demohttps://code.fanruan.com/fanruan/demo-chart-pie

Echarts

...	...
...	..

var chart = echarts.init(dom)chart.setOption(option)

1.BaseColumnFieldCollection@KeyField

DemoColumnFieldCollection

```
package com.fr.plugin.demo;

import com.fanruan.api.report.chart.field.BaseColumnFieldCollection;
import com.fr.chartx.data.annotations.KeyField;
import com.fr.chartx.data.field.ColumnField;

public class DemoColumnFieldCollection extends BaseColumnFieldCollection {

    @KeyField
    private ColumnField category = new ColumnField();

    private ColumnField value = new ColumnField();

    public ColumnField getCategory() {
        return category;
    }

    public void setCategory(ColumnField category) {
        this.category = category;
    }

    public ColumnField getValue() {
        return value;
    }

    public void setValue(ColumnField value) {
        this.value = value;
    }
}
```

2.BaseChartWithData

@FunctionRecorder

DemoChart

```
package com.fr.plugin.demo;
```

```
import com.fanruan.api.cal.FormulaKit;
import com.fanruan.api.log.LogKit;
import com.fanruan.api.report.chart.BaseChartWithData;
import com.fanruan.api.script.FineCanvas;
import com.fanruan.api.util.AssistKit;
import com.fanruan.api.util.IOKit;
import com.fanruan.api.util.StringKit;
import com.fr.base.BaseFormula;
import com.fr.base.chart.cross.FormulaProcessor;
import com.fr.chart.ChartWebParaProvider;
import com.fr.extended.chart.HyperLinkPara;
import com.fr.json.JSON;
import com.fr.json.JSONArray;
import com.fr.json.JSONFactory;
import com.fr.json.JSONObject;
import com.fr.plugin.transform.FunctionRecorder;
import com.fr.stable.xml.XMLPrintWriter;
import com.fr.stable.xml.XMLableReader;

import java.util.List;
import java.awt.Image;
import java.awt.image.BufferedImage;

@FunctionRecorder
public class DemoChart extends BaseChartWithData {

    private static final String ID = "DEMO_CHART";

    private BaseFormula titleFormula = FormulaKit.newFormula(StringKit.EMPTY);

    private PieType pieType = PieType.PIE;

    private String legendPosition = "left";

    public PieType getPieType() {
        return pieType;
    }

    public void setPieType(PieType pieType) {
        this.pieType = pieType;
    }

    public BaseFormula getTitleFormula() {
        return titleFormula;
    }

    public void setTitleFormula(BaseFormula titleFormula) {
        this.titleFormula = titleFormula;
    }

    public String getLegendPosition() {
        return legendPosition;
    }

    public void setLegendPosition(String legendPosition) {
        this.legendPosition = legendPosition;
    }

        //
    @Override
    protected Image designImage(int width, int height, int resolution, ChartWebParaProvider chartWebPara) {
        switch (pieType) {
            case PIE:
                return IOKit.readImageWithCache("com/fr/plugin/demo/pieType.png");
            default:
                return IOKit.readImageWithCache("com/fr/plugin/demo/ringType.png");
        }
    }

        //FineCanvas
```

```
    @Override
    protected Image exportImage(int width, int height, int resolution, ChartWebParaProvider chartWebPara) {
        BufferedImage bufferedImage = new BufferedImage(5, 5, BufferedImage.TYPE_INT_ARGB);
        try {
            FineCanvas canvas = new FineCanvas("/com/fr/plugin/demo/echarts-adapter.js", "/com/fr/plugin/demo
/echarts.js");
            canvas.loadText("canvas.height = " + height, "canvas.width = " + width);
            canvas.loadText("var myChart = echarts.init(canvas)");
            canvas.loadText("option = " + createAttributeConfig(chartWebPara).toString());
            canvas.loadText("myChart.setOption(option);");
            bufferedImage = canvas.paint();
        } catch (Exception ex) {
            LogKit.error(ex.getMessage(), ex);
        }
        return bufferedImage;
    }

    @Override
    public DemoChart clone() throws CloneNotSupportedException {
        DemoChart result = (DemoChart) super.clone();
        if (getTitleFormula() != null) {
            result.setTitleFormula(this.getTitleFormula().clone());
        }
        result.setPieType(this.getPieType());
        result.setLegendPosition(this.getLegendPosition());
        return result;
    }

    @Override
    public int hashCode() {
        return super.hashCode() + AssistKit.hashCode(this.getTitleFormula(), this.getPieType(), this.
getLegendPosition());
    }

    @Override
    public boolean equals(Object ob) {
        return super.equals(ob)
                && ob instanceof DemoChart
                && AssistKit.equals(this.getTitleFormula(), ((DemoChart) ob).getTitleFormula())
                && AssistKit.equals(this.getPieType(), ((DemoChart) ob).getPieType())
