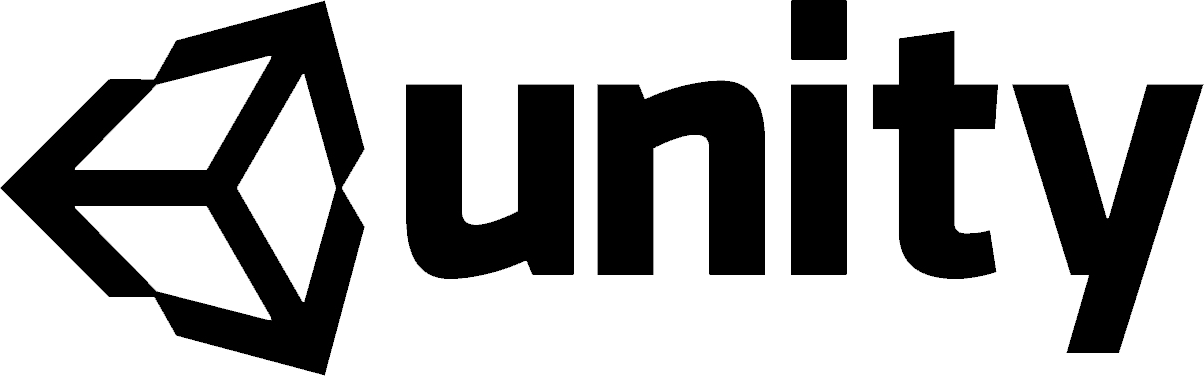
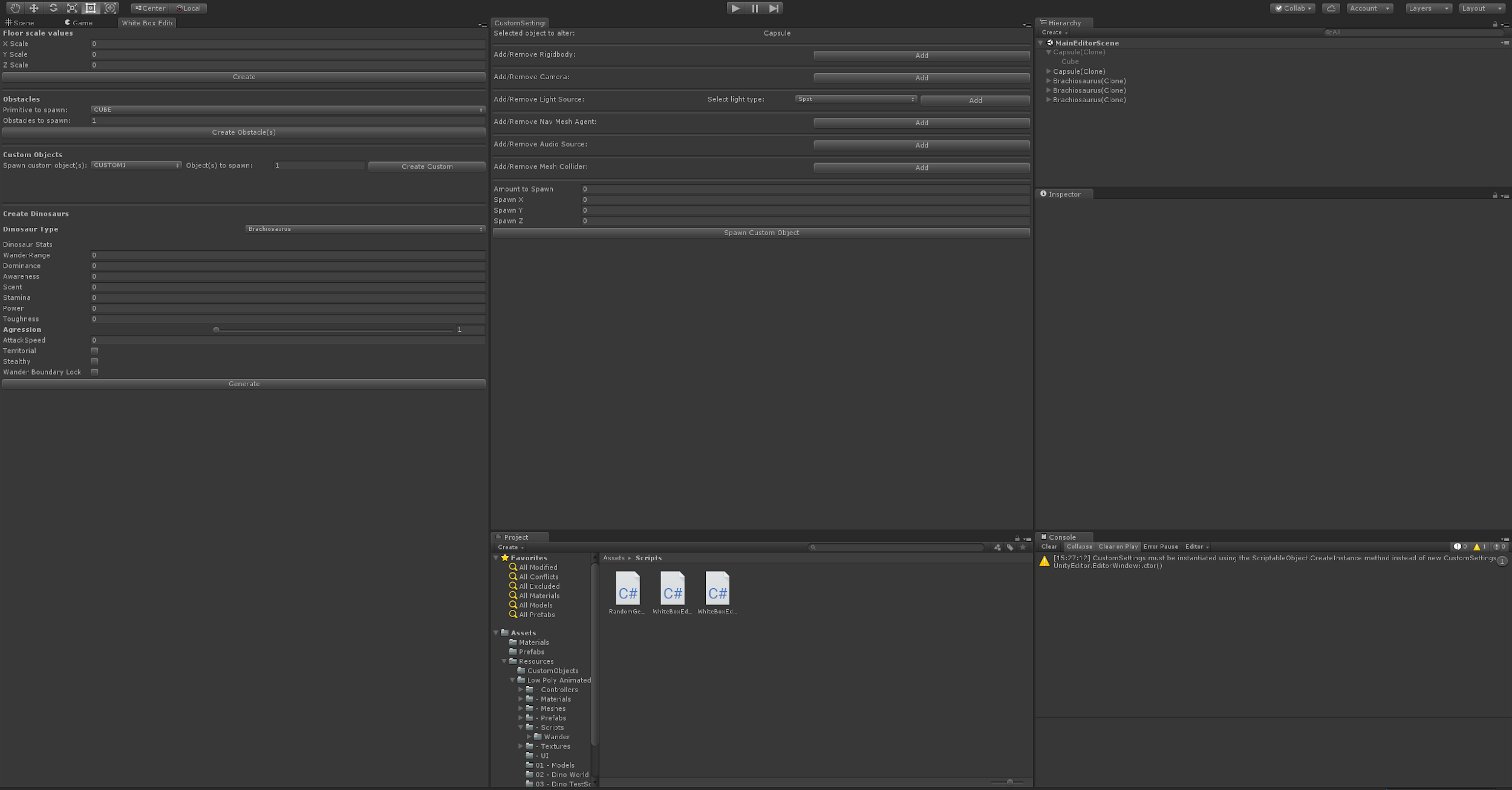
****

