SPRINGGUI

Welcome to use SpringGUI

Version v2017.1.1 f0

Author : springDong

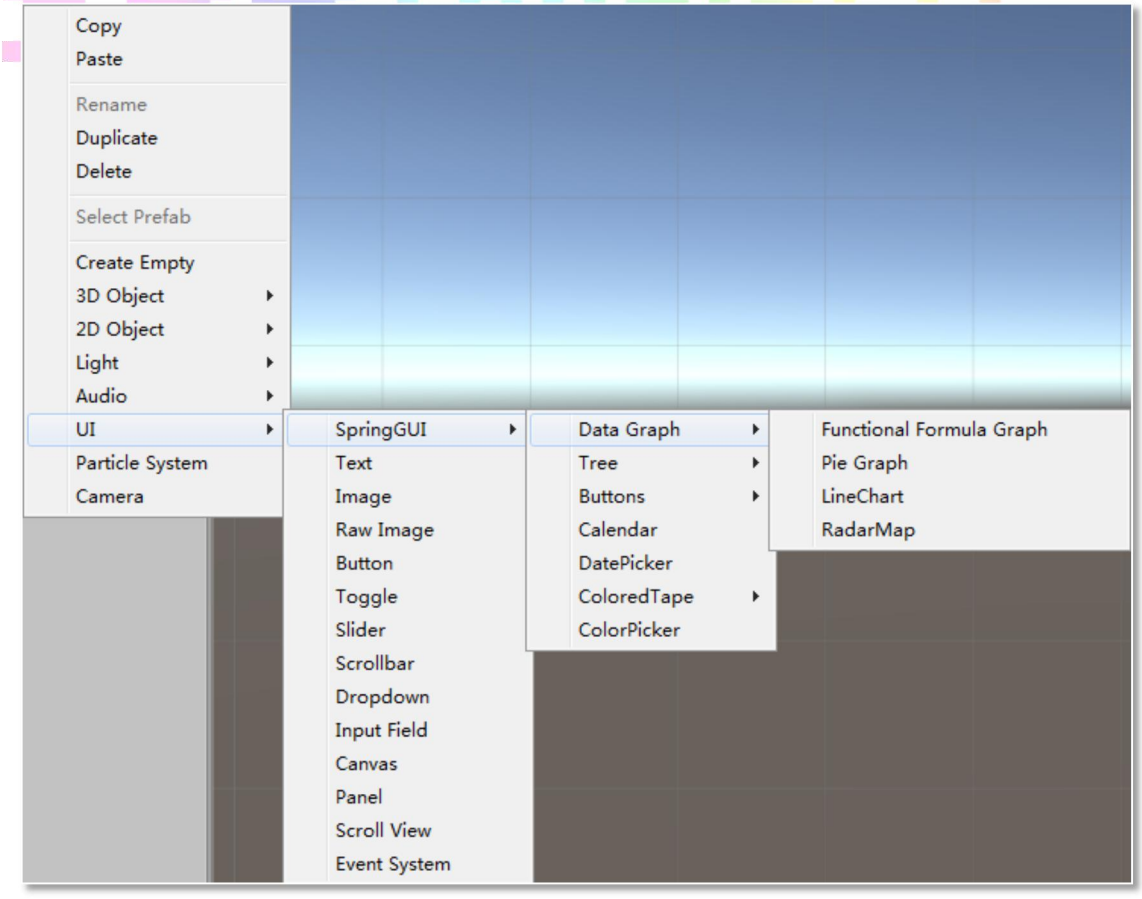
Email : springu3d@yeah.net

1. About SpringGUI

a. What is SpringGUI?

