Runtime Undo & Redo

Paint in 3D supports runtime undo & redo, but to save performance and memory it's not enabled by default.

To enable it, you need to follow these steps.

State Limits

The first step is to select your **P3dPaintableTexture** components, and set the **State Limit** setting to the value you want. A value of 10 means you can undo paint operations 10 times, and then redo them 10 times. If you paint 11 times with this setting then your initial paint state will be deleted, and you will only be able to undo 10 times to your first paint operation.

Store States

The next step is to find your painting components, and enable the **Store States** setting. This setting will automatically call the **Store State** method on each P3dPaintableTexture in your scene, letting Paint in 3D know that you're about to paint on them.

For example, the **P3dDragRaycast/Smooth** component has the **Store States** setting, as well as **P3dSprayCan**, and **P3dTapThrow**.

Buttons

Finally, you can add the **P3dUndoAll** and **P3dRedoAll** components to your UI buttons, and they will automatically be set up so when you click them they perform an undo or redo operation.

If you want to manually do this, then you can call the "**Undo All**" and "**Redo All**" methods/functions that are on these components. These methods are also available from the context menus if you want to test it in the editor.

And that's it, your game should now have Undo & Redo paint functionality!

Memory Usage

Keep in mind that each undo/redo state is stored as a full texture state. This means that if your texture is 1024x1024 using the RGBA32 format, then each undo/redo state will consume 1024x1024x4 bytes of memory, or 4 megabytes.

If your game requires a lot of undo states and you can't afford to use too much memory, then instead of using this texture state system you need to store the paint operations themselves so you can reconstruct the texture when you undo. This kind of system uses very little memory, but it's quite complex to setup and manage, so there is no implementation of it included in Paint in 3D.

Manual Undo & Redo

The undo & redo functionality listed above works well for most games, but it's not suitable for all scenarios. For example, if you have a multiplayer game where each user can paint their own object, then performing a global undo operation is unacceptable. Another problem is when using delayed painting features like the spray can, the actual paint may not have finished applying by the time you go to paint a second time, and so the save states may seem incorrect.

In these scenarios, you will likely need to manually handle undo & redo. To do this, you need to call the **P3dPaintableTexture.StoreState** method before you paint a specific texture, and then **P3dPaintableTexture.Undo** or **P3dPaintableTexture.Redo** when required. This works the same as the global methods, but because it's done on each individual paintable texture it gives you full control.