Chart And Graph

Supported Platforms:

Graph and chart supports all unity platforms including mobile , pc , html5 and all VR/AR platforms.

Quick Start

Important Notice: The world space gradient shaders require the unity toon shader from the standard assets package. For more information look at this page

The following tutorials assume you have imported the entire Chart And Graph package into your project. Notice the charts menu that has been added unders the tools menu of your editor , it can be used to add new charts to the current scene.

Folders of interest

Chart And Graph / Themes : various sample scenes from the demo project and some that are not included in the demo project.

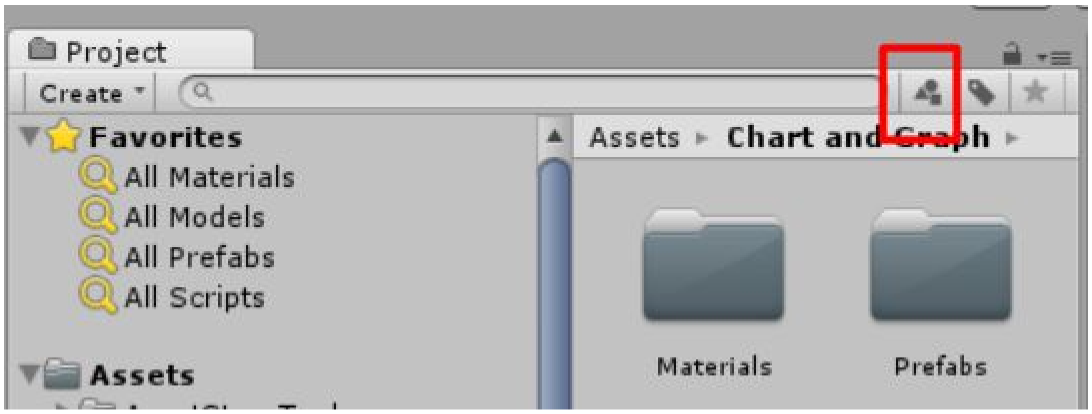
Chart And Graph/Tutorials : The quick start tutorials that appear later in this section

Chart And Graph / Prefabs: prefabs that can be assigned to the charts and labels , be sure to explore this part of the library

Chart And Graph/ Materials: useful materials for canvas charts , this include line styles and point styles for the axis and graph chart

Running the Demo scene:

The demo scene can be found under the folder Chart And Graph / Themes .
The demo scene loads other scene so in order for it to work you must add all scenes to your build. You can find scenes easily by tapping the filter button in the project tab :



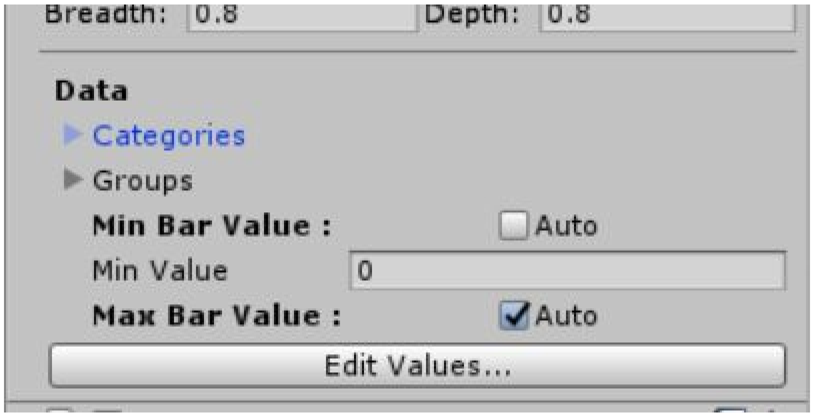
Notes for oculus

In order to make charts interactive with oculus , follow this unity blog post :
https://developer3.oculus.com/blog/unitys-ui-system-in-vr/

Also make sure all interactive charts are set with the correct unity layer.

Bar Chart

1. Go to Tools->Charts->Bar and select the type of bar chart you would like to create. A new chart object should appear in the scene (if a canvas chart is selected it should appear under a canvas).
2. Select the bar object and view it in the inspector. Notice the data section of the bar chart inspector.



3. Expand both Categories and groups. You can add groups and categories by typing a name and clicking the add button. You can remove and rename groups and categories by clicking the "..." button near each one. For the sake of demonstration , We will assume that "category *" is renamed to "player *" and "group *" is renamed to "value *"
4. To edit the chart values from the editor click the "Edit Value" button
5. To update the bar chart from your script , create a new monobehaviour named BarChartFeed and add the following code to your script :

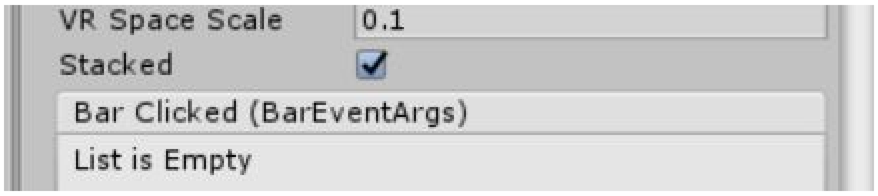
```
void Start () {
        BarChart barChart = GetComponent<BarChart>();
        if (barChart != null)
        {
            barChart.DataSource.SetValue("Player 1", "Value 1", Random.value * 20);
            barChart.DataSource.SlideValue("Player 2","Value 1", Random.value * 20, 40f);
        }
    }
```

NOTICE: calling SetValue and SlideValue must be done with your own group and category names. Notice spaces and case letters

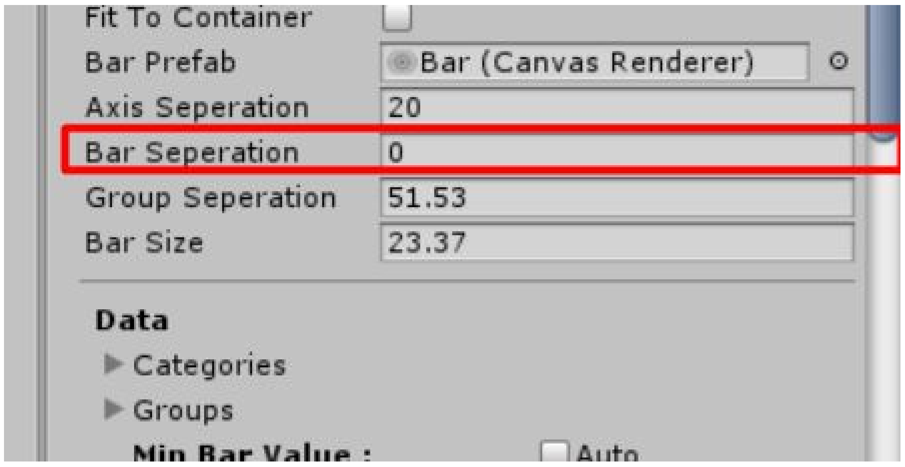
6. Add a BarFeedChart component to the bar chart object and click play.
7. The full example can be found in Assets\Chart and Graph\Demos\Tutorials\Bar

Stack Bar Chart

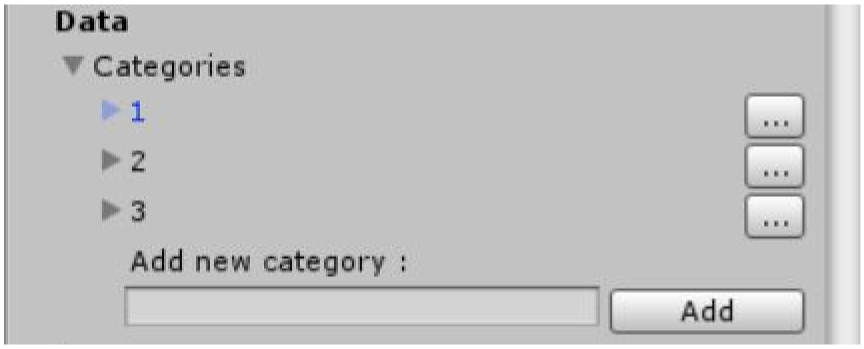
To make a bar chart stackable use the stacked feature of the bar chart:



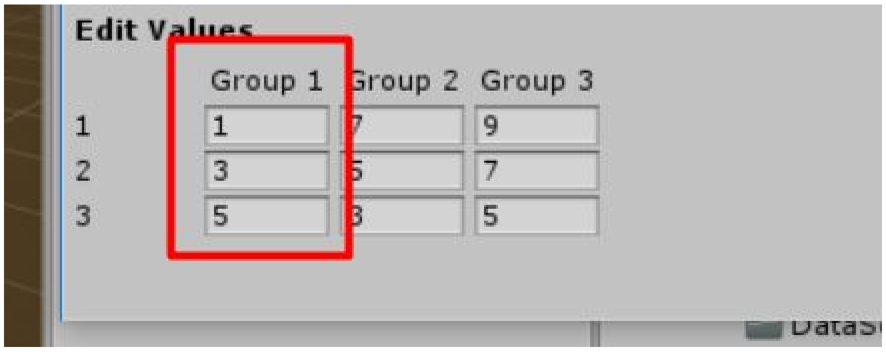
Set bar seperation to 0


Fit To Container	☐
Bar Prefab	Bar (Canvas Renderer)
Axis Seperation	20
Bar Seperation	0
Group Seperation	51.53
Bar Size	23.37