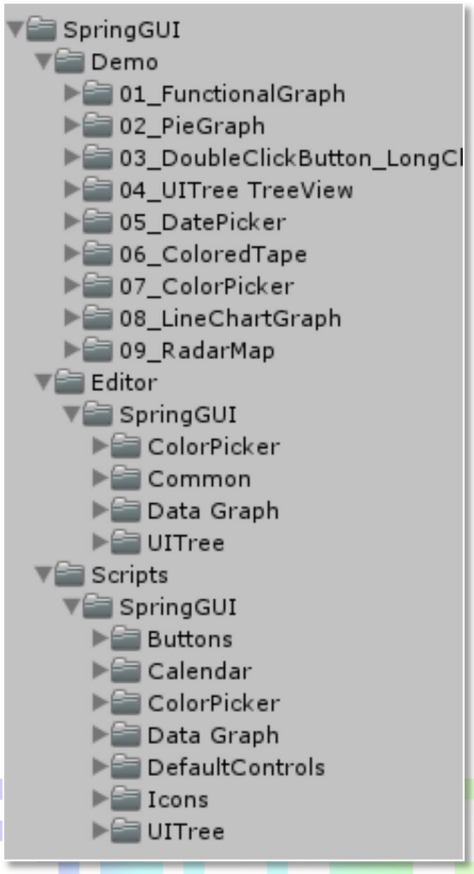
SpringGUI is based on the UGUI expand components it supports Unity5. X. SpringGUI provide some data chart and data gragh, they are very suitable for 3d visualization and virtual simulation project.I start write SpringGUI components from 2017,I will update etwo Components very month .Cut the crap, directly to the content!

b. About SpringGUI's Feature!



SPRINGGUI

You can create SpringGUI component from Unity.GameObject.UI!



Only have code !no assets!

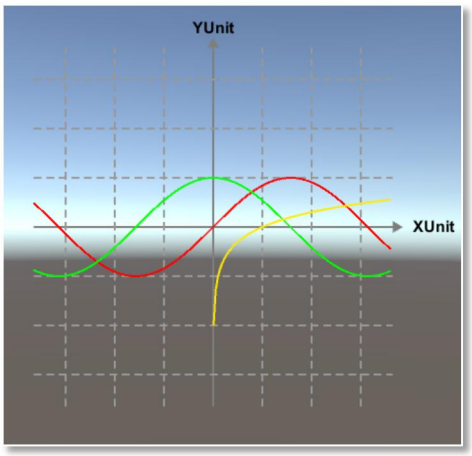
Create SpringGUI component by scripts,all icon drwa by scripts,SpringGUI can easy to transfer to other unity projects,you don't need to worry about resources repetition or some other resources problem.

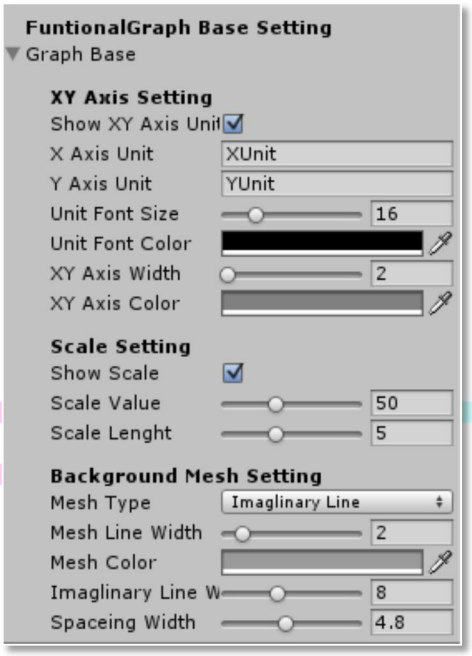
2. SpringGUI Component Effects && Description

Because i am a programmer not art workers, so it looks in general, I'm so sorry. But the appearance can free change, like using UGUI components, if your company or studio have a fine arts related staff, I'm sure he/she can adjust very good-looking.

Let's take a look :

