Documentation **V18**

Mesh Deformation Full Collection

Official Website

APi documentation

Last package overview

Last package video-documentation

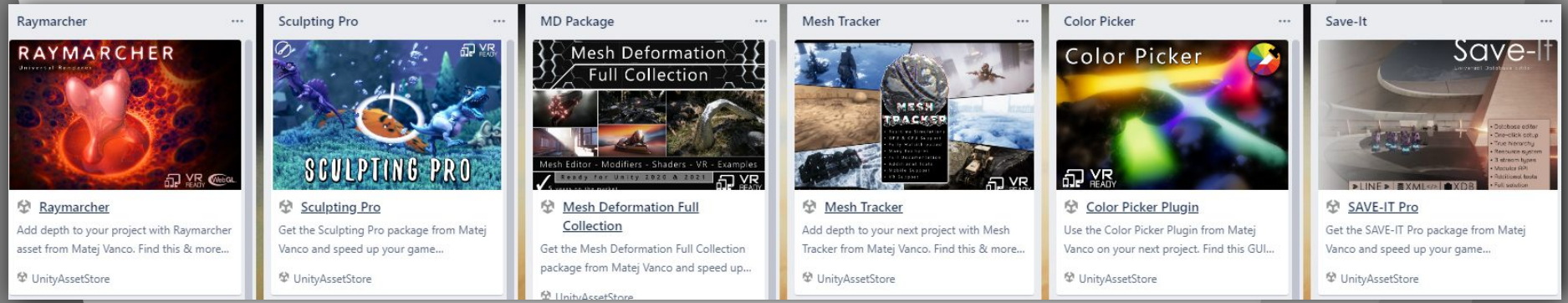
It's recommended to open the documentation in the browser to keep up-to-date

Roadmap

Official roadmap of the package and all of my assets

Trello

Curious what the future holds? Click the Trello icon!



Content

Basics

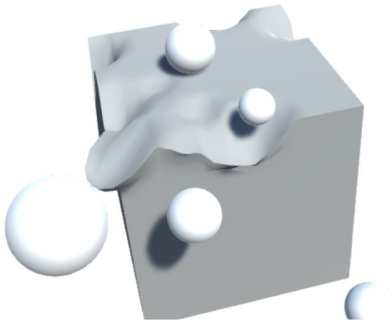
Basic information you should know

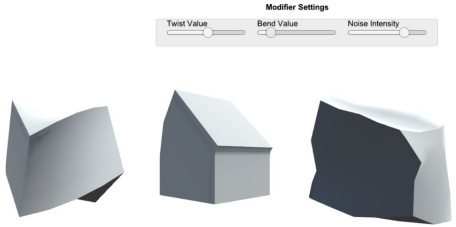
Introduction

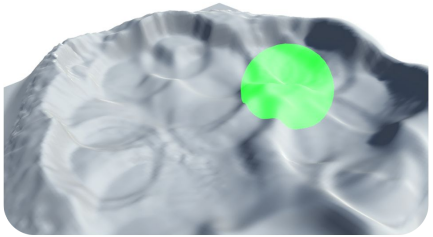
- **MD (Mesh Deformation)** is a collection of mesh manipulation tools for the Unity Engine, suitable for both beginners and advanced users. The package includes a simple & primitive vertex editor (PC, VR, Mobile), a collection of various well-known modifiers, a mesh collider refresher, shader-based deformation (Built-in RP only), physically-based procedurals, geometry primitives, and more. The MD package comes with numerous examples, detailed explanations, source code, descriptions, and real-time support.

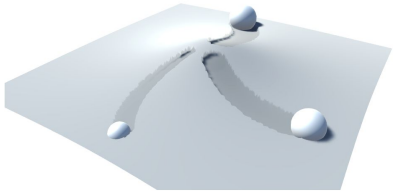
- The development of this package began in 2013 by me, Matej Vančo. It started with basic features such as generating generic points for meshes and basic controls for Skinned Mesh bones. The package was officially published on November 12, 2015. At that time, 'mesh deformation' was not very common in Unity, as Unity is not a modeling software. This package contains various systems with universal and modular functionality. It's a comprehensive collection of mesh deformations in Unity, based on well-known modifiers and mesh features that have been tested in many commercial projects by numerous users over the years.

In this documentation, you will become familiar with all essential components, modifiers, shaders, and additional tips and tricks for optimizing performance and overall use of the package.





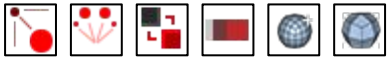




Summary

The **MD Package** is divided into **5 categories**:

- **Essentials**
  - *Contains essential components such as MeshProEditor, MeshEditorRuntime+VR & MeshColliderRefresher.*
- **Modifiers**
  - *Contains all modifiers such as Mesh Damage, Sculpting, Interactive Surface, FFD and other.*
- **Shaders (Built-In RP)**
  - *Contains a shader source called Standard Deformer*
- **Geometry**
  - *Contains all geometry such as procedural primitives, Tunnel Creator, Path Creator, Mesh Paint and other.*
- **Utilities**
  - *Contains additional utilities such as Mesh Smooth, Vertex Tools, Package Utilities and other.*



Each category has its own folder with all the required components. If you want to use the modifiers only, simply drag and drop the modifier onto any object with a MeshFilter component. To generate procedural geometry, access it via Hierarchy/Create. More details about each category are provided in the following slides.

Official APi documentation of the MD Package can be found **here**.

Essentials

Essential package components

Full description & API

Editor-tooltips available

Global namespace = **MDPackage**

Mesh Pro Editor

Mesh Editor Runtime

Mesh Editor Runtime VR

Mesh Collider Refresher

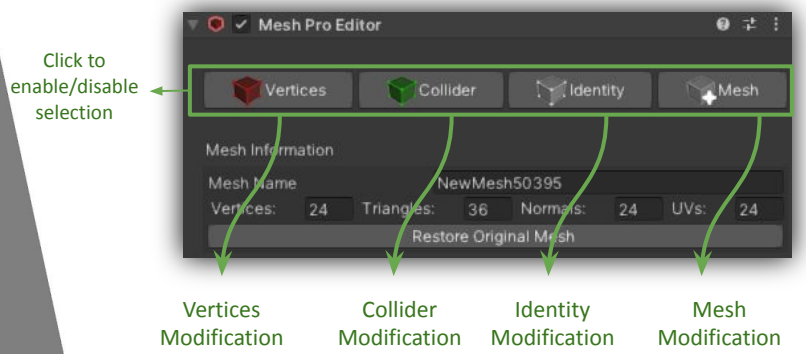
Global Preferences

Mesh Pro Editor

APi documentation

Mesh Pro Editor is an essential component that allows you to analyze specific meshes, edit their vertices, and apply external modifiers such as "smooth mesh" and "mesh subtraction." The component is divided into four subcategories: **Vertex Modification, Collider Modification, Identity Modification,** and **Mesh Modification**. At the bottom of the component, the **Mesh Information** panel displays essential data about the mesh (vertex count, tris, uvs).

The **Mesh Pro Editor** must be applied to objects that contain a **Mesh Filter** or **Skinned Mesh Renderer** (it will be converted to Mesh Filter either way). It is recommended to use the **Mesh Pro Editor** first before applying any other modifiers. The Mesh Pro Editor will prompt you to create a new mesh reference, which is a safer approach when editing specific meshes. The component also includes a public API that can be used externally.

