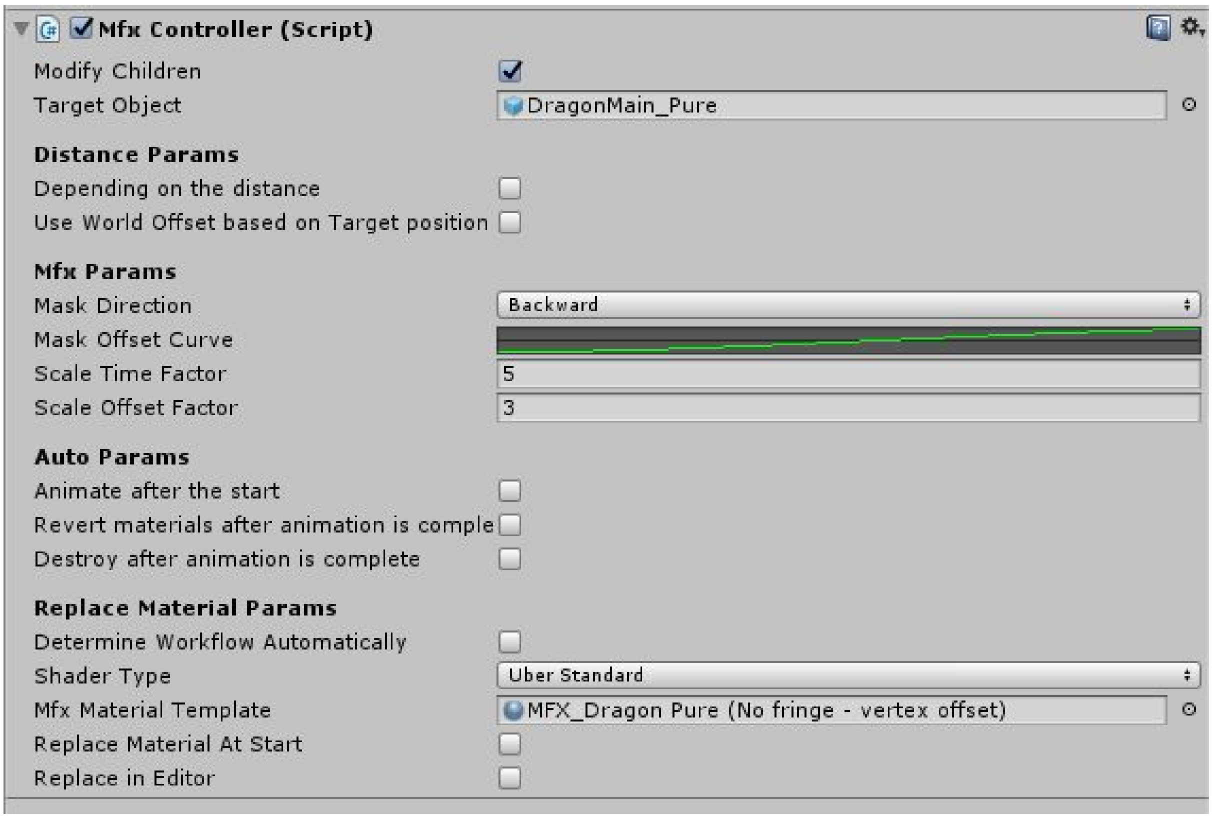

MfxController



Mfx Controller (Script)

Modify Children	✓
Target Object	DragonMain_Pure
Distance Params	
Depending on the distance	☐
Use World Offset based on Target position	☐
Mfx Params	
Mask Direction	Backward
Mask Offset Curve	
Scale Time Factor	5
Scale Offset Factor	3
Auto Params	
Animate after the start	☐
Revert materials after animation is comple	☐
Destroy after animation is complete	☐
Replace Material Params	
Determine Workflow Automatically	☐
Shader Type	Uber Standard
Mfx Material Template	MFX_Dragon Pure (No fringe - vertex offset)
Replace Material At Start	☐
Replace in Editor	☐

This script does all work.

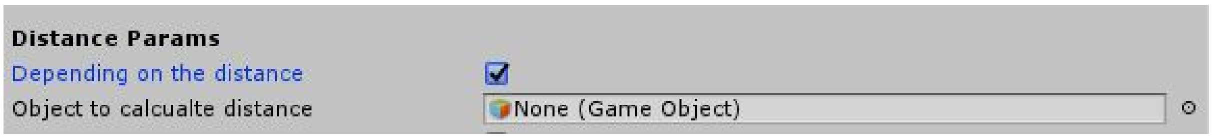
Main Params:

"Modify children" - if enabled, script will animate and replace materials of all children.

"Target object" - you can add MfxController script to "fake" object and specify target object, which should be materialized. If "Target" is not specified, the script will get the current gameObject.