A.FunctionalGraph



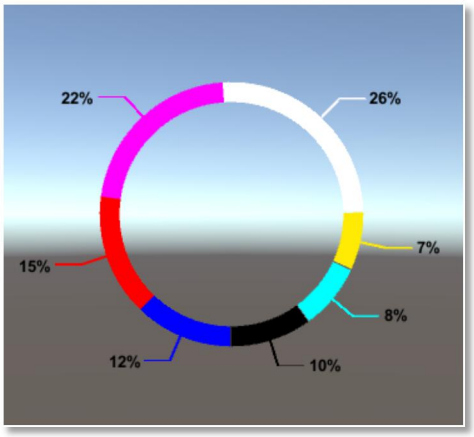


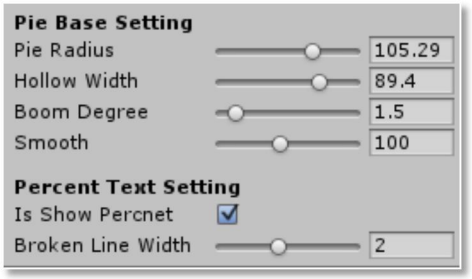
XY Axis Setting

b.Scale Setting

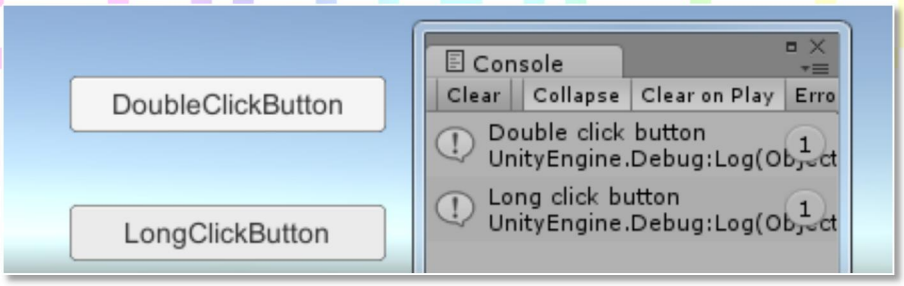
c.Background Mesh Setting

B.PieGraph

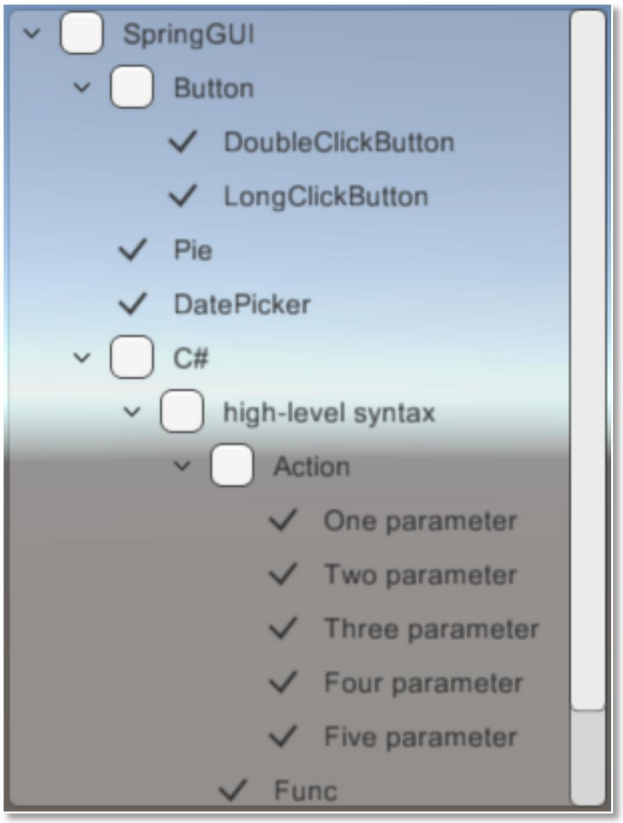




C.Buttons (Double click button && Long click button)



D.TreeView

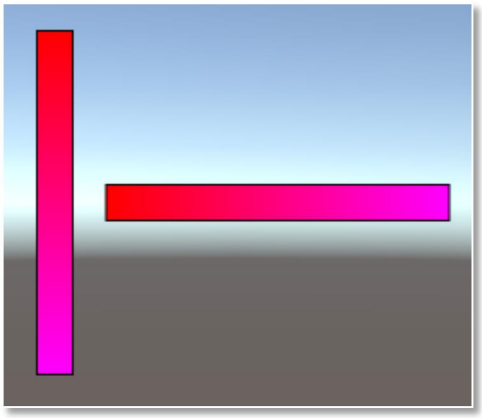


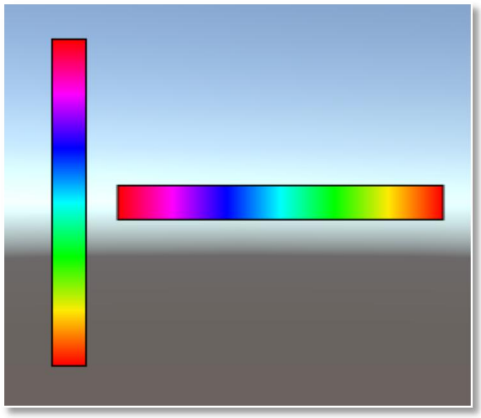
You can set the Close Icon, the Open Icon, the Last Layer Icon (similar to a binary tree Leaf node Icon) sprite in the Inspector panel UITree components . Icon is automatically in the corresponding to the components.

