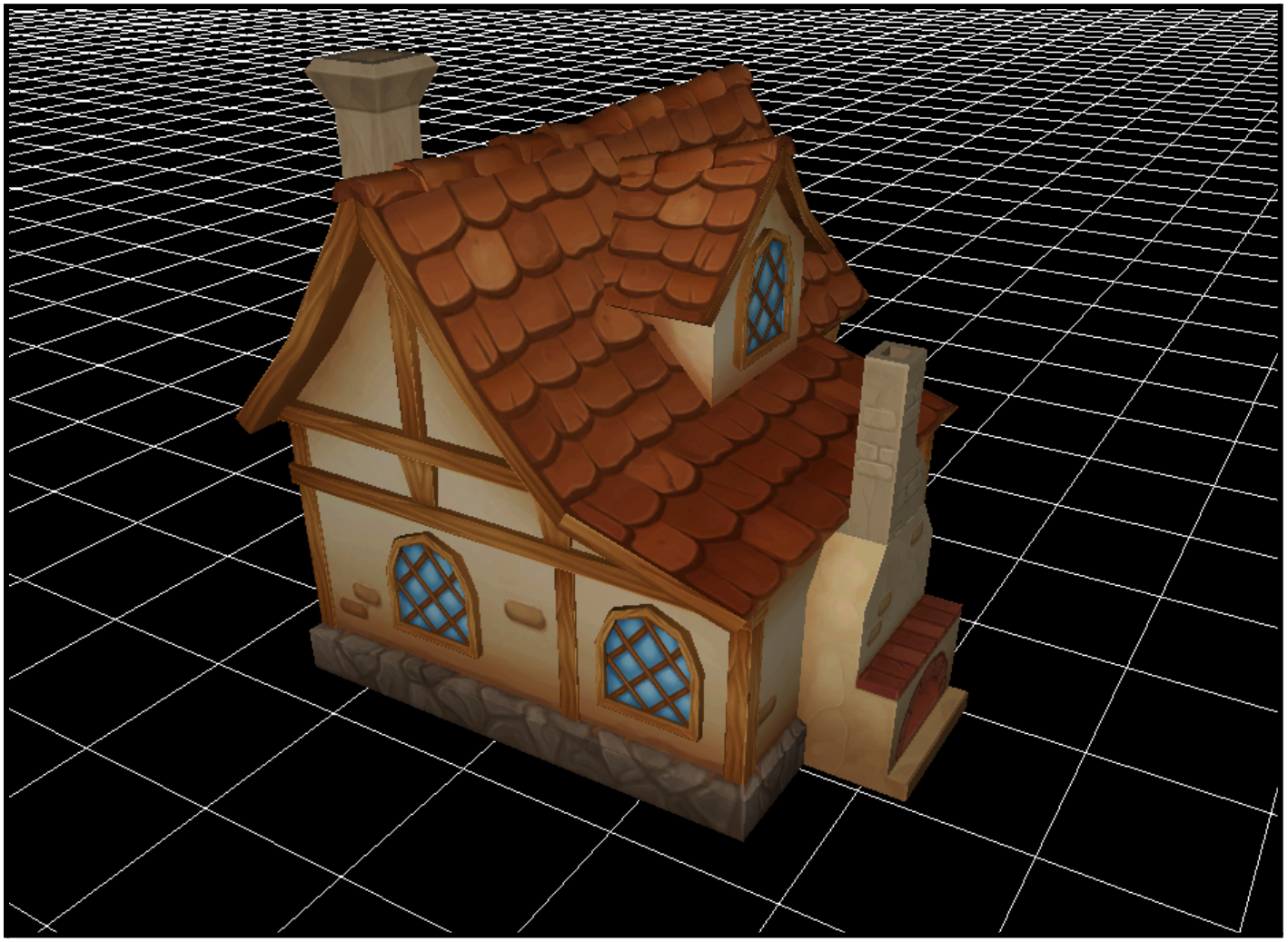
Game Engines - Assignment 1

Overview

For our first assignment, we aim to deliver a simple geometry viewer supporting drag and drop and orbital camera. The final release should load the provided model (baker_house.zip) plus other models of your choice.