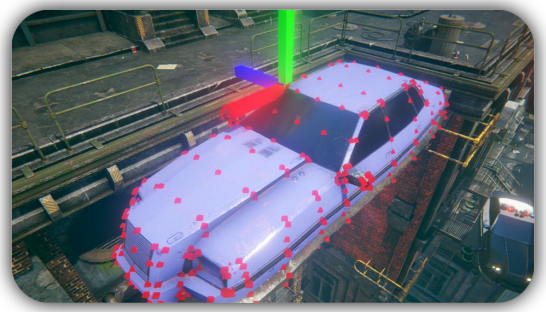


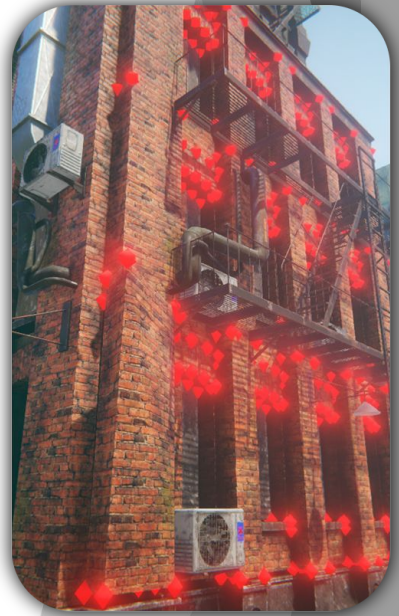
Mesh Editor Runtime

APi documentation

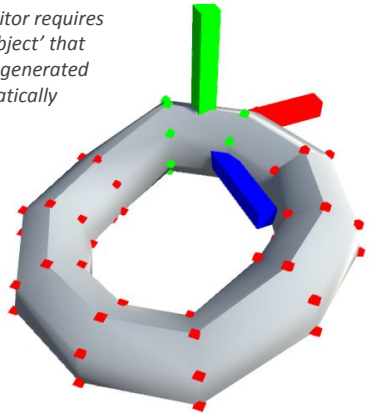
Mesh Editor Runtime enables you to edit points generated by **Mesh Pro Editor** at runtime using specified mouse input (PC) or "finger" input (Mobile). The component offers two types of runtime editor modes: **Axis Editor** and **Non-Axis Editor**.

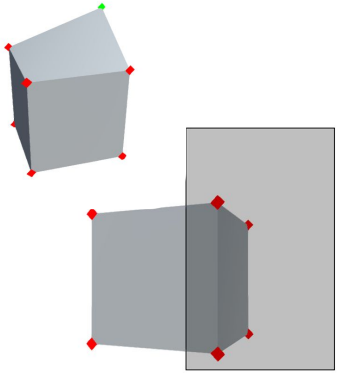
In **Axis Editor** mode, a specific object with XYZ generics represents the well-known "axis arrows," allowing for precise manipulation. Conversely, **Non-Axis Editor** mode allows for the dragging and dropping of vertices. Both modes serve the same purpose but offer different user-interactions and controls. Typically added to the main camera in a scene, **Mesh Editor Runtime** is designed for designers. Additionally, users can choose from three editor-control modes: **Grab/Drop Vertex, Pull Vertex,** or **Push Vertex**.





Axis Editor requires 'Axis Object' that can be generated automatically



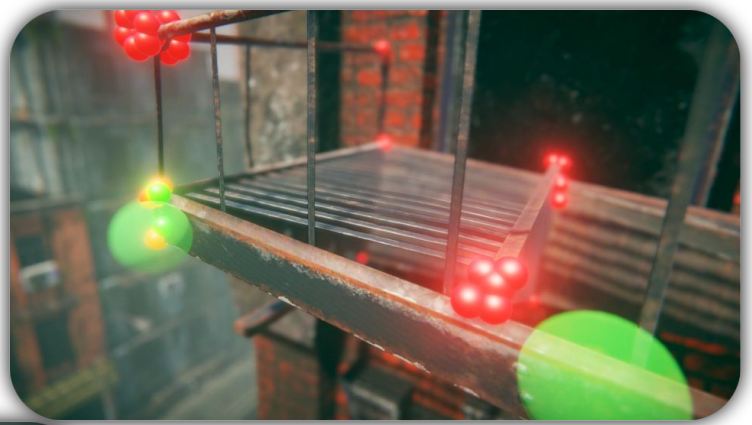


Mesh Editor Runtime VR

APi documentation

Mesh Editor Runtime VR enables you to edit points generated by **Mesh Pro Editor** at runtime within a **VR environment**. Similar to the Non-VR Mesh Editor Runtime, it performs the same functions. Full support is provided for all major VR platforms; however, your responsibility lies in **handling input**. Refer to the official API documentation for guidance on how to handle input for this component, paying particular attention to the 'Input_Hookups' properties.

The component should be added to **one of the VR controllers**. A demonstration of the overall VR setup for **Mesh Editor Runtime VR** is provided in the VR Setup slide.

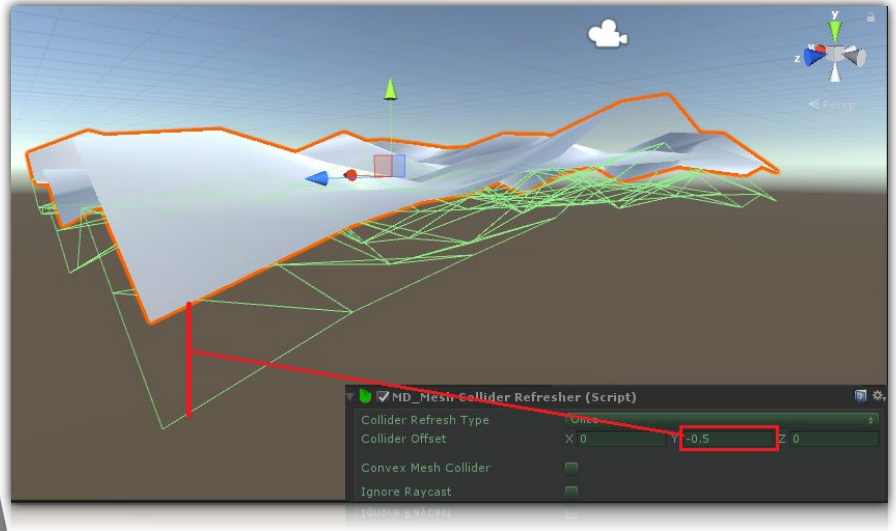


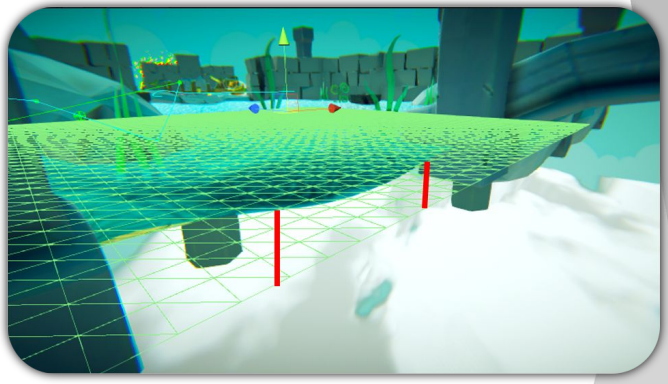
Mesh Collider Refresher

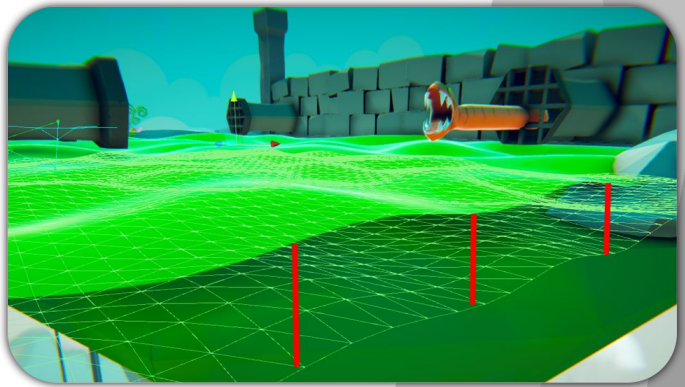
APi documentation

Mesh Collider Refresher enables you to update and refresh mesh colliders at runtime. You have the flexibility to choose from various modes, including **refreshing the mesh collider once**, **every frame**, or at **custom intervals**. Additionally, you can update the mesh collider manually or trigger it by a specific event action.

In the "**once**" refresh type, you can customize the mesh collider's position offset. A simple example is provided below to illustrate its usage.





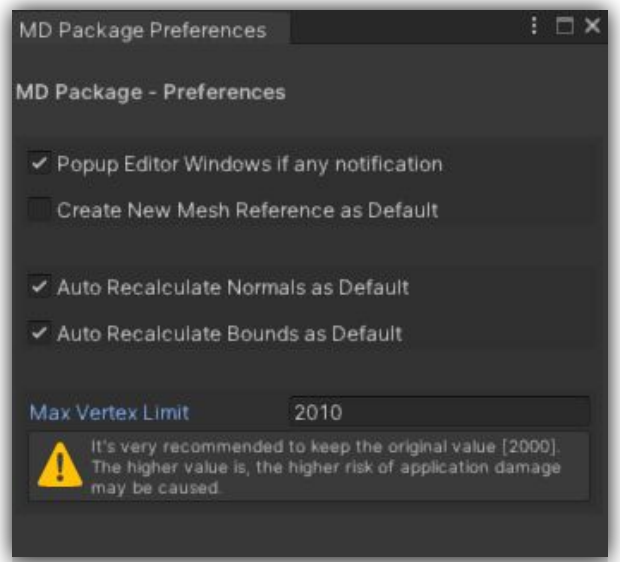


Preferences

APi documentation

The MD Package includes a global preference scriptable object designed to streamline certain tasks and features within the package. You can access the Preferences window by navigating to ***Window/MD_Package/Preferences***.