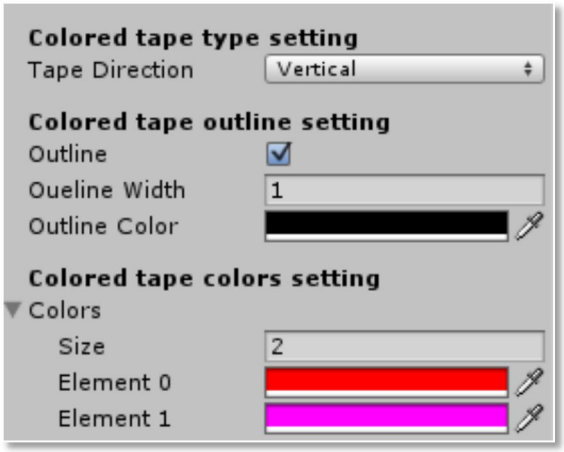
E.DatePicker

Calendar component attributes only one, i set the standard two display mode in English and Chinese, you can set the display type in the Inspector panel ,then it will run automatically.

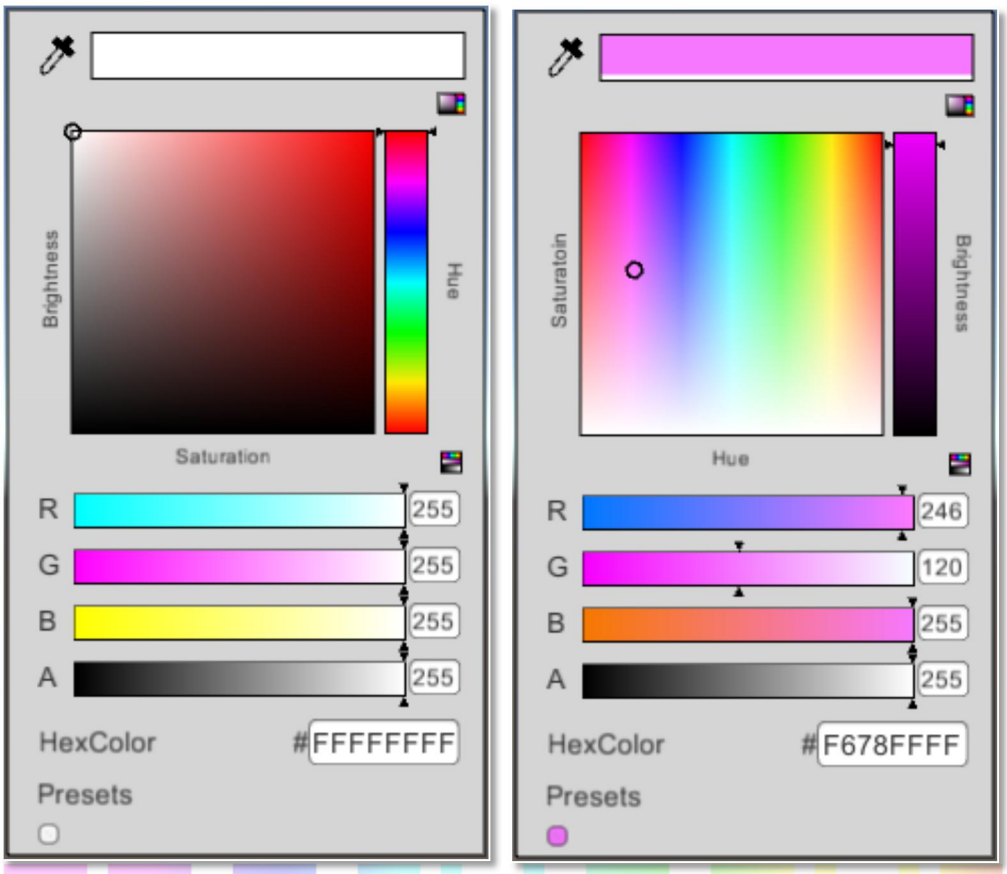
