What is Tracing and Writing?

It is an easy-to-use asset that allows you to trace and fill objects using trace paths from lines, curves and dots. All that you need to do is to add prefab on the scene, set points of curves, and a few sprites!

You can use the asset to create your shapes or different characters for the tracing using Lines, Curves, and Dots. It includes a set of built-in Editor used to manage and create your own shapes easily.

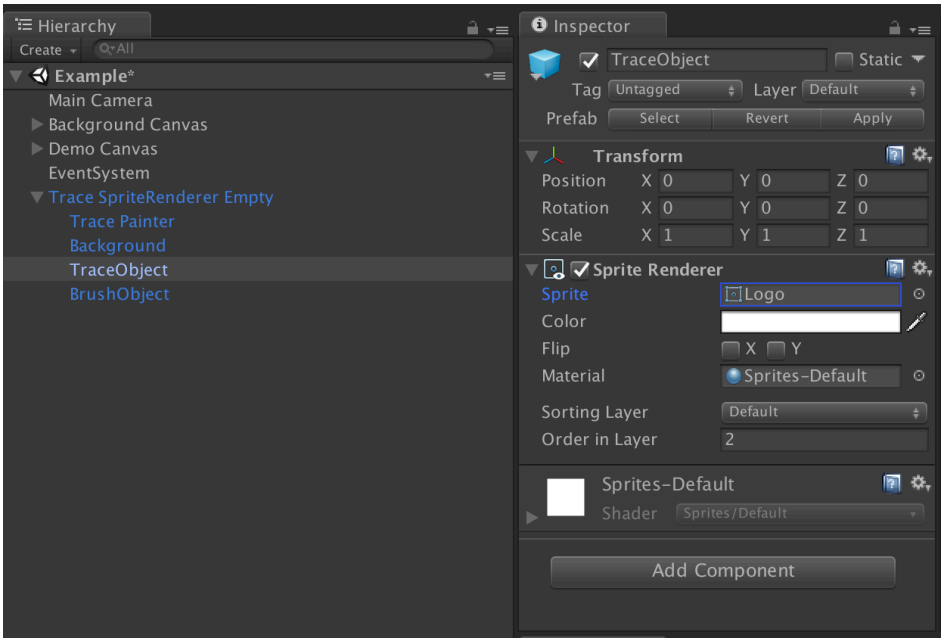
Quick start

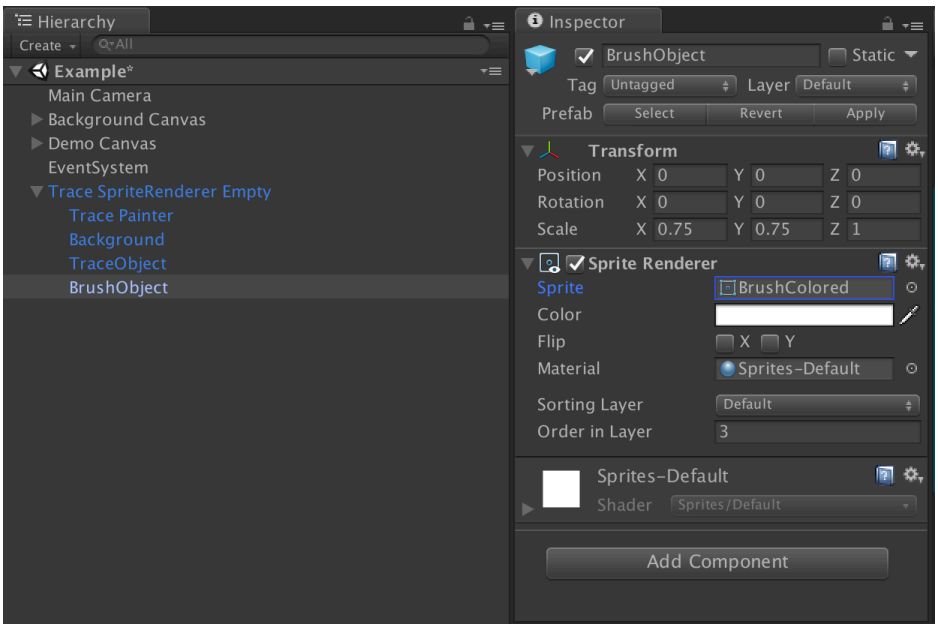
As it is mentioned above, add a prefab on the scene from the folder: "**Assets/TraceCurve/Prefabs/**". In this folder are located three prefabs:

