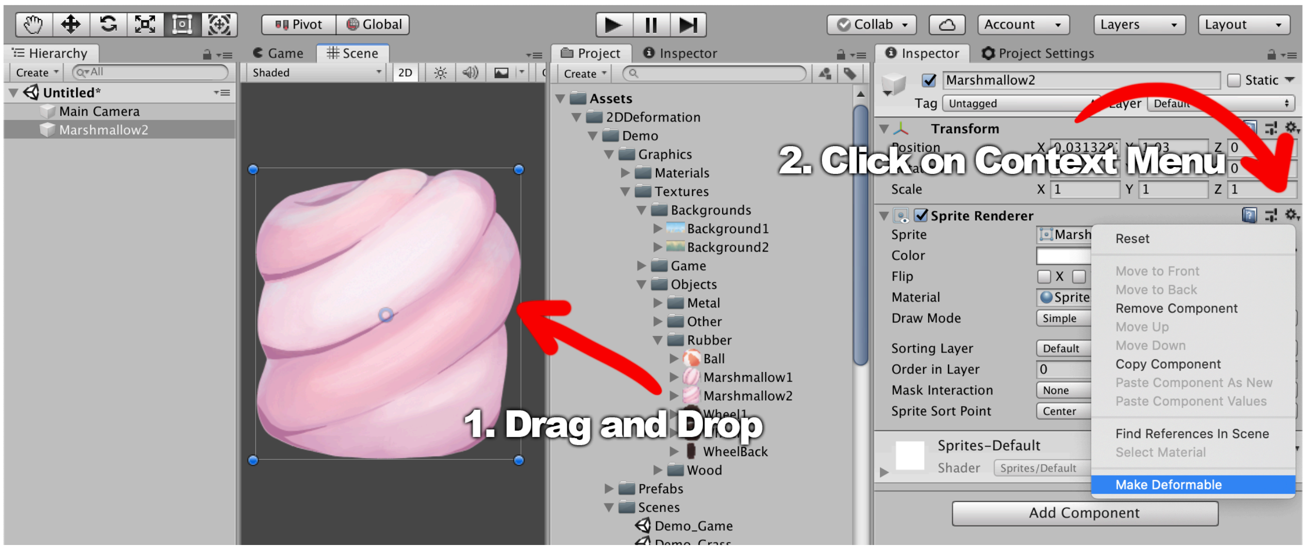
Table of content

What is 2D Deformation?

This is an asset for Unity which allows to create sprites that may be deformed using physics or just from code! To create deformable sprite drag and drop sprite to the scene or hierarchy window and click on Context Menu item.

Quick Start

To create a 2D Deformable object: drag and drop Sprite from Project window to Scene or Hierarchy window and click on Context Menu -> Make Deformable:



Afterwards the deformable object creates on the scene, user enter the play mode and test the deform physics using collisions with other objects.

How does it works?

2D Deformation changes vertices positions of MeshFilter and PolygonCollider2D points according the positions of collisions with a normal and an impulse. Using PolygonCollider2D provides the best collision accuracy with the great performance.

The DeformableSprite class creates a mesh using the MeshFilter, the MeshRenderer.

The DeformableObject class contains deformation parameters, passes collision data to the PolygonDeformationSystem instance where the MeshFilter vertices and the PolygonCollider2D points are changing accordingly collisions data and deformation parameters.

Deformation parameters can be stored in Presets for convenient editing parameters of all the objects with concrete preset name or stored regardless.

Asset has deformation parameters for quick setup many types of simulated objects, such as:

- Metal objects;
- Water (waves, streams, waterfalls);
- Soil (mud, clay);
- Vegetation (grass, trees, bushes).

