Humanizing COVID-19 Data through Visual Analysis in Unity

A thesis presented by

James Finnerty

Submitted to the Board of Mathematics & Computer Science

School of Natural and Social Sciences

In partial fulfillment of the requirements

For the degree of Bachelor of Arts

Purchase College

State University of New York

May 6rd, 2021

Sponsor: Dr. Athar Abdul-quader

Second Reader: Dr. Iuliia Chikish

Contents

Abstract

There are numerous statistical models extrapolated from data gathered on the 2020 pandemic caused by COVID-19. One such model by the USC Center for Risk and Economic Analysis of Terrorism Events projects almost $5 trillion in net GDP losses, from March 2020 through the end of February 2022 (Gibson), signaling that the preventative methods against the virus and how well they are followed will help to either prolong or shorten our economic recovery. Though highly accurate, few of these models convey the data in laymen's terms that are approachable from a non-statistical point of view. The scatter plots, heatmaps and other graphs representing this data only begin to scratch the surface of humanizing the reasoning behind flattening the curve. In this project, I developed an application which brings a visual reinterpretation of the millions of lives socially and economically displaced by the pandemic in the United States. This application generates a "virtual room" of people infected by COVID-19 and the various outcomes of the illness.

Introduction

In March 2020, the reality of COVID-19 had begun to settle into the minds of the public in the United States. Non-essential businesses, schools, and universities had begun to shut their physical locations and open new virtual ones. These shutdowns were dictated through government thanks to the models produced by experts that informed the world that without quick and decisive action, our medical institutions would soon be overwhelmed. On March 20, 2020, Governor Mario M. Cuomo signed the 'New York State on PAUSE' executive order, shutting down all non-essential businesses in New York, among other restrictions, as the first step of many to combat the COVID-19 virus (Cuomo). As of October 26, 2020, the state has been successfully engaged in the process of flattening the curve, thanks to the actions of our state's leaders and its inhabitants, including the collaborative efforts between the administration and student body at State University of New York at Purchase. As of May 4, 2021, a vaccine against COVID-19 has recently been made available for all age ranges as New York plans to fully reopen on June 1st, 2021, hopefully indicating a return to normalcy.

Prior to March, in early January 2020, I had been given word from my mother's side of the family that there was a new virus emerging from East Asia. Given her yearly trips to Taiwan to see her mother, her family urged her to stay in the United States and to not return home for Chinese New Year later in the month. My initial reaction was to dismiss my extended family's concern as them being overly cautious, while in the back of my mind I was unsettled. From late January through February 2020, amongst reports of Chinese nationals within the country and abroad returning home a new strain of coronavirus began to spread in the aftermath of Chinese New Year, my family's fears were validated. On March 2nd 2020, the virus had proven that it reached the shores of the United States, going from one to one-hundred cases ten days later, in Westchester County, where I was born and raised (CNN). I have had close friends who have fallen ill with COVID-19 and were fortunate to resolve, and associates of family members who caught the virus and passed away shortly thereafter. In the span of ten weeks, from early January to the middle of March 2020, I went from a fence-sitting skeptic of COVID-19 to accepting the stances of the professional community that gathered the data, simulated the models, and urged us to engage in the practices of social distancing, mask-wearing, and quarantine. What resolved my cognitive dissonance when rejecting the severity of the virus was not from examining the statistics independently, but from examining the statistics in tandem with the human element that personally impacted friends, family, and members of my community.

For a clearer picture of the impact of the 2020 pandemic, more is not necessarily the answer. In the words of Ernest Rutherford, Nobel Prize winner for the orbital theory of the atom, "An alleged scientific discovery has no merit unless it can be explained to a barmaid." (Whitrow). Applying this statement to the data on COVID-19, there are an exhaustive number of models showing the peaks and valleys of the curve and their statistical significance. Few of these models effectively convey this data succinctly and to the point where an individual with no prior education in statistics could grasp the concepts on the same level as a professional. By harnessing a game engine, used to create interactive and accessible experiences, the goal is to use visualization so the user can view the application and walk away with a personal understanding of the United States population directly affected by the pandemic, and how one's actions help to prevent the further spread of COVID-19 while the world's experts search for a cure.

With a passing interest in game design from a consumer standpoint and no prior experience in the languages used within the most popular game engines, Unity and Unreal Engine, designing an application in either one of these environments would prove to be a challenging task. The purpose of this project from a programmer's perspective is not to serve as a treatise in game design, but to serve as the proof-of-concept for a project developed with newly introduced technologies and proprietary toolkits, from start to finish. The design and development process will take into account some of the many economic impacts of COVID-19 and the 2020 pandemic, the game engine of choice, Unity, the programming language it uses, C#, the creation and use of character models, and the procedural generation of game objects in a 3D environment using a "save file" of open-sourced COVID-19 data from the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University. The finished state of this proof-of-concept is comparable to a "vertical slice" or game demo, where the contents within are near or entirely functional but subject to change at a later date, after further iteration.