F.ColoredTape





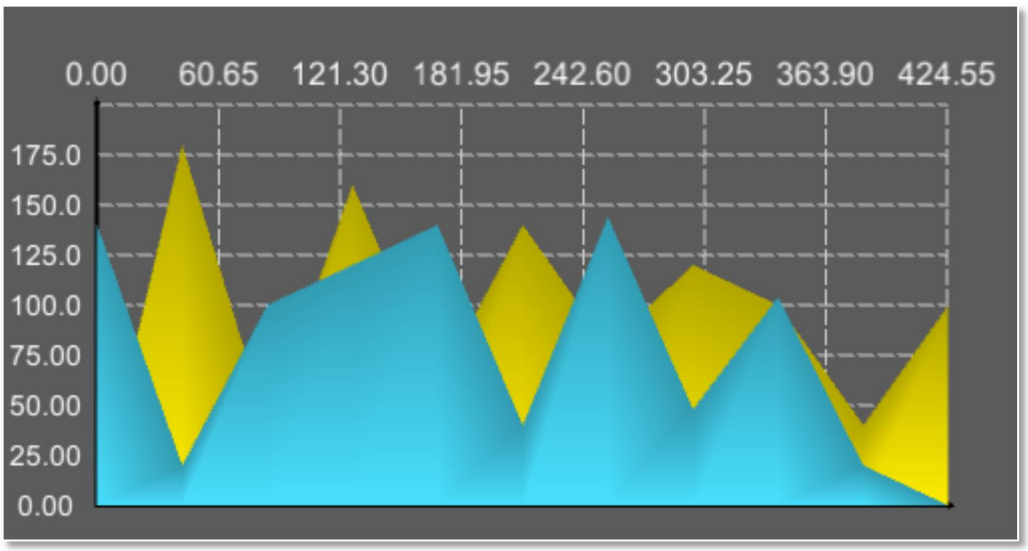


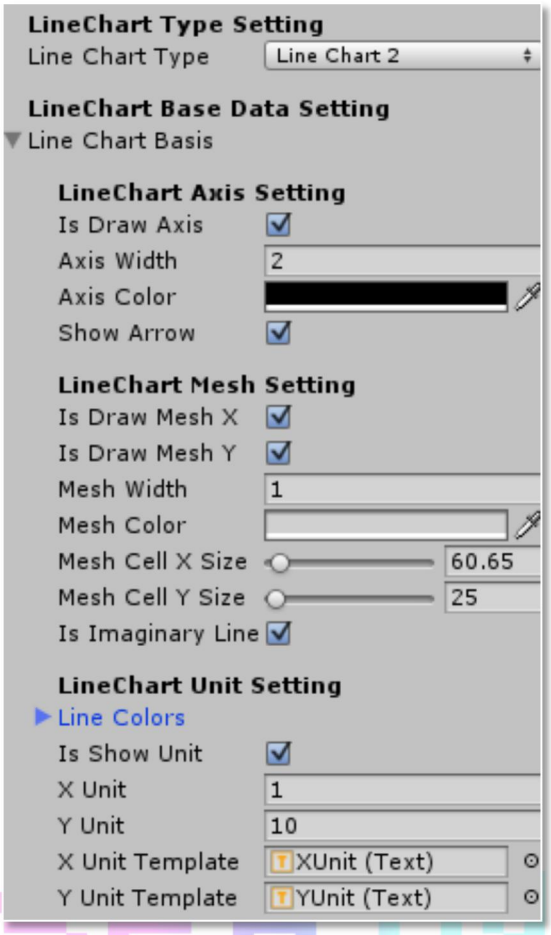
G.ColorPicker



ColorPicker is not support the HSV model, because i do not chosen a good design patterns , later I will update this component, make its function more comprehensive.