Below is an image describing the actual preferences scriptable object.



- The MD Package includes a popup editor window, activated as needed with any component related to the MD Package. For instance, if a mesh surpasses the safe level of vertex count for editing, this window will appear.
- By default, the package creates a new mesh reference, ensuring that original mesh data remains intact. Each modifier automatically generates new references.
- Auto Recalculate Normals & Bounds is enabled by default. This ensures that modified meshes have recalculated normals and bounds unless specified otherwise.

- The package also features an Allowed Max Vertex Limit field. If a mesh exceeds the specified vertex limit, a warning window will popup (assuming the popup editor window feature is enabled).

Modifiers

Collection of well-known modifiers

Full description & API

Editor-tooltips available

Global namespace = **MDPackage.Modifiers**

MDM_Bend
MDM_AngularBend
MDM_Twist
MDM_MeshNoise
MDM_FFD
MDM_MeshEffector
MDM_MeshDamage
MDM_Morpher
MDM_InteractiveSurface
MDM_SurfaceTracking
MDM_MeshFit
MDM_MeshSlime
MDM_MeltController
MDM_SculptingLite
MDM_RaycastEvent
MDM_MeshCut
MDM_SoundReact

MDM_Bend,Twist,Noise

MDM (MeshDeformationModifier) Angular/Bend, Twist, and **Noise** are fundamental modifiers within the package. Each serves a unique purpose in deforming meshes:

- Angular Bend bends the mesh vertically based on specific angle.
- Bend bends the mesh along three axes.
- Twist twists the mesh along three axes.
- Noise populates the mesh with a 'dirty' surface that can be modified in two modes: vertical and general deformation. General noise affects the entire mesh in a 3D space, while vertical noise affects the mesh in a 2D space, making it planar. All basic modifiers function both in the editor and at runtime, with parameters modifiable during runtime as well.

Each modifier can only be added to an object once. Using multiple modifiers simultaneously on the same object is prohibited due to vertex synchronization issues.

Since Angular Bend, Bend, Twist, and Noise are not multithreaded modifiers, it is not recommended to apply them to high-poly objects (~2000 vertex count and above).

APi documentation (Twist)

APi documentation (Bend)

APi documentation (Angular Bend)

APi documentation (Noise)

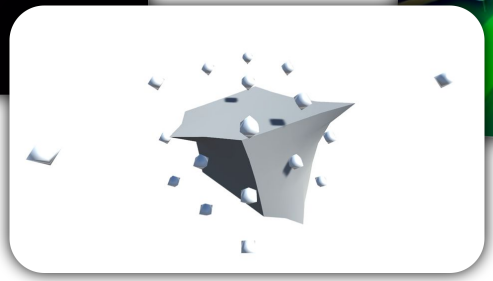
MDM_FFD

APi documentation

MDM (MeshDeformationModifier) FFD, or **Free-Form Deformation**, controls mesh deformation using weights registered in a lattice. By moving points within this lattice, the mesh smoothly deforms, allowing for quick edits to desired mesh parts.

The FFD modifier offers various lattice resolutions in cubic shapes: 2x2x2, 3x3x3, and 4x4x4. Additionally, a custom number of lattice points is available, although it's not recommended due to performance considerations.

Since FFD is not multithreaded modifier, it is not recommended to apply it to high-poly objects (~2000 vertex count and above).



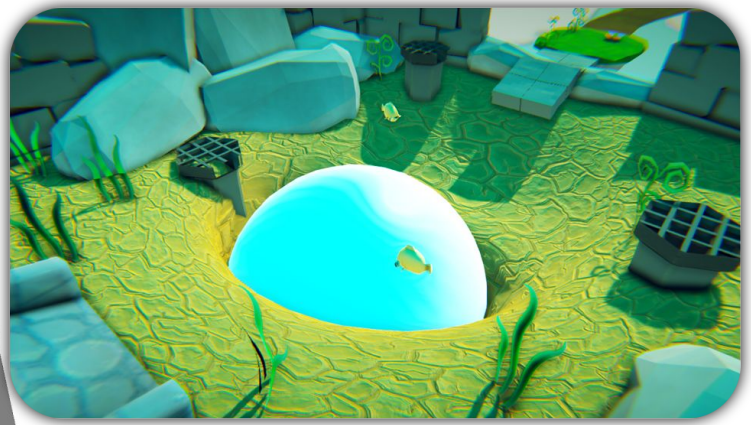
MDM_MeshEffector

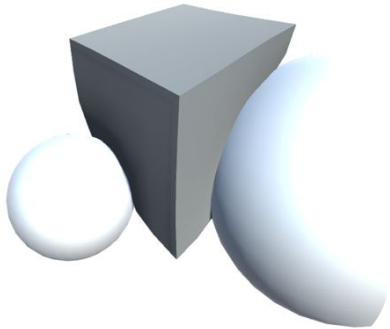
APi documentation

MDM (MeshDeformationModifier) Mesh Effector deforms a mesh using registered weight nodes and density. There are four types of effectors: One-Pointed, Two-Pointed, Three-Pointed, and Four-Pointed. These effectors create spherical radii around assigned nodes, which then influence the target mesh.

Unlike the **FFD**, the **Mesh Effector does not create a lattice**. Instead, it generates radiuses that deform the overall mesh based on parameters such as Weight value, Weight multiplier, Weight density, and Weight effector for each effector type. This modifier is excellent for quick deformations without additional effort!

Mesh Effector is a multithreaded modifier. It's safe to edit high-poly objects (~2000 vertex count and more).



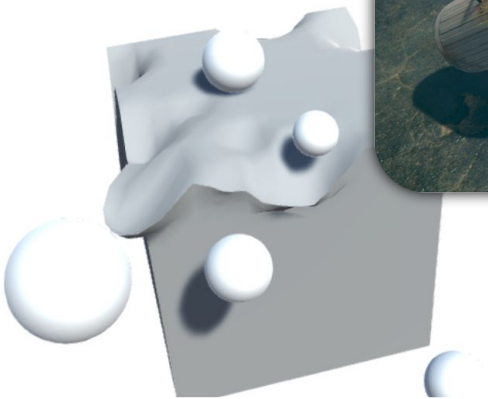


MDM_MeshDamage

APi documentation

MDM (MeshDeformationModifier) Mesh Damage enables you to deform any mesh surface using external sources such as rigidbody collision impacts or raycast events. This simple yet highly effective modifier enhances mesh interactivity with the environment.

While Mesh Damage is not a multithreaded modifier, it is safe to apply to high-poly objects (~2000 vertex count and above) since it is not computationally heavy to calculate.



MDM_Morpher

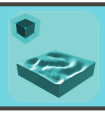
APi documentation

MDM (MeshDeformationModifier) Mesh Morpher enables you to blend between various shapes. You can choose an unlimited number of shapes and register their initial vertex states. The Morpher is commonly used for facial expressions, mimics, statue translations, and more.

Mesh Morpher is a multithreaded modifier. It's safe to edit high-poly objects (~2000 vertex count and more).

