BrainGenix Initial Engine Usability Planning Report

Document Author: Josh Coldiron

Goal: Explore BrainGenX game engine with the goals of establishing plans for Usability, QA tests, and User Experience, as well as an exercise in need-finding based on the current state of the project.

Contents:

Methods

The approach to this project was to initially try and install the engine on a Windows machine and blindly use it to see how intuitive it was and a baseline of its capabilities. After the initial test, I reviewed existing documentation and asked questions to Thomas Liao who is working on the engine to see what features exist and if any of the assumed features functioned correctly. Through this process, I took notes, logged bugs, and examined the engine for the sake of developing notes to present to the team. Through the use of the engine, I noted what features are initially available as well as created a list of suggestions that occurred to me during this research. The following sections are broken down into different departments that may aid the development of this engine and games that derive from it. This document will cover each topic in its own section. The Testing section will make suggestions for organizing a Test Plan for the engine as well as the game made from it. This section will also include some of the bugs I encountered. Some of these may already be addressed but as we develop a formal test plan, we can log these through that process. The next section will discuss ways we can enforce good UX in the development of the engine and carry that over into the game menus. This section will also touch on the User Interface. Next will propose a plan for documentation that could be essential for the engine. This section will also include some notes on proposed localization. Lastly, suggested features will be a list of features I feel are worth discussing.

My experience pertaining to this research includes various roles in game development: Quality assurance testing, Playtesting, User Experience, Production, Game Design, Level Design, Technical Documentation, Localization, Unity Development, and some development in a game engine proprietary engine. My goal is to take my departmental experience and apply it here.

Initial Use Test

Testing Usability from Intuition and Learn-ability

Install

Key findings (initial use) Testing was done on Windows 10 with a machine of the following specs: i7-9700k 3.6G, 16GB Ram, GeForce RTX 2080 Super. Installing the application was a bit of a challenge at first. It was not the fault of the engine, but issues related to flaws in Microsoft's Visual Studio install. There is a common issue with C++ libraries not installing correctly and causing a conflict. After speaking to Thomas, I downloaded a prebuilt version which launched without any issues. It may benefit for the sake of promoting the project to show videos on the BrainGenX YouTube that cover installs and workarounds.

Initial Use

I was first greeted with a large blank window (viewport), an FPS graph across the top, and a variety of windows on the left. I attempted to move the mouse to see if a 3d environment exists in the main window. Mouse navigation was initially not intuitive. Clicking the left mouse allows us to look around the viewport, but I had to discover the combination with the "asdw" keys to navigate the space. Once I realized it was the equivalent of moving around an FPS game, I was able to navigate it more easily. I then placed the location of the default sample object in the center of my viewport.

Next, I went on to click on the controls of the object to see how I can modify it. The 3d controls allow us to distort a single object in the scene. It was easy to determine the controls to change the size of the object and distort it, but not to move it. The move controls are not always present. There is a single model in the scene however the asset explorer does not list any models or textures. The scene tree did however give us access to the models, lights, and cameras. Through this, I was able to discover the selectable area for changing the position of an object as well as cloning or deleting it. I then changed the toggle settings to see what effects they had. Shadows can be toggled on and off. The color and intensity of the light source can be changed. The light's max distance seems to function. The results are on and off as I moved the range. Light rotation and scale seemed to have no visual effect. Adjusting object parameters seemed to show a change in most that were observable with the standard model.

I tried to import models through the menu. I tested ftx, obj and dae formats, which that showed they were imported. The mtl format gave a metadata missing error. None of the items appear in the model tree or in the viewport.

I tested the controller feature. I had the engine detect an Xbox One controller. The system registers movement in the controller settings. It has no effect on the scene navigation though.

I attempted to change the color theme through the settings menu, but no other themes worked. I was able to change fonts though. I started closing windows to see how it responded. First, I closed the viewport window. I was able to bring it back by going to windows > viewport. If you close all the windows the viewport will expand to fill the space.

The system log seems to reflect actions taken. I did not see a log file produced when the system crashed. I did see a section for scripts but no way to implement them. With little knowledge of what the engine does, I did feel that there was enough that utilized from standardized methods for applications that I was able to ascertain much of what I could do.

