

Unity-Multiplayer

Kurzbeschreibung

In diesem Dokument werden die einzelnen Komponenten des Projekts *Unity-Multiplayer* beschrieben. Bei dem Projekt handelt es sich um in VR-Projekt, in welchem man geplante Bauwerke gemeinsam in einer virtuellen Welt besichtigen kann. Dabei ist es möglich über die Anwendung zu kommunizieren und mit anderen Nutzern in der Welt zu interagieren. Die geplanten Bauwerke können mit Hilfe eines Web-Interfaces in die Welt eingefügt werden.

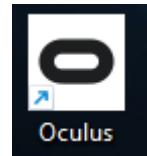
Unity-Multiplayer.....	1
Kurzbeschreibung.....	1
Quickstart guide.....	2
Webinterface.....	3
Sequenzdiagramme.....	4
Clients Interaktion	6
Modell Platzierung	6
Dynamische Sonne.....	7

Quickstart Guide

Um das Projekt in VR begehen zu können, wird SteamVR auf dem auszuführenden Computer benötigt. Dies ermöglicht die Verwendung von einer größeren Anzahl an VR-Headsets.

Nach Installation des Steam-Client (hier: <https://store.steampowered.com/about/>) und Anmeldung mit einem Account kann SteamVR über den Steam-Store heruntergeladen und installiert werden (hier: <https://store.steampowered.com/app/250820/SteamVR/>). Danach wird der Account nicht mehr benötigt. Der SteamVR Server ist nun im Hintergrund aktiv und startet SteamVR wenn eine VR-Anwendung gestartet wird. Dabei ist keine weitere Anmeldung notwendig.

Bei Verwendung eines Headsets eines anderen Herstellers als VALVE, muss zunächst die VR-Software des Herstellers gestartet werden. Im Falle von Oculus der Oculus-Client.



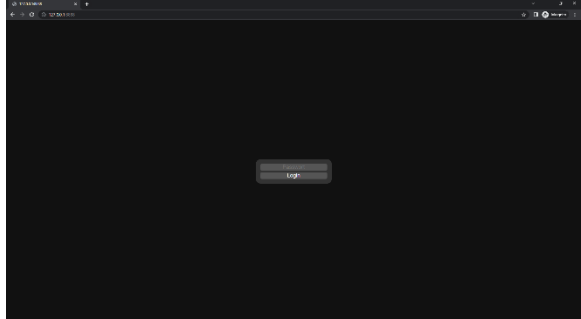
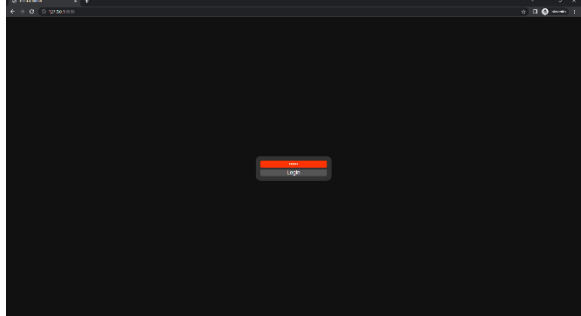
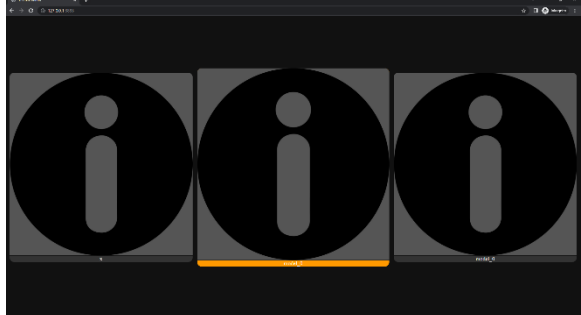
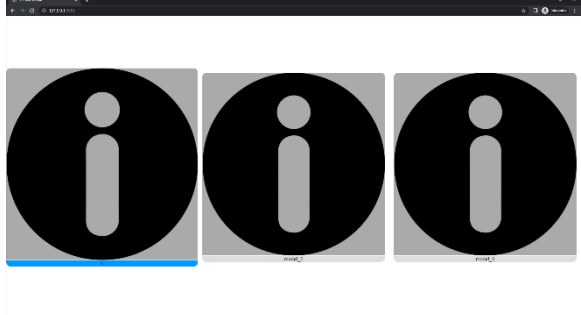
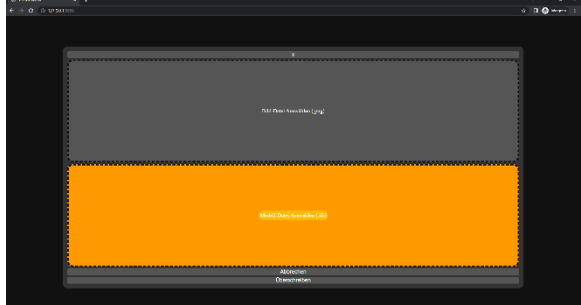
Nun kann die Software durch Ausführen der Datei *Multiplayer.exe* gestartet werden.