The order of the categories is the order at which the stacks are defined:



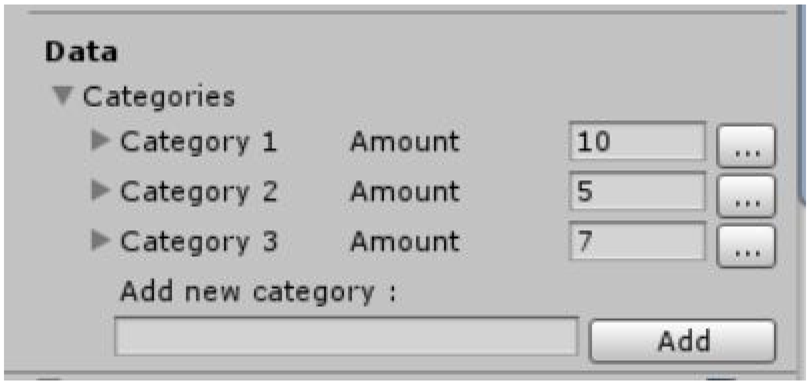
Values for the stack bar should be added in ascending order. like group1 in this picture


Edit Values

	Group 1	Group 2	Group 3
1	1	7	9
2	3	5	7
3	5	3	5

Pie Chart

1. Go to Tools->Charts->Pie Or Tools->Charts->Torus and select the type of chart you would like to create. A new chart object should appear in the scene (if a canvas chart is selected it should appear under a canvas).
2. Select the pie or torus object and view it in the inspector. Notice the data section of the chart inspector.



3. Expand the Categories foldout. You can add categories by typing a name and clicking the add button. You can remove and rename categories by clicking the "..." button near each one. For the sake of demonstration , We will assume that "category *" is renamed to "player *".
4. You can change the amount of each pie slice by editing it's input box.
5. To update the pie chart from your script , create a new monobehaviour named PieChartFeed and add the following code to your script :

```
void Start ()
 {
        PieChart pie = GetComponent<PieChart>();
        if(pie != null)
        {
            pie.DataSource.SlideValue("Player 1", 50, 10f);
            pie.DataSource.SetValue("Player 2", Random.value * 10);
        }
}
```

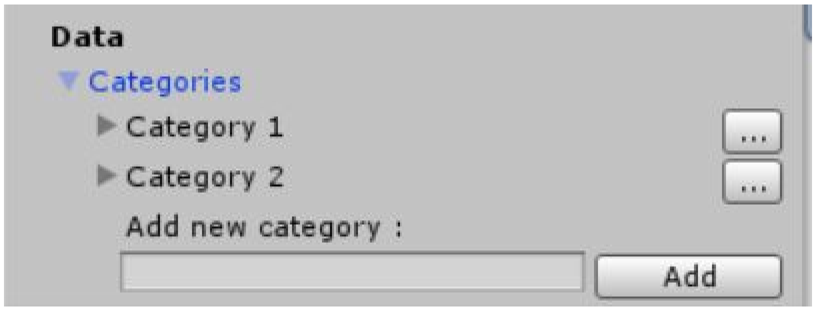
NOTICE: calling set value must be done with your own category names. Notice spaces and case letters

6. Add a PieFeedChart component to the pie chart object and click play.
7. The full example can be found in Assets\Chart and Graph\Demos\Tutorials\Pie

Graph Chart

1. Add a canvas to the scene. Go to Tools->Charts->Graph and select the type of graph you would like to create. The chart should apear under the canvas.

2. Select the graph object and view it in the inspector. Notice the data section of the chart inspector.



3. Expand the Categories foldout. You can add categories by typing a name and clicking the add button. You can remove and rename categories by clicking the "..." button near each one. For the sake of demonstration , We will assume that "category *" is renamed to "player *".
4. Graph data can only be changed through source code. The graph appearance can be customized in the editor.
5. To update the graph chart data from your script , create a new monobehaviour named GraphChartFeed and add the following code to your script :

```
void Start ()
    {
        GraphChart graph = GetComponent<GraphChart>();
        if (graph != null)
        {
            graph.DataSource.StartBatch();
            graph.DataSource.ClearCategory("Player 1");
            graph.DataSource.ClearCategory("Player 2");
            for (int i = 0; i < 30; i++)
            {
                graph.DataSource.AddPointToCategory("Player 1",Random.value*10f,Random.value*10f);
                graph.DataSource.AddPointToCategory("Player 2", Random.value * 10f, Random.value * 10f);
            }
            graph.DataSource.EndBatch();
        }
    }
```

NOTICE: calling set value must be done with your own category names. Notice spaces and case letters

6. Add a GraphFeedChart component to the graph chart object and click play.
7. The full example can be found in Assets\Chart and Graph\Demos\Tutorials\Graph

Streaming Graph Chart

Using the same setup as before just call AddPointToCategoryRealtime instead of AddPointToCategory. Realtime points can only be added at the end of the current data (their x value must be higher then previous data points). See this reference code for example :

Note: Calling Do not use StartBatch And EndBatch to batch realtime calls , this will only slow things down.

```
public class StreamingGraph : MonoBehaviour
{

    public GraphChart Graph;
    public int TotalPoints = 5;
    float lastTime = 0f;
    float lastX = 0f;

    void Start()
    {
        if (Graph == null) // the ChartGraph info is obtained via the inspector
            return;
        float x = 0f;
     /////   Graph.DataSource.StartBatch(); // do not call StartBatch for realtime calls , it will only
slow down performance.

        Graph.DataSource.ClearCategory("Player 1"); // clear the "Player 1" category. this category is
defined using the GraphChart inspector
        Graph.DataSource.ClearCategory("Player 2"); // clear the "Player 2" category. this category is
defined using the GraphChart inspector

        for (int i = 0; i < TotalPoints; i++)  //add random points to the graph
        {
            Graph.DataSource.AddPointToCategoryRealtime("Player 1", x, Random.value * 20f + 10f); //
each time we call AddPointToCategory
            Graph.DataSource.AddPointToCategoryRealtime("Player 2", x, Random.value * 10f); // each time
we call AddPointToCategory
            x += Random.value * 3f;
            lastX = x;

        }
      ////  Graph.DataSource.EndBatch(); // do not batch reatlime calls
}

    void Update()
    {
        float time = Time.time;
        if (lastTime + 2f < time)
        {
            lastTime = time;
            lastX += Random.value * 3f;
            Graph.DataSource.AddPointToCategoryRealtime("Player 1", lastX, Random.value * 20f + 10f,
1f); // each time we call AddPointToCategory
            Graph.DataSource.AddPointToCategoryRealtime("Player 2", lastX, Random.value * 10f, 1f); //
each time we call AddPointToCategory
        }

    }
}
```

Graph Chart Curves:

1. First off , you must set up a category as a curve category , this can be done via the inspector or via code :

```
graph.DataSource.ClearAndMakeBezierCurve("Player 1"); // for curved line
graph.DataSource.ClearAndMakeLinear("Player 1"); // for normal lines
```

2. Set up the initial point for the curve:

```
graph.DataSource.SetCurveInitialPoint("Player 1",1.0,1.0));
```

3. Then add segments to the curve using any of the following methods:

```
            graph.DataSource.AddLinearCurveToCategory("Player 2",
                                                                            new DoubleVector2(i *
10f/30f, Random.value * 10f + 10f)); // add a line segment to a curve category
              graph.DataSource.AddCurveToCategory("Player 2",
                                                        controlPointA,controlPointB,DestinationPoint);
        // adds a bezier curve segment to a curve category/
```

4. You can also call

```
graph.DataSource.MakeCurveCategorySmooth("Player 2");
```

This will override all currently set control points , and will make a smooth transition between all the segment points.

Bubble Chart:

The bubble chart is essentially a graph chart , the graph chart enables you to control point sizes for all data points. Feeding a bubble chart is done by passing a point size parameter to any of the methods described above , i.e :

```
AddPointToCategory
AddPointToCategoryRealtime
AddCurveToCategory
AddLinearCurveToCategory
SetCurveInitialPoint
```

For example :

```
        graph.DataSource.AddPointToCategory("Player 1", Random.value * 10f, Random.value * 10f,
PointSize);
```

Handling a large amount of data

The following example illustrates how you can use graph and chart to display a large amount of data with scrolling.
This code keeps track of the current horizontal scrolling of the graph , once the scrolling has passed some threshold , this data around the current scrolling offset is loaded , and the previous data is discarded.

