Lab Report - 01

Andrew Seba  
Game221 - 001

This lab report undertakes the programming practice of automatic folder creation. Without project folder organization teams can easily lose their project files. The main goal of the lab was to write a tool script for unity that could help project teams start off with a well-organized folder structure with .txt files in most of the folders to communicate what to put in them.

In Unity placing scripts inside the Assets/Editor folder allows unity to run them only in the editor. For the script ‘using UnityEditor’ allows us to call functions to make a new tool menu in Unity, there is a flag you call before the function definition [MenuItem(“MenuName/ActionName”)] . With this it will add your tool into the Unity window tool bar at the top. The AssetDatabse.CreateFolder() function also comes from the ‘using UnityEditor’ to create new folders. The first parameter for this is the root folder you want to create your folder in, and the second parameter is the name of the new folder. Using that we created each folder individually. ‘Using System.IO’ were able to create and write .txt files.

### Technical Outline

* Import Dependencies
  + System.IO
  + UnityEditor
* Flag the function to be a menu item
  + For each folder to be needed use the AssetDatabase.CreateFolder() function.
    - First parameter is the base folder you want to create the folder in as a string.
    - The second parameter is the new folder name as a string.
  + For each .txt file use the System.IO.File.WriteAllText() function
    - While using the Application.dataPath to start the directory path in the ‘Assets/’ Folder
  + After all your folder changes use the AssetDatabase.Refresh() function to update the folder browser in unity.

For this lab I’ve learned so much about the power of editor tools and how easy it is to set up your own ‘Menu Items’. Writing files to the system was way easier than I thought and I can see using this to save files needed for tools to make the tool your “Best Friend” in remembering what you like or what you’ve done. File structure is pretty straight forward. I will be using this to initialize my projects in the future because I know this will make my projects way more organized that they ever have been in the past.

# Post-Lab

1. What is the purpose of the Editor folder?
   1. Unity uses the editor folder only when the editor is running to load/run scripts inside the editor.
2. Menu items can be added to the inspector using what statement?
   1. [MenuItem(“MenuTitle/MenuOption”);
3. What must you watch out for when adding menu items?
   1. Can’t add a menu item without being nested.
   2. If you do two menu items before the same function it will just run the same function.
4. How do you attach an action to a menu item?
   1. Using the menu item statement before a public static void Function()
5. What is the purpose of AssetDatabase.Refresh()?
   1. It refreshes the editor to show new changes inside the editor. Such as new folders or files.
6. What is the difference between a static asset and a dynamic asset?
   1. Static assets are all the assets that aren’t loaded into the game at runtime.
   2. Dynamic assets are instantiated during run time using Resources.Load(). If they are both dynamic and static put them into the dynamic folder.
7. Why is it important to keep static and dynamic assets in different folders?
   1. Might have similarly named assets that have completely different properties. And when loading assets it’s nice to have them in one folder that just need to be loaded instead of individually keeping them separate.
8. Think about your workflow, what would be your optimal folder structure/organization?
   1. Next page.

# Personal Idea of file structure.

* Assets
  + Models
    - Models
      * Characters
        + Male\_01

Male\_01.mdl

Source

Male\_01

* + - * Environment
        + Lake

Bush.mdl

Source

Bush

* + - * Items
        + Health.mdl
        + Source

Health.blend

* + Particles
    - Explosions
      * Boom.particle
    - Spaceship
      * Jet.particle
  + Sounds
    - Music
      * Soundtrack
        + 1
    - SFX
      * Explosions
        + Boom1.wav
  + Scripts
    - Characters
    - Items
  + Scenes
  + Editor

My thoughts were you could put the source materials along side the folders. But having them in the same folder saves a long directory list and makes it easier to find.