Name	Änderungsdatum	Typ	Größe
MonoBleedingEdge	22.11.2022 11:53	Dateiordner	
Multiplayer_BurstDebugInformation_Do...	22.11.2022 11:53	Dateiordner	
Multiplayer_Data	22.11.2022 11:53	Dateiordner	
Multiplayer.exe	16.03.2022 03:18	Anwendung	639 KB
UnityCrashHandler64.exe	16.03.2022 03:25	Anwendung	1.205 KB
UnityPlayer.dll	16.03.2022 03:25	Anwendungserwe...	27.612 KB

Während die Anwendung läuft, kann unter <http://127.0.0.1:8888> das Webinterface aufgerufen werden. Das Passwort lautet 123. Darin können die Modelldaten ausgetauscht werden.

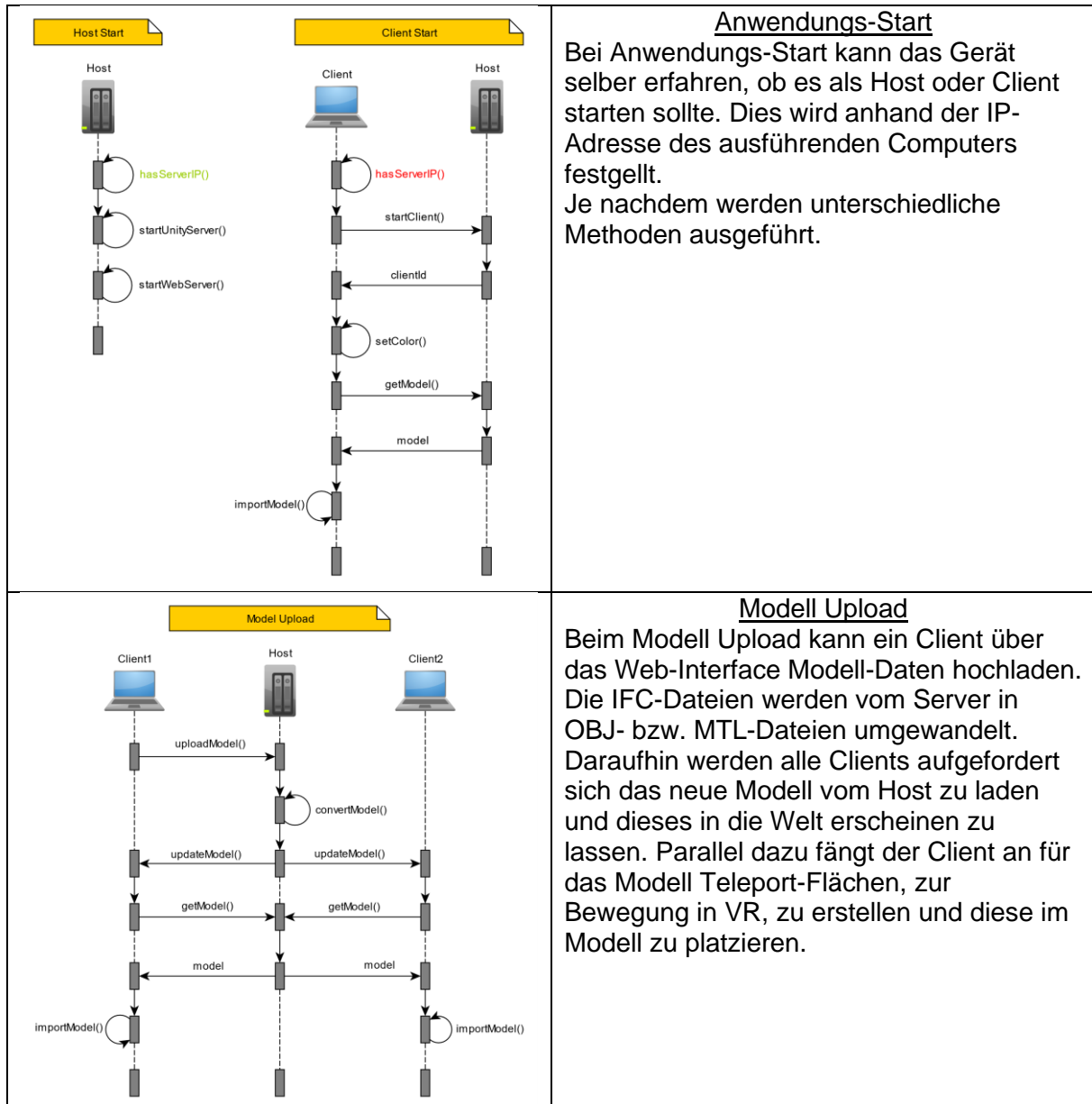
Webinterface

Das Webinterface dient zur Verwaltung der Bauwerke und Pläne, welche in der Welt dargestellt werden sollen. Dieses wird von dem Host-Computer bereitgestellt und ist mit einem Passwort vor unbefugter Nutzung geschützt.

