AUTHOR	**Michael Soler**
CONTACT	**michael.soler.beatty@gmail.com**
Unity Ver.	**2018.3.5f1**

Index

1.Description of the package.

From cardboard buddies we pretend to give the best packages to our customers with simplicity and transparency. This package allows the user to create a chart from financial data similar to a stock market that can be generated in a xml file.

This asset contains the following:

- Scripts that generate the chart from a xml file.
- Scripts that generate the data of the xml file.
- Scripts that change the appearance of the chart and its parts.
- Scripts that control the selectable line and gives the values of the chart.

The asset contains the necessary models, textures and prefabs shown in the video.

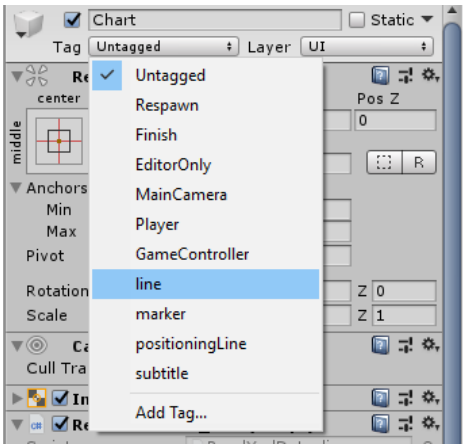
For further information please contact michael.soler.beatty@gmail.com.

2.Dependencies of the package (importing).

This package does not need any other package to work.

3.Colliders, tags and physics

UI elements are used to generate events. Please check that the following tags are in your project:



4.Scripting and Resources

The main scripts used in this asset are:

```
public class ReadXmlData_line : MonoBehaviour
{
    //this is the name of the file
    public string DataStr;

    // line parameters
    [Range(0,15)]
```

```
    public float L_width = 5f;

    [Range(0, 15)]
    public float M_width = 5f;

    [Range(2, 10)]
    public int nb__div = 5;

    [Range(0, 10)]
    public int HL_width = 1;


    // line prefab for UP and DOWN behaviour
    public GameObject prefab_UP, prefab_DOWN, prefab_HORIZ;

    //this is the list for the diccionary of x,y values
    public List<Dictionary<string, string>> allTextDic;

    //these two arrays contain the values for the line chart
    public float[] x_value;
    public float[] y_Open_value;
    public float[] y_Close_value;
    public float[] y_min_value;
    public float[] y_Max_value;

    //maximum values for x and y
    public float xmax, ymax, xmin, ymin;


    //this is the number of point to plot;
    public int nb_points;

    // in the start function the data is loaded and the chart is drawn
    void Start()
    {

        […]

        plotChart();

    }


    // this function is called when a variable is changed in the inspector
    public void replot()
    {
        clearChart();
        plotChart();

    }


    //deletes the chart data
    public void clearChart()
    {

        // get containers for the lines and the markers
        GameObject[] lines = GameObject.FindGameObjectsWithTag("line");

        for (int i = 0; i < lines.Length; i++)
        {
            Destroy(lines[i]);
```

```
        }

    }

    //draws the chart
    public void plotChart()
    {

        […]

    }


    //reads the xml document
    public List<Dictionary<string, string>> parseFile()
    {
        TextAsset txtXmlAsset = Resources.Load<TextAsset>(DataStr);
        var doc = XDocument.Parse(txtXmlAsset.text);


        //get the xml data points
        var allDict = doc.Element("data").Elements("point");
        List<Dictionary<string, string>> allTextDic = new
List<Dictionary<string, string>>();
        foreach (var oneDict in allDict)
        {
            var firstX = oneDict.Elements("x");
            var Yo = oneDict.Elements("yo");
            var Yc = oneDict.Elements("yc");
            var Ym = oneDict.Elements("ym");
            var YM = oneDict.Elements("yM");

            XElement element1 = firstX.ElementAt(0);
            XElement element2 = Yo.ElementAt(0);
            XElement element3 = Yc.ElementAt(0);
            XElement element4 = Ym.ElementAt(0);
            XElement element5 = YM.ElementAt(0);

            //these are the two x and y points
            string first = element1.ToString().Replace("<x>",
"").Replace("</x>", "");
            string second = element2.ToString().Replace("<yo>",
"").Replace("</yo>", "");
            string third = element3.ToString().Replace("<yc>",
"").Replace("</yc>", "");
            string forth = element4.ToString().Replace("<ym>",
"").Replace("</ym>", "");
            string fifth = element5.ToString().Replace("<yM>",
"").Replace("</yM>", "");

            Dictionary<string, string> dic = new Dictionary<string, string>();
            dic.Add("x", first);
            dic.Add("yo", second);
            dic.Add("yc", third);
            dic.Add("ym", forth);
            dic.Add("yM", fifth);

            allTextDic.Add(dic);
        }

        return allTextDic;
```

```
    }

}
```

```
public class OverLine : MonoBehaviour
{
    // Start is called before the first frame update
    Transform lineTF;

    void Start()
    {
        lineTF = GameObject.FindGameObjectWithTag("positioningLine").transform;
    }

    // Update is called once per frame
    public void onEnter()
    {
        lineTF.transform.localPosition = new
Vector3(transform.localPosition[0],0,0);

        //set the text and images to true
        transform.GetChild(0).GetComponent<Image>().enabled = true;
        transform.GetChild(1).GetComponent<Image>().enabled = true;
        transform.GetChild(2).GetComponent<Text>().enabled = true;
        transform.GetChild(3).GetComponent<Text>().enabled = true;
        transform.SetAsLastSibling();
    }

    public void onExit()
    {
        //set the text and images to false
        transform.GetChild(0).GetComponent<Image>().enabled = false;
        transform.GetChild(1).GetComponent<Image>().enabled = false;
        transform.GetChild(2).GetComponent<Text>().enabled = false;
        transform.GetChild(3).GetComponent<Text>().enabled = false;
    }
}
```

Please notice that your resources folder contains the xml data needed for the chart and the code that generate that xml file.

5. Video tutorials

We have a video tutorial explaining how package mechanics works.

https://youtu.be/y3Gx-68jwpE

We also comment the scripts in this video tutorial.

https://youtu.be/86e5OcJcN10

To know how to create the xml data file follow this tutorial:

https://youtu.be/hGoBl2QDNoc

6. Exporting to android

Please notice that the pointer enter and exit event work differently in Android.