Example blend between 3 meshes

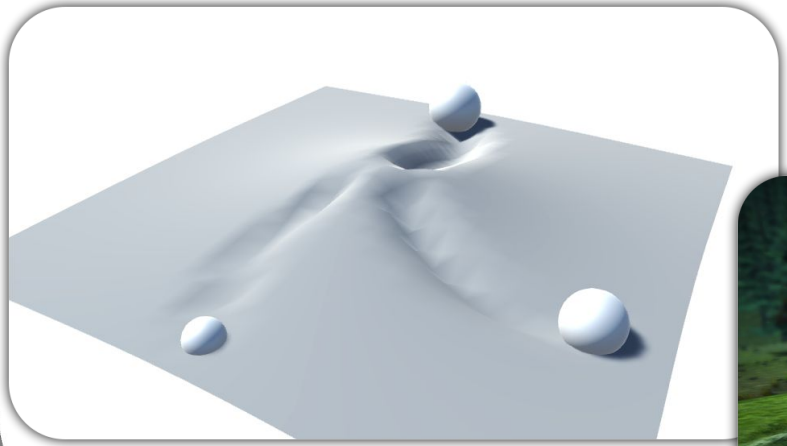
MDM_InteractiveSurface

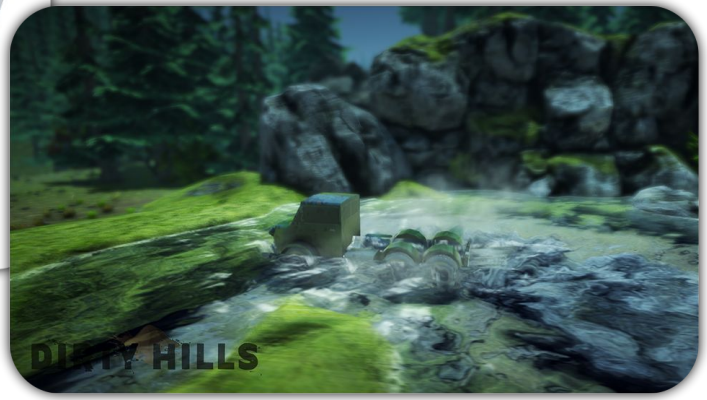


APi documentation

MDM *(MeshDeformationModifier)* **Interactive Surface** allows you to interact with any meshes and simulate ‘surface deformation’. Create dynamic tracks, snow trails, drops and more. The mesh can be ‘regenerated’ as well by specified speed value and interpolation.

Interactive Surface is a multithreaded modifier. It’s safe to edit hi-poly objects (2000 vertex count and more).



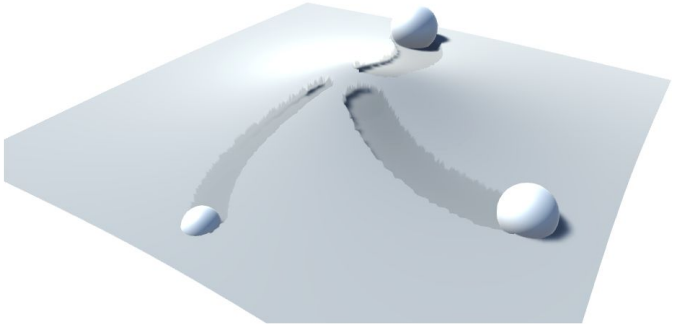


MDM_SurfaceTracking

APi documentation

MDM (MeshDeformationModifier) Surface Tracking performs a similar function to **Interactive Surface** but requires additional materials and settings for setup. It primarily operates via **GPU**, conserving performance and providing easier manipulation. This modifier necessitates the **MD_LiteMeshTracker** shader and **Render Texture**, which can be automatically generated in the Inspector. For mobile usage, utilize the **MD_LiteMeshTrackerMobile** shader. Surface Tracking is ideal for rapid surface simulations. Compared to Interactive Surface, the results may be less smooth, contingent upon the Tessellation value.

Surface Tracking is a multithreaded modifier. It's safe to edit high-poly objects (~2000 vertex count and more).

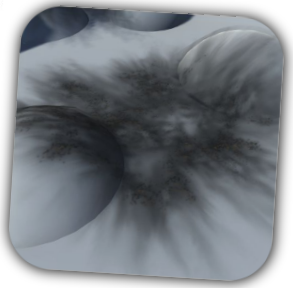


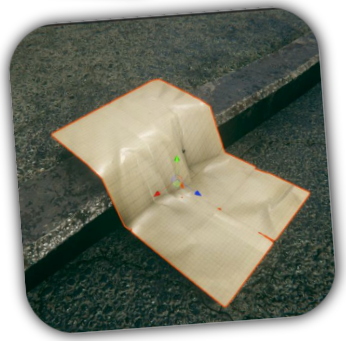
MDM_MeshFit

APi documentation

MDM (MeshDeformationModifier) Mesh Fit enables you to fit a mesh to any surface with a collider by using either generated or specified points. This modifier simplifies the creation of universal decals and dynamic meshes through raycast events. Notably, it does not rely on a projector component.

Since Mesh Fit is not multithreaded modifier, it is not recommended to apply it to high-poly objects (~2000 vertex count and above).



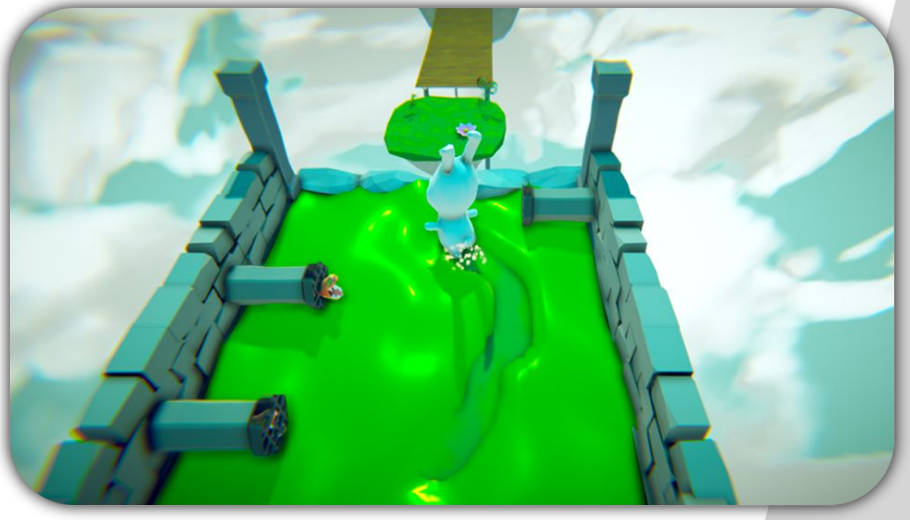


MDM_MeshSlime

APi documentation

MDM (MeshDeformationModifier) Mesh Slime enables you to create "slime-like" surfaces with additional input. It performs a similar function to the Interactive Surface but offers more settings and delves deeper into the process. Refer to the official API documentation for guidance on how to handle input for this component, paying particular attention to the **'Input_Hookups'** properties.

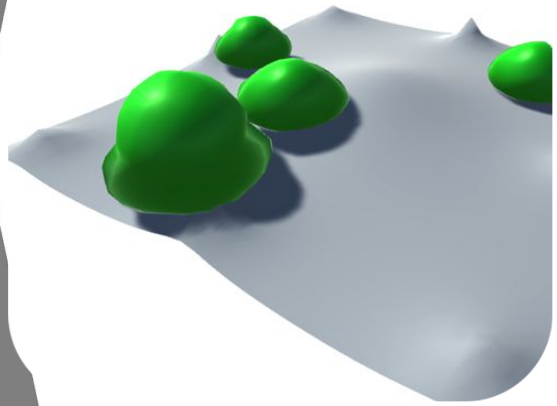
Since Mesh Slime is not multithreaded modifier, it is not recommended to apply it to high-poly objects (~2000 vertex count and above).



MDM_MeltController

APi documentation

MDM (MeshDeformationModifier) Melt Controller facilitates the creation of a simple melting effect. This controller requires a special material with the **Melting** shader, provided within the package. Users can configure parameters such as melting zone, melt transition, melt amplification, and more. **It's important to note that this modifier is not compatible with HDRP or URP.**



MDM_SculptingLite

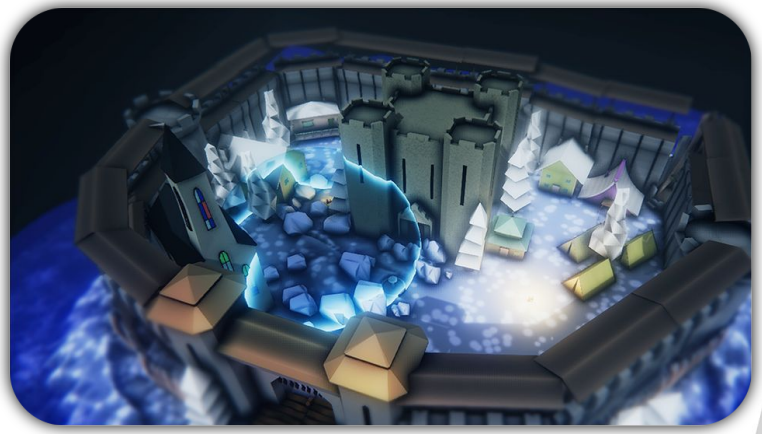
APi documentation

MDM (MeshDeformationModifier) Sculpting Lite offers a unique approach to mesh deformation, allowing for more artistic manipulation. This system introduces a mesh manipulation through radial brushes with various customizable features. Users can adjust brush type, appearance, radius, and intensity to suit their needs. Additionally, an internal API is available for users to modify their own methods.