- **Trace Canvas Empty**
- **Trace MeshRenderer Empty**
- **Trace SpriteRenderer Empty**

These prefabs are intended to work with three types of graphic components: **Trace Canvas Empty** - working with Unity UI Canvas, **Trace MeshRenderer Empty** - working with component MeshRenderer, and **Trace SpriteRenderer Empty** - working with component SpriteRenderer. For example, add prefab "Trace SpriteRenderer Empty" on the scene.

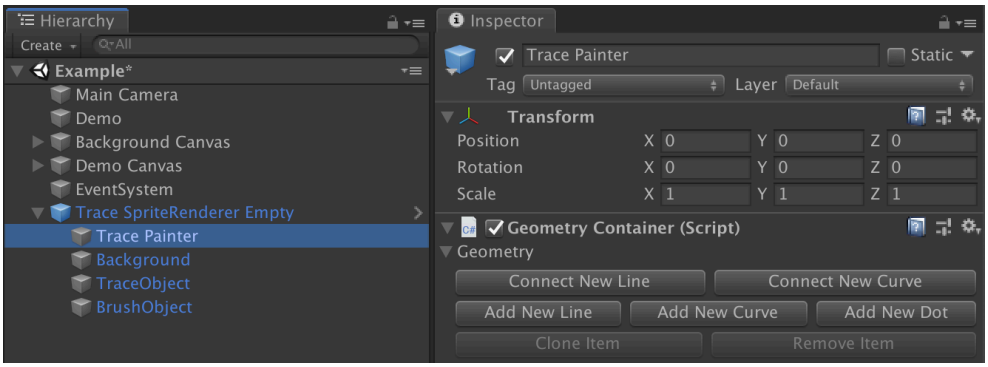
After adding prefab set sprite of trace object and brush, for that set them sprite or texture in the Inspector tab:





Object TraceObject contains a sprite, which will be displayed as the user traces/draws, the BrushObject object contains a graphic element to display the current progress of the tracing/drawing - the brush. These objects are necessary for the TracePainter component for initialization and further interactions.

Next set the curves/lines/dots for the user tracing/drawing path. To do this, click on the GameObject, which contains the component “GeometryContainer”:



Consider the "GeometryContainer" component in more detail. The component stores the geometric shapes for the trace path. Two types of objects can be used as shapes: lines, curves, and dots. Lines represent an object with the position of the beginning of the segment and the position of the end. Curves represent an object with a start position, an initial tangent position, an end tangent position, and an end position. Dots represent an object with a position to click on (for example, it can be a lowercase “i” letter with a dot at the top). Add the corresponding object using the keys in the Inspector window:

- **Connect New Line** - creates a new line whose beginning is the end of the previous object;
- **Connect New Curve** - creates a new curve, the beginning of which is the end of the previous object;

- **Add New Line** - creates a new line, independent of previous object;
- **Add New Curve** - creates a new curve independent of previous object;
- **Add New Dot** - creates a new dot independent of previous object;
- **Clone Item** - clones an existing line or curve selected in the Inspector window;
- **Remove Item** - deletes an existing line, curve or dot selected in the Inspector window.

Hold down the Option, Alt, or Shift key and click on the add/clone object button - the item will be added after the selected item, instead of being added to the end of the list.