	<p><u>Login</u></p> <p>Das Web-Interface bietet einen Login an welches mit einem Passwort abgesichert ist.</p>
	<p><u>Falsche Passwort</u></p> <p>Bei falschem Passwort wird der Nutzer durch Rot-Werden des Textfeldes informiert.</p>
	<p><u>Dunkles Thema</u></p> <p>Das Webinterface hat ein dunkles Thema mit der Kontrastfarbe orange.</p>
	<p><u>Helles Thema</u></p> <p>Das Webinterface hat ein helles Thema mit der Kontrastfarbe blau.</p>
	<p><u>Bearbeitungsmodus</u></p> <p>Im Bearbeitungsmodus können der Titel, der auszustellende Plan und das Model, welches Präsentiert werden soll, ausgewählt werden.</p>

Sequenzdiagramme

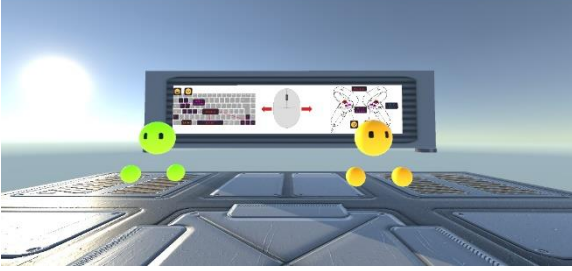


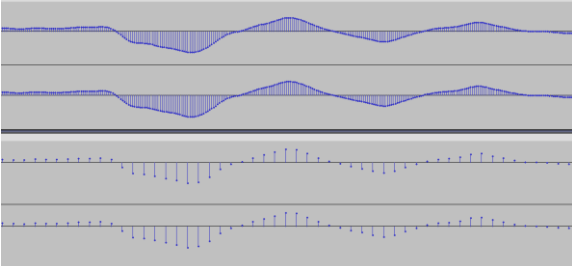
Hier werden einige Sequenzdiagramme zum besseren Verständnis der Anwendung dargestellt.



<p>Client Syncing</p> <pre> sequenceDiagram participant Client1 participant Host participant Client2 Client1->>Host: uploadMovement() Host->>Client1: shareMovement() Host->>Client2: shareMovement() Client1->>Client1: isOwner() Client2->>Client2: isOwner() Client2->>Client2: interpolatePosition() Client2->>Client2: updateClientPosition() </pre>	<p><u>Übertragung der Position</u></p> <p>Beim Client Syncing überträgt ein Client seine Bewegungen an den Host. Dieser verteilt anschließend die Bewegungsdaten an alle Clients, die mit ihm verbunden sind. Jeder Client, der die Bewegungsdaten erhält, prüft ob die empfangenen Bewegungsdaten seine eigenen sind, und updatet diese gegeben falls entsprechend.</p>
<p>Voice Chat</p> <pre> sequenceDiagram participant Client1 participant Host participant Client2 Client1->>Client1: recordAudio() Client1->>Client1: isRelevant() Client1->>Client1: downSampleAudio() Client1->>Host: uploadAudio() Host->>Client1: shareAudio() Host->>Client2: shareAudio() Client1->>Client1: isOwner() Client2->>Client2: isOwner() Client2->>Client2: playAudio() </pre>	<p><u>Übertragung des Sprachchat</u></p> <p>Beim Sprachchat wird kontinuierlich Audio aufgenommen. Dabei wird über mit Hilfe der maximalen Amplitude überprüft, ob das Audio relevant ist. Wenn das der Fall ist, wird die Qualität des Audios heruntergesetzt, um Bandbreite beim Übertragen zu sparen. Abgesehen vom Client, welcher das Audio gesendet hat, können alle die Nachricht hören.</p>
<p>Reaction</p> <pre> sequenceDiagram participant Client1 participant Host participant Client2 Client1->>Host: uploadReaction() Host->>Client1: shareReaction() Host->>Client2: shareReaction() Client1->>Client1: isOwner() Client2->>Client2: isOwner() Client2->>Client2: playReaction() </pre>	<p><u>Übertragung der Reaktionen</u></p> <p>Wenn ein Client eine Reaktion teilen möchte, wird eine Nachricht an den Host geschickt. Dieser leitet die Nachricht an alle Clients weiter. Abgesehen vom Client, welcher die Reaktion getätigt hat, können alle die Reaktion sehen.</p>