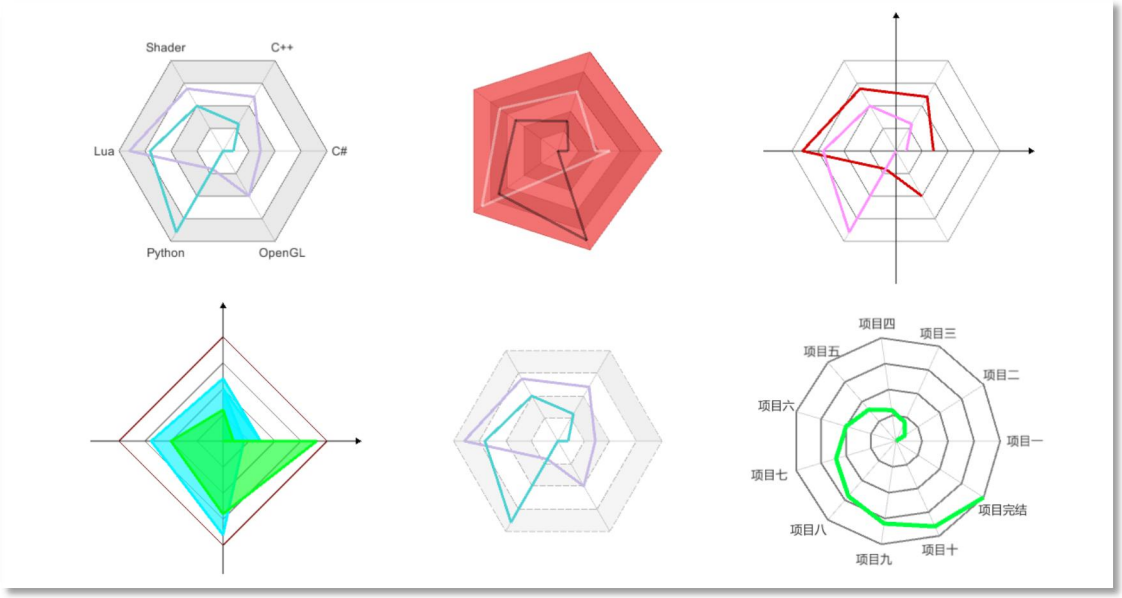
H.LineChart

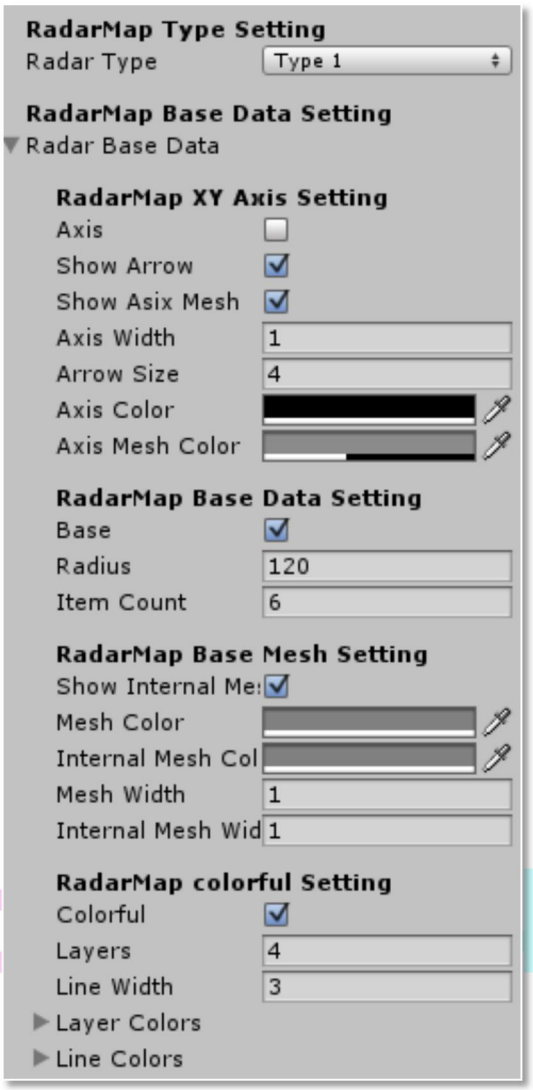




Line chart with a number of customizable properties, you can set the attribute values in the Inspector panel and observe the effect of it, choose that you like, the code user factory pattern design, you can have a extension.

I.RadarMap





Radar chart with more attribute can customize the Settings, go to try it.