Economic Impact

Justifying the statistics behind COVID-19 and the precautions taken to prevent the spread cannot be accomplished without considering how the pandemic has altered United States fiscal policy and discussing some of the proposed solutions to encourage economic recovery. From increased unemployment rates, to losses in tax revenue, to government funding and newly realized vaccines, these are some of the most pertinent factors in addressing and resolving the pandemic from an economic perspective.

In early 2020, the United States' economy came to a screeching halt. In an article on the National Tax Journal, Tracy Gordon, Lucy Dadyan and Kim Rueben released a report in September 2020, detailing some of the losses caused by the COVID-19 pandemic at that point in time. Between February 2020 to April 2020, the unemployment rate rose from single digit to double digit values, hitting economies that relied upon tourism the hardest. (Gordon et al.). One such example being Nevada, home of the tourist attraction Las Vegas, seeing unemployment shift from 3.6% to 30.1% within the same two-month interval (Gordon et al.). The hospitality industry was hit hard as well. As many as twenty-three states including Washington D.C. were reported to have "lost more than half of their leisure and hospitality jobs between February and April 2020" (Gordon et al.). Drastic fiscal policy changes were needed at this time to sustain the US economy throughout the early stages of the pandemic.

In addition to the need for fiscal policy change due to unemployment, state and local governments were experiencing another funding issue; a lack of tax revenue. The first of note was a decrease in revenue from sales tax. Due to delayed tax filing, payment deadlines, waived penalties and interest, collected sales taxes decreased “by more than $6 billion, or 21 percent, in May 2020 compared to one year earlier” (Gordon et al.). In twenty-seven states revenue estimators predicted losses of $114 in tax revenue through end of fiscal year 2021. Altogether, the tax revenues collected were estimated to be unable to offset the losses in taxable income from increased unemployment (Gordon et al.).

In light of record unemployment and significant losses in tax revenue at the state and local level, federal intervention was deemed necessary. Three pieces of legislation, *The Coronavirus Preparedness and Response Supplemental Appropriations Act* (Public Law 116-123), *Families First Coronavirus Response Act* (PL. 116-127), and *The Coronavirus Aid, Relief, and Economic Security Act* ("CARES," PL. 116-136) infused $250 billion worth of capital into state and local governments and as stimulus checks in response to the economic effects of the pandemic (Gordon et al.). On March 11, 2021, President Joe Biden signed a $1.9 trillion dollar stimulus bill into law, titled the *American Rescue Plan Act,* infusing additional capital into local and state governments and issuing a third round of stimulus checks (Smartasset). Ideally, these stimulus packages will address the worst aspects of the pandemic as we transition our economy back to normal as vaccination for COVID-19 is full steam ahead. As of April 2021, there are three vaccines available against COVID-19 from Pfizer-BioNTech, Moderna and Johnson & Johnson/Janssen with anyone over the age of eighteen now eligible for the shot (CDC).

In their report, the *USC Center for Risk and Economic Analysis of Terrorism Events* projected $4.8 trillion or 23% in net GDP losses, from March 2020 through the end of February 2022 (Gibson). These estimates were taken into account under a worst-possible case scenario for the United States economy during the COVID-19 pandemic. Two stimulus packages and a few readily available vaccines later, it appears worst of the pandemic is over and the United States economy can begin to recover. This can only continue to happen if as many people as possible can easily understand and follow the reasoning behind the preventative practices used against COVID-19, the primary goal of this project.

Game Engine

Unity is a game engine and Integrated Development Environment (IDE) designed in the C and C++ programming languages. It is used to create interactive forms of media typically reserved for video games, with software developed with Unity releasing on desktop and mobile platforms including Mac, PC, Android and iOS. Unity was decided as the choice for this project seeing as other languages used to handle data, such as Python, would not provide the visual components necessary to convey the statistics of COVID-19 in layman's terms. Unity uses files organized by type, with Figure 2.1 displaying the file types used, such as assets, scenes, materials, models, prefabs and scripts, which are written in the C# language.