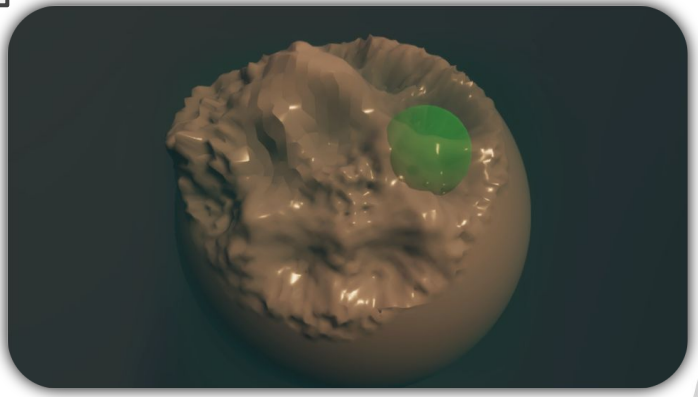
The system is particularly useful for advanced terrain editing (excluding built-in Unity terrain), offering various brush types such as smooth (HCFilter, Laplacian Filter), noise filters, stylization, and more. Sculpting Lite is compatible with Mobile and VR platforms and is fully optimized for complex meshes thanks to multithreading.

*Please note that development of Sculpting Lite has ended. For a more comprehensive sculpting solution, we recommend exploring our brand new plugin, **Sculpting Pro**.*

Used in a PC & VR game The God



Oculus Quest 2 with hand tracking

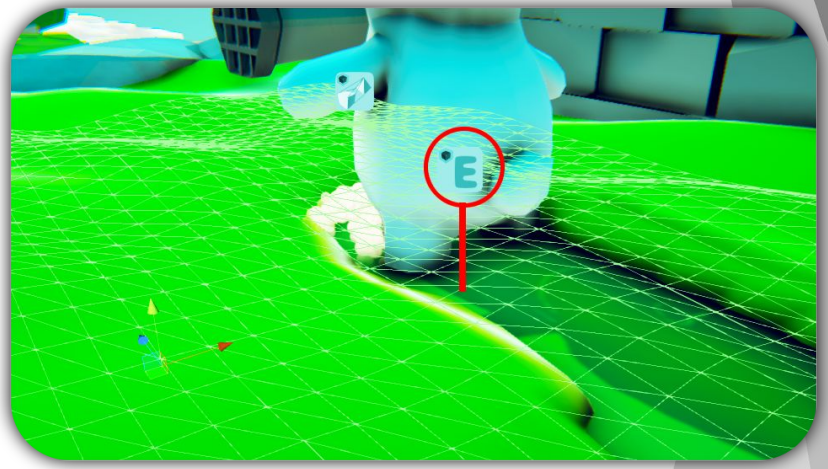


MDM_RaycastEvent

APi documentation

MDM (MeshDeformationModifier) Raycast Event is a straightforward modifier designed to trigger specific events when a raycast intersects with an object. Users can configure their own event systems and raycast logic without requiring programming skills. This raycast event is integrated with nearly all other modifiers, enabling access to their methods via the Raycast Event interface.

Blabibo tracks its position for Mesh Slime



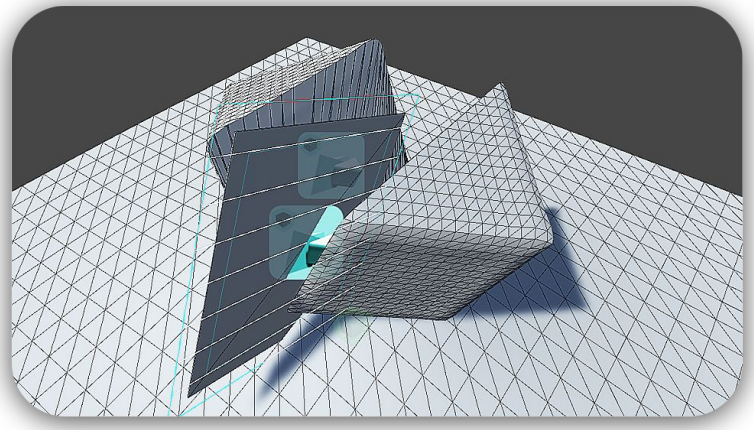
Used as a 'step-tracker' for Surface Tracking

MDM_MeshCut

APi documentation

MDM (MeshDeformationModifier) Mesh Cut is a straightforward modifier designed for one-directional mesh cutting. Users have the option to manually process cut operations via script or utilize the physically-based solution provided by MeshCut_Cutter, which handles the task automatically without the need for coding.

While Mesh Cut is not a multithreaded modifier, it is safe to apply to high-poly objects (~2000 vertex count and above) since it is not computationally heavy to calculate.

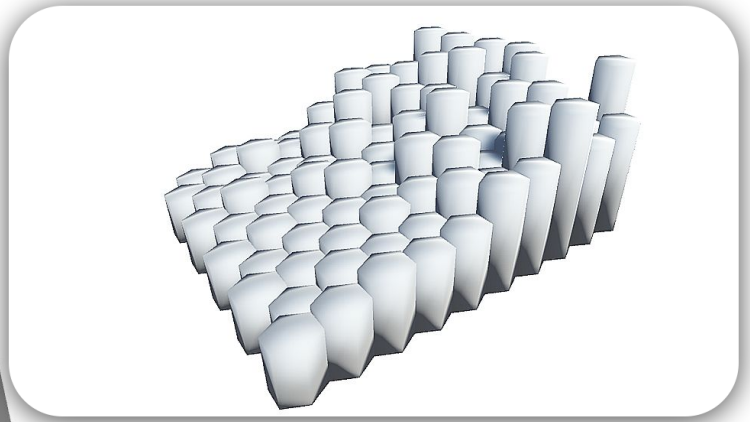


MDM_SoundReact

APi documentation

MDM (MeshDeformationModifier) Sound React enables you to convert audio clip data into a single floating value. This value can be utilized for various purposes, including musical audio visualizations. Access the output variable in the SoundReact script and apply the value to unleash your creativity in any scenario!

Used in a hexagon grid - grid dances to the music!



Shaders

Package shader source

Full description & API

Standard Deformer
Lite Mesh Tracker
Melt Shader

The following slides contain gif images. Please open the docs in browser, gif images don't work in PDF.

Standard Deformer

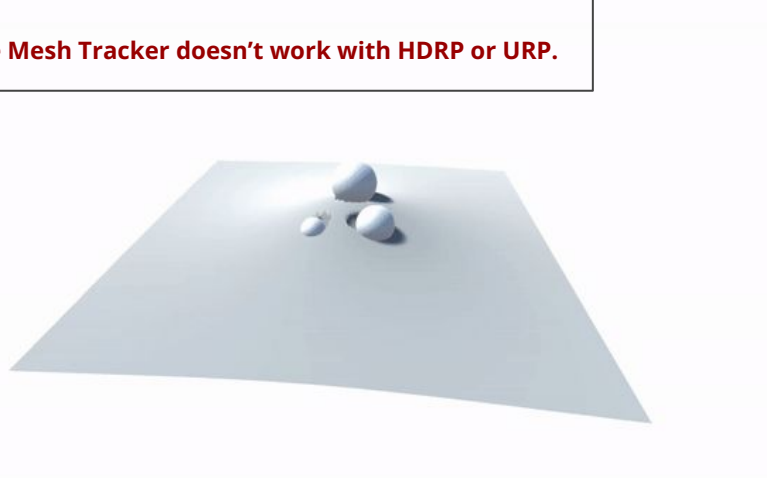
Standard Deformer stands as a pivotal shader within the MD Package. This shader empowers you to animate any object with a renderer component in diverse ways. Generate motion with countless possibilities, including jumping, fish swimming, simple swaying in all directions, noise simulation, and more. Notably, the game **Sea Orchestra** was created exclusively using the Standard Deformer. Explore the settings and features available in the right panel to unleash the full potential of the Standard Deformer.