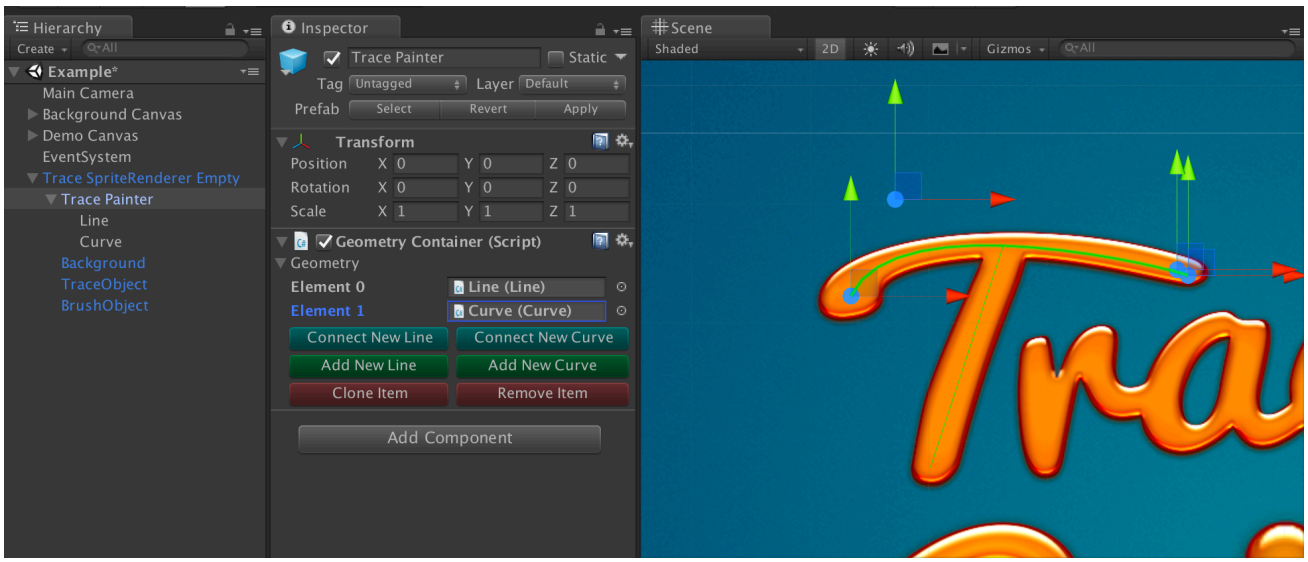
Using the pressed Option, Alt, Shift, or Control key allows moving the selected object in the Scene window.

Added objects are displayed in the Scene window at default positions and must be moved to create a trace path.

To change the position of an object, click on the corresponding element in the "**Geometry**" list of the "**GeometryContainer**" component, as a result, its handles will be displayed on the scene to change the positions that form the geometry of the object.

It should be remembered that the geometry of objects may be discontinuous and objects may be present that are not connected to each other.

Example: 2 objects were added - Line and Curve, which are not connected to each other, but represent the trace path of the letter "T":



TracePainter script

TracePainter contains the logic for creating a RenderTexture in R8 format and transforming world coordinates into texture drawing coordinates. Drawing in RenderTexture occurs via **TraceBrushRenderer** class.