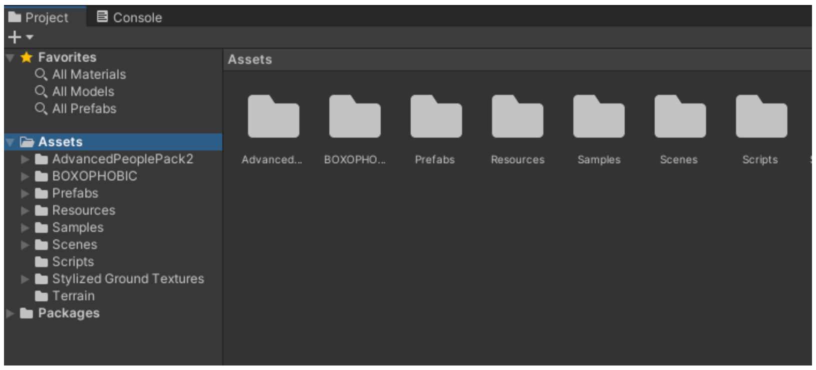
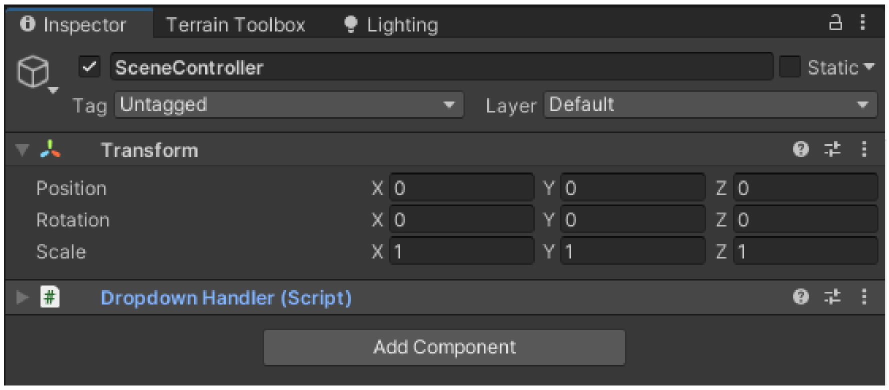


Figure 2.1. Project window.

These file types are then organized into groups called “components” used to generate and manage the interactive assets, from objects in motion to the setting displayed in the background. These components are controlled through scripts in C# by extending *Monobehaviour* which is the base class that enables compatibility with the Unity engine. Figure 2.2 shows the Transform user interface and the Dropdown Handler script, both of which are components.



SceneController
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Dropdown Handler (Script)

Figure 2.2. Inspector showing the components of SceneController.

The first component used in this project involves a script that converts JavaScript Object Notation (JSON) files containing COVID-19 data into information that can be interpreted by Unity to reflect what visuals are displayed on screen. The converted data is then used by one of two dropdown menus with all fifty US states as separate selections. The next component manages the instantiation and destruction of the game objects used to depict real persons affected by COVID-19 and the various outcomes by state, on a varying scale, e.g., 1:10; 1:100; 1:1000.

Interactive Game Objects

Unity is primarily designed around the use, interaction and application of game objects. Game objects at their core embody the concepts of object-oriented programming with an added UI layer used to load components which involve scripts and other game objects. In Unity, all objects are treated as game objects. As described in their documentation, “GameObjects are the fundamental objects in Unity that represent characters, props and scenery. They do not accomplish much in themselves but they act as containers for Components, which implement the functionality.” (Unity). Unity also provides many built-in components for those without prior programming knowledge to use, and to enhance the information provided for those that do have prior experience. For this application, many of the components being used will be added through scripting in C#, while some aspects were managed through the in-engine components provided by Unity, such as the size and font of the dropdown menus and the on-screen display, to the mesh and light components used in the game objects representing the cumulative COVID-19 totals by month and per state.

COVID-19 Data

The COVID-19 data provided by the Center for Systems Science and Engineering at Johns Hopkins University comes in comma-separated value format (CSV), a text file that separates entries using commas and is commonly used with Microsoft Excel. In order to use this information in my project with Unity, I used an external website to rapidly convert the files from CSV format to JavaScript Object Notation (JSON). Once converted, copies of the remaining JSON files are stored within the Unity Project folder to be accessed as needed. The current converted format of the JSON files contain columns listing all 50 US states and the total confirmed and deceased cases of COVID-19 by month, omitting all other column data from the original CSV files that had incomplete or missing values. The method by which Unity interprets JSON file data would have required variables to be generated for each additional column even if that data was not used in the project. Converting directly from CSV to JSON while omitting incomplete data columns helped to save significant time developing and testing the code for this portion of the project.

Loading a JSON file into Unity is as simple as declaring a public TextAsset variable in a C# script with Unity's FromJson method, and then dragging and dropping the JSON file from the Unity Project folder onto the Inspector window of an object that contains the script as a component. For this project I needed access to not one single JSON file, but one out of ten as needed. As a result, I implemented a C# script named JsonFileReader using a tutorial on serialization in Unity with one method, LoadJsonAsResource(string) in Figure 3.1.

```
public string LoadJsonAsResource(string path) //loads the .json file using its full name
from the Resources folder.
    {
        string jsonFilePath = path.Replace(".json", ""); //TextAsset accepts json
filetypes without the file designation (.json)
        TextAsset loadedJsonFile = Resources.Load<TextAsset>(jsonFilePath);
        return loadedJsonFile.text;
    }
```

Figure 3.1 LoadJsonAsResource() method of the Serializable class JsonFileReader.