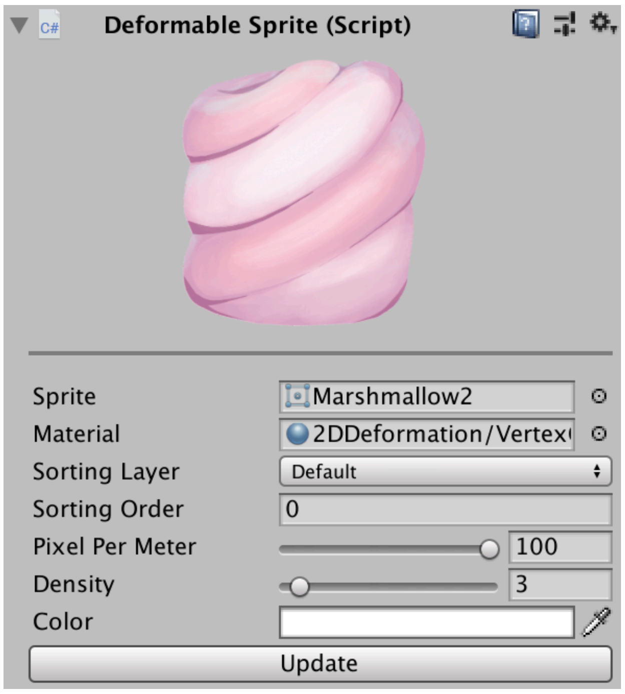
DeformableSprite parameters

DeformableSprite component creates a mesh using the MeshFilter, the MeshRenderer.

Consider the following parameters of the DeformableSprite component:

- Sprite - the source sprite;
- Material - material of the object;
- Sorting layer and Sorting order - sort by layer and depth;
- Pixel Per Meter - analogue Sprite's pixelPerUnit;
- Density - vertex density, affects the number of the vertices that will be generated. It is recommended to set the density value in that way: approximately each point of the PolygonCollider2D lays on the separate triangle. Shaded Wireframe mode in the Scene window can help to achieve that;
- Color - sprite color.

The "*Update*" button updates the sprite parameters.



Deformable Sprite (Script)

Sprite	Marshmallow2
Material	2DDeformation/Vertex(
Sorting Layer	Default
Sorting Order	0
Pixel Per Meter	100
Density	3
Color	

DeformableObject parameters

DeformableObject changes MeshFilter vertices and PolygonCollider2D points using an instance of the PolygonDeformationSystem class to simulate deformations. Deformation parameters has a few curves field for maximum customization and simulation many physical objects. Curves allow to create linear, non-linear and exponential models. Curves x-axis (time) are using as input-value, y-axis (value) are used as output-value.

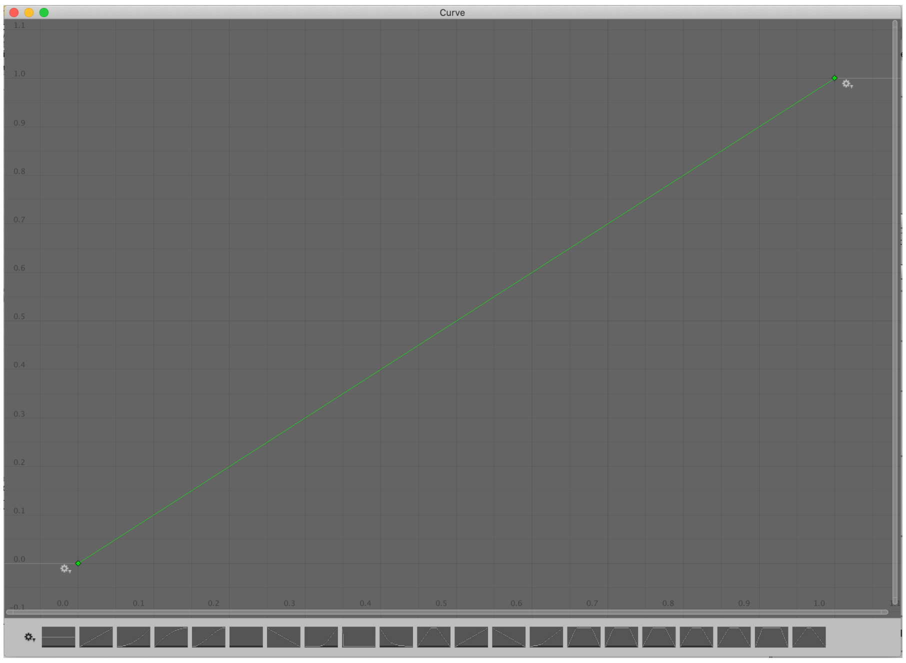
Impulse is the data of contact with colliders and contains: collision position, impulse value and direction.

Elasticity is an ability of object to restore an initial shape after deformation.

