
Leap VR Castle Defense

User Guide

Valerie Gadjali
Timothy Tong
Alexandre Wimmers
Mario Yepez

Last Updated: May 27, 2015

Contents

1	Introduction	2
2	Software Requirements	2
2.1	Windows	2
2.2	OSX	2
2.3	Linux	2
3	Game Installation	3
4	Gameplay	3
5	Further Development	4
5.1	Extending from the Current Leap VR Castle Defense . . .	4
5.1.1	Load the Project	4
5.1.2	Edit the Map	4
5.1.3	Modify Enemies	4
5.2	Starting from Scratch	4

1 Introduction

This project was created using the following versions. Previous versions of runtime environments are outdated and will not suffice. Updates will be required.

- Unity 5.0.1f1
- Oculus Runtime 0.6.0.0-beta
- Leap Motion 2.2.5
- Leap Motion Core Assets 2.2.4

2 Software Requirements

2.1 Windows

1. Download and install the Oculus Runtime for Windows at <https://developer.oculus.com/downloads/>.
2. Download and install the Leap Motion Setup for Windows at <https://www.leapmotion.com/setup>.
3. Download and install the Unity 5 engine for Windows at <https://unity3d.com/get-unity/download>.

2.2 OSX

1. Download and install the Oculus Runtime for OSX at <https://developer.oculus.com/downloads/>.
2. Download and install the Leap Motion Setup for OSX at <https://www.leapmotion.com/setup>.
3. Download and install the Unity 5 engine for OSX at <https://unity3d.com/get-unity/download>.

2.3 Linux

1. Download and install the Oculus Runtime for Linux at <https://developer.oculus.com/downloads/>.
2. Download and install the Leap Motion Setup for Linux at <https://www.leapmotion.com/setup>.
3. Download and install the Unity 5 engine for Linux at <https://unity3d.com/get-unity/download>.

3 Game Installation

No installation required. The Unity system behaves very similarly to Java. The Unity runtime environment acts like the Java Virtual Machine so installation of the game engine will suffice.

*****Critical:** Development of game was perform strictly in Windows 7 and not extensively tested on OSX or Linux.

4 Gameplay

1. First, make sure that both the Oculus Rift and the Leap Motion devices are on and connected to your computer.
2. To run the game, start by executing:
`LeapVRCastleDefense_DirectToRift.exe`¹
3. Put on the Oculus Rift.
4. A game menu will appear. Move your hands in front of the Leap Motion device and press the **Start** button.
5. Enemies will begin spawning. The enemies will constantly run towards the castle and attempt to damage it.
6. You, as the player, must defend your castle. To defend your castle, you can use your hands to pick up the enemies and toss them into the air. When they fall and hit the ground, they will receive damage.
7. Enemies can also spawn from any direction so you will need to rotate your head and/or body to constantly look around.
8. When all enemies on the current level are defeated, a countdown is shown at the top. This is the countdown until the next wave of enemies spawn.
9. The game continues until player runs out of health.

¹ On OSX, DirectToRift.exe is not supported, to run the game mirror your display to the Rift, execute `LeapVRCastleDefense.app`, and position the window correctly.

5 Further Development

5.1 Extending from the Current Leap VR Castle Defense

*****This guide is written with the assumption that the developer is familiar with the Unity game engine.**

5.1.1 Load the Project

1. Clone the Bitbucket repository:
`git@bitbucket.org:ecs193/project.git`
2. Load the scene – `main.unity` – into the Unity game engine.

5.1.2 Edit the Map

1. Using Unity’s terrain tools, modify the map.
2. To add new entities, drag and drop Prefabs from the Project windows into the Scene window or the Hierarchy window.

5.1.3 Modify Enemies

1. Under directory `Custom/Prefabs/Enemies`, enemy entities are notated by the `.prefab` extension in this folder.
2. To change the model, modify the renderer component.
3. To modify the enemy behavior, modify any of the scripts attached to the enemies.²
 - `EnemyAttack.cs` (Enemy script to attack player)
 - `EnemyHealth.cs` (Enemy script to handle enemy health)
 - `EnemyMovement.cs` (Enemy script to handle movement)

5.2 Starting from Scratch

² All scripts are written in in C#