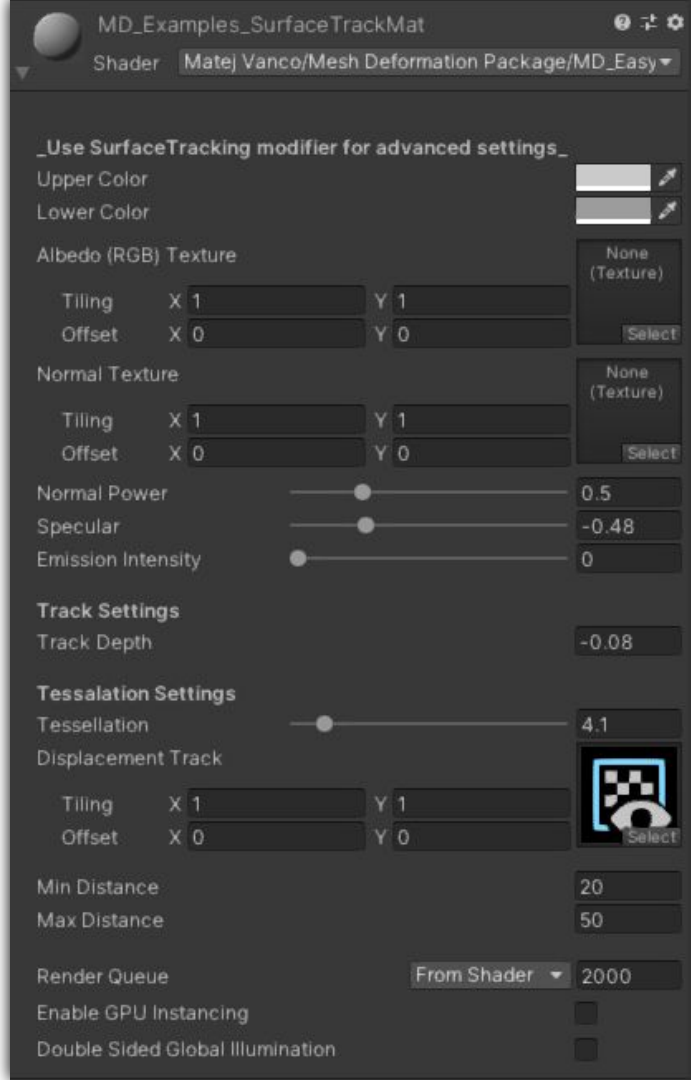
It's essential to note that the Standard Deformer is not compatible with HDRP or URP.

Lite Mesh Tracker

Lite Mesh Tracker is an essential shader required for utilizing the **Surface Tracking modifier**. This modifier operates with **Render Textures**, which provide instructions to the shader regarding the height or depth of the mesh. Setting up Lite Mesh Tracker is simple and efficient, making it an easy process. For mobile applications, utilize **Lite Mesh Tracker_Mobile**. However, please note that the **mobile version does not support Tessellation**. Instead, you can employ **Procedural Plane** to customize vertex count as needed.

The Lite Mesh Tracker doesn't work with HDRP or URP.

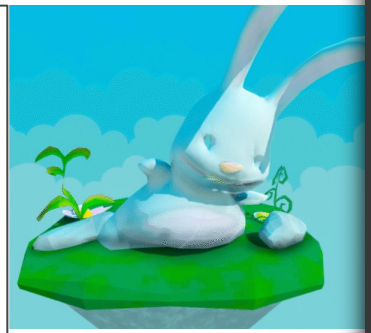


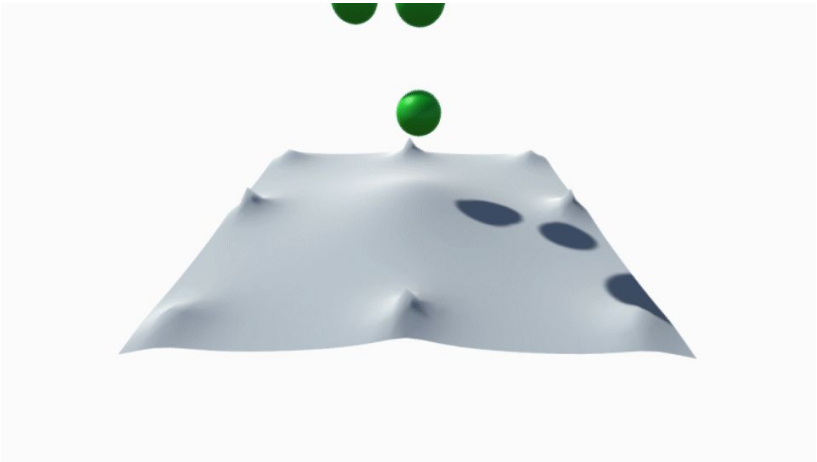


Melt Shader

The Melt Shader is a vital component required for utilizing the **Melt Controller**, although it can also be employed independently of the modifier. With the Melt Shader, you can generate a 'melting' effect on any renderer object, adjusting settings as needed. This shader is compatible with all platforms, offering versatility in your visual effects creation.

The Melt Shader doesn't work with HDRP or URP.





Geometry

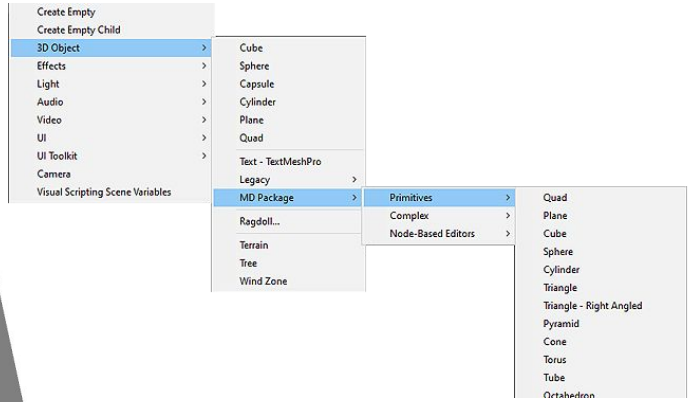
Package geometry primitives & generators

Full description & API

Editor-tooltips available

Global namespace = **MDPackage.Geometry**

Geometry can be found in Hierarchy/Create



MDG_ProceduralExtendedPlane

MDG_HexagonGrid

MDG_MeshPaint

MDG_PathCreator

MDG_TunnelCreator

Other Primitives

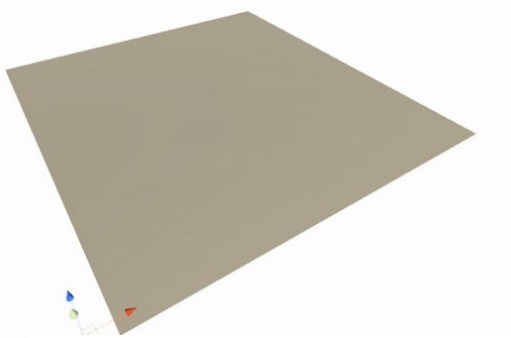
The following slides contain gif images. Please open the docs in browser, gif images don't work in PDF.

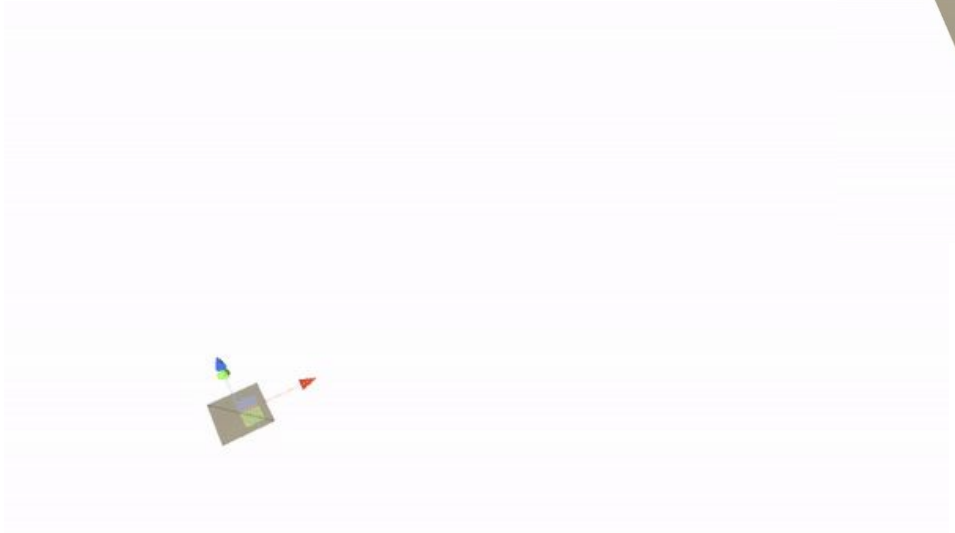
MDG_ProceduralExtendedPlane

APi documentation

Procedural Extended Plane is an advanced shape component designed to generate procedural planes with custom vertex counts and sharp normals. Unlike traditional planes, the generated plane lacks smoothing, resulting in sharp-looking final results. Additionally, it includes an "Angle Property" feature, enabling you to bend the plane up or down for added versatility.