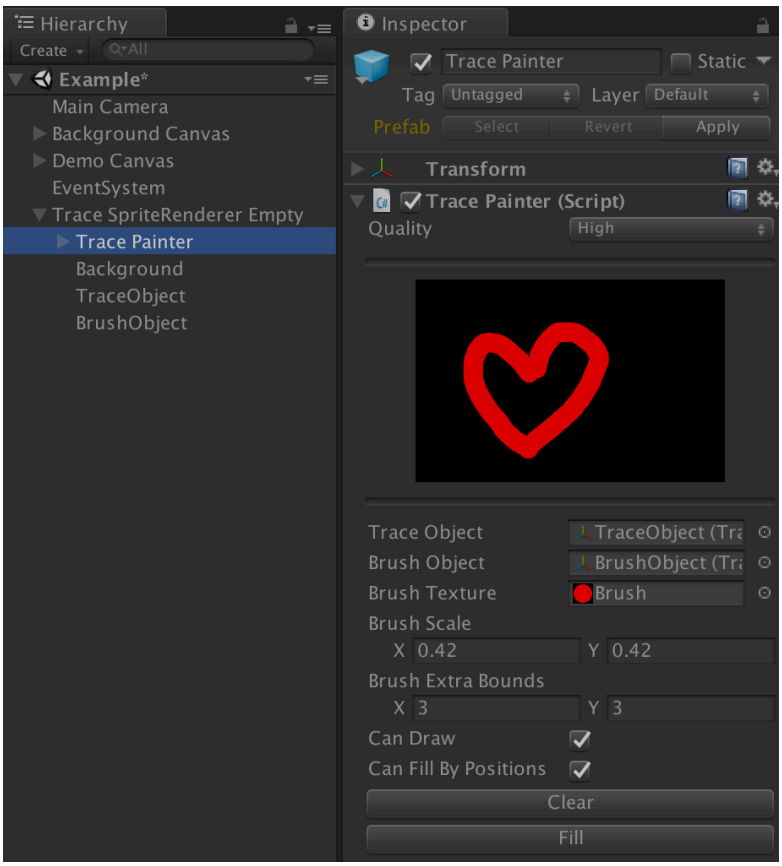
Variables:

- **public Quality RenderTextureQuality** - the quality of the RenderTexture, affects the size of the created RenderTexture, where High is the initial size of the texture of the TraceObject field, Medium is half, Low is quarter;
- **public RenderTexture RenderTexture** - RenderTexture for masking TraceObject;
- **public Transform TraceObject** - object for un/masking using RenderTexture;
- **public Transform BrushObject** - paint/trace brush object;
- **public Texture BrushTexture** - texture in R8 format for drawing on RenderTexture;
- **public Vector2 BrushScale** - scale of the BrushTexture texture, affects the size of the drawing/tracing area;
- **public Vector2 BrushExtraBounds** - additional size of BrushTexture bounds, used to change the pressed area size of the brush;
- **public bool CanDraw** - is it possible to draw/update RenderTexture;
- **public bool CanFillByPositions** - it is possible to draw on RenderTexture by changing the position of BrushObject. It can be used for tracing without using a path using lines/curves/dots;
- **public bool SetBrushAngleFollowPath** - whether to rotate brush following trace path;
- **public float BrushAngleOffset** - brush angle offset;
- **public RectTransform BrushRectTransform** - RectTransform component of the brush. Used when working with Canvas;
- **public Vector2 BrushBoundsSize** - brush sizes in world coordinates;
- **public bool IsCanvasOverlay** - whether RenderMode.ScreenSpaceOverlay is used by Canvas;
- **public bool UpdateOnce** - whether draw in RenderTexture one time in next MonoBehaviour.Update() invoking.

Methods:

- **public void Fill()** - setting red color for RenderTexture;
- **public void Clear()** - setting of black for RenderTexture;
- **public void AddToRenderQueue(Vector2[] positions)** - adding lines positions to the queue for drawing in Update method when CanDraw flag is active.

Using the texture along the path "**Assets/TraceCurve/Demo/Graphics/Textures/BrushInverse**" as BrushTexture can erase the drawn. The default texture is the path "**Assets/TraceCurve/Demo/Graphics/Textures/Brush**". Textures for painting should be in R8 format.



It should be distinguished that BrushTexture is the texture used to paint on RenderTexture, and BrushObject is the graphic display of the brush, which is not used while painting on RenderTexture, but is used only to find the position of painting.

GeometryContainer script

The GeometryContainer contains lines, curves, and dots data for tracing the path and transforms them according to their position, rotation, and scale.

Variables:

- public List<Geometry> Objects - data about lines, curves, and dots;
- public List<GeometryData> SegmentsData - transformed data about lines, curves and dots;
- public bool UpdateSegmentsData - whether to update the transformed data in the Update method once.