**White Box Editor Tool**

**Created by: Jake P, Mitch H, & Ryan V**

* **Description**: This is a Unity 2018 editor tool for designers and programmers. This tool is designed to be a quick, whitebox setup of a Unity scene; as well as instantiating primitive and custom objects directly from the Assets folder.

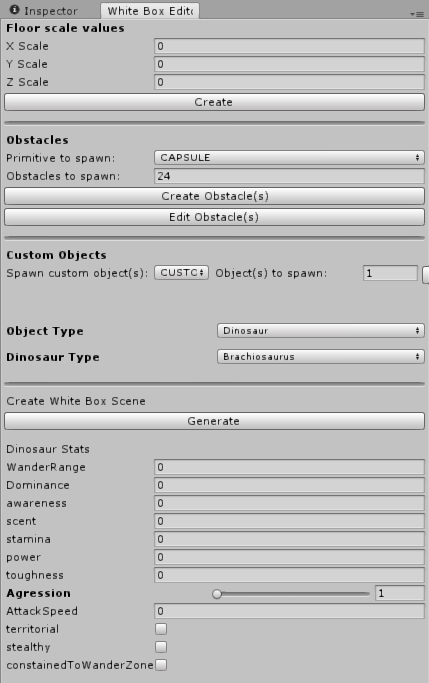
* **With this tool you’ll be able to:** 
  + Create a floor for to spawn your objects on
  + Spawn Unity primitive gameObjects (Cube, Cylinder, Capsule, and Sphere) as placeholders
  + Spawn custom Prefabs
  + Edit and customize the spawning objects
  + Spawn fully functional Low Poly Dinosaurs
* **How it’s made (Script is broken down by the Regions that they appear)**



* **Set up**

1. The first step in creating the tool is to create an empty project
2. Once in the empty scene, two folders were created: CustomObjects & Scripts
   1. Just to house the assets and scripts
3. Inside of the Scripts folder, the script “WhiteBoxEditorWindow” was made
4. First we had to reference the UnityEditor library
5. Then set the class to inherit from EditorWindow rather than Monobehaviour
6. Now that we are able to use all of Unity’s editor functionalities, we set the editor window to be opened as a menu item
7. Also we set the window’s minimum pixel size, in order to ensure the layout’s entegrity
8. Finally used Unity’s “Show()” function to open the window once it’s selected in the window menu