Note that in order for this to work properly , you have to use non automatic horizontal view. You should also disable horizontal auto scrolling.

```
using UnityEngine;
using ChartAndGraph;
using System.Collections.Generic;
using System;

public class GraphChartFeed : MonoBehaviour,IComparer<DoubleVector2>
{
    List<DoubleVector2> mData = new List<DoubleVector2>();
    double pageSize = 50f;
    double currentPagePosition = 0.0;
    GraphChartBase graph;
    void Start ()
    {
        graph = GetComponent<GraphChartBase>();
        double x = 0f;
        for (int i = 0; i < 250000; i++)    // initialize with random data
        {
            mData.Add(new DoubleVector2(x, UnityEngine.Random.value));
            x += UnityEngine.Random.value * 10f;
        }
        LoadPage(currentPagePosition); // load the page at position 0
    }

    int FindClosestIndex(double position) // if you want to know what is index is currently
displayed . use binary search to find it
    {
        //NOTE :: this method assumes your data is sorted !!!
        int res = mData.BinarySearch(new DoubleVector2(position, 0.0),this);
        if (res >= 0)
            return res;
        return ~res;
    }


    void findPointsForPage(double position,out int start,out int end) // given a page
position , find the right most and left most indices in the data for that page.
    {
        int index = FindClosestIndex(position);
        int i = index;
        double endPosition = position + pageSize;
        double startPosition = position - pageSize;

        //starting from the current index , we find the page boundries
        for(start = index; start >0; start--)
        {
            if (mData[i].x < startPosition) // take the first point that is out of the page.
so the graph doesn't break at the edge
                break;
        }
        for(end = index; end < mData.Count; end ++)
        {
            if (mData[i].x > endPosition) // take the first point that is out of the page
                break;
        }
    }
    private void Update()
    {
        if (graph != null)
        {
            //check the scrolling position of the graph. if we are past the view size , load
a new page
            double pageStartThreshold = currentPagePosition - pageSize;
            double pageEndThreshold = currentPagePosition + pageSize -
graph.DataSource.HorizontalViewSize;
            if (graph.HorizontalScrolling < pageStartThreshold || graph.HorizontalScrolling
> pageEndThreshold)
            {
                LoadPage(graph.HorizontalScrolling);
            }
        }
    }
    void LoadPage(double pagePosition)
    {

        if (graph != null)
        {

            Debug.Log("Loading page :" + pagePosition);
            graph.DataSource.StartBatch(); // call start batch
            graph.DataSource.HorizontalViewOrigin = 0;
            int start, end;
            findPointsForPage(pagePosition, out start, out end); // get the page edges
            graph.DataSource.ClearCategory("Player 1"); // clear the cateogry
            for(int i=start; i<end; i++) // load the data
                graph.DataSource.AddPointToCategory("Player 1",mData[i].x,mData[i].y);
            graph.DataSource.EndBatch();
            graph.HorizontalScrolling = pagePosition;
        }
        currentPagePosition = pagePosition;
    }

    public int Compare(DoubleVector2 x, DoubleVector2 y)
    {
        if(x.x < y.x)
```

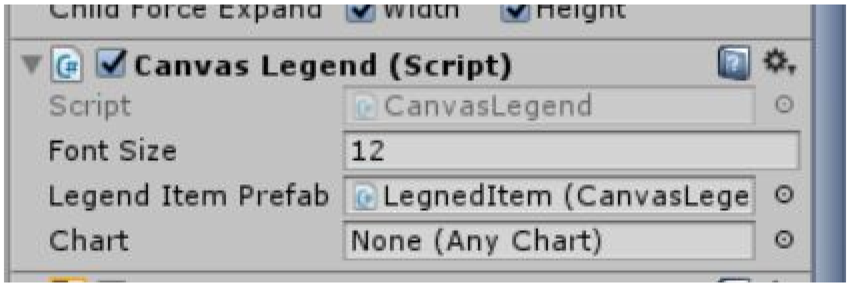
```
            return -1;
        if (x.x < y.x)
            return 1;
        return 0;
    }
}
```

Dynamically adding and remove categories

Every chart cs type (BarChart,GraphChartBase,RadarChart,PieChart) has a DataSource property. This data source has methods for dynamically adding and removing categories and groups.
Also for best performance , use the datasource's StartBatch and EndBatch methods.
These methods suppress the redraw of the chart until all changes are made.

Legend

1. Add a canvas to the scene. Go to Tools->Charts->Legend. A new legend object will be created under the canvas
2. Select the legend object in the inspector



Canvas Legend (Script)
Script: CanvasLegend
Font Size: 12
Legend Item Prefab: LegnedItem (CanvasLege
Chart: None (Any Chart)

3. Drag any chart gameobject to the chart field of the inspector.

Overview

Chart And Graph contains a few base component that can be used in order to create charts and customize them.

The basic chart components are :
WorldSpaceBarChart - 3d bar charts
CanvasBarChart - canvas bar charts
WorldSpacePieChart - 3d pie and torus charts
CanvasPieChart - canvas pie charts

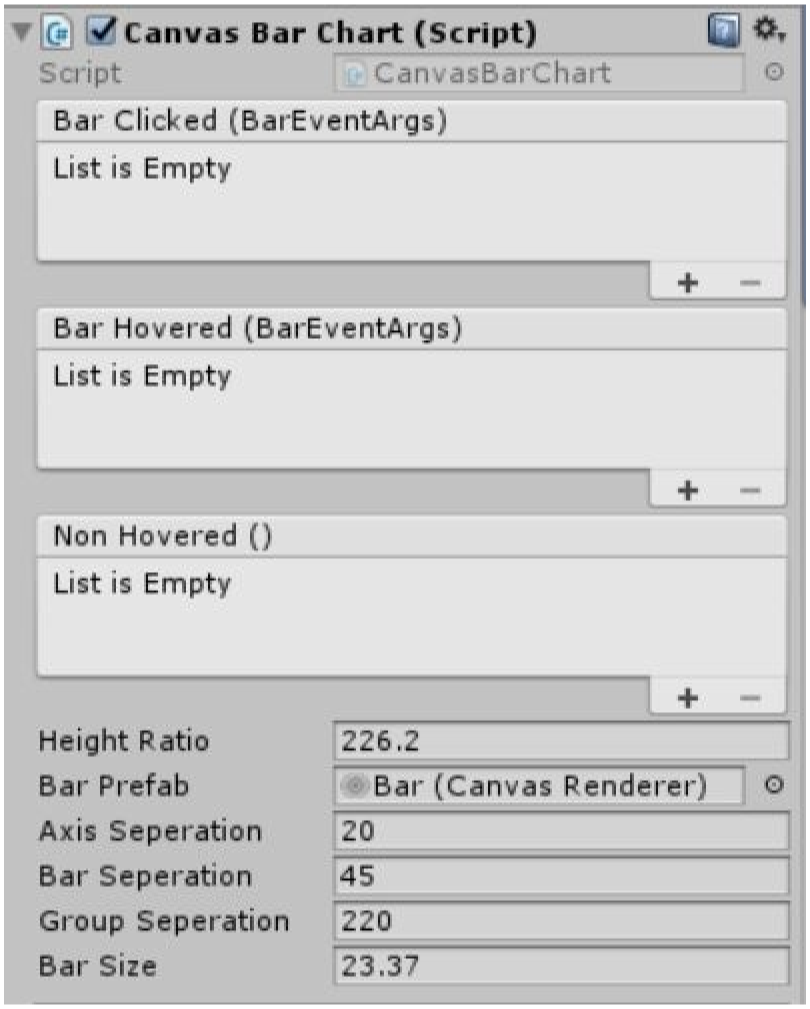
GraphChart - canvas graph charts

Once you have added one of the base components to the a gameobject , you can add the following additional setting components :

HorizontalAxis/VerticalAxis - both bar and graph charts support axis display. Add these components to define the axis display.
ItemLabels/CategoryLabels/GroupLabes - these components define lables settings for the chart, item labels appear for each item of the chart (for example each bar or pe slice). Category and group labels correspond to the categories and groups of the chart data.

Bar Chart

Canvas



Canvas Bar Chart (Script)

