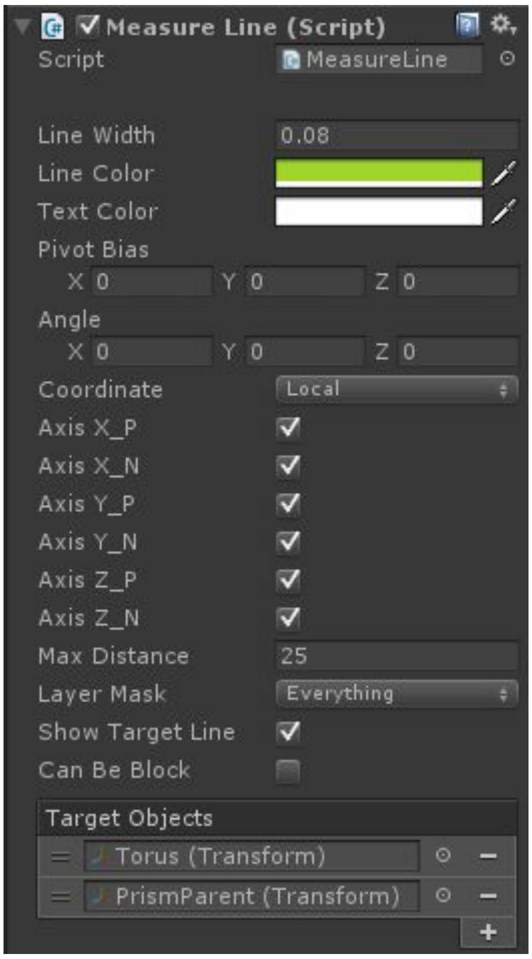
Thank you for choosing Measure Lines.

It's a program for dynamic measure distance between object.
Lines are base on mesh, draw as usual object.
You can use it on a GameObject with collider or a empty GameObject.
It use raycast to find out what's distance between objects.
I hope you enjoy the program and wish it can help you with your game.
If you have any question, you can email me: unicoea@gmail.com

How to use?

It's very simple to use, just attach it to your object like this.



Measure Line (Script)

Script	MeasureLine
Line Width	0.08
Line Color	🟩
Text Color	⬜
Pivot Bias	X 0 Y 0 Z 0
Angle	X 0 Y 0 Z 0
Coordinate	Local
Axis X_P	✓
Axis X_N	✓
Axis Y_P	✓
Axis Y_N	✓
Axis Z_P	✓
Axis Z_N	✓
Max Distance	25
Layer Mask	Everything
Show Target Line	✓
Can Be Block	☐

Target Objects
- Torus (Transform)
- PrismParent (Transform)

I will introduce the parameters of it:

"Line Width" : The line's width show in game window.

"Line Color" : The line's color.

"Text Color" : The distance text's color.

"Pivot Bias" : The bias of line's center,it's the pivot of GameObject by default, but when you have a complex mesh, the pivot may need to adjust.

"Angle" : The angle you can modify to all lines, you can use custom direction instead of normal Up/Down/Right/Left/Forward/Back.

“Coordinate” : Local or World, it will affect “Angle” only. The lines and pivot center will rotate with object when it’s Local.

“AxisX_P” : Enable to show AxisX_Positive lines, Disable to Hide them.

“AxisX_N” : Enable to show AxisX_Negative lines, Disable to Hide them.

“AxisY_P” : Enable to show AxisY_Positive lines, Disable to Hide them.

“AxisY_N” : Enable to show AxisY_Negative lines, Disable to Hide them.

“AxisZ_P” : Enable to show AxisZ_Positive lines, Disable to Hide them.

“AxisZ_N” : Enable to show AxisZ_Negative lines, Disable to Hide them.

“Distance” : Lines will be show only when distance below this value.

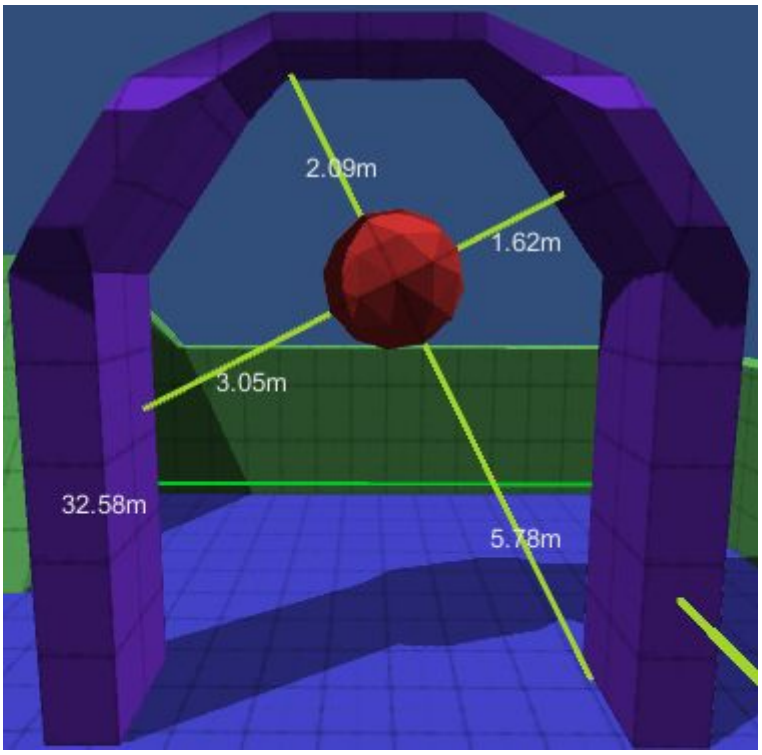
“LayerMask” : Sometimes we need to ignore some objects to measure the distance.