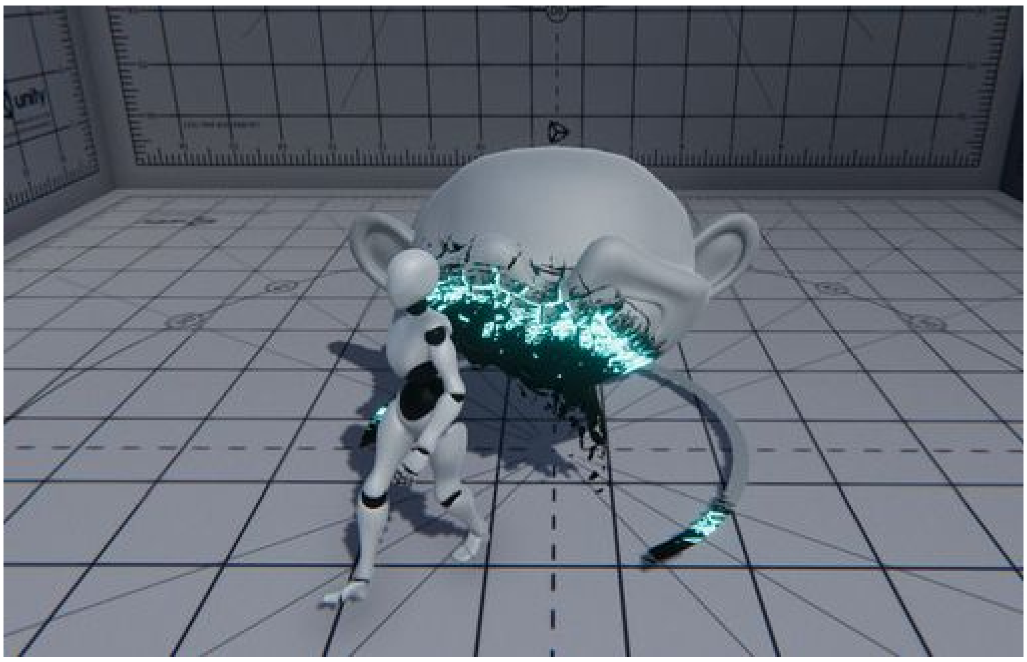
Distance Params:

"Depending on the distance" - if enabled, you need to specify the object to calculate distance so the shader will work in "Distance Based" mode. It will materialize or disintegrate the object based on the distance between objects.



Distance Params	
Depending on the distance	✓
Object to calcualte distance	None (Game Object)

It will look like this:



“Depending on the distance”: if enabled, the script will provide the position of “Target Object” in the world coordinates into the shader.

Mfx Params:

“Mask Direction” : Direction of the animation curve**:** you don’t need to change the curve if you want to change only the direction.

“Mask Offset Curve” : The animation Curve of the “Mask Offset” shader parameter: the object materializes and disintegrates depended on this parameter.

“Scale Time Factor” : the value to scale time.

“Scale Value Factor” : the value to scale value.

Auto Params:

“Animate after the start” : if enabled, the script will do all the work immediately after the start.

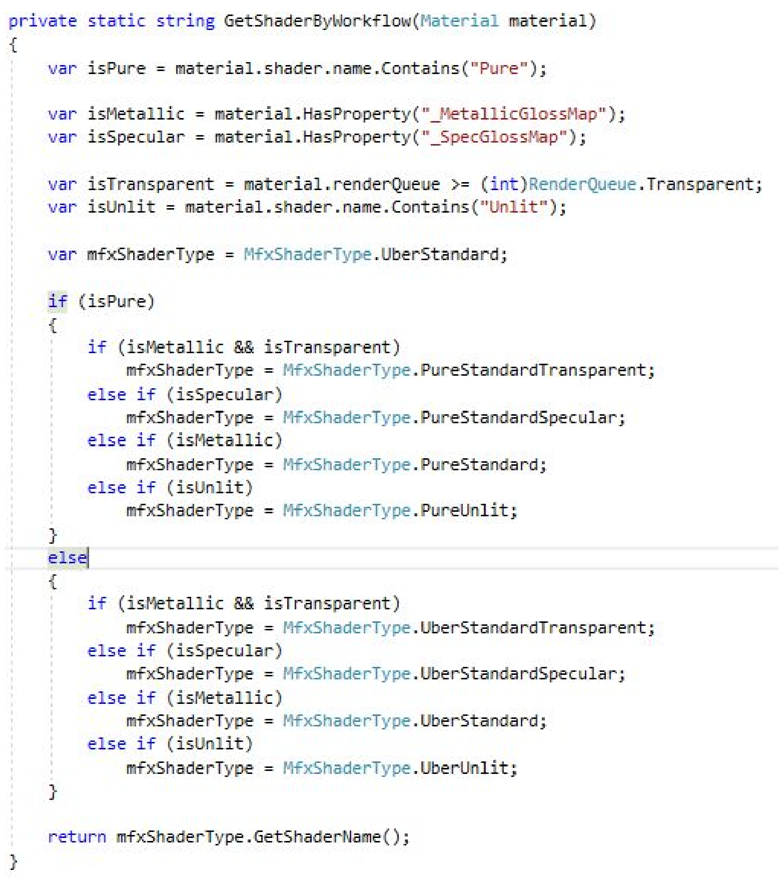
“Revert materials after animation is complete” : When the curve will end, the script will revert materials to original (materials on object before they were replaced).