Methods:

- public void UpdateSegments() - transformation of lines/curves/dots data.

TraceInput script

TraceInput processes user input and contains Drag&Drop logic for the brush contains events and handlers.

Variables:

- public Camera Camera - scene camera used to transfer from screen coordinates to world coordinates;
- public TracePainter TracePainter - link to the component TracePainter;
- public GeometryContainer GeometryContainer - link to the component GeometryContainer;
- public TraceBrushMoverBase BrushMover - link to the component TraceBrushMoverBase;
- public Move MoveDirection - the direction of the trace when using Drag&Drop, can take the values: Move.Forward to trace forward, Move.Backward to trace back;
- public float ConnectRadius - the radius of the brush connection with the vertices of the lines using Drag&Drop, is used to move the brush approaching the vertices of the lines;
- public float BreakRange - the radius from the brush position to the offset position using Drag&Drop, at which Drag&Drop is interrupted. Using the value -1, Drag&Drop won't be interrupted;
- public bool UpdateOnDown - whether to draw brush on Input.GetMouseButtonDown(0)/TouchPhase.Began events.

Events:

- public event Handler OnBrake - interruption of movement while increasing the value is above the permissible radius BreakRange;
- public event GeometryHandler OnGeometryStart - start of line/curve/dot tracing;
- public event GeometryHandler OnGeometryFinish - end of line/curve/dot tracing;
- public event GeometryHandler OnGeometryCancel - user trace interruption;
- public event GeometryHandler OnAllGeometryFinish - completion of tracing of all lines/curves/dots;
- public event ProgressHandler OnProgress - trace progress.

Methods:

- public void SetBrushPosition() - setting the brush position to the position of the current vertex of the line/curve/dot;
- public void SetProgress(int geometry, int point) - setting the index of the current geometry object and the vertex index;
- public void InitRanges() - line/curve/dot length calculation, used for the ProgressHandler OnProgress event.

TraceBrushMoverBase script

TraceBrushMoverBase - basic class of brush moving logic at the end of an intermittent object. For implementing the new logic to move the brush, class should used as a base class.

Variables:

- public Action OnMoveStarted - delegate at the beginning of the movement of the brush;
- public Action OnMoveFinished - delegate at the end of the brush movement;
- public bool IsBusy - property whether the brush moves;
- protected Transform BrushTransform - Transform of the brush;
- protected Vector3 From - starting position of movement;
- protected Vector3 To - ending position of movement;
- protected bool MoveStarted - whether movement of the brush started.

Methods:

- public virtual void Init(Transform t) - initialization of the object, it takes as an argument Transform of the brush;
- public virtual void StartMove(GeometryData geometryFrom, Vector3 positionFrom, GeometryData geometryTo, Vector3 positionTo) - start brush movement;
- public abstract void Move() - brush moving logic.

TraceBrushMover script

TraceBrushMover inheritor to TraceBrushMoverBase, contains the logic for moving the brush when using interrupted lines/curves/dots from the end of the previous object to the beginning of the next at a given speed.

Variables:

- public float Speed - speed of movement.

TraceFiller script

TraceFiller - component for filling TracePainter object using progress value (from 0 to 1) or index of geometry.

Variables:

- public TracePainter TracePainter - link to the component TracePainter;
- public GeometryContainer GeometryContainer - link to the component GeometryContainer;

Methods:

- public void Init() - initialization of the component;
- public void UpdateProgress(float progress, out int geometry, out int point) - update TracePainter filling progress, out values - index of geometry and point index;
- public void UpdateGeometryProgress(int geometryId) - update TracePainter filling progress by index of geometry.

TraceCanvasScaler script

TraceCanvasScaler updates line, curve and dot data while scaling Canvas.

TraceCanvasHelper script

TraceCanvasHelper sets the camera for Canvas, in case there is no link to the camera.

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