“Show Target lines” : Whether show lines between this object to targets or not.

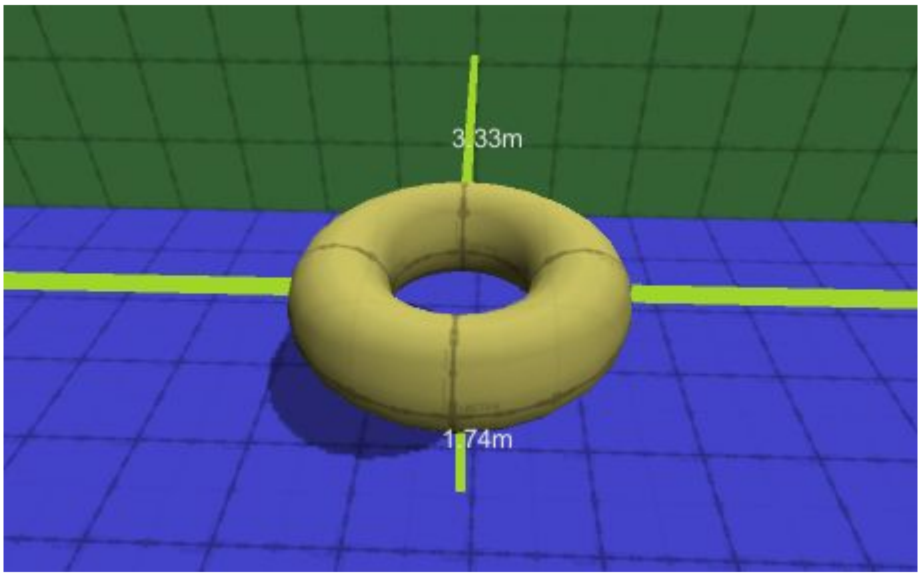
“Can Be Block” : Can the line between this object to targets be block by collider.

“Target Objects” : Distance can be measure by set target objects(Support change in runtime).

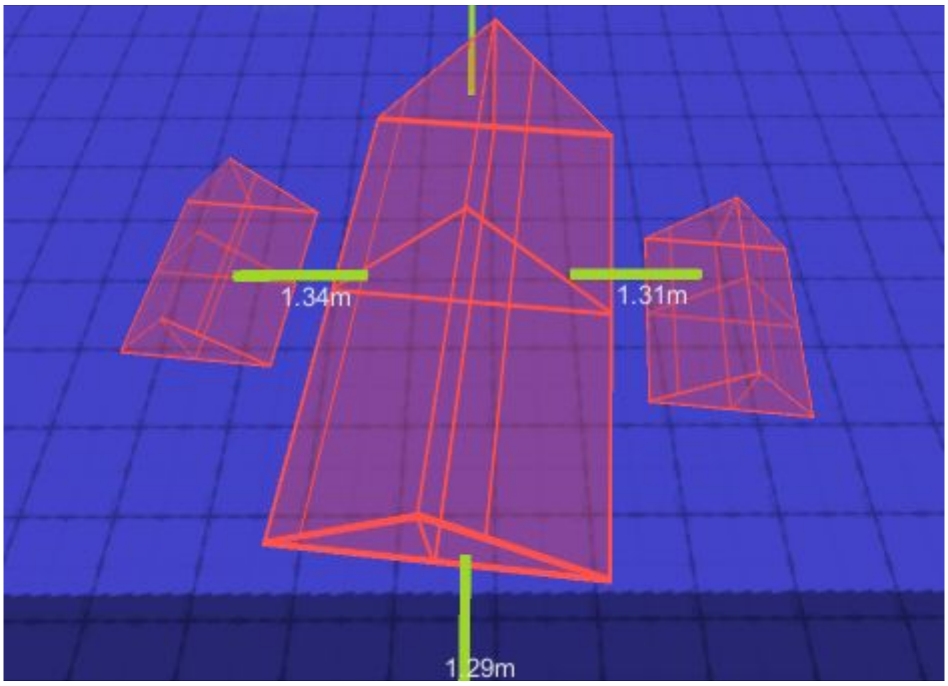
Let’s explain the Demo Scene:



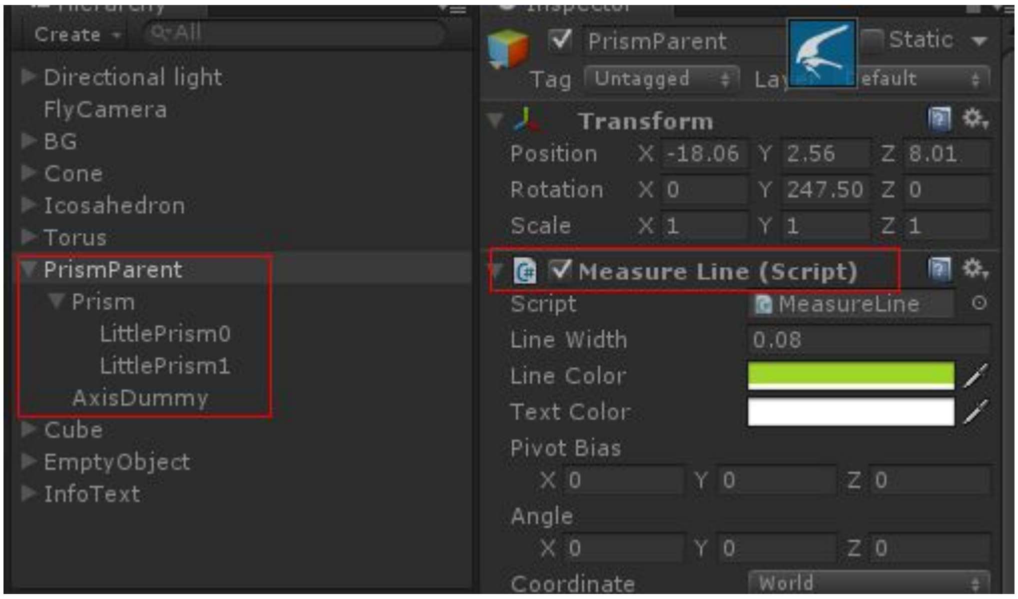
This Icosahedron show how to use local coordinate. When in local coordinate mode, the line will rotate and move with the object.



This Torus show the line will auto adapt the change of object's scale.

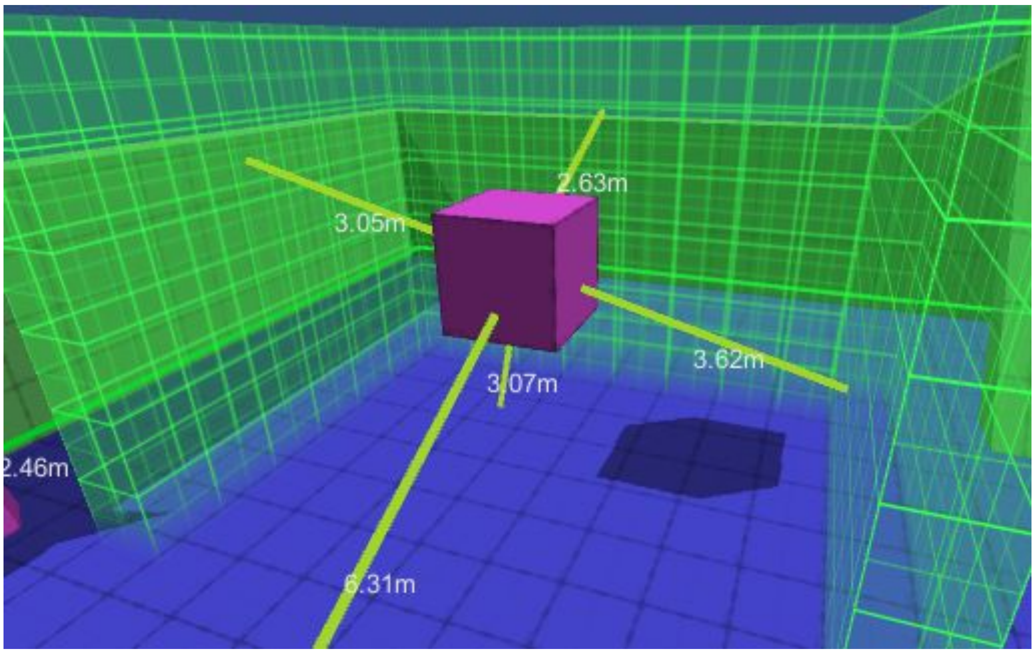


Two little prisms is the childs of Big Prism,and the MeasureLine.cs is attach to parent of them. And it use world coordinate mode,so the lines will not rotate with object, but still will move with it.

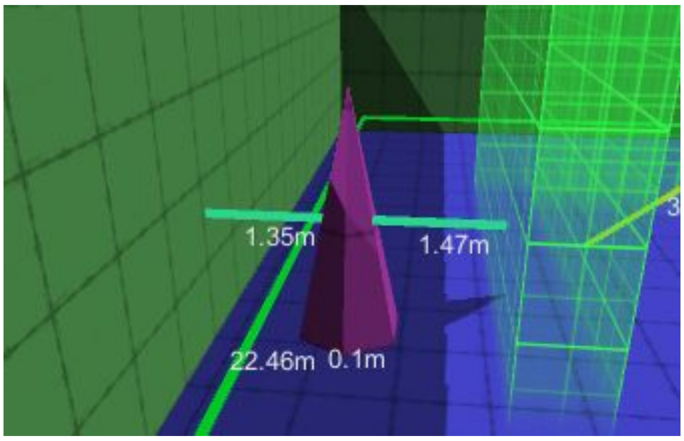

Create
Directional light
FlyCamera
BG
Cone
Icosahedron
Torus
PrismParent
Prism
LittlePrism0
LittlePrism1
AxisDummy
Cube
EmptyObject
InfoText