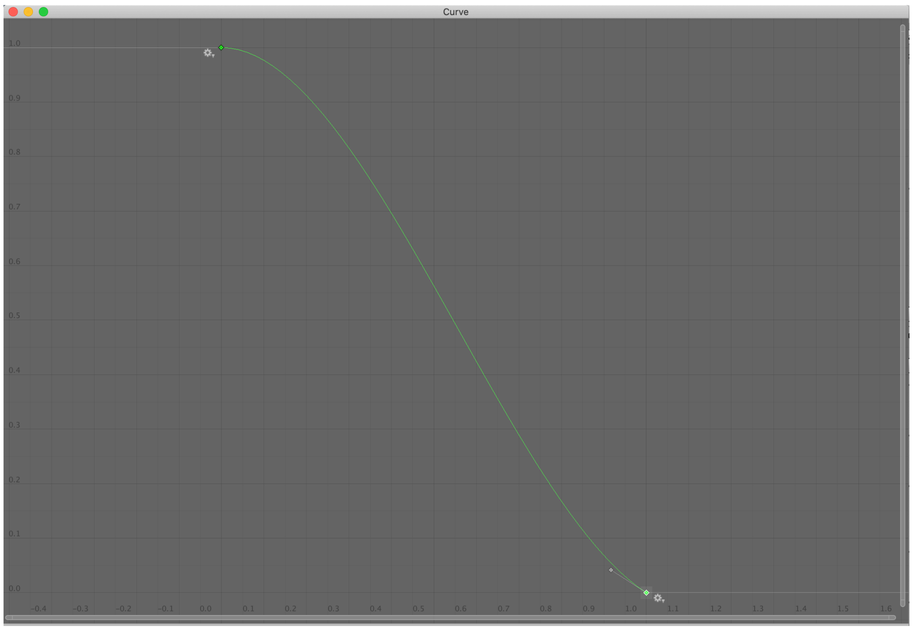
Resistance is an ability of object to resist vertex deformation when they are deformed already.

Consider the parameters of the DeformableObject component:

- **Preset** - deformation parameters preset. If Preset is set as "*Custom*", deformation parameters use from component, otherwise, deformation parameters loads from Presets ScriptableObject. Note that if some preset parameters are changed, other objects parameters with the same preset will be changed too;
- **Mass** - a parameter that affects the level of deformation of the object, the higher value means the less deformation of the object. Note that Rigidbody2D.mass parameter has different meaning;
- **Hardness** - a parameter describing the hardness of the object, the higher means less external forces impact to the object;
- **Impulse Curve** - a curve describing the affect of external impulses on the object. For an example: to simulate metal object, where small impulses (less than 24) should not affect the object, impulse ratio growing from 0 value at impulse 24 to 1 value at impulse 32 and greater (it's x-axis - input impulse value, y-axis - output value):

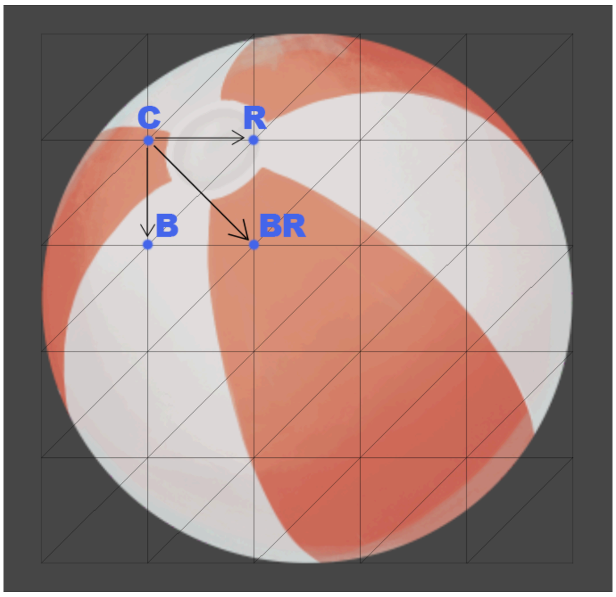


- **Impulse Ratio** - a ratio of Impulse Curve value;
- **Force Damping** - a ratio of the force damping. For example, value is 0, forces applies in one frame, if value is 0.9, force applies in each frame with that value until forces fades. It can be used for smooth deformation;
- **Force Curve** - a curve describing an affect of external forces on the object vertices. As an example, when object hits DeformableObject, it gets impulses, but it has some physical parameters that should deform only nearly vertices from the hit point. Force Curve describes how external forces should affect the vertexes, takes the input parameter - the distance from the collision to the each vertex. As an example, zero-distance from the hit point to vertex gets full force, one meter distance gets zero force, more than one meter won't get any external force:



- **Force Ratio** - a ratio of the Force Curve value;
- **Elasticity** - an elasticity value, means that object seeks to restore initial shape when value larger than 0. Larger value means faster shape restoring;
- **Damping** - a ratio for decreasing velocity of each frame;
- **Elasticity Curve** - a curve describing an affect of Elasticity on the object. Can be used for simulating object that have elasticity (rubber, jelly, vegetation, etc). ElasticityCurve is used to change elasticity value when vertices moved to some distance - it can be marshmallow that should decrease elasticity value when it clench, as an example. Curve minimal value is 0, when x-axis (time) - it is different between neighbor vertices distance (C-R, C-B, C-BR) and start neighbor vertices distance (C-R, C-B, C-BR).

Vertices iteration step for vertex (1:1), where C - current vertex, R - right vertex, B - bottom vertex, BR - bottom right vertex:



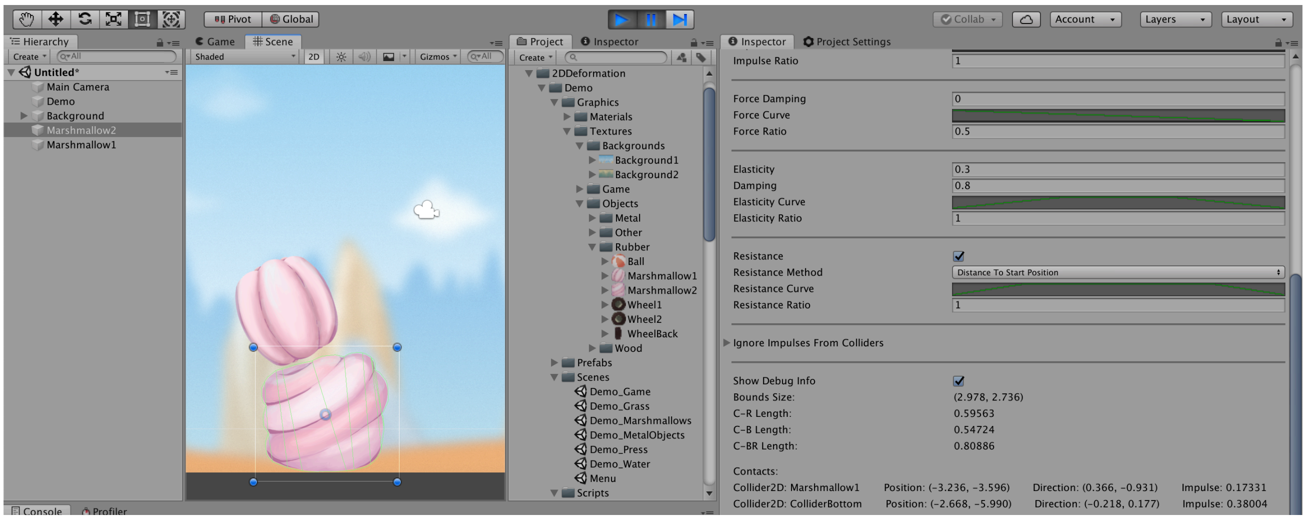
- **Elasticity Ratio** - a ratio of the Elasticity Curve value;
- **Resistance** - whether object vertices have resistance;
- **Resistance Method** - a resistance method: <u>Distance To Center</u> or <u>Distance To Start Position</u>.

<u>Distance To Center</u> means that vertex resists using information about the difference between the initial vertex distance to mesh center and the current vertex distance to mesh center. Distance information will be send to Resistance Curve.

<u>Distance To Start Position</u> means that vertex resists using information about the difference between current vertex position and start vertex position. Distance information will be send to Resistance Curve;

- **Resistance Curve** - a curve describing the resistance of external impulses on the object. It can be used for simulating durable metal object, which cannot be compressed to the very small object and should resists to the external forces after compressing. Curve x-axis (time) receive vector length - vertex position minus start vertex position when **ResistanceMethod** set as <u>DistanceToStartPosition</u>, when **ResistanceMethod** set as <u>DistanceToCenter</u> x-axis (time) receive difference between vertex distance to mesh center and start distance to mesh center;
- **Resistance Ratio** - a ratio of the Resistance Curve value;
- **Ignore Impulses From Colliders** - a list of colliders that should ignore impulses;
- **Show Debug Info** - debug information of the DeformableObject: <u>Mesh</u> <u>bounds</u> size, distances from current vertex to right vertex (C-R), from current vertex to bottom vertex (C-B), from current vertex to bottom right vertex (C-BR), Contacts data (Collider2D, Position, Direction, Impulse). Debug information shows only in Unity Editor and helps to setup object DeformationParameters (such as ForceCurve) to correctly describe of force spreading with deformation. As an example, a ball has vectors C-R, C-B with length equals 0.6 meters, vector C-BR with length equals 0.8 meters. Debug info also contains Contacts data:
  - Collider2D GameObject name;

- Position - position of contact;
- Direction - direction of contact;
- Impulse - impulse of contact.



