Weekly breakdown - Week 6

Weekly objective: Students will understand how to distinguish what objects the user is clicking on, add more complexity to their prefabs and open up more options to the user through the UI.

Goals:
Students will be able to:

- Create their own physics layers
- Use physics layers to distinguish what user is clicking on
- Create cleaner hierarchies for better navigation
- Edit data sent back to UI for better usability
- Organize UI elements into clear delineations
- Upgrade their prefabs allowing more control to user
- Understand 3D manipulations of rotations and translations
- Make use of parenting to simplify math of 3D manipulations
- Understand more of standard shader, emission and transparency

Lecture Topics:

- Physics layers to seperate click objects and distinguish different behaviors
- Navigation habits for moving through deep hierarchies
- Repositioning UI elements in consideration of UI layouts
- Visual feedback to user, ie color changes
- Naming conventions for components and their data members, ie text.text
- Converting floats from 0-1 to 0-100
- Using float.ToString method to truncate decimal length
- Grouping UI elements for easier organization
- Updating color and timer text elements to inform user
- Concatenating dynamic and static strings together for UI elements
- Switching from CreatePrimitive to Instantiate for custom prefabs
- Creating new material for the custom prefabs to handle random colors, transparencies, and emissions
- Public gameobject variables to reference external prefab files
- Ensure prefabs have zero position to ensure no offset from spawning location
- Adding physics layer on clock to prevent deleting those parts
- Adding physics layers on prefabs to identify that they are valid raycast targets
- Creating user-generated physics layers
- Checking physics layers during raycasts and user input events
- Rebuilding prefabs with user generated materials
- Using transparency rendering mode on standard shader
- Relationship of the fourth float in color data structure, alpha, to transparency
- Reusing initial color information to build the emission color for a balanced material
- Updating prefabs and ensuring the external file representation gets updated

- Ensuring prefabs have position zero, custom physics layer, and new material
- Seperating functioning on left clicks and left drags
- Relationships of colliders and layers for raycasting checks
- GameObject variables as references to external prefabs and instantiated game objects
- Color variables to hold our initialized random colors and emission colors
- Using a float EmissionStrength to build emission color and allow user to change it in UI
- Rotating sun light source through inspector slider and within UI slider
- Parenting game objects to simplify rotation and translation math for user input
- Figure out easiest way to rotate sun 360 degrees connected to the 24 hours from the clock
- Using eulerAngles of Quaterions for easier rotation handling
- Rotation vs localRotation when objects are parented
- How to connect the system time to the daylight rotation
- Checking real time connection to light rotation and faked time for testing
- Setting up 3D model parts of the clock to be clickable parts of the UI
- Using the rotations of clock parts to figure out the time of day they should relate to
- Passing information between two scripts
- How to update the hour hand of the clock when time of day is changed by user
- How to setup the AM/PM button to adjust time of day by 12 hours
- Removing unused variables and methods, casting variables
- Finishing the clock script

Assignment: Update control of emissive color, and transparency of prefabs to UI, finalize clock parts to act as UI elements to control daylight system