Script	CanvasBarChart

Bar Clicked (BarEventArgs)
List is Empty

Bar Hovered (BarEventArgs)
List is Empty

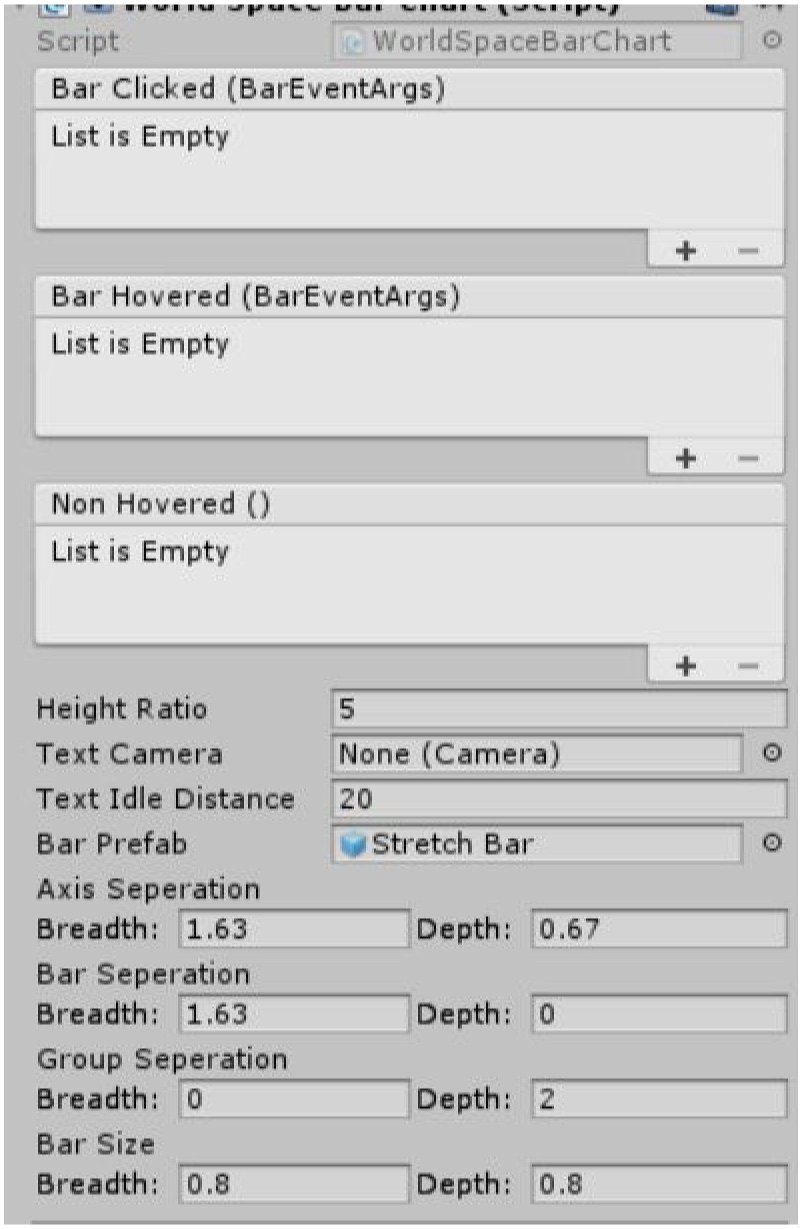
Non Hovered ()
List is Empty

Height Ratio	226.2
Bar Prefab	Bar (Canvas Renderer)
Axis Seperation	20
Bar Seperation	45
Group Seperation	220
Bar Size	23.37

Bar Clicked - occurs when the a bar is clicked. Receives BarEventArgs with data.

Bar Hovered - occurs when the a bar is hovered. Receives BarEventArgs with data.
Non Hovered - occurs when the pointer leaves the current bar and no other bar is hovered.
Height Ratio - The height ratio of the chart , the width is calculated by the other properties. Once the width and height ratio is set, the chart is fit into the rect transform.
Bar Prefab - Use the default prefabs in Assets/Chart And Graph/Prefabs/Canvas/Bar. You can use your own prefab , make sure to set it’s size to a unity size and it pivot to bottom center.
Axis Separation - the gap between the bars and the chart edges
Bar Separation - the gap between bars within the same group.
Group Separation - the gap between different groups.
Bar Size - the size of each bar.

World Space



Script: WorldSpaceBarChart

Bar Clicked (BarEventArgs)
List is Empty

Bar Hovered (BarEventArgs)
List is Empty

Non Hovered ()
List is Empty

Height Ratio: 5
Text Camera: None (Camera)
Text Idle Distance: 20
Bar Prefab: Stretch Bar

Axis Seperation
Breadth: 1.63 Depth: 0.67

Bar Seperation
Breadth: 1.63 Depth: 0

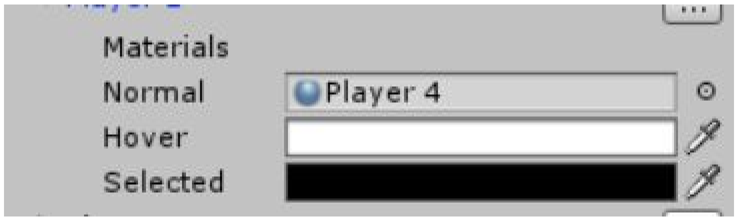
Group Seperation
Breadth: 0 Depth: 2

Bar Size
Breadth: 0.8 Depth: 0.8

Bar Clicked - occurs when the a bar is clicked. Receives BarEventArgs with data.
Bar Hovered - occurs when the a bar is hovered. Receives BarEventArgs with data.
Non Hovered - occurs when the pointer leaves the current bar and no other bar is hovered.
Height Ratio - The height ratio of the chart , the width is calculated by the other properties.
Bar Prefab - Use the default prefabs in Assets/Chart And Graph/Prefabs/3d/. You can use your own prefab , make sure to set it's size to a unity size and it pivot to bottom center.
Axis Separation - the gap between the bars and the chart edges. This includes depth and breadth for 3d effects

Bar Separation - the gap between bars within the same group.This includes depth and breadth for 3d effects
Group Separation - the gap between different groups.This includes depth and breadth for 3d effects
Bar Size - the size of each bar. This includes depth and breadth for 3d effects

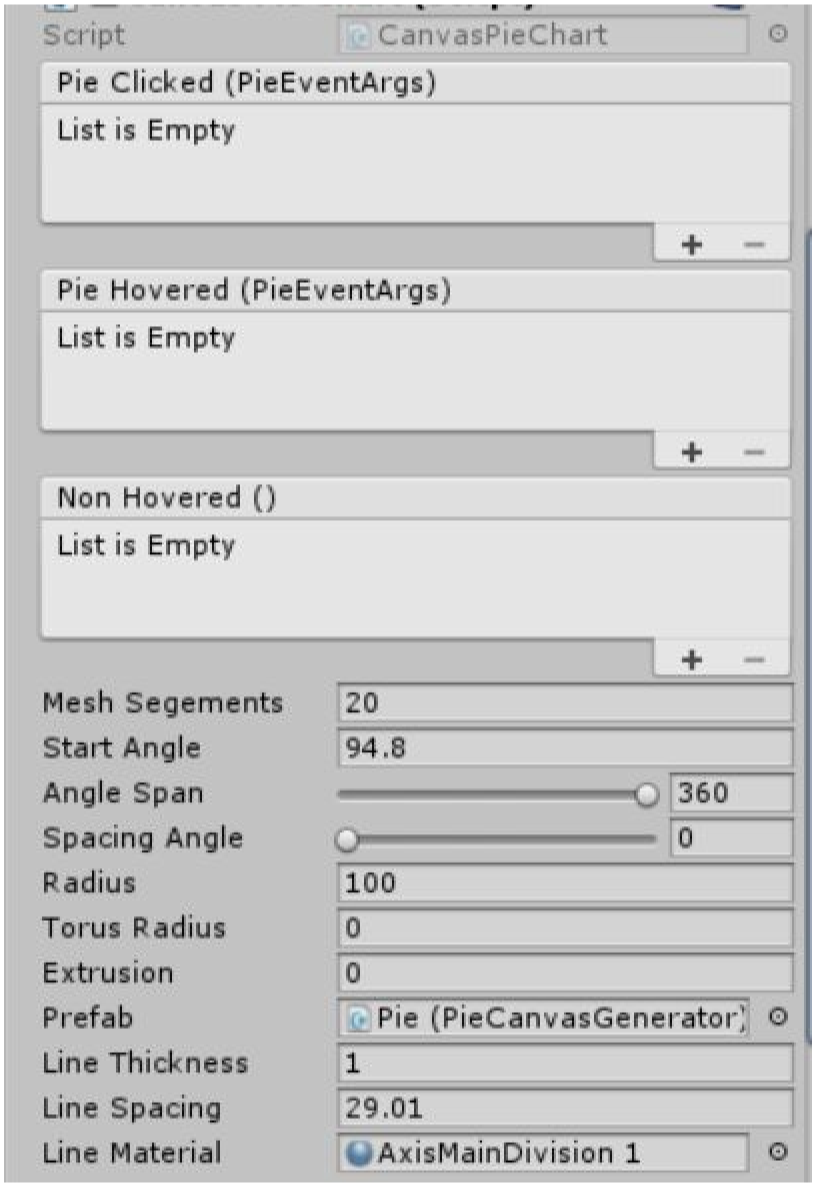
Category settings



Normal - the standard material for this category
Hover - this color is assigned to the material when the bar is hovered. Use Color.clear for no effect
Selected - this color is assigned to the material when the bar is selected. Use Color.clear for no effect.