PrismParent Static
Tag Untagged Layer Default
Transform
Position X -18.06 Y 2.56 Z 8.01
Rotation X 0 Y 247.50 Z 0
Scale X 1 Y 1 Z 1
Measure Line (Script)
Script MeasureLine
Line Width 0.08
Line Color
Text Color
Pivot Bias
X 0 Y 0 Z 0
Angle
X 0 Y 0 Z 0
Coordinate World

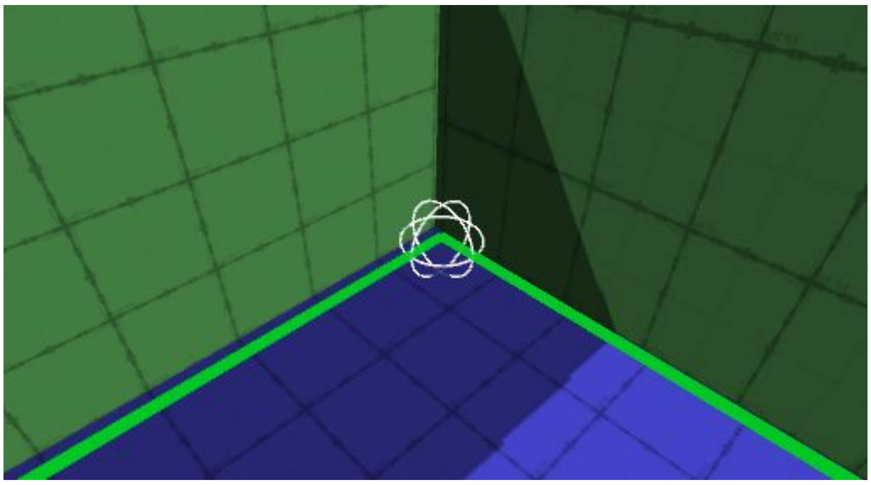
You can use a empty object to be the parent, and all child's collider can combine.



This one show how the "Angle" work,This cube is not rotate and you can adjust rotation of Lines through Angle parms.You can use "Pivot Bias" to move the center of line as well.

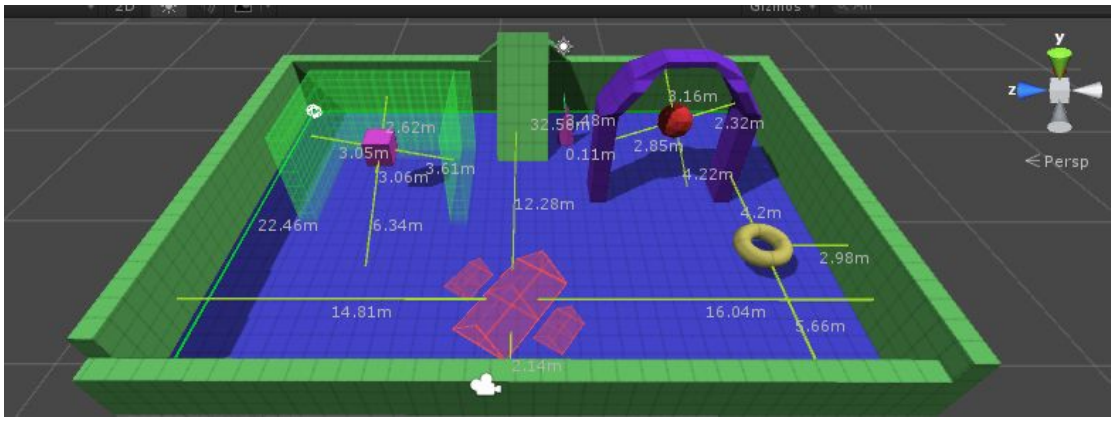


This little one show how to use “Distance”, it’s distance is set to 5,so the lines will only appear when get very close to other objects.



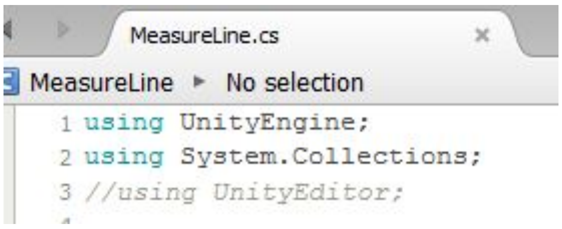
This one show Use it on an empty object, and disable some lines.

