Lake Ontario Documentation

**Cloning the repo and running in Unity:**

1. Go to <https://github.com/btomaszewski/project_lake_ontario>
2. Click the green CODE button and click the clipboard icon under clone section.
   1. If you choose to download the zip, unzip it where you want it and skip to step 6
3. Open Command Prompt or a Terminal. On Windows, click search icon on taskbar and type ‘cm’
4. Navigate to the folder you want to clone the project to. `CD <directory here>`
5. Type `git clone [https://github.com/btomaszewski/project\_lake\_ontario.git`](https://github.com/btomaszewski/project_lake_ontario.git%60) to clone the project
6. Download Unity Hub from here: <https://unity3d.com/get-unity/download>
7. Launch Unity Hub.
8. Then go to this site and download v2021.1.12 <https://unity3d.com/get-unity/download/archive>
   1. Click the Unity Hub button for that version and click open in Unity Hub.
9. After the Unity version is downloaded, go to Project tab in Unity Hub and click Add.
10. Select the cloned project of Lake Ontario and add it to Unity Hub.
11. Click on the newly added project to launch it in Unity.

**Overview of project codebase:**

The Prefab folder holds most of the assets that is utilized in the scene. By using prefabs, we can make changes and edits the prefabs themselves without having to manually edit every instance of the GameObject in the scene. We also have a scene for each developer and 1 main scene. This way, everyone can develop within their own scene and plan out merging their changes into the main scene to help prevent any merge conflict if two people were to edit the main scene at the same time.

The main scene is made of several GameObjects/Prefabs:

* 1 player
* Terrain
* Houses (with propane tanks included)
* Telephone Poles
* Cars
* Trees
* Lakeside and Pondside sandbags
* Lake and its masks (for showing and hiding water)
* Lighting
* Cameras (non-player cameras)
* Event System

Sandbags are controlled in two ways. For the pondside/lakeside sandbags already set in the scene, those sandbags are grouped into two GameObjects and the sandbags are tagged as ‘defaultSandbag’. One for lakeside and the other for pondside with the sandbags as child GameObjects for them. The SandbagControls script is attached to the ControlSystem GameObject. The script has a reference to the two sandbag groups GameObjects and to the toggle buttons for them on the GUI. SandbagControls script is what reads the P and L keys to turn on or off those sandbag groups. The second way sandbags are controlled is by the PlayerSandbag script attached to the ControlSystem GameObject with a reference to the sandbag prefab to use. When right clicking, the script grabs the currently active camera, raytrace a few meters in front, checks if it hit land, and spawns a sandbag there with the tag ‘playerSandbag’. If shift+right clicked, the script raytrace again but looks for if it hits a sandbag and if so, destroys the sandbag if it is tagged as ‘playerSandbag’ and disables the sandbag if it is tagged as ‘defaultSandbag’ so toggling the lakeside/pondside buttons will restore those ‘defaultSandbag’ tagged sandbags.

Water level is controlled by the FloodController script attached to the Lake GameObject with references to the water level slider, the water masks, and the two sandbag groups. When the user presses and holds the K or O key, the script will increment or decrement the water slider on the GUI. The slider’s value is used to raise and lower the Lake GameObject within the FloodController script. The script will also draw onto the GUI a box and say what the estimated water level is.

CameraSwitch script is attached to the prefabCameraSwitch GameObject which is a child of the Cameras GameObject. The script holds a list with a reference to every camera in the scene. Any new camera needs to have their reference be added to this list in prefabCameraSwitch by inspector mode and the script edited. The script disables all cameras and keeps the current camera active. Clicking a number key will enable the new camera the user wants. This script also hides the mouse cursor and locks it to the center of the screen to make the player camera and flying camera be able to look around more easily.

CameraMovement script is attached to the camFlyAround GameObject which allows that camera to be controlled by the WASD/Arrow keys to fly around when active. It also has a reference to the camera’s collider so that users cannot fly the camera through terrain or houses.

SimpleCameraController script is unused.

The editorGameobjectReplace script is a unique script that can replace all non-prefab GameObjects in a scene with a prefab at the same position and rotation. Useful for correcting a mistake such as accidentally placing 100+ non-prefab sandbags but then realized you need to make them into a prefab. At top of Unity, click EditorScripts, click ReplaceGameobjects, then drag in the prefab to use from the Asset window and then drag in the GameObject whose children GameObjects are the one you want to replace with the prefab. Note, CopyValue is not implemented so if you want to sync extra data from the non-prefab to the prefab variants besides position and rotation, you’ll have to edit the script.