Trevor Brown

ITLS 6245 Final Project Documentation

**Credits**

I worked with my engineering department in obtaining the flow of how each tutorial would go. We discussed certain needs and worked examples. We developed the visuals that are embedded in the tutorials.

In terms of the code, specifically for this project, I used Unity’s API website a lot to understand documentation. I created most of the code with the help of the API website.

Some of the code and assets were created for last term’s project. In terms of the code, I don’t recall where I got some of it. All of the code was written by me, but snippets inside each file possibly came from either stackoverflow.com or the Unity forums / Unity API. However, everything this term I wrote and mostly used Unity’s API code if I had specific questions for the code.

**Team Member Contributions**

Only I did this project. I didn’t have any team members helping.

**Next Steps**

There are two main steps for this project.

The first is creating this application as a web project. That might take a little bit to figure out or it will be very basic.

The second step is to create all of the assets. Currently the variable letters, operations, and music were taken from various sources online. I don’t know exactly where I got each of them. However, I will recreate each one to stay safe from copyright issues.

**Timeline Structure**

The main events in this game consist of clicking the mouse, as well as dragging the mouse. I want this game to be web and app friendly, particularly with the mouse and a touch screen. Using keys on a keyboard does not occur because of the inconvenience to the mobile user.

In terms of clicking the mouse, buttons are used to create the events. For the actual game portion (not part of this project), there is a click and drag event. The main purpose of click and drag is in each level, but you can still click and drag the variables and operations in any scene with those present.

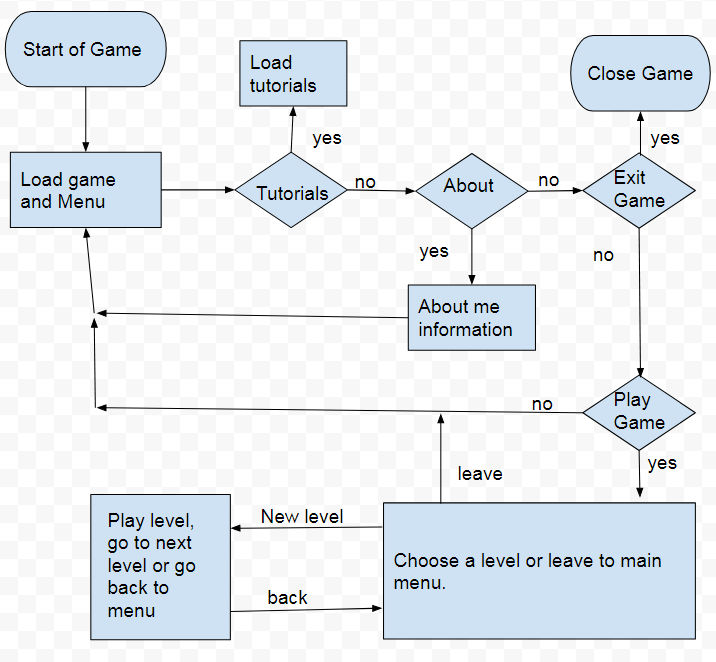
The main navigation is through buttons, which can be clicked. Each button in the menu screens leads to another scene. In either the tutorial scenes or in the level scenes, there are some buttons. The buttons in the tutorial scenes either start the tutorial from the beginning by resetting values to go back to the beginning, go to the previous portion of the tutorial, or go to the next portion of the tutorial. There is a common button in every scene except the first main menu, and that is to go back to the previous menu screen.

In the level scenes, there is a button to check your answer, a button to reset the level, a button to go back to the menu, and a hidden button that appears when you get the correct answer that takes you to the next numerical level.