Researched Use Test

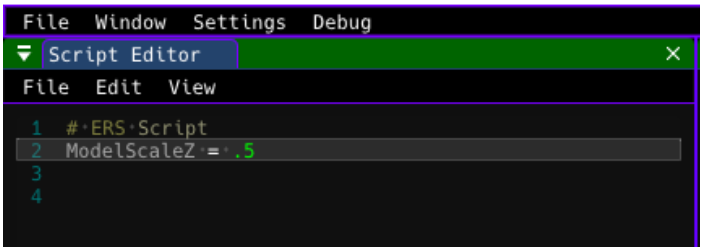
Testing Usability from Existing Knowledge Base

Knowledge Base

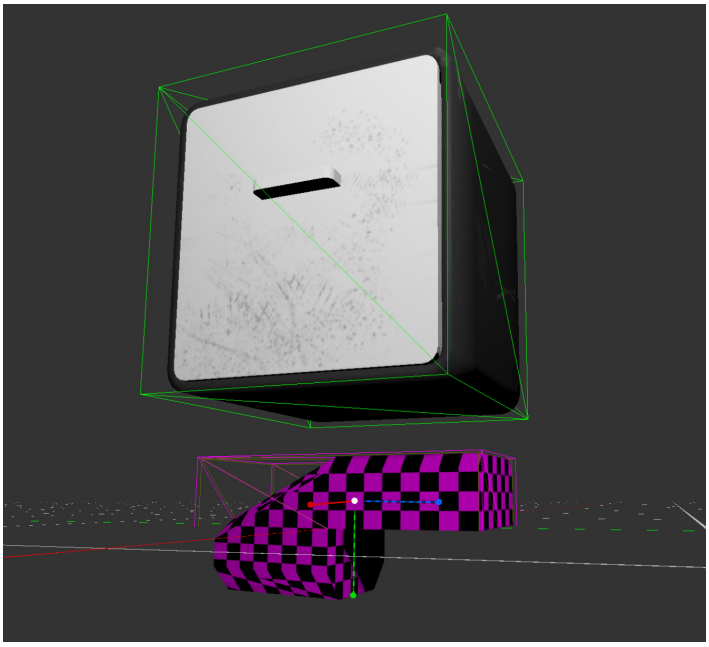
Use after research After my blind reaction, I contacted Thomas for some insight, information in the GIT instructions, and read the included readme files. Having a better idea of the capabilities of this state of the application I was then able to test it further and examine its usability. While the documentation needs to expand some, it was enough to give me an idea of how to approach this application.

Testing

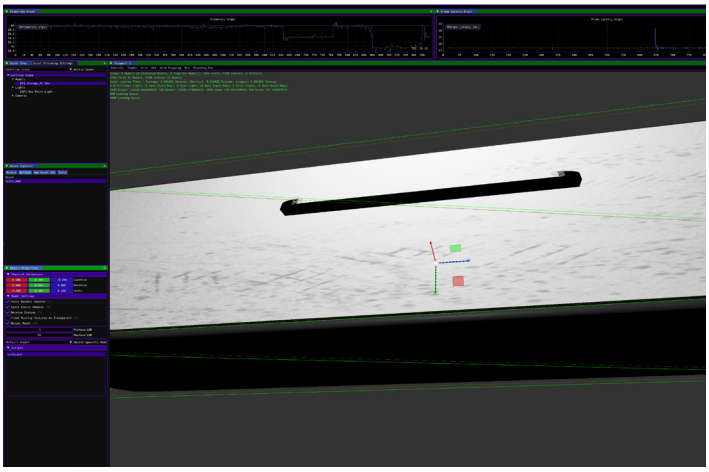
The models were still loaded into the project tree. I was then able to drag the listed model into the scene. It appeared with a checkered texture, but it was in the scene and could be edited just like the demo model that was already there. I was able to duplicate the model and see it in the scene as well.



Next, I attempted to attach a script. I loaded the script editor window. Through its menu, I was able to create a new script. I was unsure of if there was special syntax needed. I tried to look through the folder to see if any existing scripts existed but noticed all files were of a (.ers) file type. I was able to look through them but the file management system might make external editing or file management a challenge. I simply added this code and saved the script. (ModelScaleZ = .5). I then saved the script. In the script list, I had an unnamed script that I changed the name of. From there I was able to drag the script to the model in the object tree and link it. I was unsure if it had been linked. Nothing in the display port changed. I pressed f5 to see if the system would run the script and immediately the object stretched along one axis. I had assumed the scale would be cut in half; however, it did show me that it did have an effect on the demo model.



With what I experienced from this trial run I was able to see some current capabilities and to develop suggestions that will be covered in later sections.



Usability Review

Usability results of this research

Usability

The first factor we will examine is learnability. We want to know how easy it is to complete simple tasks when we first use the engine. The initial navigation and windowed system rely on established ideas which makes that aspect easy to learn and complete simple tasks. The naming convention of the windows and actions also can be said to be the same. Some areas I feel that may be on a more difficult scale would be the addition of a model and texture into a scene. It is not a difficult task, but without prompts, one may not assume how to do it correctly. The same can be said about the scripting system. Creating, renaming, and applying a script is one area that we may be able to streamline. The same goes for running a script with the F5 key. This is an item that maybe belongs in the menu system and elements that use shortcuts may need those shortcuts listed beside their menu button.

Efficiency

Next, we can consider efficiency. This is when we know how to do something, and how quickly we can do it. Knowing how to do something, most tasks I tried to complete were not very long. I think some things can be streamlined such as the scripting, and navigating the viewport. It can be a little cumbersome navigating and I will add a section for suggestions that may improve this experience.