About Publish To Runtime

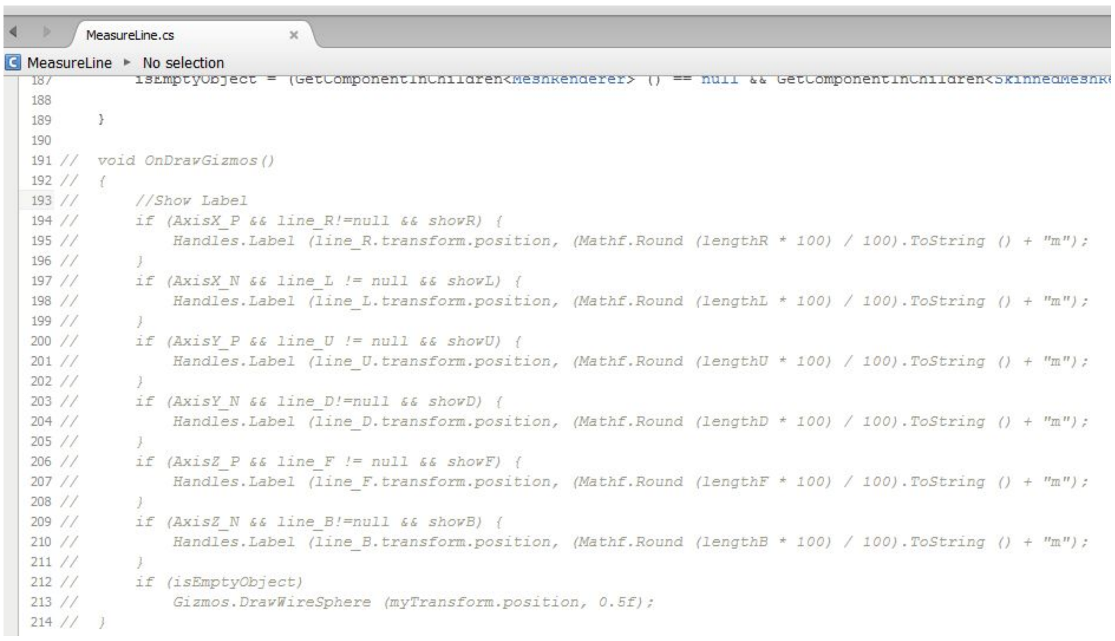


Important Note:

You can see value in either editor or game,but when you want to publish,you need to disable the using UnityEditor and OnDrawGizmos in MeasureLine.cs.



```
using UnityEngine;
using System.Collections;
//using UnityEditor;
```

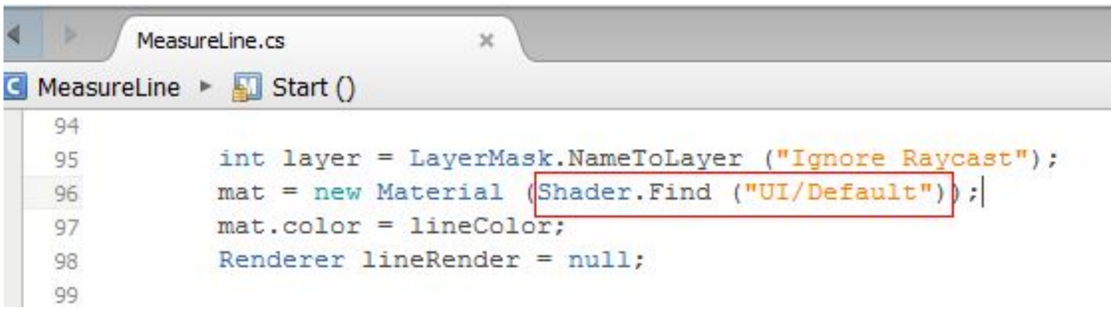


```
        isEmptyObject = (GetComponentInChildren<MeshRenderer> () == null && GetComponentInChildren<SkinnedMeshRe

    }

//  void OnDrawGizmos()
//  {
//      //Show Label
//      if (AxisX_P && line_R!=null && showR) {
//          Handles.Label (line_R.transform.position, (Mathf.Round (lengthR * 100) / 100).ToString () + "m");
//      }
//      if (AxisX_N && line_L != null && showL) {
//          Handles.Label (line_L.transform.position, (Mathf.Round (lengthL * 100) / 100).ToString () + "m");
//      }
//      if (AxisY_P && line_U != null && showU) {
//          Handles.Label (line_U.transform.position, (Mathf.Round (lengthU * 100) / 100).ToString () + "m");
//      }
//      if (AxisY_N && line_D!=null && showD) {
//          Handles.Label (line_D.transform.position, (Mathf.Round (lengthD * 100) / 100).ToString () + "m");
//      }
//      if (AxisZ_P && line_F != null && showF) {
//          Handles.Label (line_F.transform.position, (Mathf.Round (lengthF * 100) / 100).ToString () + "m");
//      }
//      if (AxisZ_N && line_B!=null && showB) {
//          Handles.Label (line_B.transform.position, (Mathf.Round (lengthB * 100) / 100).ToString () + "m");
//      }
//      if (isEmptyObject)
//          Gizmos.DrawWireSphere (myTransform.position, 0.5f);
//  }
```

Then you should check here:

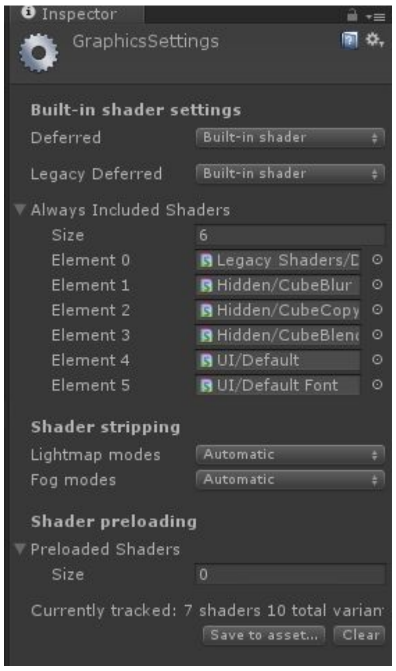


```

        int layer = LayerMask.NameToLayer ("Ignore Raycast");
        mat = new Material (Shader.Find ("UI/Default")) ;
        mat.color = lineColor;
        Renderer lineRender = null;

```

The measure lines use UI/Default as shader, so if you changed the shader name,you should Open Graphics Setting(Edit->Project Setting->Graphics) to ensure the shader is included.

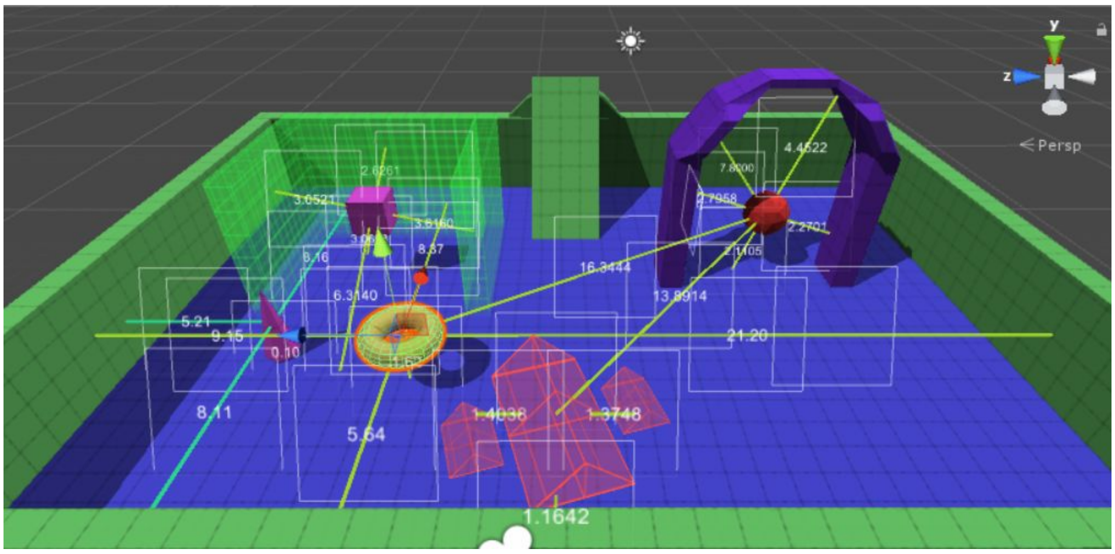


```
Built-in shader settings
Deferred                 Built-in shader
Legacy Deferred          Built-in shader
Always Included Shaders
   Size                  6
   Element 0             Legacy Shaders/D
   Element 1             Hidden/CubeBlur
   Element 2             Hidden/CubeCopy
   Element 3             Hidden/CubeBlen
   Element 4             UI/Default
   Element 5             UI/Default Font
Shader stripping
Lightmap modes           Automatic
Fog modes                Automatic
Shader preloading
Preloaded Shaders
   Size                  0
Currently tracked: 7 shaders 10 total varian
                         Save to asset...  Clear
```

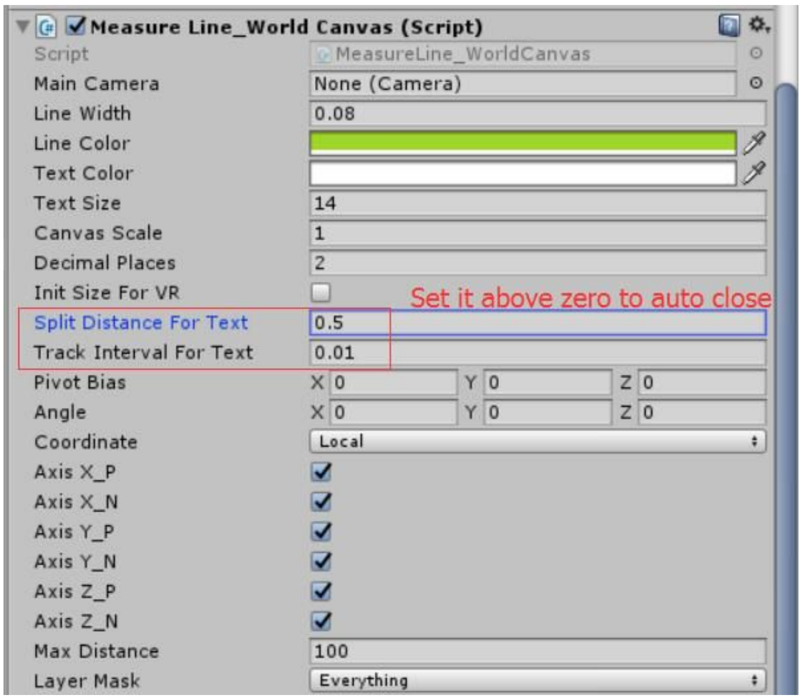
-New Feature for v1.3

MeasureLine_WorldCanvas.cs support for VR:

The text now is in world space canvas, so you can blend it for VR display.