LoadJsonAsResource(string) accepts a string which will be the file path of the JSON file and removes the file designation ".json", otherwise the method to load a JSON file as a TextAsset provided by Unity will not function properly. This method is used in combination with LoadDropdownOnValueChanged(int) which takes the position of the month selected in the first of two dropdown menus, chooses the corresponding file path of COVID-19 data located in a JSON file containing file paths, and stores the data in a data structure. This dynamic allocation of JSON file data was the first step in representing COVID-19 in a visual format, providing the values for instantiating the objects displayed on screen.

Dropdown Menu

The dropdown menus guide the main behavior of the program, with dynamically loaded dropdown options, data stored in a data structure known as a Dictionary, and objects that are instantiated and destroyed upon change of selection within the dropdown menu. The first dropdown menu contains a list of ten available months, starting from April and ending with January. These months range from the year 2020 through 2021 and are based on the dates of the original COVID-19 files obtained in CSV format. The first dropdown's options are dynamically inserted into the dropdown menu, using the DateTime struct and a loop that is ran before the application starts (Microsoft). All further behavior in this application relies on the functionality of the selection in the first dropdown containing these dates. Figure 4.1 and 4.2 show the first and second dropdown menu.

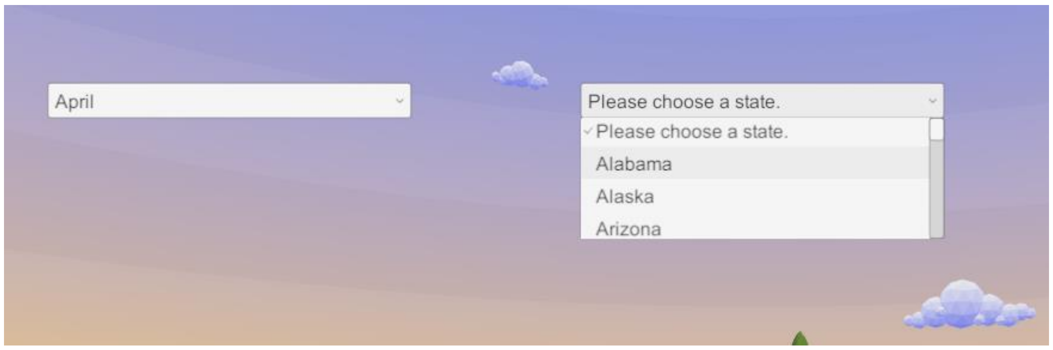


Figure 4.1. First dropdown menu with second dropdown disabled.

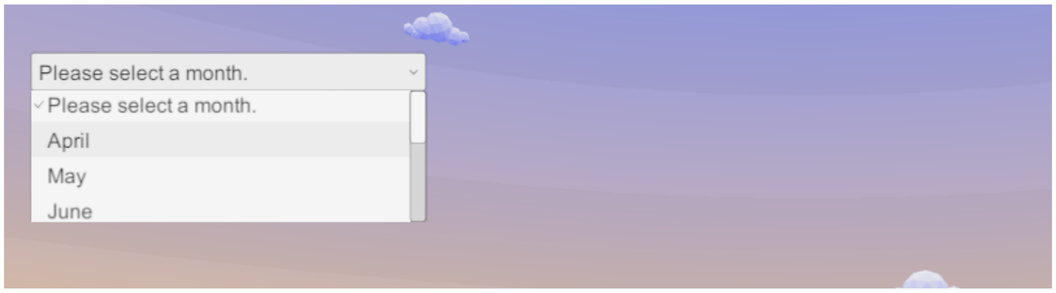
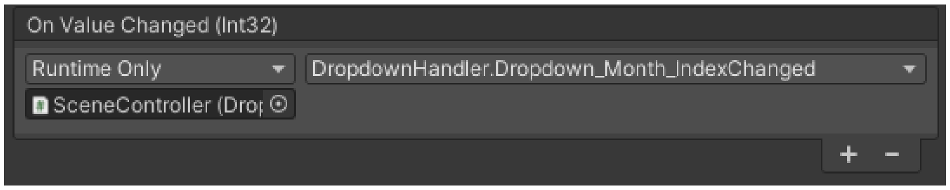


Figure 4.2. First and second dropdown menus enabled.