Noise modifier applied to the plane





MDG_HexagonGrid

APi documentation

Hexagon Grid enables you to generate procedural hexagon grids with full customization options. Each hexagon segment's height can be randomized using the random height feature. Additionally, you can flip hexagon faces, make them planar, or adjust the overall size of the grid to suit your needs.

MDG_MeshPaint

APi documentation

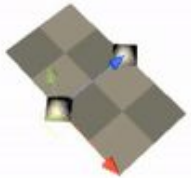
Mesh Paint offers a comprehensive mesh painting solution, allowing users to paint meshes in three shapes: Plane, Triangle, and Cube. Designed with simplicity and completeness in mind, this component is fully equipped with advanced settings for planar drawing or spatial drawing. It is compatible with all platforms, including VR and mobile, making it accessible for designers seeking versatile mesh painting capabilities. Please read the APi documentation of how the input can be handled.

MDG_PathCreator

APi documentation

Path Creator is a comprehensive tool for creating and editing procedural paths, offering both triplanar and regular UV mapping options. With user-friendly nodes, you can easily create simple paths, edit tracks, and apply additional modifiers for smoothing features.

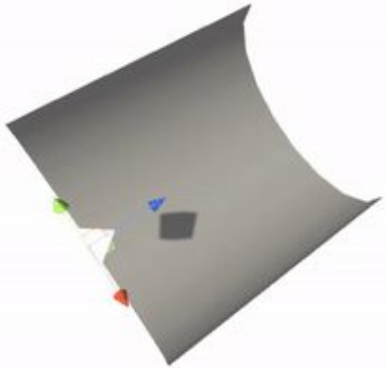
Whether you prefer to create custom paths with modular nodes or generate paths along existing nodes, Path Creator provides the flexibility you need for your projects.



MDG_TunnelCreator

APi documentation

Tunnel Creator is a robust tool for generating and modifying procedural tunnels, featuring triplanar UV mappings for enhanced visual quality. The system is built on modular nodes and weights that can be adjusted both in the editor and at runtime. Additional features enable you to create turns, straight lines, or connect other tunnels seamlessly. To streamline your workflow, Tunnel Creator includes its own editor window for improved organization and efficiency. Two essential components are associated with Tunnel Creator: **MDG_TunnelCreator** (the root) and **MDG_TunnelNodeUVData** (the node UV data controller).



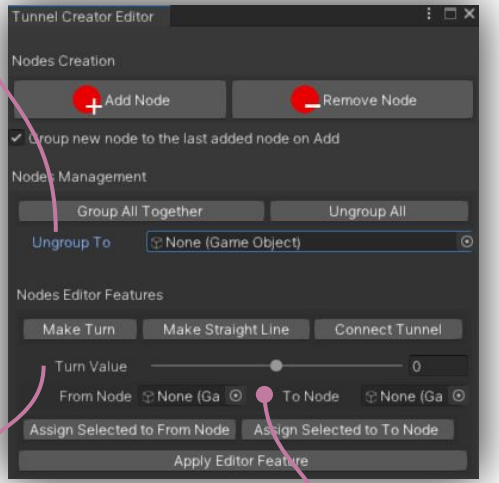
MDG_TunnelCreator

APi documentation

Tunnel Creator detailed description.

Ungroup all nodes to the desired object

Tunnel Editor window



Make turn in degrees (Left/Right)

Process 'Turn' from NODE to NODE including all nodes in between these two nodes. (The more nodes included, the smoother the turn evaluates)

Tunnel Vertex count <
Everytime when vertex count is changed, the Apply button must be pressed & you will lose all your created nodes.
Overall Tunnel Radius <
Overall Node Size <

If enabled, the tunnel will be split into half <
If enabled, the tunnel's normals will be inverted (facing out/in) <
If enabled, the scale of nodes will partially affect tunnel radius <

Currently created nodes (just for debug purposes) <

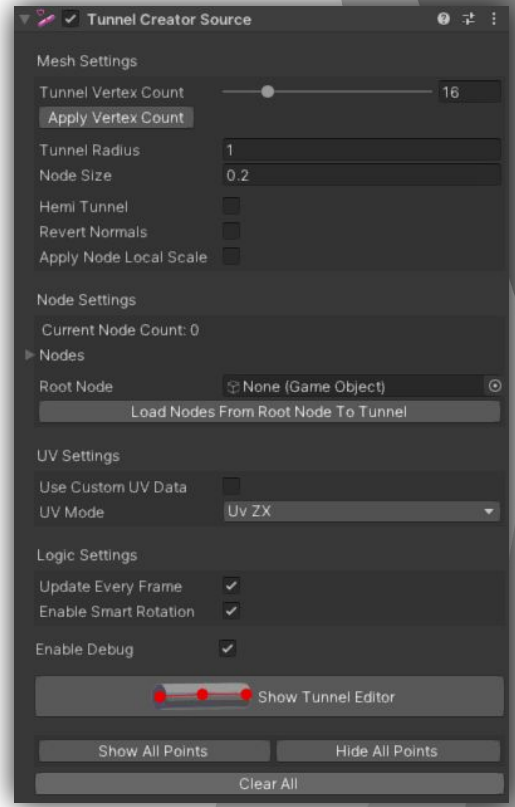
Root node of loaded nodes <
If the button is pressed, the nodes will be automatically loaded into Nodes list. RootNode field must be assigned. This helps you to restore lost tunnels with stored nodes.

Custom UVData option <
If the field is enabled, each tunnel chunk may contain *TunnelNodeUVData* behaviour to customize UV map. Otherwise choose UV_Mode option.

Update Every Frame (takes more performance if bigger tunnels) <
Enable Smart Rotation <
If enabled, the nodes will rotate towards its neighbour. Makes the node editor easier.

Advanced tunnel editor which helps you to do turns and straight lines a bit easier. <

Inspector view

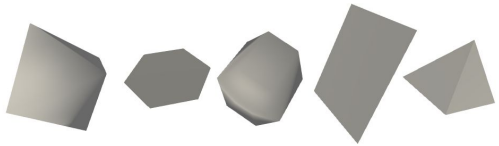


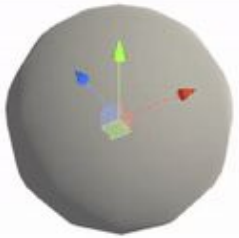
Other Primitives

APi documentation

The Mesh Deformation Package includes various procedural primitives that can be easily created via **Hierarchy/Create 3D/MD Package**.

Users can generate procedural plane, octahedron, triangle, pyramid, spatial and planar hexagon, cone, tube, and more. Additionally, these primitives are accessible through code via **MD_FullPackage/MD_Core/Geometry.**





Technical Info

Technical information about the package & general tips

VR Setup

The MD Package offers full support for all major VR platforms, including Oculus Quest. However, if you wish to interact with and send input to certain modifiers or mesh tools provided by the package, you'll need to set up your **own input system**.

Unity provides various methods for handling input, whether it's the Legacy Unity Input System, Unity Input System v2, or others. You have the freedom to integrate them with the MD Package. As mentioned in previous slides, some modifiers/tools contain **"Input Hookup"** properties that can be used to pass input events. You'll need to handle this integration yourself. For more information, please refer to the API documentation.

To learn how this integration can be achieved, please explore the example content under **MD_Examples_Scripts/InputWrappers**.