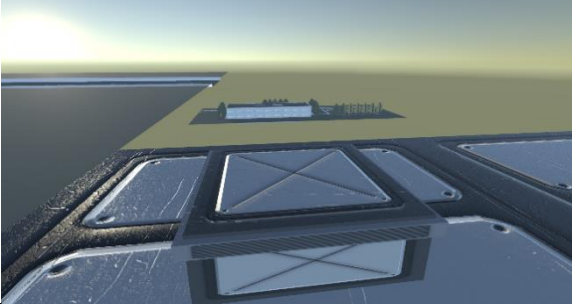
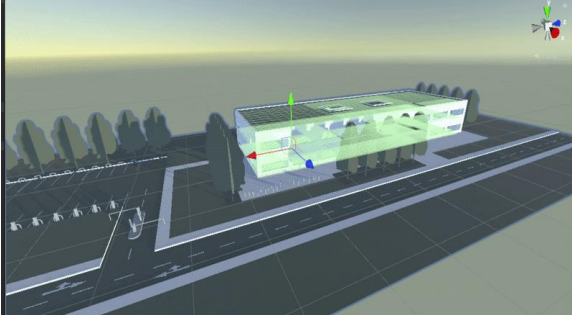
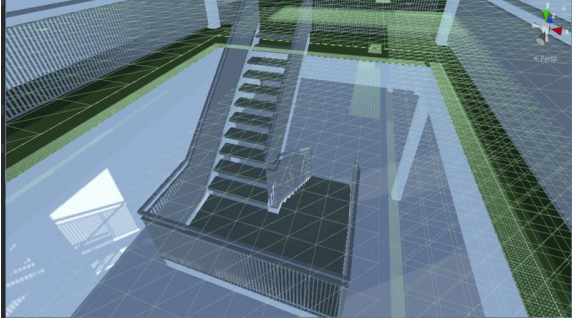
Clients Interaktion

Nutzer können sich gegenseitig in der Welt sehen und miteinander sprechen und durch einen Tastendruck Reaktionen ausführen, welche vom Gegenüber gesehen werden können. Die Sprache eines Nutzers wird zu allen anderen Nutzern übertragen. Dabei wird die Lautstärke der Sprache an die Distanz des Gegenübers angepasst. Das bedeutet, umso weiter eine Person von einer anderen entfernt steht, umso leiser ist diese zu hören, bis sie komplett verstummt.

	<p><u>Client Aussehen</u></p> <p>Jeder Client besitzt zwei Hände, welche als kleine Kugeln dargestellt werden. Die größere Kugel entspricht dem Kopf, wobei die beiden schwarzen Punkte die Augen und damit Vorderseite des Clients anzeigen. Jeder Client wird automatisch eingefärbt, wobei er die gleiche Farbe bei allen Clients hat.</p>
	<p><u>Positive Reaktion</u></p> <p>Eine positive Reaktion wird bei den anderen Clients als Wolke von fröhlichen Emojis angezeigt.</p>
	<p><u>Negative Reaktion</u></p> <p>Eine negative Reaktion wird bei den anderen Clients als Wolke von traurigen Emojis angezeigt.</p>
	<p><u>Sprachchat Resample</u></p> <p>Das Audio des Sprachchats wird durch ein downsample komprimiert. <i>(erstellt mit Audacity)</i></p>

Modell Platzierung

Die Modelle werden auf zwei Arten präsentiert. Die erste ist das diese auf einem kleinen Podium als Miniaturversion dargestellt werden. Die zweite ist die Darstellung in normaler Größe, welche besichtigt werden können.

	<p><u>Podium Modell</u></p> <p>Das Modell wird zur Orientierung runterskaliert und auf einem Podest dargestellt.</p>
	<p><u>Teleportflächen</u></p> <p>Teleportflächen werden dynamisch während des Besichtigens platziert.</p>
	<p><u>Schräge Teleportflächen</u></p> <p>Schräge Flächen und Treppen werden dabei ebenfalls unterstützt. Dabei werden die Teleport Flächen rotiert und skaliert, um das Benutzen von Schrägen deutlich zu erleichtern.</p>

Dynamische Sonne

Die Sonne und somit das Licht in der Welt werden abhängig von der Systemzeit des ausführenden Computers dargestellt. Somit ist es möglich die Bauwerke an verschiedenen Tageszeiten zu besichtigen beziehungsweise darzustellen.

	<p>Sonnenstand</p> <p>Der Sonnenstand um 10 Uhr Systemzeit.</p>
	<p>Sonnenstand</p> <p>Der Sonnenstand um 13 Uhr Systemzeit.</p>