Unity's Inspector is a user interface that shows the visible attributes of a selected object prior to or during runtime (Unity). When selecting a dropdown object in Unity, the Inspector provides OnValueChanged(integer) which allows the developer to call an assigned method that takes the current index in the dropdown as an integer and executes a set of instructions each time the selected option in the dropdown menu changes. The first and second dropdown menus perform this function by calling the methods Dropdown_Month_IndexChanged(integer) and Dropdown_State_IndexChanged(integer) for purposes of this project. The inspector view of OnValueChanged() and code snippets are located in Figures 5.1, 5.2, and 5.3.



On Value Changed (Int32)

Runtime Only — DropdownHandler.Dropdown_Month_IndexChanged

SceneController (Drop

Figure 5.1. OnValueChanged() listed in Unity's Inspector.

```
public void Dropdown_Month_IndexChanged(int location) //When dropdown_months selection
changes, if index > 0, generate the states list with JSON data and load the states into
the dropdown menu dynamically.
    {
        monthMenuIndex = location;
        SetDropdownActive();
        LoadDropdownOnValueChanged(monthMenuIndex);

        State currentState = new State();

        int adjustedIndex = stateMenuIndex - 1;
        int monthIndex = monthMenuIndex - 1;
        if(adjustedIndex >= 0 && monthIndex >= 0)
        {
            currentState = statesPool[adjustedIndex];
            SetOnScreenDisplay(currentState);
            ClearSpawnLists();
            BuildSpawnListsAndGrid();
            AdjustCameraPos();
        }
        else
        {
            ClearSpawnLists();
            osd_date.SetActive(false);
            osd_stats.SetActive(false);
        }
       }
```

Figure 5.2. OnValueChanged() method for first dropdown menu.

```
public void Dropdown_State_IndexChanged(int index) //when the dropdown of the list of
states changes, update the accessible values for other classes to access
    {
        /** To-Do List:
         * 1. Cast index to inner variable (int indexDropdown)
         * 2. Get State object fields from statesPool using index (int indexDropdown)
         * 3. Update OSD (textbox)
         * 4. Instantiate pools of objects for statistical representation
         * 5. Track objects instantiated
         * 6. Instantiate or Destroy if objects are lesser or greater than new State
object fields.
         **/
        stateMenuIndex = index;
        int adjustedIndex = stateMenuIndex - 1;
        State activeState = new State();
        if(adjustedIndex >= 0)
        {
            activeState = statesPool[adjustedIndex];
            SetOnScreenDisplay(activeState);
            ClearSpawnLists();
            AdjustCameraPos();
            BuildSpawnListsAndGrid();

        }
        else
        {
            ClearSpawnLists();
            osd_stats.SetActive(false);
            osd_date.SetActive(false);
        }
       }
```

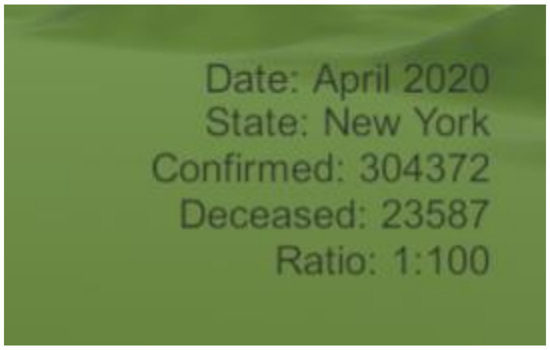
Figure 5.3. OnValueChanged() method for second dropdown menu.

Dropdown_Month_IndexChanged(integer) manages the display of the second dropdown menu, rendering it inactive if the first position of the index is selected or active if an index greater than the first position is selected. At the same time an index greater than the first position in the first menu is selected, the COVID-19 JSON file data is loaded into a Dictionary, corresponding to the month selected under the first dropdown menu. A Dictionary is a data structure that contains "key and value pairs" used for efficient retrieval, making its use ideal for quickly accessing the data needed to represent the current selection based on the two dropdown menus displayed (Microsoft). One caveat of using a Dictionary in Unity is that Dictionaries are not serializable, meaning Unity nor its components such as the Inspector can see what is inside the data structure, requiring the key pair values to be inserted into an object viewable by the game engine. Dropdown_State_IndexChanged(int) handles this by using a dictionary of pairs of ints and State class objects. The current selected index is used as an int to retrieve the object State containing the COVID-19 statistics and assigns it to a new State object variable that can be serialized or accessed by Unity.

Both of the methods, Dropdown_Month_IndexChanged(int) and Dropdown_State_IndexChanged(int) are involved in instantiation and destruction of objects on screen. If the index selected in either dropdown is greater than zero, the objects representing the statistics on screen are destroyed and replaced with an updated number of objects instantiated on screen, according to the newly loaded State object and/or dictionary. If the index selected is the first position in either dropdown menu, all objects representing statistics are destroyed.