Pie Chart

canvas

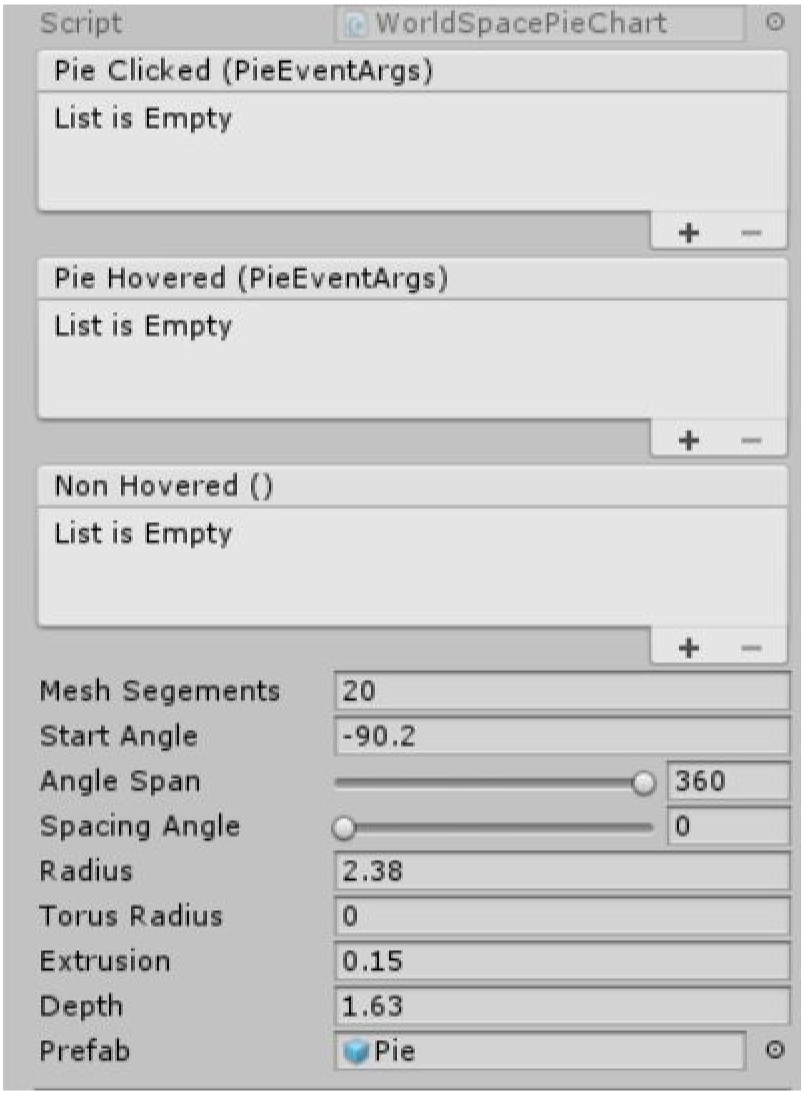


Script	CanvasPieChart
Pie Clicked (PieEventArgs)	List is Empty
Pie Hovered (PieEventArgs)	List is Empty
Non Hovered ()	List is Empty
Mesh Segements	20
Start Angle	94.8
Angle Span	360
Spacing Angle	0
Radius	100
Torus Radius	0
Extrusion	0
Prefab	Pie (PieCanvasGenerator)
Line Thickness	1
Line Spacing	29.01
Line Material	AxisMainDivision 1

Pie Clicked - occurs when the a pie is clicked. Receives PieEventArgs with data.
Pie Hovered - occurs when the a pie is hovered. Receives PieEventArgs with data.
Non Hovered - occurs when the pointer leaves the current pie and no other pie is hovered.
Mesh segments - the amount of vertex segments in the pie mesh.
Start Angle - the start angle of the pie chart. Value is in degrees
Angle Span - the angle span of the pie chart. Value is in degrees

Spacing Angle- the spacing angle between pie slices. Value is in degrees
Radius - the radius of the pie chart
Torus Radius - the inner radius of the pie chart. Set this value to 0 for a pie chart , set this value to more than 0 for a torus chart.
Extrusion - Translates the pie slices outwards for a non 0 value
Prefab - Use one of the default prefabs in Assets/Chart And Graph/Prefabs/Canvas/Pie. Or use your own prefab by adding PieCanvasGenerator to it.
Line Thickness - thickness for label lines in the pie chart.
Line Spacing - spacing for label lines in the pie chart
Line Material - material for labels lines in the pie chart

World Space

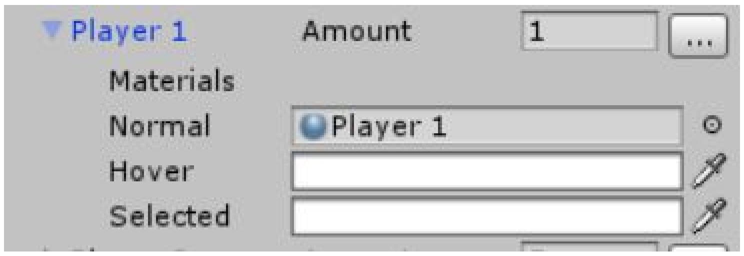


Script	WorldSpacePieChart
Pie Clicked (PieEventArgs)	List is Empty
Pie Hovered (PieEventArgs)	List is Empty
Non Hovered ()	List is Empty
Mesh Segements	20
Start Angle	-90.2
Angle Span	360
Spacing Angle	0
Radius	2.38
Torus Radius	0
Extrusion	0.15
Depth	1.63
Prefab	Pie

Pie Clicked - occurs when the a pie is clicked. Receives PieEventArgs with data.
Pie Hovered - occurs when the a pie is hovered. Receives PieEventArgs with data.

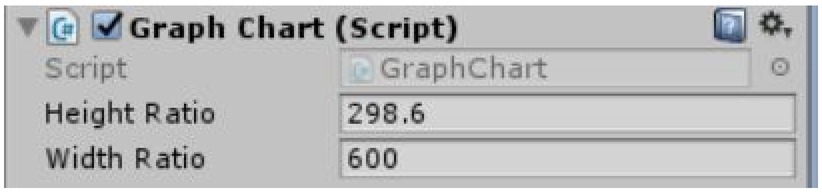
Non Hovered - occurs when the pointer leaves the current pie and no other pie is hovered.
Mesh segments - the amount of vertex segments in the pie mesh.
Start Angle - the start angle of the pie chart. Value is in degrees
Angle Span - the angle span of the pie chart. Value is in degrees
Spacing Angle- the spacing angle between pie slices. Value is in degrees
Radius - the radius of the pie chart
Torus Radius - the inner radius of the pie chart. Set this value to 0 for a pie chart , set this value to more than 0 for a torus chart.
Extrusion - Translates the pie slices outwards for a non 0 value
Prefab - Use one of the default prefabs in Assets/Chart And Graph/Prefabs/3d. Or use your own prefab by adding WorldSpacePieGenerator to it.

Pie Category


Player 1 Amount 1
Materials
Normal Player 1
Hover
Selected

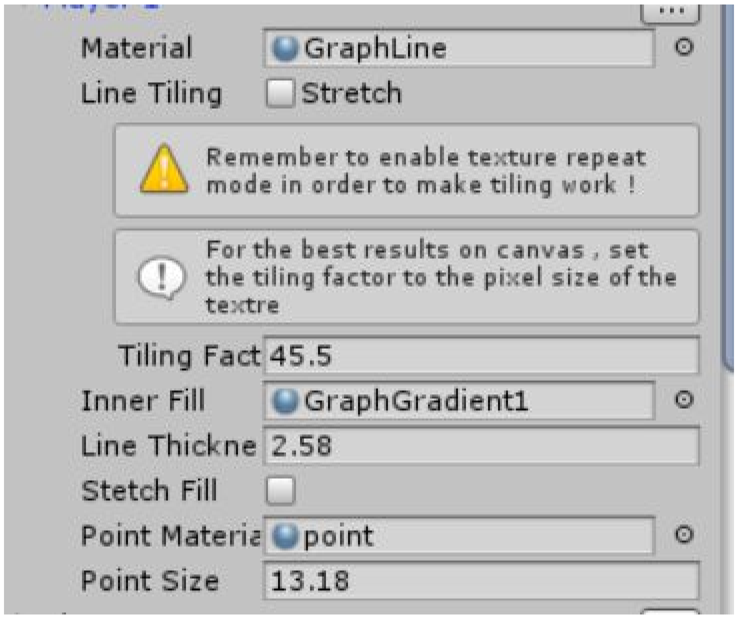
Amount - the value of the category
Normal - the standard material for this category
Hover - this color is assigned to the material when the bar is hovered. Use Color.clear for no effect
Selected - this color is assigned to the material when the bar is selected. Use Color.clear for no effect.

Graph Chart


Graph Chart (Script)
Script GraphChart
Height Ratio 298.6
Width Ratio 600

Height/Width Ratio - the height and width ratio of the chart. For example if width is twice the size of height then the chart will have a 2:1 ratio. Note that the only thing that matters is the ratio between these values, because the chart is fitted into the rect transform it belongs to.