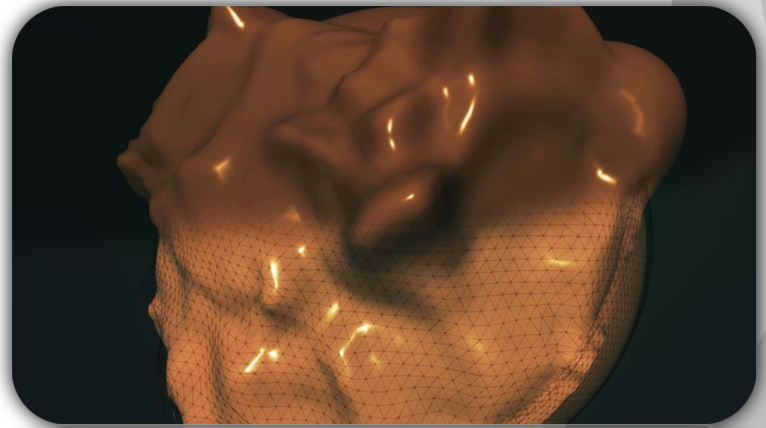
- See how custom input is handled in Sculpting Pro.
- See how custom input is managed in the Mesh Paint modifier.
- See how custom input is utilized in Mesh Editor Runtime.

Multithreading

The MD Package includes many modifiers that support multithreading, allowing certain tasks to run on separate threads. This feature enables manipulation of meshes with over ~2000 vertices *(which is the recommended value to prevent performance degradation)*. However, not all modifiers support multithreading, so it's important to optimize your mesh or refrain from using certain modifiers for project safety. It's also advisable not to use many *(10 and over)* modifiers with multithreading enabled, as this may result in performance slowdown. Multithreading functionality works well on all devices, including PC, VR, and mobile platforms. See multithreading in practice.

Car model - 50k vertex count

Planet editor - 16k vertex count



Vertex Tool Window

APi documentation

The Vertex Tool Window is an additional feature-tool included in the MD Package, accessible via **'Window/MD_Package/VertexTool'** or within the **'Mesh Pro Editor/Vertices Modification'** interface. It serves as an expansion for manipulating meshes, vertices, and elements, offering a dedicated APi for internal use known as **MD_VertexToolModifier**. See Vertex Tool Window in practice.

Attach 2 or more meshes [Meshes will be combined and will share the same material]

Clone selected mesh [Parameters below - Count, Position Offset and Rotation Offset]

Weld selected vertices [Vertices will be weld – they will split into one]

Relax selected vertices [Vertices will be normalized and their offset will be multiplied]

Group selected objects into included object below

Group enabled vertices [Additional group function to group enabled vertices in Zone-Generator Mode in MeshProEditor]

Performance & Complexity

The MD Package offers limitless possibilities but is subject to performance limitations, particularly with higher-poly meshes and certain modifiers/components. **Here are some important limitations to consider before purchasing:**

Essential Component (Mesh Pro Editor):
Mesh Pro Editor enables direct editing of mesh vertices in the editor and at runtime. However, there's a condition: if a mesh exceeds a specified vertex count (N-count, set in preferences), you'll be advised not to generate mesh points.
Mesh Pro Editor isn't designed for advanced mesh editing; it provides basic manipulation capabilities.
Editing meshes beyond 10,000 vertices isn't supported; for such meshes, use multithreaded modifiers like FFD, Mesh Effector, Sculpting Pro, and other.

Modifiers:
While multithreaded modifiers can handle extremely high-poly meshes, there's still a performance bottleneck. Using multiple multithreaded components simultaneously may lead to performance issues. It's advisable to organize and optimize your project well and use as few multithreaded modifiers as possible.

FAQ

• Is the MD Package available for Mac or Linux?
Yes, it's available for all operating systems & devices.

• Can I use the MD Package in my mobile game/application?
Yes, you can, but the mobile performance may vary.
(depends on use of tools & how you organize your project).

• Can I edit complex meshes at runtime/ in editor?
Yes you can, but it depends which modifier you'll choose to work with. Please read Performance & Complexity slide for more info.

• Is it possible to export my mesh to OBJ format?
No, it is not possible with MD Package. Use a different external plugin.
But you can save your mesh to Assets and create a prefab in the Unity editor.

• Do I need any programming skills to use the MD Package?
No, you don't need any programming skills. But if you would
like to make complex operations or custom modifiers, you can use the internal APi.

• Am I able to edit a Skinned Mesh object?
Yes, you can edit a Skinned Mesh object using the MD Package, but the process is not straightforward. Skinned Mesh Renderers control mesh deformation via bones. However, in the MD Package, ***Skinned Mesh Renderer objects need to be converted into Mesh Filters****. This results in having two copies: the original Skinned Mesh object and the editable Mesh Filter. Changes made to the Mesh Filter are then applied to the original Skinned Mesh object. While this approach may not be the most user-friendly or practical solution, it can work for simple edits and minor deformations.* ***In summary, the MD Package does not officially support a user-friendly editor for skinned meshes.***

• Do I need latest Unity version to use the MD Package?
No, you don't. But it's much safer and recommended to use
the latest Unity Version. If you are going to use the older Unity version, the code conversion and compilation will be required and you may get some warning/error messages.

• Is the Mesh Tracker included in the MD Package?
No, it's not included in the package. But the package contains a similar modifier called Surface Tracking.

• Does the MD Package work with WebGL?
Yes, but the multithreading feature is not supported.

• Does the MD Package work with Unity HDRP or URP?
Yes, ***it does****, but* ***shaders won't work*** *as they use* ***Unity standard pipeline****. All the other components and modifiers will work as they are logical part of the package.*

• Does the MD Package support editing of Unity terrains at runtime?
No, MD package doesn't work with Unity terrains at all. Meshes only.

FAQ

• **Are there any project sources of available free demos that use the MD Package?**
No, for privacy & license purposes, there are no project sources of free demos that use the MD Package, you can only try them & test them for free on your own.

• **Do you plan to add URP/HDRP support for shaders in the MD Package?**
Yes, I plan to add URP/HDRP support for shaders in the MD Package.
Hopefully in the 2025.

Commercial Products

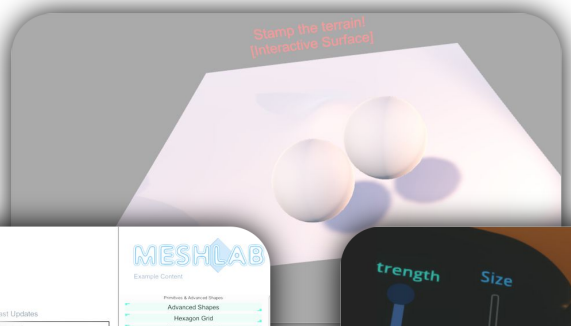
Some game projects utilize the MD Package tools and all of its features. Explore these projects below. [Click the image]

Downloadable Content

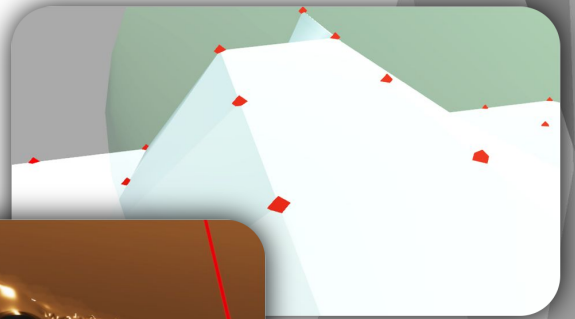
You are welcome to download the official example content of the MD Package.
Currently available for Windows OS only. [Click the image]

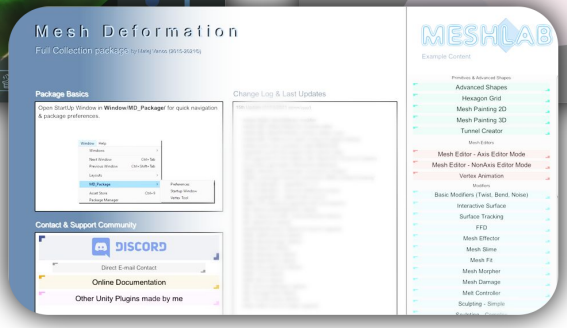
Terrain Sculpting PC [Win]

Modifiers in VR [Win, Unity-XR]



Mesh Editor in VR [Win, Unity-XR]





MD Package official examples [Win]

Terrain Sculpting in VR [Win, Unity-XR]

Extras

The MD Package contains additional short demo games built entirely from the package's modifiers and shaders. Explore these games for free below! **Just click the GIF image.**

Thank you!

Thank you for your attention. I hope the general documentation was helpful! If you have any questions, suggestions, or issues, please don't hesitate to **join my official Discord server** for quick and real-time support.

If you don’t like Discord, you can still contact me **here**.
(may take some time to respond)