“Destroy after animation is complete” : when the curve will end, “Target” object will be destroyed.

Replace materials Params:

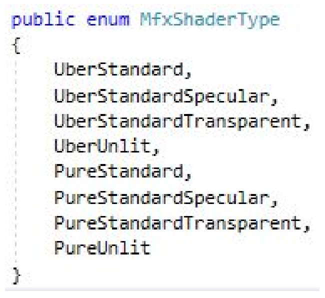
<u>Determine workflow automatically</u> : if enabled, the script will try to determine workflow of shader, you don't need to specify shader of replacement material.

If you added new version of shader, or want to change it, you need to modify the function: ***GetShaderByWorkflow*** in the ***MfxMaterialUtil*** script



```
private static string GetShaderByWorkflow(Material material)
{
    var isPure = material.shader.name.Contains("Pure");

    var isMetallic = material.HasProperty("_MetallicGlossMap");
    var isSpecular = material.HasProperty("_SpecGlossMap");

    var isTransparent = material.renderQueue >= (int)RenderQueue.Transparent;
    var isUnlit = material.shader.name.Contains("Unlit");

    var mfxShaderType = MfxShaderType.UberStandard;

    if (isPure)
    {
        if (isMetallic && isTransparent)
            mfxShaderType = MfxShaderType.PureStandardTransparent;
        else if (isSpecular)
            mfxShaderType = MfxShaderType.PureStandardSpecular;
        else if (isMetallic)
            mfxShaderType = MfxShaderType.PureStandard;
        else if (isUnlit)
            mfxShaderType = MfxShaderType.PureUnlit;
    }
    else
    {
        if (isMetallic && isTransparent)
            mfxShaderType = MfxShaderType.UberStandardTransparent;
        else if (isSpecular)
            mfxShaderType = MfxShaderType.UberStandardSpecular;
        else if (isMetallic)
            mfxShaderType = MfxShaderType.UberStandard;
        else if (isUnlit)
            mfxShaderType = MfxShaderType.UberUnlit;
    }

    return mfxShaderType.GetShaderName();
}
```

If you want to add new version of shader, you need to add it to the ***MfxShaderType*** enum.



```
public enum MfxShaderType
{
    UberStandard,
    UberStandardSpecular,
    UberStandardTransparent,
    UberUnlit,
    PureStandard,
    PureStandardSpecular,
    PureStandardTransparent,
    PureUnlit
}
```

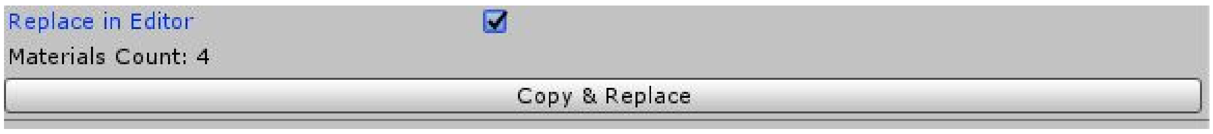
<u>"Shader type"</u> : the script will set this shader in the replaced material.

<u>"Mfx Material Template"</u> : the template of the MaterializeFx material. All MaterializeFx properties of the shader will be copied from this template to the replaced material.

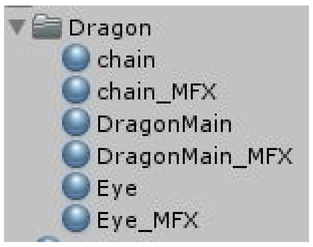
“Replace Material At Start” : material on target object will be replaced after the start.

“Replace in editor” : adds opinion to Copy and Replace original material. All materials on target object will be copied and replaced by a copy (mfx params will set by template material).

The copies will store in the original material folder with “MFX” postfix.


Replace in Editor ✓
Materials Count: 4

Example:


Dragon
chain
chain_MFX
DragonMain
DragonMain_MFX
Eye
Eye_MFX

Public Api:
----

void ReplaceMaterials() - replace materials.

void RevertMaterials() - revert materials.

void Reset() - reset animation curve time.

void Activate() - start animation.

void SetHitPosition(Vector3 hitPositionWorldPos) - set position of the mask in the world coordinates.

Shooter Scene:

MfxActivator - Starts animation and provides the hit position. You need to add it to the “Shootable” object.

RayCastShoot - Simple shoot script, which activates MfxActivator if it hits the object.