Submission rules

The delivery must be a **Build** zipped in its folder inside *"First Assignment"* named after your engine. The release should be a **zip** containing:

1. **README.md**:
    - Short description of the engine and link to the github.com page of the project
    - Llist of team members and github accounts
    - How to use the engine, detailing the controls and any specific action
    - Additional funcionality in he engine outside of the assignment requirements
    - Additional comments for the teachers to understand some part of the engine

2. **LICENCE.md**:
    - Chose a licence that fits your project - https://choosealicense.com/

3. **Project files:**
    - Executable compiled in Release with all necessary DLL files
    - A resource folder with all the media files (fbx, png, dds,...)
    - *No other file must be there!* - Be sure to remove any code and unnecessary files
    - Maximum of **30 Mb** of zipped build

The folder structure should be the following:

```
> EngineName_v.X
    > EngineName
        - Game / assets / resource files
        - .dll files
        - EngineName.exe
    > README.md
    > LICENCE.md
    > Licences (folder with all third party libraries' licences)
```

Assignment Content

1. The build should be compiled in Release with all *(and only)* the material needed for execution (including two other sample geometries to load).
2. Baker_house should be automatically loaded at the start.
3. Your own two models must be in FBX format with one channel diffuse texture in DDS format.
4. It should accept drag&drop of FBX files and textures (only to show the last dropped) from anywhere in the Hard Drive. Remove the current geometry if another one is dropped.
5. To find a texture try (LOG every step):
    a. First check on the path described in the FBX
    b. Then check on the same folder you loaded the FBX
    c. Last, try in your own "Textures/" folder
6. It should feature Unity-like camera controls:
    a. While Right clicking, "WASD" fps-like movement and free look around must be enabled.
    b. Mouse wheel should zoom in and out.
    c. Alt+Left click should orbit the object.
    d. Pressing "f" should focus the camera around the geometry.
    e. Holding SHIFT duplicates movement speed.
7. Have a console window that should LOG the geometry loading process from ASSIMP
    a. This means that all the debug output from ASSIMP must be captured on our console.
8. There must be a configuration window containing at least:
    a. A graph for the frames per second
    b. Configuration for all variables on each module (renderer, window, input and textures)
    c. There must be information output with FPS graph, memory consumption, hardware detection and software versions (SDL, OpenGL, Devil).
9. The camera must adapt to the size of the geometry after being dropped to move far or close depending on the scale of the geometry.

10. There must be a general menu with option to quit, visit the github page, and get info about the engine (“About”). It should also be able to turn on/off editor windows
11. A properties window with three sections: transformation, geometry and texture. All should give read-only information about the current loaded meshes and texture (triangle count, texture size).
12. If the FBX file contains multiple meshes, the system must be able to load all of them without applying transformations. It should only need to load one texture at a time.
13. If the window changes its aspect ratio the graphics should not deform.
14. The window size of the editor must be in relation to the desktop size (you can request the desktop screen size from SDL).

The **zip** must be submitted before **December 8th 23:59** (folder closes automatically). The build must **also** be uploaded to the github repository under the Releases section.

Acceptance Criteria

1. It followed the submission rules stated above.
2. The code compiles and uses only english.
3. It should be **original**. Since most of the functionality will be the same, this will be monitored closely.
4. The release did not crashed while testing.
5. The FBX provided can load and renders correctly.

Grading Criteria

- 10%: Repository & Commit structure (small commits with clear description)
- 60%: Code structure / Good choice of containers / Const-correctness and use of pointers/references/ No resource leaks
- 30%: Unity-like camera controls & Editor tools (main menu, console, editor info and configuration, properties)

To achieve the maximum grade on each area the code is expected to be: const-correct, correct choice of STL containers, efficient, simple and readable