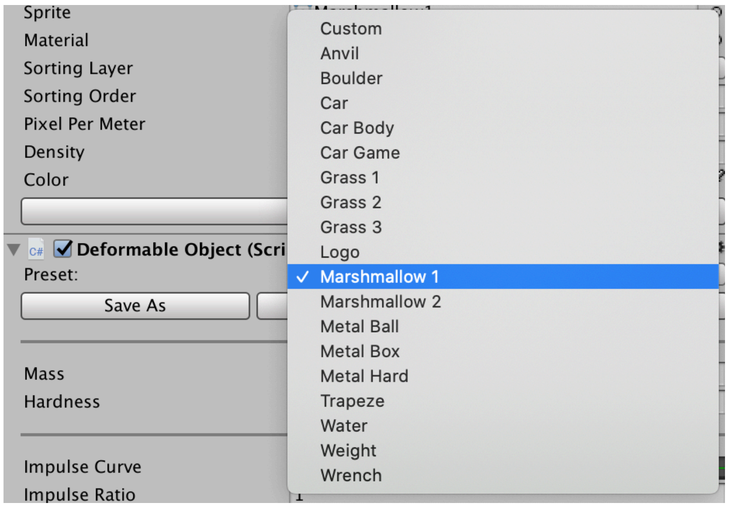
```
Impulse Ratio              1

Force Damping              0
Force Curve
Force Ratio                0.5

Elasticity                 0.3
Damping                    0.8
Elasticity Curve
Elasticity Ratio           1

Resistance                 ✓
Resistance Method          Distance To Start Position
Resistance Curve
Resistance Ratio           1

▶ Ignore Impulses From Colliders

Show Debug Info            ✓
Bounds Size:               (2.978, 2.736)
C-R Length:                0.59563
C-B Length:                0.54724
C-BR Length:               0.80886

Contacts:
Collider2D: Marshmallow1     Position: (-3.236, -3.596)   Direction: (0.366, -0.931)   Impulse: 0.17331
Collider2D: ColliderBottom   Position: (-2.668, -5.990)   Direction: (-0.218, 0.177)   Impulse: 0.38004
```

Contacts data helps to setup Curves parameters and tests collisions with objects to determine physical behaviour.

Presets

Assets has preset system to store deformation parameters in ScriptableObject, that allow to reuse them with many DeformableObject's. Presets located in the file at: Assets/2DDeformation/Resources/DeformationPresets.asset. User can save deformation parameters with name and apply it to other objects from the list. If user changes non-custom parameters of DeformableObject, data will be written to Presets and all objects with the same selected preset load new deformation parameters automatically.

Asset has built-in presets that can be reused or modified.

