Mesh Materializer API is accessible:

- (c#) `using VacuumShaders.MeshMaterializer;`
- (java) `import VacuumShaders.MeshMaterializer;`

Simple and Skinned mesh conversion

```
static public Mesh MaterializeMesh(Renderer _renderer, params MMData[] _data)
```

```
static public Mesh MaterializeMesh(Renderer _renderer, ref MM_INFO[] _buildInfo, ref string[]
_buildInfoFull, params MMData[] _data)
```

Function returns converted mesh.

_renderer – active gameobject renderer.

_buildInfo – this variable will contain conversion info.

_buildInfoFull – this variable will contain conversion info with detail explanation.

_data – array of conversion data. Available data type and their parameters are exactly same as inside **Mesh Materializer** window:

- MMData_SurfaceInfo
- MMData_MeshTintColor
- MMData_MeshMainTexture
- MMData_MeshSecondTexture
- MMData_MeshVertexColor
- MMData_MeshDisplace
- MMData_UnityAmbient
- MMData_IBL
- MMData_Lightmap
- MMData_AmbientOcclusion
- MMData_Optimize

Note:

Textures and Models need to be readable.
Unity readable texture formats are - ARGB32, RGBA32, BGRA32, RGB24, Alpha8 and DXT.

Check “`Runtime_Materializer_Example`” script inside Example Scenes folder.

Combine Mesh Conversion

```
static public Mesh MaterializeMeshGroup(Transform _parent, params MMData[] _data)
```

```
static public Mesh MaterializeMeshGroup(Transform _parent, ref MM_INFO[] _buildInfo, ref string[]
_buildInfoFull, params MMData[] _data)
```

Function returns converted and combined mesh.

_parent – parent of hierarchy that should be combined. Hierarchy should contain only MeshFilter components and not Terrain or SkinnedMeshRenderers.

_buildInfo – this variable will contain conversion info.

_buildInfoFull – this variable will contain conversion info with detail explanation.

_data – array of conversion data. Available data type and their parameters are exactly same as inside **Mesh Materializer** window:

- MMData_SurfaceInfo
- MMData_MeshTintColor
- MMData_MeshMainTexture
- MMData_MeshSecondTexture
- MMData_MeshVertexColor
- MMData_MeshDisplace
- MMData_UnityAmbient
- MMData_IBL
- MMData_Lightmap
- MMData_AmbientOcclusion
- MMData_Optimize

Terrain Conversion

```
static public Mesh MaterializeTerrain(Terrain _terrain, params MMData[] _data)
```

```
static public Mesh[] MaterializeTerrain(Terrain _terrain, ref MM_INFO[] _buildInfo, ref string[]
_buildInfoFull, params MMData[] _data)
```

Function returns converted terrain as mesh.

_terrain – active terrain object.

_buildInfo – this variable will contain conversion info.

_buildInfoFull – this variable will contain conversion info with detail explanation.

_data – array of conversion data. Available data type and their parameters are exactly same as inside **Mesh Materializer** window:

- MMData_SurfaceInfo
- MMData_TerrainData
- MMData_TerrainTexture
- MMData_UnityAmbient
- MMData_IBL
- MMData_Lightmap
- MMData_AmbientOcclusion
- MMData_Optimize

Helper Functions

static public void GetMeshInfo(Mesh _mesh, MM_SURFACE_TYPE _surfaceType,
ref int _genVertexCount, ref int _genTrinalgeCount)
Calculates mesh vertex and triangle count (for simple and skinned meshes) based on surface type.

static public void GetCombinedMeshInfo (Mesh _mesh, MM_SURFACE_TYPE _surfaceType,
ref int _genVertexCount, ref int _genTrinalgeCount)
Calculates mesh vertex and triangle count (for combined mesh) based on surface type.

static public void GetTerrainInfo(MMData_TerrainData _terrainData, MM_SURFACE_TYPE _surfaceType, ref int _genVertexCount, ref int _genTrinalgeCount)
Calculates mesh vertex and triangle count (for terrain) based on surface type and desired width/length.