3. API && Hot to use

A.FunctionalGraph

External call interface

```
/// <summary> inject formula by base datas
public void Inject( Func<float , float> formula , Color lineColor , float lineWidth = 2.0f )...

/// <summary> inject one formula
public void Inject( FunctionFormula formula )...

/// <summary> inject multi formulas the same time.
public void Inject( IList<FunctionFormula> formulas )...
```

User case

```
// method one
FunctionalGraph.Inject(Mathf.Sin , Color.red , 2.0f);

// method two
FunctionalGraph.Inject(new FunctionFormula(Mathf.Cos , Color.green , 2.0f));

// method three
FunctionalGraph.Inject( new List<FunctionFormula>()
{
    new FunctionFormula(Mathf.Log10,Color.yellow,2.0f),
    new FunctionFormula(Mathf.Abs,Color.cyan,2.0f)
} );
```

B.PieGraph

External call interface

```
public void Inject( IList<float> percents,IList<Color> colors )...

public void Inject( IList<float> percents )...

public void Inject( IList<PieData> pieData )...

public void Inject( Pies pies )...
```

SPRINGGUI

User case

```
// method 1:
PieGraph.Inject(new Pies(new List<PieData>()
    {
        new PieData(26 ,Color.white)
    }));

// method 2:
PieGraph.Inject(new List<PieData>()
{
    new PieData(22,Color.magenta),
    new PieData(15 ,Color.red),
});

// method 3:
PieGraph.Inject(
    new List<float>() { 12 , 10 } ,
    new List<Color>() { Color.blue , Color.black });

// method 4:
PieGraph.Inject(new List<float>() { 8 , 7 });
```

C.Buttons

External call interface

```
[SerializeField]
private DoubleClickedEvent m_onDoubleClick = new DoubleClickedEvent();
public DoubleClickedEvent onDoubleClick
{
    get { return m_onDoubleClick; }
    set { m_onDoubleClick = value; }
}
```

```
[SerializeField]
private LongClickEvent m_onLongClick = null;
public LongClickEvent onLongClick
{
    get { return m_onLongClick; }
    set { m_onLongClick = value; }
}
```

User case

```
DoubleClickButton.onDoubleClick.AddListener(() =>
{
    Debug.Log("Double click button");
});
LongClickButton.onLongClick.AddListener(() =>
{
    Debug.Log("Long click button");
});
```

D.TreeView

External call interface

```
public void Inject( UITreeData rootData )...

// insert new node method. The next version will add this funcion.
[Obsolete("Next version will add this funcion")]
public void Inject( UITreeData insertData , UITreeData parentData )...

[Obsolete("This method is replaced by Inject.")]
public void SetData( UITreeData rootData )...
```

User case

```
var data = new UITreeData("SpringGUI",new List<UITreeData>()
{
    new UITreeData("Button",new List<UITreeData>()
    {
        new UITreeData("DoubleClickButton"),
        new UITreeData("LongClickButton")
    }),
    new UITreeData("Pie"),
    new UITreeData("DatePicker"),
    new UITreeData("C#",new List<UITreeData>()
    {
        new UITreeData("high-level syntax",new List<UITreeData>()
        {
            new UITreeData("Action",new List<UITreeData>()
                {
                    new UITreeData("One parameter"),
                    new UITreeData("Two parameter"),
                    new UITreeData("Three parameter"),
                    new UITreeData("Four parameter"),
                    new UITreeData("Five parameter")
                }),
            new UITreeData("Func"),
            new UITreeData("delegate")
        }),
        new UITreeData("Reflect")
    })
});
//UITree.SetData(data);
UITree.Inject(data);
```

E.DatePicker

External call interface

```
private DayClickEvent m_onDayClickEvent = new DayClickEvent();
public DayClickEvent onDayClick
{
    get { return m_onDayClickEvent; }
    set { m_onDayClickEvent = value; }
}
private MonthClickEvent m_onMonthClickEvent = new MonthClickEvent();
public MonthClickEvent onMonthClick
{
    get { return m_onMonthClickEvent; }
    set { m_onMonthClickEvent = value; }
}
private YearClickEvent m_onYearClickEvent = new YearClickEvent();
public YearClickEvent onYearClick
{
    get { return m_onYearClickEvent; }
    set { m_onYearClickEvent = value; }
}
```