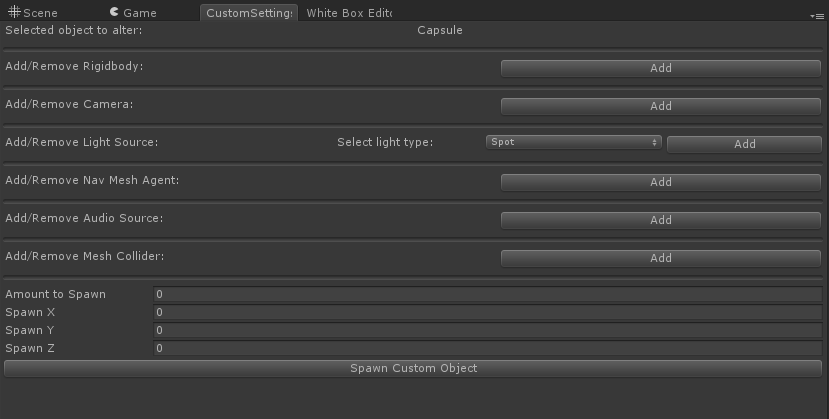
* **First Window**



* + **Floor Spawning**

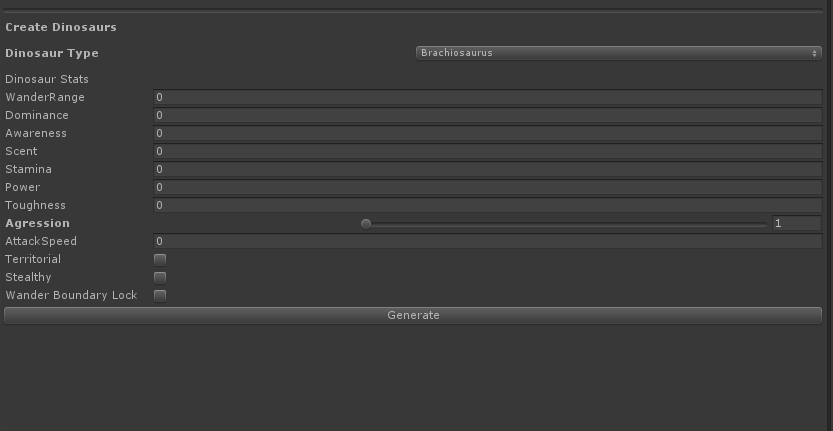
1. Open the script called “WhiteBoxEditorWindow”
2. Then we have a function to spawn a Cube primitive to act as a white boxed floor
   1. The function takes in three variable parameters
      1. One for the X scale of the cube
      2. One for the Y scale of the cube
      3. One for the Z scale of the cube
3. Once the three variables have been set, there’s a button labeled “Create”
4. The button then takes in the three float fields and applies them to the transform of the cube object
5. The cube is always spawned at (0, 0, 0) world location
   * **Primitive Spawning**
6. First, there’s an Enum Popup which correlates to which of the four (Cube, Sphere, Capsule, and Cylinder) primitives to spawn as obstacles
7. There’s also an Int field which dictates how many of the selected primitive gameObject(s) are to be spawned
8. Once all the fields have been filled out, the user is able to spawn the primitive object at (0,0,0) with the properties that it was given
   * **Setting Custom Object**
9. First, getting the custom objects, by using the Resources.LoadAll() function
10. The five custom prefabs are to be placed in the “Resources” folder (Assets/Resources). They will be gathered by the Resources.LoadAll() function finding all of the prefabs at the resources data path
11. Then by using the Custom Objects Enum Popup, the custom object will be selected by a Switch function that works with the Custom Objects enumeration
12. Once the custom object is selected, the user is given the option to vary the amount of prefabs that are spawned
13. By using the “Create Obstacle(s)” button, the user is able to spawn the selected amount of custom objects in scene
14. By using the “Edit Obstacle(s)” button, the user is able to open a second editor window that allows the user to customize their selected prefab further before instantiating it/them

* **Second Window (Edit Object Window)**

****

1. Like previously stated, to access this window you must use the “Create Custom” button
2. When the window is opened, you are greeted with the name of the selected object you are editing
3. This window is laid out in a list format, with buttons to add or remove specified components
4. Components:
   1. Rigidbody
   2. Camera
   3. Light source -
      1. With an Enumeration drop down menu to select the light type (Spot, Directional, Point, & Area)
   4. Navmesh Agent
   5. Audio Source
5. Once the user has edited the custom object as much as they’d like, they’re able to alter the amount of objects to spawn and their spawning coordinates

* **Dinosaurs**

****

1. Needs to be placed on a Navmesh
2. Needs the Animal Manager in the scene
   * Types of Dinosaurs
     + Brachiosaurus
     + Pteranodon
     + Stegosaurus\_Green
     + Stegosaurus\_Red
     + T-Rex
     + Triceratops
     + Velociraptor
   * Stats for dinosaurs
     + Wander Zone
       - How far away from its origin this animal will wander by itself
     + Dominance
       - How dominant this animal is in the food chain, aggressive animals will attack less dominant animals.
     + Awareness
       - How far this animal can sense a predator.
     + Scent
       - How far this animal can sense its prey.
     + Stamina
       - How many seconds this animal can run for before it gets tired.
     + Power
       - How much this damage this animal does to another animal.
     + Toughness
       - How much health this animal has.
     + Aggression
       - Chance of this animal attacking another animal.
     + Attack Speed
       - How quickly the animal does damage to another animal
     + Territorial
       - If true, this animal will attack other animals of the same species.
     + Stealthy
       - Stealthy animals can't be detected by other animals.
     + Constrained to Wander Zone
       - If true, this animal will never leave it's zone, even if it's chasing or running away from another animal.

* Marketing
  + This project will be an open source tool that other new Unity developers will have access to
  + The White Box Editor Tool will be available on the Unity Assets store for **Free**
  + By having this tool be free, we hope it encourages other developers to add on and make this tool more appropriate for their own personal use
  + Also we hope this helps new developers get a jump start to some of their first projects as well as teaching them a little bit about the Unity Editor functionality