages**:
   * Go to **Window > Package Manager**.
   * Install:
     + XR Interaction Toolkit.
     + TextMeshPro.
   * Import Normcore:
     + Download from Unity Asset Store or normcore.io.
     + Import to Assets/Normcore.
4. **Configure XR**:
   * Go to **Edit > Project Settings > XR Plug-in Management**.
   * Enable **OpenXR**.
   * Enable **XR Device Simulator** in OpenXR Features.
5. **Set Normcore App Key**:
   * Open Assets/NormcoreAppSettings.asset.
   * Paste your Normcore App Key (from normcore.io dashboard).
6. **Configure Build Settings**:
   * Go to **File > Build Settings**.
   * Add scenes:
     + Assets/Scenes/0\_PreGame.unity
     + Assets/Scenes/1\_CommonRoom.unity
     + Assets/Scenes/2\_Showroom.unity
     + Assets/Scenes/3\_Clinic.unity
     + Assets/Scenes/4\_Nature.unity
7. **Verify Scene Setup**:
   * **0\_PreGame**:
     + Canvas (World Space, scale ~0.001) with TMP\_InputField (name, room), TMP\_Dropdown (color), Buttons (Create, Join).
     + PreGameManager with PreGameCustomization, linked to UI and 5 materials (Assets/Materials/Red.mat, etc.).
     + XR Setup with XR Origin tagged “XRRig”.
   * **1\_CommonRoom, 2\_Showroom, 3\_Clinic, 4\_Nature**:
     + XR Setup (XR Origin tagged “XRRig”).
     + AvatarManager with RealtimeAvatarManager, linked to Assets/Prefabs/AvatarPrefab.
     + RoomManager (root) with RoomManager, linked to NormcoreAppSettings, PopupCanvas, PopText, object prefab names (“SpherePrefab”, “CubePrefab”).
     + SceneSwitchCanvas with SceneSwitchPanel, linked to four buttons.
     + Table at (0, 1, 0).
8. **Prefab Setup**:
   * **AvatarPrefab** (Assets/Prefabs/AvatarPrefab):
     + Components: RealtimeView, RealtimeTransform, NetworkedAvatar.
     + Model: Capsule (body, (0, 0.75, 0)), Sphere (head, (0, 1.5, 0)).
     + Head GameObject, NameTagCanvas (TMP\_Text at (0, 1.7, 0)).
     + NetworkedAvatar: Linked to TMP\_Text, MeshRenderers, Head, 5 materials.
   * **SpherePrefab, CubePrefab**:
     + Components: RealtimeView, RealtimeTransform, Rigidbody (non-kinematic), XRGrabInteractable, SyncGrab.
9. **Tags**:
   * Create “XRRig” tag (Edit > Project Settings > Tags and Layers).
   * Assign to XR Origin in all scenes.

**Running the Project**

1. **Start PreGame**:
   * Open Assets/Scenes/0\_PreGame.unity.
   * Click Play.
   * Enter name, color, room name (e.g., “TestRoom”).
   * Click Create (creator) or Join.
2. **Interact in CommonRoom**:
   * Use WASD (move), Mouse (look), Q/E (rotate), Left Click (grab/UI) via XR Device Simulator.
   * Creator: Click SceneSwitchPanel buttons to switch scenes.
3. **Test Multiplayer**:
   * Build project (File > Build and Run).
   * Editor: Create “TestRoom”
   * Build: Join “TestRoom”
   * Verify avatar sync, object grabbing, scene switching.

**Troubleshooting**

* **“Local avatar not spawned”**:
  + Check Normcore App Key.
  + Ensure AvatarManager links to AvatarPrefab.
  + Verify internet connection.
* **Duplicate Avatars**:
  + Confirm RoomManager is root and uses singleton pattern.
  + Check Console for “Destroying local avatar before scene switch.”
* **XR Rig Issues**:
  + Verify “XRRig” tag on XR Origin.
  + Ensure XR Device Simulator is enabled.
* **Object Grabbing Fails**:
  + Confirm XRGrabInteractable on prefabs, Interaction Layer Mask set to Default.
* **Connection Errors**:
  + Check Normcore dashboard for server status.
  + Try a different room name.
* **Debugging**:
  + Add logs in RoomManager.CheckAvatarSpawn or NetworkedAvatar.LinkXRRig.
  + Monitor Console for “Avatar linked to XR Rig at…”.

**Maintenance Tips**

* **Normcore Updates**: Check normcore.io for plugin updates, as networking changes may affect sync.
* **XR Compatibility**: Test with new OpenXR versions or VR headsets.
* **Performance**: Optimize avatar models (currently capsule/sphere) for larger rooms.
* **Extending Scenes**: Add new scenes by duplicating 1\_CommonRoom, updating Build Settings, and ensuring same setup (AvatarManager, RoomManager, XR Setup).

**Resources**

* **Normcore Docs**: [normcore.io/documentation](https://normcore.io/documentation)
* **XR Interaction Toolkit**: [docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit](https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit)
* **Unity Manual**: [docs.unity3d.com/Manual](https://docs.unity3d.com/Manual)

**Contact**

For questions, contact [sivaraghavi6103@gmail.com](mailto:sivaraghavi6103@gmail.com)