User case

```
Calendar.onDayClick.AddListener(time =>
{
    Debug.Log(string.Format("Today is {0}Yeah{1}Month{2}Day" ,
        time.Year , time.Month , time.Day));
});
Calendar.onMonthClick.AddListener(time =>
{
    Debug.Log(string.Format("This month is {0}Yeah{1}Month" ,
    time.Year , time.Month));
});
Calendar.onYearClick.AddListener(time =>
{
    Debug.Log(string.Format("This yeah{0}Yeah" , time.Year));
});
```

DatePicker type has been listening in the above three kinds of interface, so the datepicker does not need to configure the available date, datepicker provides a external access interface.

```
public DateTime DateTime
{
    get { return _dateTime; }
    set
    {
        _dateTime = value;
        refreshDateText();
    }
}
```


F.ColoredTape

External call interface

//None

User case

//None

G.ColorPicker

External call interface

```
[SerializeField]
private ColorPickerEvent m_onPicker = new ColorPickerEvent();
public ColorPickerEvent onPicker
{
    get { return m_onPicker; }
    set { m_onPicker = value; }
}
```

User case

```
ColorPicker.onPicker.AddListener(color => { Image.color = color; });
```

H.LineChart

External call interface

```
public void Inject<T>( IList<T> vertexs )
{
    if( null == m_dataMediator)
        m_dataMediator = new LineChartDataMediator();
    LineChartBasis.AddLine(m_dataMediator.Inject(vertexs));
}

public void Inject( IList<Vector2> vertexs )
{
    if ( null == m_dataMediator )
        m_dataMediator = new LineChartDataMediator();
    LineChartBasis.AddLine(m_dataMediator.Inject(vertexs));
}
```

User case

```
var data1 = new List<TestData>()
    {
        new TestData(0.0f,0.0f),
        new TestData(0.1f,0.9f),
        new TestData(0.2f,0.2f),
        new TestData(0.3f,0.8f),
        new TestData(0.4f,0.3f),
        new TestData(0.5f,0.7f),
        new TestData(0.6f,0.4f),
        new TestData(0.7f,0.6f),
        new TestData(0.8f,0.5f),
        new TestData(0.9f,0.2f),
        new TestData(1f,0.5f),
    };
var data2 = new List<TestData>()
    {
        new TestData(0.0f,0.7f),
        new TestData(0.1f,0.1f),
        new TestData(0.2f,0.5f),
        new TestData(0.3f,0.6f),
        new TestData(0.4f,0.7f),
        new TestData(0.5f,0.2f),
        new TestData(0.6f,0.72f),
        new TestData(0.7f,0.24f),
        new TestData(0.8f,0.52f),
        new TestData(0.9f,0.1f),
        new TestData(1f,0.0f),
    };
LineChart.Inject<TestData>(data1);
LineChart.Inject<TestData>(data2);
LineChart.ShowUnit();
```

Provides a dynamic data stream input cases.

Please check the source code of 08 _linechartgraph

I.RadarMap

External call interface

```
public void Inject<T>( IList<T> datas )
{
    var radardata = RadarDataProxy.Convert2RD(datas);
    RadarBaseData.Adddata(radardata);
    OnEnable();
}

public void Inject<T>( IList<T>[] datas )
{
    var radardatas = RadarDataProxy.Convert2RD(datas);
    RadarBaseData.Adddata(radardatas);
    OnEnable();
}
```

User case

```
IList<RMExampleData> radarone = new List<RMExampleData>()
    {
        new RMExampleData(0.36f),
        new RMExampleData(0.6f),
        new RMExampleData(0.69f),
        new RMExampleData(0.9f),
        new RMExampleData(0.2f),
        new RMExampleData(0.5f)
    };
IList<RMExampleData> radartwo = new List<RMExampleData>()
    {
        new RMExampleData(0.1f),
        new RMExampleData(0.3f),
        new RMExampleData(0.5f),
        new RMExampleData(0.7f),
        new RMExampleData(0.9f),
        new RMExampleData(0.0f)
    };

RadarMap1.Inject(radarone);
RadarMap1.Inject(radartwo);

RadarMap2.Inject(radarone);
RadarMap2.Inject(radartwo);
```

If you have other questions please call me by Email(springu3d@yeah.net),my english is not really good ,i am so sorry !