On-Screen Display (OSD)

Prior to becoming accessible within Unity, the JSON file data was curated, removing column data that contained incomplete or missing data, and to meet time constraints for the project. What remained were the values of total confirmed cases and deaths by US state. The on-screen display shows this data dynamically, updating when the selection in the dropdown menu changes, and is rendered inactive when all currently active dropdowns have at least one option selected with an index of zero. Figure 6.1 shows an example of the on-screen display when set to active.



Date: April 2020
State: New York
Confirmed: 304372
Deceased: 23587
Ratio: 1:100

Figure 6.1. On-Screen Display.

Instantiation & Destruction

Visualizing the magnitude of COVID-19 would not be possible without the key aspect of instantiation. Instantiation is the most crucial and costly operation in Unity, cloning an object and returning the cloned object for use on or off screen. When cloning a GameObject, its position and rotation are given, and the object is then placed into the scene and automatically set to active. In an effort to streamline instantiation, Unity allows for the creation of Prefabs, or the blueprint of a game object that contains, “all its components, property values, and child game objects as a reusable asset” (Unity). Some of the reused components in this application use a prefab for the sake of convenience. For example, a “cube object” is stored as a prefab to represent the total confirmed cases and deaths in the virtual room in one version of the project.

The objects representing the real people affected by COVID-19 needed to be in a presentable format while still resembling a crowd. Random instantiation of the currently selected cumulative total was the initial intent, but required too many resources from the engine. The number of objects instantiated on screen would not match the numbers they were intended to represent within a reasonable margin of error of five percent. To resolve this, I settled upon a mixed approach; while the total confirmed cases were fixed in position and instantiated in the format of a grid, the death totals could be randomly instantiated without significant room for error. To accomplish this, BuildSpawnListsAndGrid() calculates the boundaries of the grid treating the number of confirmed cases as an area, finds the square root to determine the perimeter of one side of the grid, and then uses those dimensions to instantiate all of the confirmed cases as game objects while adding the game object’s positions to a list. A loop is then run over the list, randomly selecting a cube to change its mesh and therefore its color from blue to red, attempting up to twenty times if the same game object at that position has already been selected, up until the total number of deaths is reached.

For the scope of this project, a one-to-one ratio of statistics represented on screen was not possible for some states, with New York and California's cumulative totals exceeding 1.4 and 3 million respectively. After roughly 300,000 instantiated cubes used as models for the statistics, the application would cease to render everything on screen and begin to lag when switching from state to state, increasing in the time necessary to load beyond practical means, taking as long as five minutes to load New York in January 2021 if a crash did not occur. Fortunately, by using a one-to-ten ratio, all objects on screen for each state can be rendered in a reasonable time of 30 seconds or less on my current home desktop.

Conclusion

From bug fixes to rewrites, a substantial effort was undertaken to reach the outlined project constraints, including expanding those constraints to add additional features within the allotted time. The initial project contained a single JSON file of the fifty US states with its data accessible via dropdown, but only through a for-each loop that was repeated every time the index of the dropdown was changed. The dropdown itself did not use the integrated OnValueChanged(integer) option available in the inspector, but a listener method that was constantly looking for a change in the dropdown selection, a seemingly more costly operation after testing. A majority of the issues were encountered during the second rewrite which is the most similar to the code in the application today.

Two prominent bugs in my second rewrite involved the loading of JSON data through LoadJson() and an error with a dictionary value not being returned. Unity is constantly in flux, and while many of the old concepts and tutorials still apply, there are some conflicts when working with the C# language that only an intermediate to experienced developer would know about. I had received an error with Unity's LoadJson() method which assists in converting JSON file data into text, where the State object containing the fields for State, total confirmed and deceased cases was not being loaded with values, with no data contained in any dictionary or the second dropdown menu. Hours later, a comment I discovered on Unity's forums from an independent developer explained what I had not been able to figure out; Unity cannot serialize a loaded Json file that uses a constructor with properties, also known as "getters and setters". Once the properties were removed, the data was being successfully stored into the fields of the State object. The second issue with the Dictionary returned a "KeyNotFoundException" where the index of the Dictionary containing the key pair values of integers and State objects containing COVID-19 data kept growing ad-infinitum upon additional dropdown selections. While the issue persisted over two days and several hours of work, the fix was simple, the Dictionary was not being cleared of previous key value pairs before new ones were added. Now, the Dictionary.Clear() method for Dictionary is called prior to loading in new JSON data.

The limit on instantiating objects was not a bug, but an issue that I did not want to acknowledge until I was face-to-face with it. The intended functionality of the application is visual, but there is next to no functionality if the application can no longer run properly or comes to a screeching halt if the objects on screen exceed a limit on the engine's ability to render them. In order to achieve more efficiency with instantiation and maximum objects rendered, a third rewrite is necessary in order to incorporate the features provided by Unity's new design implementation, Data Oriented Technology Stack (DOTS) (Unity).