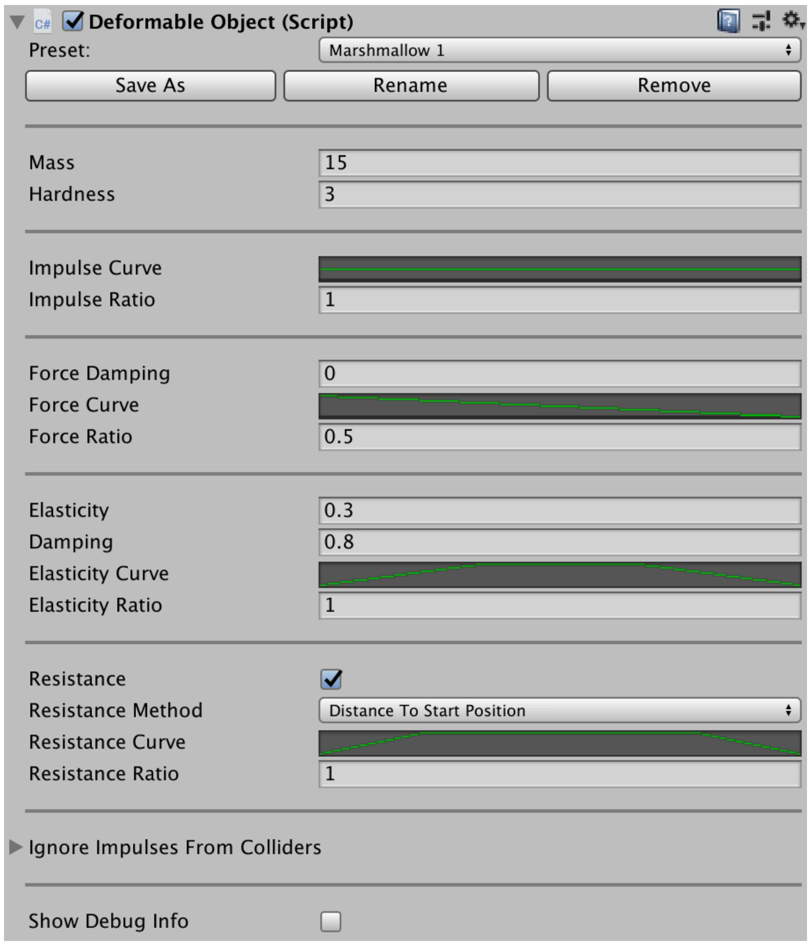


Preset: ✓ Marshmallow 1

Preset can be re-saved to a new preset, for this click on “*Save As*” button.

Preset can be renamed, for this click on “*Rename*” button and choose a new name and click on “*Save*” button.

Preset can be removed, for this click on “*Remove*” button and confirm it using dialogue box.



Preset:	Marshmallow 1
Mass	15
Hardness	3
Impulse Ratio	1
Force Damping	0
Force Ratio	0.5
Elasticity	0.3
Damping	0.8
Elasticity Ratio	1
Resistance	✓
Resistance Method	Distance To Start Position
Resistance Ratio	1
Show Debug Info	☐

DensityPolygonCollider2D

DensityPolygonCollider2D component designed for increasing the number of points of PolygonCollider2D using interpolation between neighbor points. If PolygonCollider2D shape generated by Unity does not have enough points for MeshFilter or mesh should have more points for deformation, DensityPolygonCollider2D can increase number of points. It has two modes:

- Constant - adds points every N meters between points;
- Divide - adds N points between every two points.

DeformableObject script

DeformableObject changes MeshFilter vertices and PolygonCollider2D points using an instance of PolygonDeformationSystem class to simulate deformations. DeformableObject can deform only existing PolygonCollider2D points, it does not add new points.

DeformableObject script has public fields:

`public delegate void OnDeformationHandle (Vector3 contactPoint, Vector3 direction, float impulse)` - delegate for OnDeformation event;

`public event OnDeformationHandle OnDeformation` - event of deformation;

`public DeformationParameters Parameters` - object deformation parameters. If preset with the same GUID is found in Presets, parameters will be loaded from Presets;

`public Collider2D[] IgnoreImpulsesFromColliders` - array of `Collider2D`. DeformableObject ignores getting impulses from this colliders;

DeformableObject script has public methods:

`public void AddImpulse(Vector3 contactPoint, Vector3 direction, float impulse)` - adds impulse with direction to contactPoint. Can be used for object deformation from the code;

`public void SetDeformationSystemParameters()` - sets deformation parameters to an instance of PolygonDeformationSystem class;

`public List<Contact> GetContacts()` - returns all contacts saved in OnCollision method.

DeformableSprite script

DeformableSprite creates a mesh using MeshFilter, MeshRenderer.

DeformableSprite script has public methods:

`public void GetVerticesDimension(out int horizontal, out int vertical)` - sets vertices count by horizontal and vertical in argument values;

`public void ForceUpdate()` - updates the created mesh for MeshFilter, material parameters and MeshRenderer.

DensityPolygonCollider2D script

DensityPolygonCollider2D component designed for increasing the number of points of PolygonCollider2D using interpolation between neighbor points.

DensityPolygonCollider2D script has public fields:

`public DensityPlacementMethod ColliderDensityType` - mode of adding extra point to PolygonCollider2D, where

– Constant - adds points every N meters between neighbor points;
– Divide - adds N points between every two points;

`public int ColliderDensity` - density of PolygonCollider2D points, used when `ColliderDensityType` set as `DensityPlacementMethod.Divide`;

`public float ColliderDensityDistance` - minimal distance between two points of PolygonCollider2D, used when `ColliderDensityType` set as `DensityPlacementMethod.Constant`;

`public bool UsePhysicalShape` - whether to use Sprite physical shape generated by Unity;

DensityPolygonCollider2D script has public method:

`public void UpdateCollider()` - update PolygonCollider2D points.