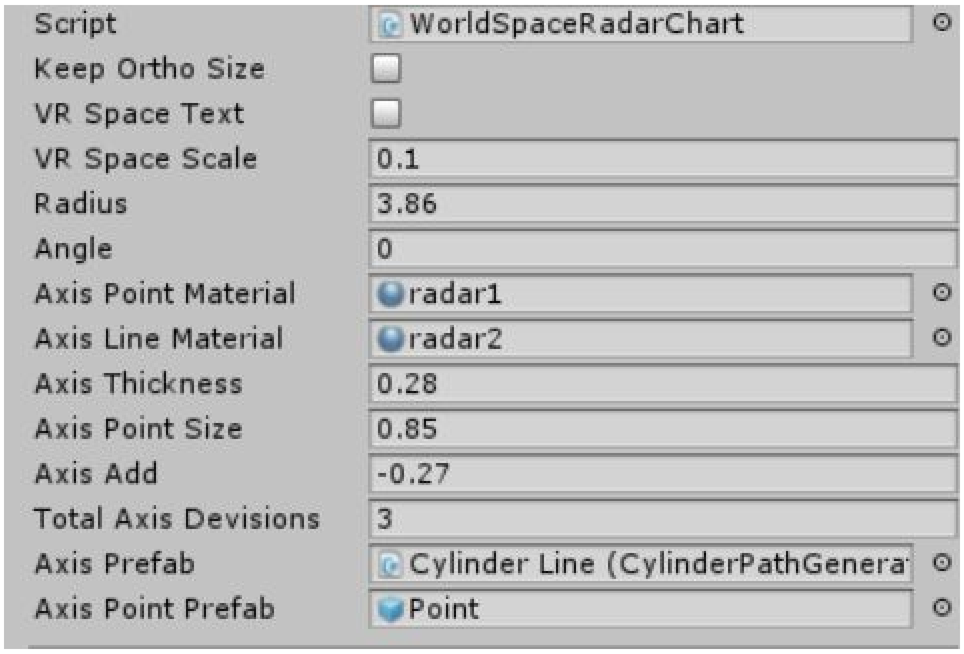
Graph Category



Material GraphLine
Line Tiling Stretch
Remember to enable texture repeat mode in order to make tiling work !
For the best results on canvas , set the tiling factor to the pixel size of the textre
Tiling Fact 45.5
Inner Fill GraphGradient1
Line Thickne 2.58
Stetch Fill
Point Materia point
Point Size 13.18

Material - The Material for the category graph line
Line Tiling - line tiling can be used to create effects such as dotted lines. Use Stretch to stretch the texture over the entire line , or set the tile factor to make the texture repeat itself. Remember that the texture repeat mode must be enabled.
Inner Fill - Material for the fill under the category line.
Stretch Fill - If true the fill is stretched along the graph shape , if false the material is masked by the line shape.
Point Material - the material used for the category points.
Point Size - the size of the category points.

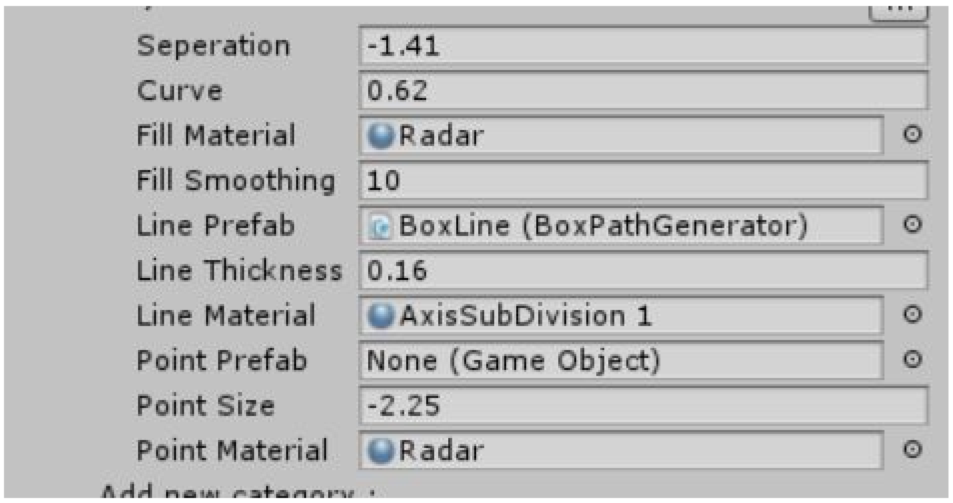
Radar Chart



Script	WorldSpaceRadarChart
Keep Ortho Size	☐
VR Space Text	☐
VR Space Scale	0.1
Radius	3.86
Angle	0
Axis Point Material	radar1
Axis Line Material	radar2
Axis Thickness	0.28
Axis Point Size	0.85
Axis Add	-0.27
Total Axis Devisions	3
Axis Prefab	Cylinder Line (CylinderPathGenera
Axis Point Prefab	Point

Radius - the radius of the radar chart in units for 3d or pixels for 2d
Angle - Use this to set the starting angle of the radar chart. This like rotation only that it keeps text labels aligned with the viewer
Axis Point Material - the material used for radar axis points , for 3d chart this should be used along with a prefab. Leave this blank and the radar chart axis will have no dots
Axis Line Material - the material used for radar axis lines, for 3d chart this should be used along with a prefab. Leave this blank and the radar chart axis will have no lines
Axis Thickness - the thickness of the radar chart axis lines
Axis Point Size - the size of a radar axis point
Axis Add - for 3d radar chart , use this to create a 3d effect for the axis.
Total Axis Devisions - the number of axis devisions for the radar chart.
Axis Prefab - for 3d charts , use this to set a prefab for the axis lines
Axis Point Prefab - for 3d charts , use this to set a prefab for the axis points

Radar Category



Seperation	-1.41
Curve	0.62
Fill Material	Radar
Fill Smoothing	10
Line Prefab	BoxLine (BoxPathGenerator)
Line Thickness	0.16
Line Material	AxisSubDivision 1
Point Prefab	None (Game Object)
Point Size	-2.25
Point Material	Radar

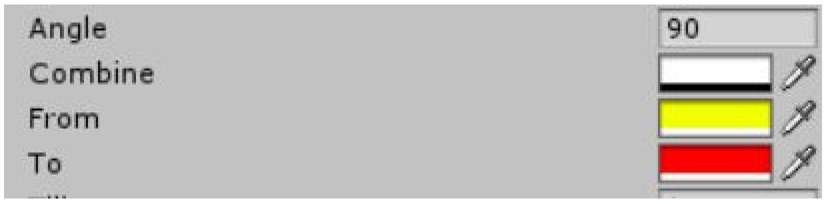
Add new category :

Separation - for 3d radar only. Set the depth of the category relative to the axis
Curve - for 3d radar only , make the category mesh go along a curve instead of a plane
Fill Material - The material used for the fill part of the radar category
Fill Smoothing - for 3d radar only. Set the smoothing for a curved radar category
Line Prefab - the prefab used for the line part of the category mesh
Line Thickness - the thickness of the line part of the category mesh
Line Material - the material for the line part of the category mesh
Point Prefab - the point prefab for point part of the category mesh
Point Size - the point size for the point part of the category mesh
Point Material - the point material for the point part of the category mesh.

Gradient Materials

The gradient shaders can be found under the Chart menu item in the shader selection menu

Both world space and canvas chart contain the following settings



Angle	90
Combine	⬜
From	🟨
To	🟥

Angle - the angle of the gradient. 0 for vertical gradient, 90 for horizontal , etc.
Combine - this value is used by the chart components in order to change the color of the gradient. The combined value is interpolated with the gradient material by it’s alpha value. Setting the alpha value to 0 has no effect, and is the default value.
From - the start color of the gradient

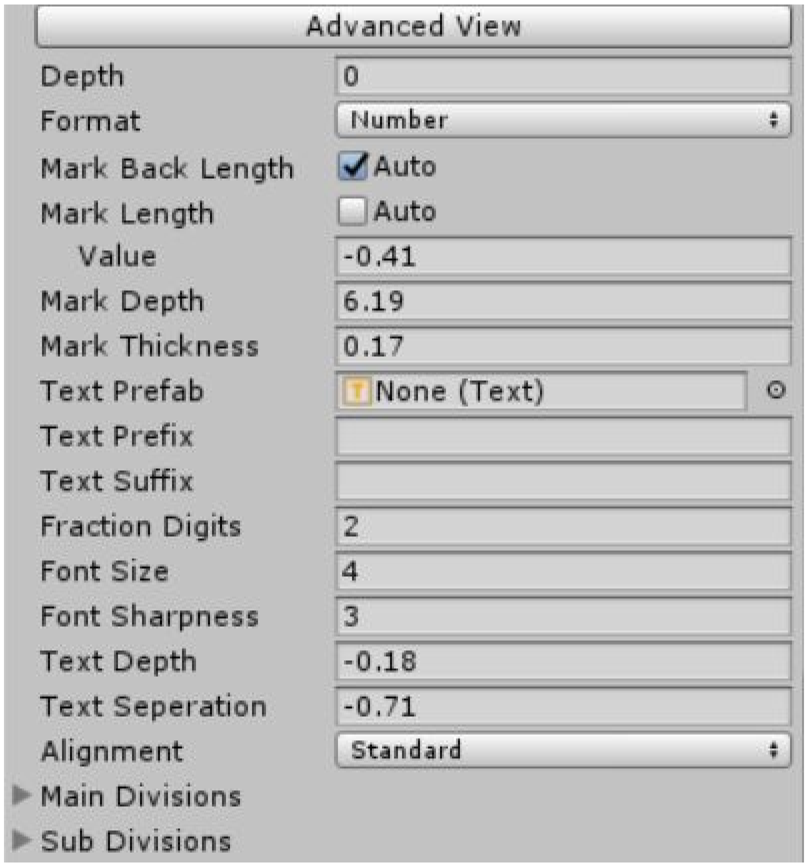
To - the end color of the gradient

Animations

Use the BarAnimation component to animate Bar Charts.
User the PieAnimation component to animate Pie Charts.