However, given realistic time constraints and the release state of Unity's DOTS which is in beta, this is an approach best saved for future projects.

At the last minute, I was able to add an additional level of polish and accomplish integration of asset store assets, replacing the scenery and the "cube objects" representing total confirmed cases and deaths, with the before and after shown in Figures 7.1 and 7.2 respectively.

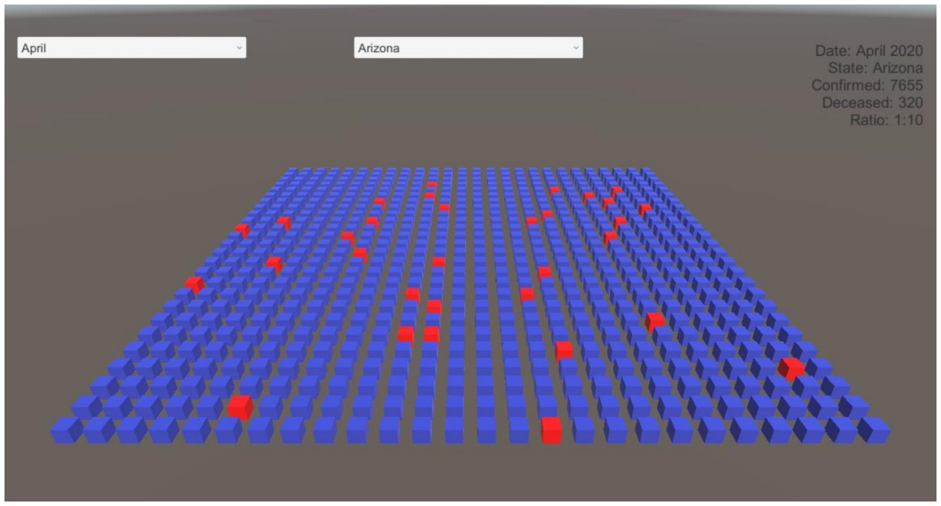


Figure 7.1. Virtual Room (before)

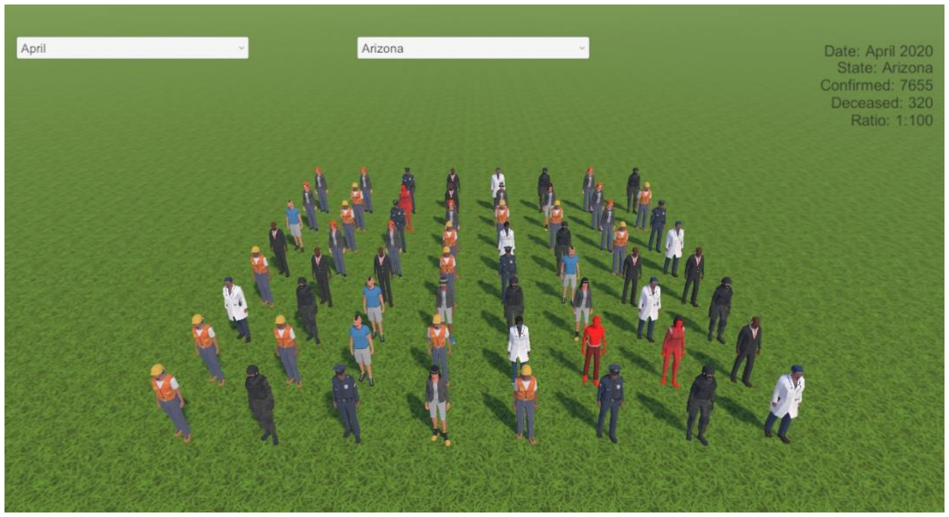


Figure 7.2. Virtual Room (after)

To improve upon this project even further, the updated character models would be first priority. The variance in outfits would be removed, allowing for increased visibility of death statistics. Ideally, the before and after examples of the application would be accessible through the same version of the application, to allow for user choice between graphics and performance. I would also consider online functionality to provide live updates to the repository of COVID-19 that would be reflected in the dropdown menus. The visual representation of COVID-19 through a virtual room fulfilled the scope of the project constraints as a demo or vertical slice. The application is not a ready-to-ship product, but serves as a first-look into visualizing the numbers behind COVID-19 and the people who have been negatively impacted by the virus.