Anchor script

Anchor component designed to attach object to MeshFilter triangle. Can be used for attaching AnchoredJoint2D or object Transform.

Anchor script has public fields:

`public void MeshFilter` - MeshFilter for attaching object;

`public AnchoredJoint2D AnchoredJoint2D` - AnchoredJoint2D for attaching, can be empty;

`public Vector3 AttachPoint` - attach point, can be changed by moving Handle from Scene window or from Inspector tab;

`public bool UseOffset` - whether to use initial position as offset to AttachPoint position.

PolygonDeformationSystem class

PolygonDeformationSystem class contains vertices object deformation logic - changing vertices of MeshFilter and PolygonCollider2D points.

PolygonDeformationSystem class has public methods:

`public void SetParameters(DeformationParameters deformationParameters)` - sets deformation parameters;

`public void Deform(Vector2 contactPoint, Vector2 direction, float impulse)` - adds impulse with direction to contactPoint;

`public void Update()` - updates MeshFilter vertices and PolygonCollider2D points.

Objects modeling

For modeling such surfaces as metal, water, cloth, vegetation, soil and others, user can use build-in presets or create new. This is recommended parameters, they can be ignored depends on the desired objects behaviour.

For creating **metal** objects use next parameters:

- Mass - from 40 to 120;
- Hardness - from 50 to 100;
- Elasticity - 0;
- Damping - 1;
- Resistance - true;
- Rest parameters can be setup optionally;

For creating **water** objects use next parameters:

- Mass - from 4 to 24;
- Hardness - from 10 to 20;
- Elasticity - 1;
- Damping - ~0.01;
- Resistance - false;
- Rest parameters can be setup optionally;

For creating **vegetation** objects use next parameters:

- Mass - from 4 to 12;
- Hardness - from 15 to 35;
- Elasticity - more than 0.7, less than 0.95;
- Damping - less than 0.2, more than 0.05;
- Resistance - false;
- Rest parameters can be setup optionally;

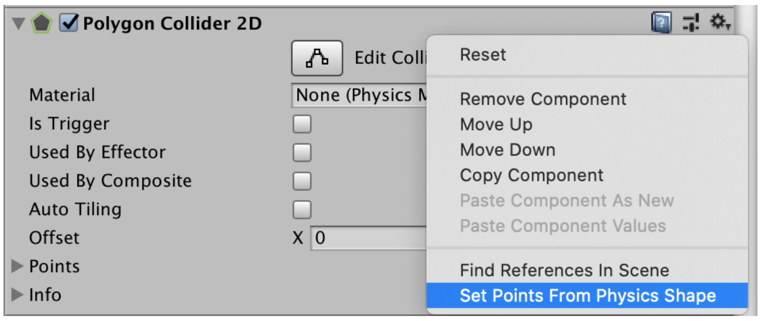
For creating **rubber** objects use next parameters:

- Mass - from 10 to 30;
- Hardness - from 2 to 20;
- Elasticity - more than 0.1, less than 0.95;
- Damping - less than 0.5, more than 0.1;
- Rest parameters can be setup optionally;

Tips

- Make sure that Rigidbody2D component has correct mass value: small mass value can change the behavior of objects so that objects with the large mass value will pass through objects with the small mass. This trick can be used to stimulate grass/ bushes;

- DeformableObject Debug info shows information about the object and contacts in runtime. This information can help setup object DeformationParameters, such as ForceDampingCurve to correctly describe force spreading after deformation, etc;
- Mesh triangles can overlap each other, it is normal behavior and this was done in order to reduce performance issues, especially on mobile devices. To fix it, check DeformationParameters and resistance configuration;
- For convenient reuse DeformationParameters curves, user can save curves presets. Unity also have manuals how to work with curves: Editing Curves and Using Animation Curves;
- User can reset PolygonCollider2D points by pressing on Context Menu -> Set Points From Physics Shape:



Material None (Physics M
Is Trigger ☐
Used By Effector ☐
Used By Composite ☐
Auto Tiling ☐
Offset X 0

Please let me know if you have any questions.
E-mail: unitymedved@gmail.com