Memorability

Next, we can examine how we remember how to use features after a long break. The basics were a non-issue in this however, I did have to remind myself of the order of windows to create scripts as well as deal with model importing. I think we can streamline this process and reduce the number of steps and or windows needed to accomplish the user's goal.

Errors

When encountering errors, one can track their actions in the log. Knowing an error will rely on the user knowing what the action can do and then seeing if it took effect. I think the implementation of some error feedback and error prevention may help train the user. Actions such as a tone when an element is dragged to a place that it cannot be placed would inform the user of a failed attempt. Other systems can be discussed as they are needed.

Satisfaction

The next question is how pleasant the product is to use. That, of course, is subjective, but I think if we consider the quality of life items and how to streamline tasks and better inform the user of actions we can easily maintain a result of satisfaction in the user.

Feature Suggestions

Features to discuss with team and further questions

Features

The following are features or settings I feel will help inform the user and make the application easier to use.

• When the engine is loaded the viewport camera should be pointed at the demo object and light. This will indicate the viewport is showing a 3D scene and draw the user in to interact with the available elements.

• The ability to turn on viewport poly count as well as selected object poly count. This could simply help with debugging when scene cost is trying to be determined when optimizing a game.

• Offer template downloads just as Visual Studio Code offers. While we can have the designed themes available the user being able to download and add needed themes may encourage community development and growth.

• The navigation in the viewport works fine however a single-axis arrow movement would be helpful to position the scene in a more precise manner. An example of this would add the use of the arrow keys to move the view on a single axis. The up and down arrows would move you on the Y-axis. The left and right would move you along the X-axis. Other ways of doing this can be explored but it would help avoid the user having to position themselves in a way to get to the view they desire.

• Folder system for managing files for each project. This is something we can work on internally rather than leaving it up to the user, however, we want to make sure it is standardized and logical for their exploration.

• A system for naming scripted files so they can be located easier in the file system.

• Export log option.

• Error message system as well as a notification system for when things work as intended.

• Toggle viewport elements on and off such as grids and frames.

• Report crash button

Questions

Some remaining questions after the application review:

• How are physics implemented or controlled?

• How much functionality is controlled through scripts? Are there built-in functions that can be activated through scripts? Example: A collision system already exists, and a script can apply it to an object or be used to modify it.

• Can scripts be linked to a variable editor in the engine? Example: We have a script that controls player movement. Through a field, in the editor linked to the script we can change that speed without having to modify the script. Similar to the variables we can change that are applied to models(scale, position, etc..).

Supplemental Dev Plans

Proposed Plans for Various Departments

QA Test Plan

The QA plan will be a team wide responsibility. The initial goal is to establish a set of elements that are regularly checked when each member uses the engine and game. The next would be a simple reporting system that allows anyone on the team to report a bug. We will need to follow this with any component that we can automate for the dev team to add that functionality into the engine. If we are able to acquire some more dedicated QA volunteers we can implement a more stringent testing plan. The following at a list of bugs I encountered during this research:

• Theme: Only a single-color theme can be selected the rest have no effect.

• Font: When a font is changed the “Reload” option does not reset the font to default.

• Adding a controller profile crashes the engine.

• No log in the folder after the crash. Logs are still viewable in the log window. If there is a log it could be under a name and file type that I did not know to identify.

• When reloaded the engine no longer renders the model but shows the green bounding box. I was able to resolve this by exiting the engine and relaunching it.

User Experience Plan

A UX plan would be to make the user’s goals easier to achieve. At this level of development, I think the engine is bare bones enough that it is not as much of an issue but as the program grows we will need to keep in mind of our UX methodologies and system management so that we can retain the ease of use. I think the primary goal is for us to create a list of all the current features and list out the steps to accomplish that text. From there we can examine how to streamline those processes then we can use that to set our standardized UX principles to carry through the rest of the development.

User Interface

While the interface is linked to UX, we still need to handle it on its own. The first element we want to explore is how much we can edit the UI. While we may be limited to themes simply because of the OS cross-compatibility limitations, we may be able to change the things that would be crucial for the overall user experience.

Documentation

I understand that our documentation is fairly sparse, but I do feel we need to develop a set of essential documentation and possibly supporting videos on the BrainGenX YouTube channel that can better show others how to use our open-source tools. The initial plan should be to establish a list of all functionality supported in the engine as well as a guide to all inputs and controls. Another would be a detailed set of install notes.

Localization

While localization might be far off, once we have established some base documentation that explains the tools better, at the very least we should consider some localized pages that will describe the application and tools and inform the user the further primary information will all be in English for the time being. Google translate is not a reliable translation tool, but between it and chatgtp we may be able to get it close enough that another staff member who is fluent in that language can proofread it.