Chart Axis

Each axis has both main divisions and sub divisions. Use the simple view to quickly set values for both of the division types. Use advance view to customize each division type.
Sub divisions are for each main division. So if there are 5 main division and 5 sub division there will be a total of 5*5 = 25 divisions.



Depth	0
Format	Number
Mark Back Length	✓ Auto
Mark Length	Auto
Value	-0.41
Mark Depth	6.19
Mark Thickness	0.17
Text Prefab	None (Text)
Text Prefix	
Text Suffix	
Fraction Digits	2
Font Size	4
Font Sharpness	3
Text Depth	-0.18
Text Seperation	-0.71
Alignment	Standard

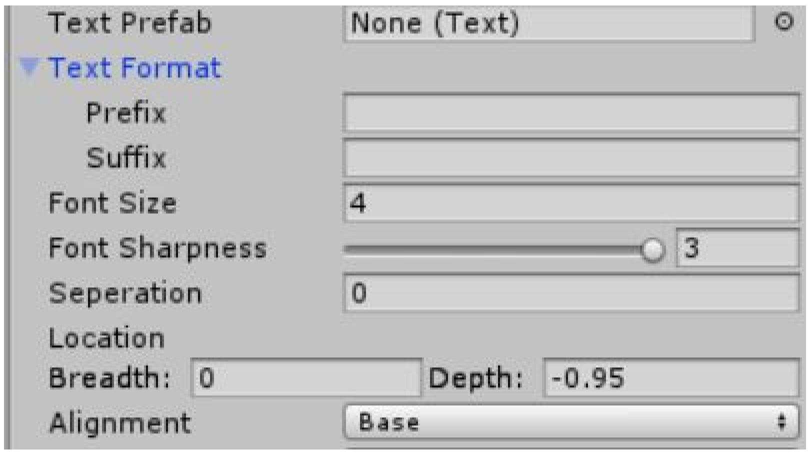
Depth- the depth of the axis relative to the chart.
Format- the format of the axis labels. Can be either a number, a date , a time or a date time. Use the custom function of DataSource and DateUtillity to set date and time values.
Mark Back Length - when mark depth in a non zero. This value is the size of the far side of the mark.

Mark Length - the size of the near side of the mark

Mark Depth - the depth size of the mark
Mark Thickness- the thickness of the mark
Text Prefab- prefab for text labels. You can use the defualt prefabs in Assets/Chart And Graph/Prefabs
Text Prefix/Suffix- added to each axis label text. (a use case would be adding a “$” sign to indicate an amount of money)
Fraction Digits - the maximum amount of fraction digits in the text lables.
Font size - the font size of the text lables.
Font Sharpness - adjust this to make the fonts look sharper.
Text Depth - set the label position along the z axis
Text Separation - set the label position along the line of the axis.
Alignment - change the label side (right or left)

Labels

Label components can be added for categories, items and groups :



Text Prefab	None (Text)
Text Format	
Prefix	
Suffix	
Font Size	4
Font Sharpness	3
Seperation	0
Location	
Breadth: 0	Depth: -0.95
Alignment	Base

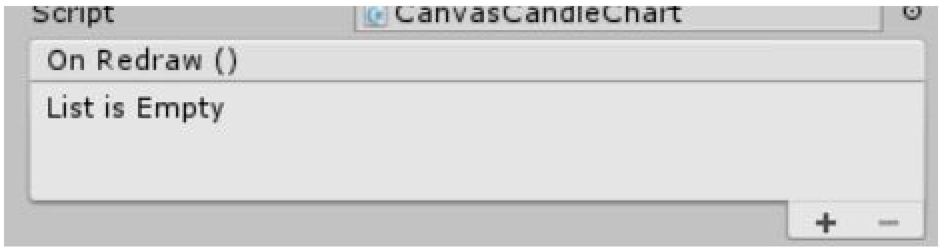
Text Prefab- prefab for text labels. You can use the defualt prefabs in Assets/Chart And Graph/Prefabs
Text Prefix/Suffix- added to each axis label text. (a use case would be adding a “$” sign to indicate an amount of money). You can also use some macros like <?category> <?group> \n (see Assets/Chart And Graph/Demos/DemoScenes/2d pie for example).
Font size - the font size of the text lables.
Font Sharpness - adjust this to make the fonts look sharper.
Separation - set the label elavation relative to the chart
Location - set the label translation relative to the chart
Alignment - align the label to either Base or Top of the chart items.

Markers and areas on charts

Some applications require adding some markers / images or other unity objects on top of charts , graph and chart provide easy access methods to cover these cases.

When to reposition markers

All the charts contain an event called OnRedraw , This event is invoked any time the data or appearance of the chart changes ,use this to reposition any markers on the chart.



```
chart.OnRedraw.AddListener(myListener);
```

AnyChart Components

All the chart components contain properties that can be easily queried in order to get the chart dimensions :

```
var chart = GetComponent<AnyChart>();
float width = chart.TotalWidth;
float height = chart.TotalHeight;
float depth = chart.TotalDepth;
```

Bar Chart

The bar chart has these utility methods :

1.

```
public bool GetBarTrackPosition(string category, string group,out Vector3
trackPosition)
```

Call this method to get the worldspace position of the top of the selected bar. The bar is specified using a category and a group. This method will return true on success or false on failure (means the category or group are non existant)

2.

```
public bool GetBarBottomPosition(string category, string group, out Vector3 bottom)
```

Same as the above method only that it returns the base position of the bar

3.

```
public bool PointToWorldSpace(string category, string group, double value,bool
allowOverflow, out Vector3 point)
```

Call this method to get the worldspace position of a bar regardless of it's value. This is like the above methods only that the value is specified by the call. Also set `allowOverflow` to true in order to specify a value which is not in the range of the chart axis

ScrollableAxisChart Comoponent

ScrollableAxisChart is a component that is the base class for graph and buble charts. It has the following methods for placing markers:

1. PointToWorldSpace - gets a point in axis units and converts it to a point in world space

2. PointToClient - gets a worldspace point and converts it to axis units

3. TrimRect - trims a rect in axis units into the visible position of it on the chart (taking scrolling and view size to considartion). The resulting rect is also in axis units

4. IsRectVisible - returns true if any portion of a axis unit rect is visible, returns false if the rect is completly off the view

5.RectToCanvas- Sets a rect transform into the position of the specified axis unity rect.

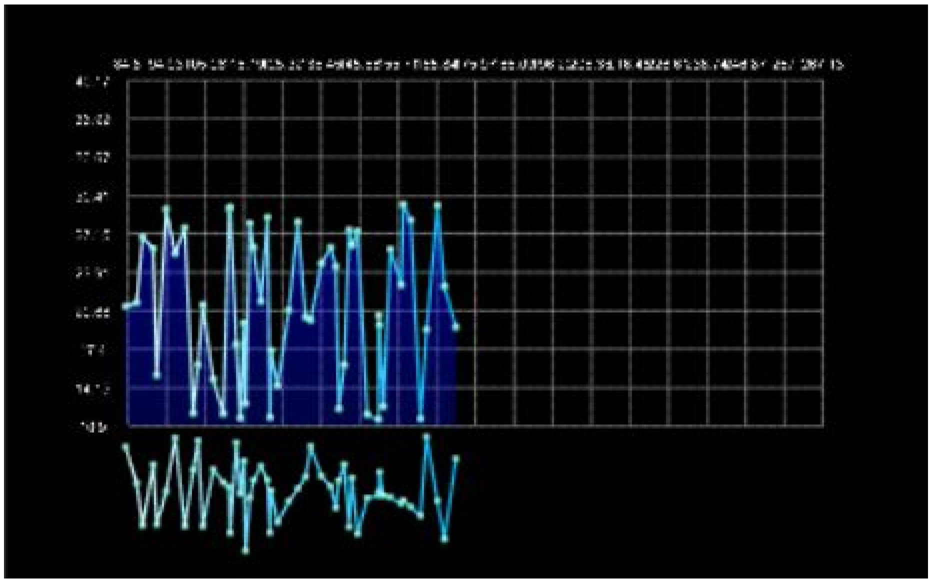
6.MouseToClient - gets the mouse position on the graph in axis units. returns true if the mouse is in bounds of the chart , false otherwise

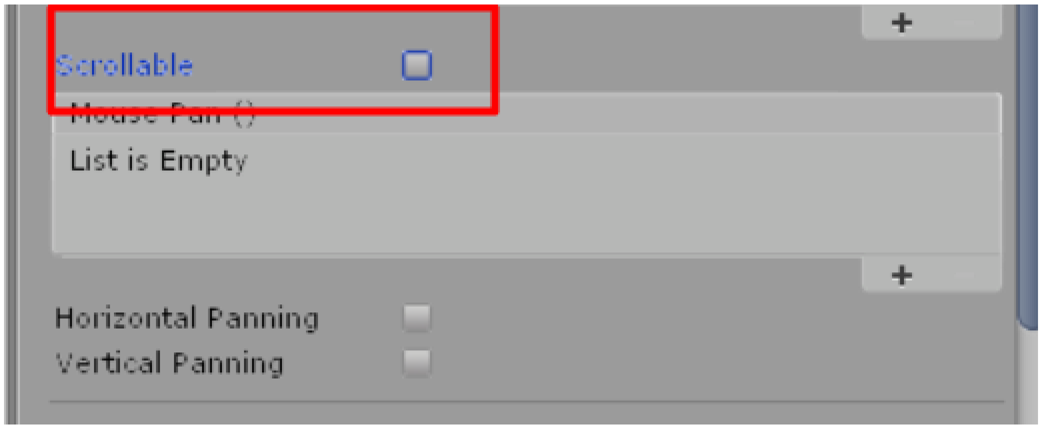
TroubleShooting

This section contains common issues users encounter during development.