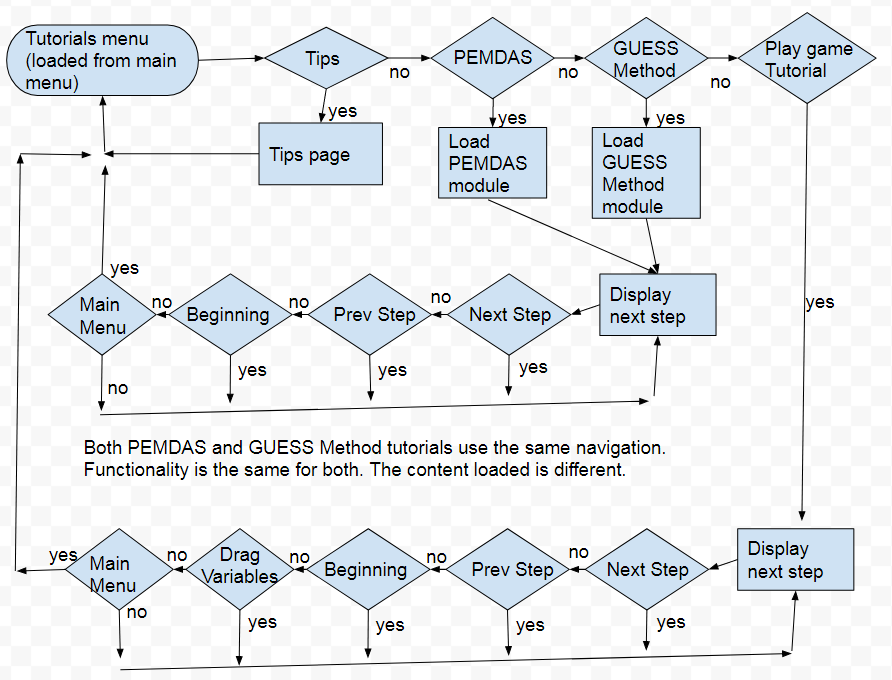
**Flow Chart**

Here are a couple flow charts depicting how to navigate through the game.

Main menus for the game (last term’s project mostly) \* note: Load tutorials better shown in following flow chart.



Menu for tutorials (from Load tutorials from first flow chart)



* Flow chart – This graphic can be an excellent means for diagramming some of the main system interactions.  Depending on the complexity of your project,  you might want to use a series of small flowcharts or one large flow chart.  (If your layers are well organized and named well, you might collapse the timeline into the flow chart, by providing the frame number(s) of the various portions of your project).

**Naming Convention**

For variable names, script names, and names for game objects found in the inspector, all use camel case. The names given to the objects and variables relate to how the object or variable is used. Because there is a lot of text and transitions, there are many hard-coded values. Creating an array of these values could have worked, but it could have been confusing which phrase contains certain messages. Most variables are made global to the file so they can be accessed throughout the script.

For the previous project, the scenes use a specific naming to help the developer know what each piece does and how it impacts the game.

**Important Variables**

There are key variables used in the different tutorial scenes’ scripts. One of the main variables is itemNumber. This variable holds the current stage of the tutorial. As you press buttons, it changes the value of this to go through the tutorial. Each script contains a text variable called instructions, and the GuessTutorial script contains an additional text variable called problemText. As you work through the tutorial, these variable’s main text changes to give instructions to the learner.

It is important to note that there is a lot of interaction between game objects via script. Most scripts can manipulate other game objects through the GameObject.Find() function. Because of this, I haven’t stored a lot of variable for these game objects.

The previous project has many important variables, including the assigned number on each game object in the level. These numbers help determine whether the player is victorious or not. There is a variable for music. The target game object variable is stored as a variable so the game knows which variable to solve for.

**Code Segments**

This term’s project has four scripts that were created for it. These scripts are called GameTutorial1.cs, GameTutorial2.cs, GuessTutorial1.cs, and PemdasTutorial1.cs. each of these files work similar to the other. They use basic navigation to work through the tutorial. As you press the buttons, they take you to the desired location of the tutorial.

The other scripts were created for the other project. They go into depth on linking to other objects, parenting to those objects, storing scores, playing music, cloning operations, checking win conditions, etc. These are all important functions and without them, the game wouldn’t be functional.

See any comments for specifics on what each portion does.

**Known Bugs**

In this term’s project, there are no known bugs. Each button does what I want for it to do. You can proceed forward and backwards without any problem. Images and text update accordingly. There were a few bugs that are no longer there based on this.

The only bug that currently exists from last term’s project is keeping the score from the player during the level portion of the game. In order to test the game effectively with keeping score from level to level, I stored the score in the playerprefs object. However, since I was testing it, it constantly updated the score without my knowledge and my score didn’t make sense. The action I took was to reset the score to 0 when you start level 1. Playing level 1 multiple times erases your score. This is currently the only bug I know of in the game. I will want a few people to test out the game in case there are any more bugs.