References

1. CSSEGISandData. *CSSEGISandData/COVID-19*. 2020. 2020. *GitHub*, https://github.com/CSSEGISandData/COVID-19.
2. “Governor Cuomo Signs the ‘New York State on PAUSE’ Executive Order.” *Governor Andrew M. Cuomo*, 20 Mar. 2020, https://www.governor.ny.gov/news/governor-cuomo-signs-new-york-state-pause-executive-order.
3. CNN, Sheena Jones and Christina Maxouris. “New York Officials Traced More than 50 Coronavirus Cases Back to One Attorney.” *CNN*, https://www.cnn.com/2020/03/11/us/new-rochelle-attorney-containment-area/index.html. Accessed 26 Oct. 2020.
4. Whitrow, G. J., and British Broadcasting Corporation. Third Programme. *Einstein, the Man and His Achievement*. New York, Dover Publications, 1973. *Internet Archive*, http://archive.org/details/einsteinmanhisac00whit.
5. Technologies, Unity. *Unity Real-Time Development Platform | 3D, 2D VR & AR Engine*. https://unity.com/. Accessed 29 Sept. 2020.
6. *Unity - Manual: GameObject*. https://docs.unity3d.com/560/Documentation/Manual/class-GameObject.html. Accessed 26 Oct. 2020.
7. *CSV To JSON Converter*. https://www.convertcsv.com/csv-to-json.htm#inputfile. Accessed 6 Oct. 2020.
8. “How to Read .Json File.” *Unity Forum*, https://forum.unity.com/threads/how-to-read-json-file.401306/. Accessed 6 Oct. 2020.
9. *Unity C# Creating and Deleting Objects*. 2019. *YouTube*, https://www.youtube.com/watch?v=XO-E6QaTniQ.
10. Microsoft. “Dictionary<TKey,TValue> Class (System.Collections.Generic).” Accessed March 30, 2021. https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2.
11. Microsoft. “DateTime Struct (System).” Accessed March 30, 2021. https://docs.microsoft.com/en-us/dotnet/api/system.datetime.

12. “DOTS - Unity’s New Multithreaded Data-Oriented Technology Stack.” Accessed March 30, 2021. https://unity.com/dots.
13. Stack Overflow. “Generics - C# Dictionary with Two Values per Key?” Accessed November 4, 2020. https://stackoverflow.com/questions/1500835/c-sharp-dictionary-with-two-values-per-key.
14. Matt MirrorFish. *Procedural Generation Basics: Spawn A Grid In C# [Unity Tutorial]*, 2019. https://www.youtube.com/watch?v=DRWE_6GoNuI.
15. CodingWithRus. *Procedural Placement : Beginners Guide EP 3 - Unity3D*, 2020. https://www.youtube.com/watch?v=vfl2HyEar68&list=PLu2uAkIZ4shpPdCTIjEpvhD8U-RRM3Y2F&index=3.
16. Unity Forum. “’Random Is Is Ambiguous Reference between ‘Unity.Random’ and ‘System.Random.’” Accessed March 4, 2021. https://forum.unity.com/threads/random-is-is-ambiguous-reference-between-unity-random-and-system-random.916637/.
17. Snowdrama. *Reading Files - Unity JsonUtility Serialization Tutorial [Episode 1]*, 2017. https://www.youtube.com/watch?v=6F6HnBHVAtE&list=PLTERDPLzAj3qdy14XoAjpZr1gD44G6msl.
18. ChilledCow. *Spawning Random Objects without Collisions in Unity [RNDBITS-039]*, 2020. https://www.youtube.com/watch?v=ENEtzLePZbQ.
19. “Unable to Delete Object from a List Using Foreach Loop - Unity Answers.” Accessed March 9, 2021. https://answers.unity.com/questions/1736086/unable-to-delete-object-from-a-list-using-foreach.html.
20. Jayanam. *Unity 5 UI Tutorial - Dropdown List*, 2016. https://www.youtube.com/watch?v=Q4NYCSIOamY.
21. Unity Forum. “C# Properties Not Getting Serialized Properly?” Accessed March 30, 2021. https://forum.unity.com/threads/c-properties-not-getting-serialized-properly.266239/.
22. Gibson, Kate*. Pandemic’s Toll to Economy Nearly $5 Trillion over Two Years*. https://www.cbsnews.com/news/pandemics-toll-on-u-s-economy-up-to-4-8-trillion-over-two-years/. Accessed 11 May 2021.

23. "American Rescue Plan: Inside Biden's $1.9 Trillion Stimulus." *SmartAsset*, 11 Mar. 2021, https://smartasset.com/financial-advisor/american-rescue-plan-biden-third-stimulus.

24. Gordon, Tracy, et al. "STATE AND LOCAL GOVERNMENT FINANCES IN THE COVID-19 ERA." *National Tax Journal*, vol. 73, no. 3, National Tax Association, Sept. 2020, pp. 733–58. *go-gale-com.ezproxy.purchase.edu*, doi:10.17310/ntj.2020.3.05.

25. CDC. "Different COVID-19 Vaccines." *Centers for Disease Control and Prevention*, 23 Apr. 2021, https://www.cdc.gov/coronavirus/2019-ncov/vaccines/different-vaccines.html.