Graph Chart Layer Mask

A. Make sure the materials you are using are canvas compatible and support stencil operations for masking. Graph and Chart comes with two of those (solid and gradient)
B. If you want your graph to be clipped using a layer mask , check the scrollable option





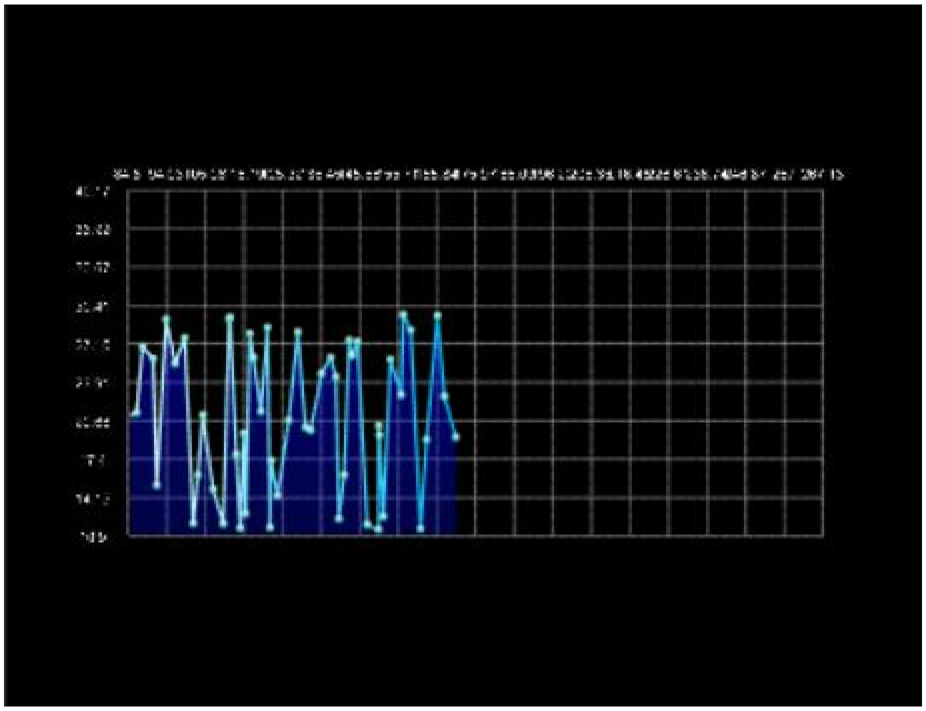
Scrollable ☐

Mouse Pan ()

List is Empty

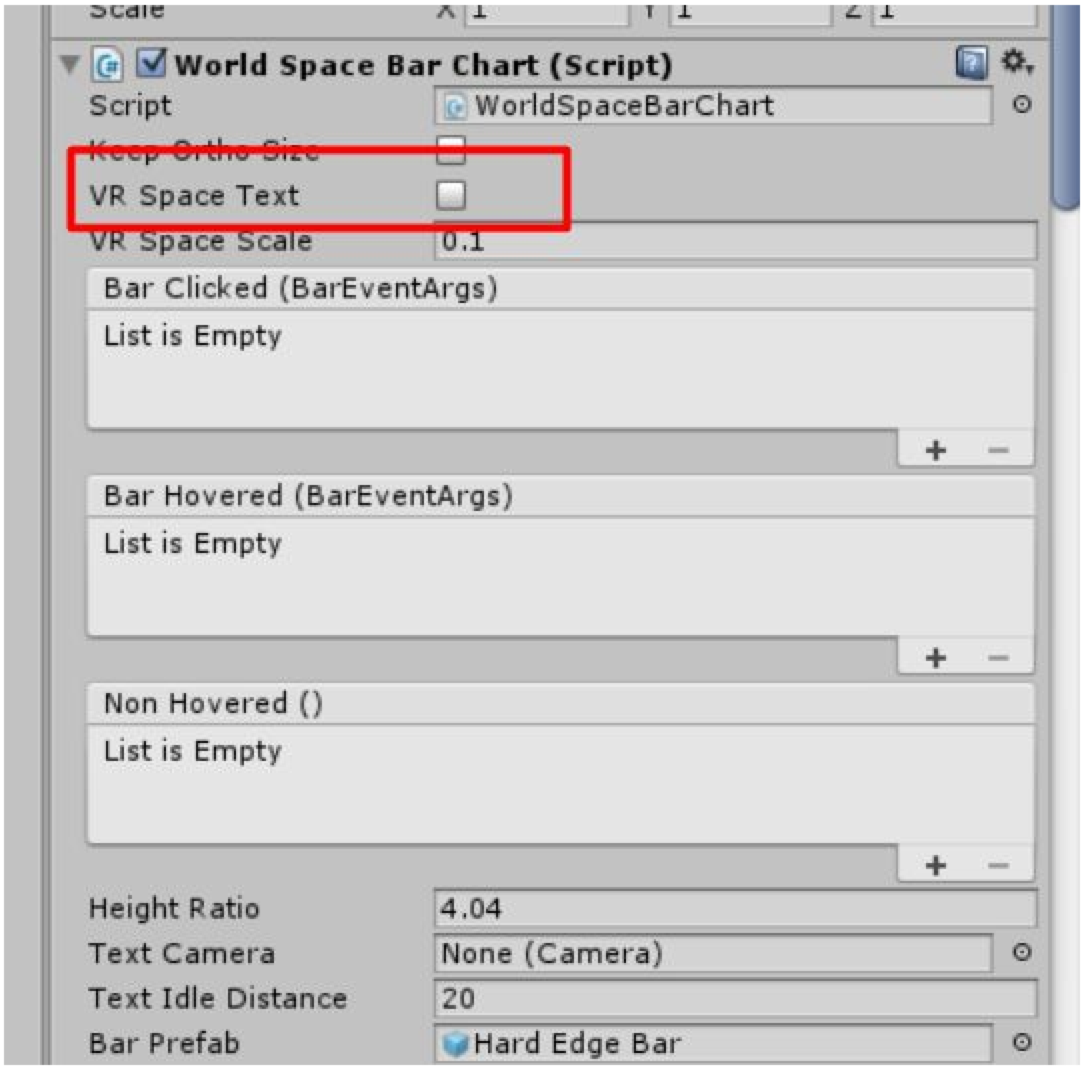
Horizontal Panning ☐

Vertical Panning ☐



Text rotation in VR and 3D

Notice that the default text billboarding in graph and chart rotates the text to face the use at all time. If you wish to change this behavior check the VR Space Text option for your chart.



World Space Bar Chart (Script)

Script	WorldSpaceBarChart
Keep Ortho Size	☐
VR Space Text	☐
VR Space Scale	0.1

Bar Clicked (BarEventArgs)

List is Empty

Bar Hovered (BarEventArgs)

List is Empty

Non Hovered ()

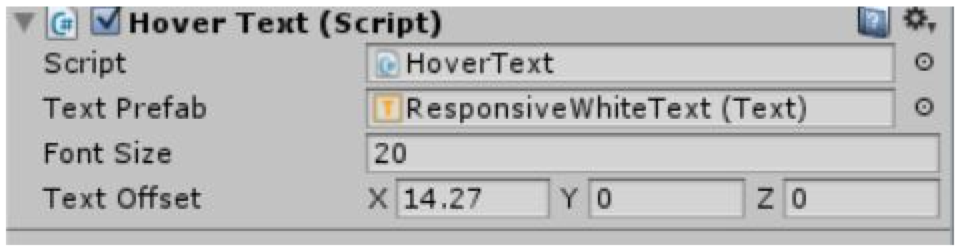
List is Empty

Height Ratio	4.04
Text Camera	None (Camera)
Text Idle Distance	20
Bar Prefab	Hard Edge Bar

Performance

If you are experiencing any type of performance issue please try the following options:

A. If you have many item and you are using ItemLabels for the graph chart or bar chart consider using HoverText component instead :



Hover Text (Script)

Script	HoverText
Text Prefab	ResponsiveWhiteText (Text)
Font Size	20
Text Offset	X 14.27 Y 0 Z 0

B. make sure you are using the latest version of unity

Support

The online version of this document can be found here

For support mail us at : support@prosourcelabs.com