XR App 1

Your first XR app is an early prototype of what could become a 3D design app in VR. You will practice implementing basic XR interactions and add some key functionality to your app, as described below.

Environment (2pts)

Your environment should use a different skybox and floor than the ones used in the template project posted on Canvas. The more customizations you make the better. At the very least, your environment should resemble the one shown in the demo, with the platform/table and shelf included. All interactable objects should be placed on the shelf (you need to use at least four different shapes from the VR Interactable Objects package provided on Canvas). The idea is that users will pickup the objects from the shelf and place them on the platform/table.

Navigations and Controllers (2pts)

The right controller will be used for direct interaction only and shouldn't have any teleportation capabilities. The left controller will be used for both teleportation and direct interaction. Therefore, users should be able to toggle the ray interactor by *touching* the joystick on the controller. When the ray is active, users should teleport to their visualized destination by moving the joystick forward.

The ray should display a reticle to visualize indicate the destination of the user.

When teleportation is not active on the left controller, it should act as a direct interactor.

Feedback (2pts)

Your app should provide carefully-designed feedback to increase users' sense of presence in the environment. For controllers, use appropriate Hover and Select feedback by playing appropriate sound clips and by applying vibrations to the controllers. For instance, vibrations for hovering should be less intense and shorter than those applied when users select an object.

All interactable objects should play a sound clip when they collider with anything, similar to what we did in class. They should all have a somewhat bouncy material so they look more realistic when they are dropped on to a solid object.

Interactions (9pts)

You will add a number of interactions to your interactable objects, as explained below:

1. **Highlight Object**
   - When selected (grabbed by the controller), objects should be highlighted by changing the color applied to their material. This custom highlight color should be modifiable through the Inspector.
   - When deselected (detached from the controller), objects' color should be changed back to the original.

2. **Clone Object**
   - Users should be able to clone a selected object by pressing the A button (primary button) on the right controller. This interaction should clone the current version of the object, not a pre-determined prefab.
   - After the cloning operation, the original object must remain intact.

3. **Destroy Object**
   - Users should be able to destroy a grabbed object by pressing the B button (secondary button) on the right controller.

- When destroying an object, a visual effect should be played (you can use the particle effect provided and modify it however you see fit), along with an appropriate sound clip.

Color Wand (10pts)

- A rather interesting aspect of your app will be the availability of a color wand, as demonstrated in the video. The color wand must be placed on the platform when the game begins. Users will be able to pick it up with either controller and color the interactable objects with it.
- For the color wand, no prefabs are provided. You can make one by using a thin cylinder and a sphere and turning them into a prefab. You should be comfortable with Unity's basic shapes, and this is your chance to do so.
- Users will be able to change the color of the tip by pressing the trigger button on either controller. The color will be a random one each time the trigger is pressed. An appropriate sound clip should be played when the color is changed.
- When the user touches an interactable object with the **tip** of the Color Wand, the interactable object's material color should be changed to that of the tip. Hence, the name Color Wand. 😊 The color of the object should change when it is touched by the tip only, not the any other part of the wand.
- Refer to the demo video to better understand how this is expected to work.

App Polish (5pts)

Part of your grade will come from how polished your app is. The polish of your app refers to the look and feel of the app and reflects how much effort you've put into making the best app possible given the requirements. For the most part, this is a subjective quality of your app; you'll know it when you see it. Here is the scale that will be used to assess the polish of your app:

Excellent: goes above and beyond the requirements to improve mechanics/interactions; provides aesthetically pleasing look; adds extra interactions/behaviors when applicable to make the app interesting/innovative

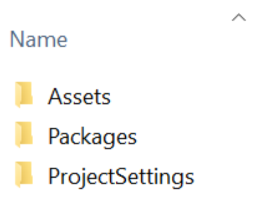
Satisfactory: meets the requirements; graphics look good but can be more polished; no extra effort to make the most interesting/innovative/different app; no extras; does a good job implementing the minimum

Half-baked: one or more requirements is missing; problematic mechanics/weird controls/behaviors; graphics/colors/assets don't look very good; no extras; doesn't convey an effort to produce the best app possible

Dull: bare minimums in all aspects; multiple problems with mechanics/graphics/assets. No effort to make the app look and feel good at all.

Submission Requirements

1. Submit an apk build of your prototype. We should be able to install and run it on our headsets. Your app name (which you can specify in Unity) should use the following naming convention: CS5097_XRA1_LastNameFirstName
2. You should also submit a zipped folder of your Unity Project (please delete redundant assets as they will increase file size). When zipping your Unity Project, include the following folders **only**:



Simply, go to your project folder (you can open it while in Unity). Then select these three folders and zip them. This is different from zipping the entire Unity folder (Unity files aren't included in this method, which will mean smaller files). Use the same naming convention for the folder.

3. **Also attach all your script files to your submission. The number of script files will depend on how you structure your game. All must be attached! Failing to do this will result in a grade of 0.**

Failing to meet submission requirements will result in up to 25% penalty above and beyond other point deductions.