Color Adjustment Functions

```
public class MMColorAdjustment

{

    static public Color Adjust_BrightnessContrast(Color _srcColor, float _brightness, float _contrast,

                                                  float _redCoeff, float _greenCoeff, float _blueCoeff)

    static public Color Adjust_HueSaturationLightness(Color _srcColor, float _hue, float _saturation, float _lightness)

    static public Color Adjust_Level(Color _srcColor, float _inputMin, float _inputMax, float _inputGamma,

                                     float _outputMin, float _outputMax)

    static public Color Adjust_Level(Color _srcColor,

                                     float _inputMinR, float _inputMinG, float _inputMinB,

                                     float _inputMaxR, float _inputMaxG, float _inputMaxB,

                                     float _inputGammaR, float _inputGammaG, float _inputGammaB,

                                     float _outputMinR, float _outputMinG, float _outputMinB,

                                     float _outputMaxR, float _outputMaxG, float _outputMaxB)

    static public Color Adjust_ColorSpace(Color _srcColor, MM_COLORADJUSTMENT_COLORSPACE _colorSpace)

    static public Color Adjust_ColorOverlay(Color _activeColor, Color _backgroundColor,
                                            MM_COLORADJUSTMENT_BLEND_MODE _blendMode,
                                            float _blendIntensity)
     static public Color Adjust_Invert(Color _srcColor)

}
```

MMEnums

enum MM_SURFACE_TYPE { Original, Flat }

enum MM_TEXTURE_SAMPLING_TYPE { Smooth, FlatHard, FlatSmooth, FlatSmoother }

enum MM_TEXTURE_ALPHA { MainTextureAlpha, MainTextureAlphaInvert, One, Zero, SeconTextureAlpha, SeconTextureAlphaInvert, BlendAdd, BlendMultiply, BlendDecal }

enum MM_TEXTURE_BLEND_TYPE { Add, Multiply, Decal, Detail, MainTextureAlpha, MainTextureAlphaInvert, SecondTextureAlpha, SecondTextureAlphaInvert, VertexColorAlpha, VertexColorAlphaInvert }

enum MM_DISPLACE_READ_CHANNEL { R, G, B, A, Grayscale }

enum MM_DISPLACE_SAVE_TYPE { DisplaceVertex, SaveToColor, DisplaceVertexAndSaveToColor}

enum MM_COLOR_SAMPLING_TYPE { Smooth, Flat }

enum MM_COLOR_ALPHA { ColorAlpha, One, Zero }

enum MM_SAVE_CHANNEL { RGB, Alpha }

enum MM_AO_DIRTTEXTURE_TYPE { None, RGB, RGBA }

enum MM_COLORADJUSTMENT_CHANNEL { RGB, R, G, B }

enum MM_COLORADJUSTMENT_COLORSPACE { Original, Gamma, Linear }

enum MM_COLORADJUSTMENT_BLEND_MODE { Normal, Darken, Multiply, ColorBurn, LinearBurn, Lighten, Screen, ColorDodge, LinearDodge, Overlay, HardLight, VividLight, LinearLight, PinLight, HardMix, Difference, Exclusion, Subtract, Divide}

```
enum MM_INFO
{
        Ok,
        Renderer_Is_Not_Valid,
        Mesh_Is_Null,
        Vertex_Count_Limit_21666_Exceeded,
        Mesh_Is_Not_Readable,
        Material_Is_Null,
        Difference_Between_Submesh_And_Material_Count,
        Material_Has_No_Variable_MainTex,
        Mesh_Has_No_UV,
        Unsupported_Texture_Format,
        MainTex_Is_Null,
        MainTex_Is_Not_Readable,
        MainTex_Has_No_MipMaps,
        Material_Has_No_Variable_Color,
        Material_Has_No_Second_Texture,
        Second_Texture_Is_Null,
        Second_Texture_Is_Not_Readable,
        Second_Texture_Has_No_MipMaps,
        Material_Has_No_Displace_Texture,
        Displace_Texture_Is_Null,
        Displace_Texture_Is_Not_Readable,
        Displace_Texture_Has_No_MipMaps,
        Invalid_Vertex_Color,
        Invalid_Lightmap_Index,
        Lightmap_Texture_Is_Null,
        Lightmap_Is_Not_Readable,
        Lightmap_Has_No_MipMaps,
        Mesh_uv2_Have_Problems,
        Invalid_IBL_Cubemap,
        IBL_Cubemap_Is_Not_Readable,
        Mesh_Has_No_Normals_For_IBL_Calculation,
        Mesh_Has_No_Normals_For_AO_Calculation,
        Mesh_Has_No_Normals_For_Displace_Calculation,
        AO_Dirt_Texture_Is_Not_Readable,
        AO_Dirt_Texture_Is_Null,
        Terrain_Is_Not_Valid,
        Terrain_Has_No_Textures,
        Terrain_Texture_Is_Null,
        Terrain_Texture_Is_Not_Readable,
        Terrain_Texture_Has_No_MipMaps,
        Terrain_Invalid_Lightmap_Index,
        Terrain_Lightmap_Is_Not_Readable,
        Procedural_Material_Is_Not_Readable,
        Procedural_Material_Has_Unsupported_Format
    }
```