Because in VR you need to adjust the size of text, so there provide a speed way, just check on "Init Size for VR", it will set some parameter for you in VR.



Measure Line_World Canvas (Script)

Script	MeasureLine_WorldCanvas
Main Camera	None (Camera)
Line Width	0.08
Line Color	🟩
Text Color	⬜
Text Size	14
Canvas Scale	1
Decimal Places	2
Init Size For VR	☐
Split Distance For Text	0.5
Track Interval For Text	0.01
Pivot Bias	X 0 Y 0 Z 0
Angle	X 0 Y 0 Z 0
Coordinate	Local
Axis X_P	✓
Axis X_N	✓
Axis Y_P	✓
Axis Y_N	✓
Axis Z_P	✓
Axis Z_N	✓
Max Distance	100
Layer Mask	Everything

Set it above zero to auto close

Text Auto Close To Camera:

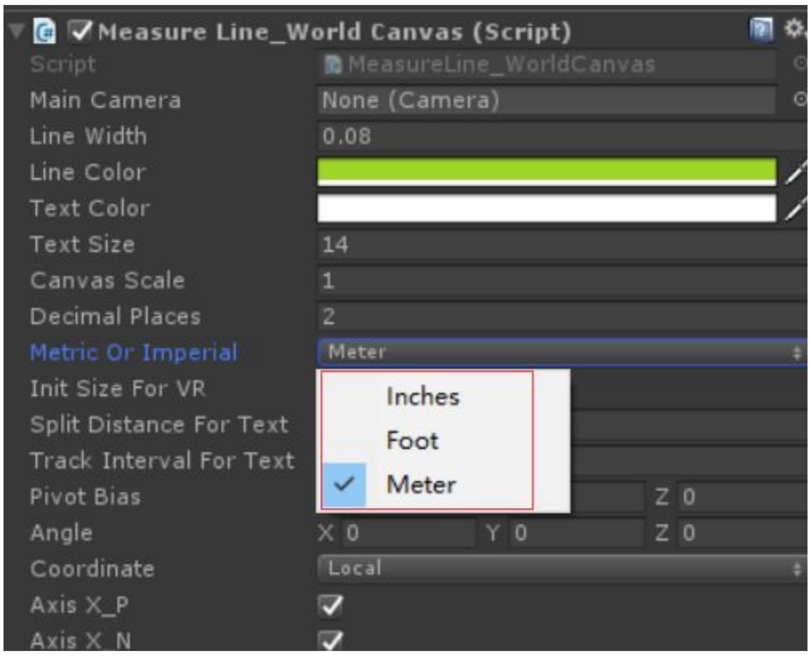
If a measureLine is too long, the text will be so far and user can't see the text, so there are a new feature for this: Text can keep themself close to MainCamera automaticly.

Set Parameter: "**splitDistanceForText**" and "**trackIntervalForText**":

"splitDistanceForText" is what distance to use to divide of a line(for prepare position for find closest one). It is suggest to **above 0.1f.**

"trackIntervalForText" is interval time when a uiText is update it's position for close you. It is suggest to **above 0.01f.**
The default "splitDistanceForText" is zero, so it will not try to close user automaticly. So set it when you need it.

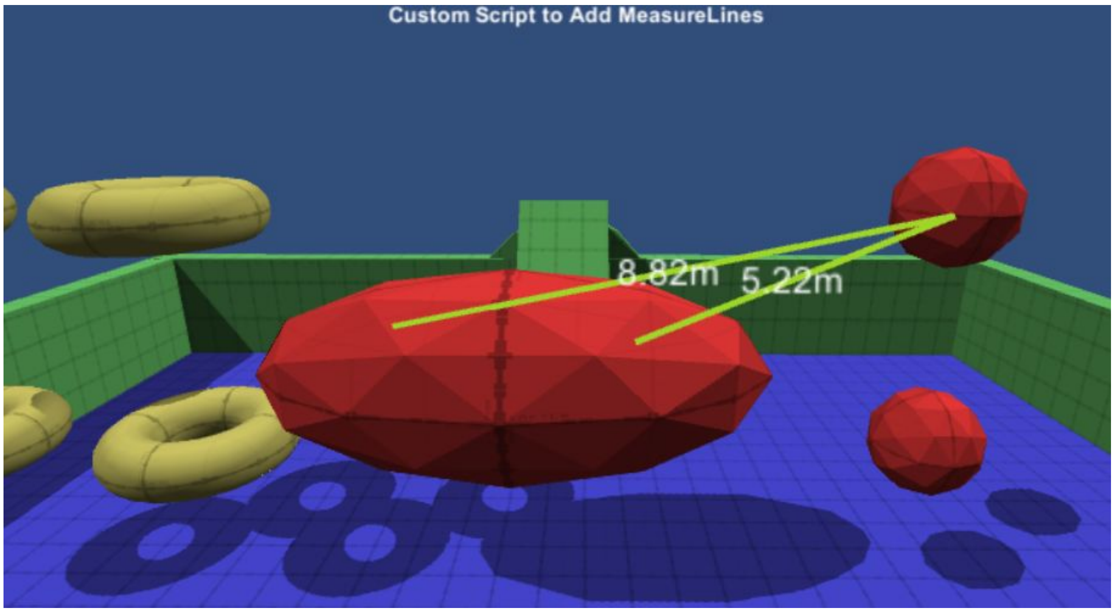
-New Addon and Change for v1.3c:
Now you can select Metric Or Imperial:



Measure Line_World Canvas (Script)

Script	MeasureLine_WorldCanvas
Main Camera	None (Camera)
Line Width	0.08
Line Color	🟩
Text Color	⬜
Text Size	14
Canvas Scale	1
Decimal Places	2
Metric Or Imperial	Meter
Init Size For VR	
Split Distance For Text	
Track Interval For Text	
Pivot Bias	Z 0
Angle	X 0 Y 0 Z 0
Coordinate	Local
Axis X_P	✓
Axis X_N	✓

Now user can add measureLine use script and delete them dynamically. And **The "OnSurface" measure now supported:**
When you have a big object, you can measure the distance between different parts of it with other objects, you need to add a parameter to define "OnSurface" lines.



1. MeasureLine_WorldCanvas.cs

```
static public void AddLine(Transform obj1, Transform obj2, bool InitSizeForVR, bool canBeBlock,
bool onSurface)
static public void DeleteLine(Transform obj1, Transform obj2, bool onSurface)
static public void DrawLine(Transform newTarget, bool InitSizeForVR, bool canBeBlock, bool
onSurface)
static public void EndDrawLine()
```

2. MesureLine.cs

```
static public void AddLine(Transform obj1, Transform obj2, bool canBeBlock, bool onSurface)
static public void DeleteLine(Transform obj1, Transform obj2, bool onSurface)
static public void DrawLine(Transform newTarget, bool canBeBlock, bool onSurface)
static public void EndDrawLine()
```

Here are an example, you can add measurelines using raycast, i use some trick here to add lines on surface of two object：

```
using UnityEngine;
using System.Collections;

public class DynamicAddLineSample_WorldCanvas : MonoBehaviour {

    public bool OnSurface = false;
    private Vector3 hitPos;

    void Update ()
    {
        if (Input.GetKeyDown(KeyCode.Mouse0))
        {
            GameObject hitObj = MouseRayer.GetMouseRayHit(Camera.main, out hitPos);
            if (hitObj!=null)
            {
                if (!OnSurface)
                {
                    MeasureLine_WorldCanvas.DrawLine(hitObj.transform, false, false,
OnSurface);
                }
                else
                {
                    GameObject hitDummy = new GameObject("SurfaceLineDummy");
                    hitDummy.transform.position = hitPos;
                    hitDummy.transform.SetParent(hitObj.transform);
                    MeasureLine_WorldCanvas.DrawLine(hitDummy.transform, false, false,
OnSurface);
                }
            }
        }
        else if (Input.GetKeyDown(KeyCode.Mouse1))
        {
            MeasureLine_WorldCanvas.EndDrawLine();
        }
        else if (Input.GetKeyDown(KeyCode.Mouse2))
        {
            MeasureLine_WorldCanvas.DeleteLine(GameObject.Find("Icosahedron
(1)").transform, GameObject.Find("Icosahedron (3)").transform, OnSurface);
        }
    }
}
```

That’s all, wish you have a good time.

If you have any problems or want to share your opinions, you can email me:

unicoea@gmail.com

V1.3c ChangeLog:

-Add Support Metric Or Imperial.

-Support “OnSurface” Measure Lines.

V1.3b ChangeLog:

-Add Support for Add MeasureLine with Custom Script. It can be use to add line, delete line between objects.

V1.3 ChangeLog:

-Add Support for VR (new MeasureLine_WorldCanvas.cs)

-Text can keep themself close to MainCamera automaticly.

-Upgrate OnGUI() to UGUI for origin MeasureLine.cs

V1.2d ChangeLog:

-Fix compatible with Unity5.4.3 and up(line display issue)

V1.2c ChangeLog:

-Add an editor customize interface support for target objects.

V1.2b ChangeLog:

- Now can measure distance from an object to others.
- Add support to modify target objects in runtime.
- Add support to Enable/Disable MeasureLine in runtime.
- Disable display text when lines behind main camera.
- Fixed some other Bugs.

V1.2a ChangeLog:

-Can enable/disable to all six axis now.

V1.2 ChangeLog:

- Line width/Color, Text Color.
- Pivot Bias/ Angle/Coordinate adjustment.
- distance and layermask selector.

V1.1 ChangeLog:

- show and hide lines by Axis.

-Fix line's length bug.

V1.0 Changelog:

-Init complete the six direction mesh line.