                && AssistKit.equals(this.getLegendPosition(), ((DemoChart) ob).getLegendPosition());
    }

        //Echartsoptionsoptions
        //FieldCollectiongetFieldCollection
    @Override
    public JSONObject createAttributeConfig(ChartWebParaProvider chartWebPara) {
        JSONObject jsonObject = super.createAttributeConfig(chartWebPara);

        jsonObject.put("title", JSONFactory.createJSON(JSON.OBJECT).put("text", getTitleFormula().getResult()).
put("x", "center"));
        jsonObject.put("tooltip", JSONFactory.createJSON(JSON.OBJECT).put("trigger", "item").put("formatter",
"{b}: {c} ({d}%)"));

        DemoColumnFieldCollection columnFieldCollection = getFieldCollection(DemoColumnFieldCollection.class);
        List<Object> category = columnFieldCollection.getCategory().getValues();
        List<Object> values = columnFieldCollection.getValue().getValues();
        JSONArray legendData = JSONFactory.createJSON(JSON.ARRAY);
        JSONArray seriesData = JSONFactory.createJSON(JSON.ARRAY);
        for (int i = 0; i < category.size(); i++) {
            legendData.put(category.get(i));
            seriesData.put(JSONFactory.createJSON(JSON.OBJECT).put("name", category.get(i)).put("value", values.
get(i)));
        }
        jsonObject.put("legend", JSONFactory.createJSON(JSON.OBJECT).put("orient", "vertical").put("left",
getLegendPosition()).put("data", legendData));
        JSONObject series = JSONFactory.createJSON(JSON.OBJECT);
        series.put("type", "pie");
        switch (pieType) {
            case PIE:
                series.put("radius", "55%");
```

```
                break;
            default:
                series.put("radius", JSONFactory.createJSON(JSON.ARRAY).put("35%").put("55%"));
                break;
        }
        series.put("data", seriesData);
        jsonObject.put("series", series);
        return jsonObject;
    }

        //
    @Override
    public void dealFormula(FormulaProcessor formulaProcessor) {
        if (titleFormula != null) {
            formulaProcessor.dealWith(titleFormula);
        }
        super.dealFormula(formulaProcessor);
    }

    @Override
    public String getID() {
        return ID;
    }

        //xml
    @Override
    public void readAttr(XMLableReader xmLableReader) {
        super.readAttr(xmLableReader);

        this.setPieType(PieType.parseInt(xmLableReader.getAttrAsInt("pieType", 0)));
        this.setTitleFormula(FormulaKit.newFormula(xmLableReader.getAttrAsString("title", "")));
        this.setLegendPosition(xmLableReader.getAttrAsString("legendPosition", StringKit.EMPTY));
    }

        //xml
    @Override
    public void writeAttr(XMLPrintWriter xmlPrintWriter) {
        super.writeAttr(xmlPrintWriter);
        xmlPrintWriter.attr("pieType", pieType.ordinal())
                .attr("title", titleFormula.toString())
                .attr("legendPosition", legendPosition);
    }

        //Categorydataname
    private static final HyperLinkPara Category = new HyperLinkPara() {
        @Override
        public String getName() {
            return "";
        }

        @Override
        public String getFormulaContent() {
            return "Category";
        }

        @Override
        public String[] getProps() {
            return new String[]{"data", "name"};
        }
    };

    @Override
    protected HyperLinkPara[] hyperLinkParas() {
        return new HyperLinkPara[]{
                Category,
        };
    }
}
```

3.BaseCellDataFieldsPaneBaseDataSetFieldsPaneDemoColumnFieldCollection

1

UIFormulaTextField

DemoCellDataFieldsPane

```
package com.fr.plugin.demo;

import com.fanruan.api.design.DesignKit;
import com.fanruan.api.design.chart.field.BaseCellDataFieldsPane;
import com.fanruan.api.design.ui.component.formula.UIFormulaTextField;

import java.awt.Component;

public class DemoCellDataFieldsPane extends BaseCellDataFieldsPane<DemoColumnFieldCollection> {

    private UIFormulaTextField categoryPane;
    private UIFormulaTextField valuePane;

    public void initComponents(){
        categoryPane = new UIFormulaTextField();
        valuePane = new UIFormulaTextField();
        super.initComponents();
    }

        //
    @Override
    protected String[] fieldLabels() {
        return new String[]{
                DesignKit.i18nText("Fine-Plugin_Demo_Category"),
                DesignKit.i18nText("Fine-Plugin_Demo_Value")
        };
    }

        //
    @Override
    protected Component[] fieldComponents() {
        return new Component[]{
                categoryPane,
                valuePane
        };
    }

        //
    @Override
    protected UIFormulaTextField[] formulaPanes() {
        return new UIFormulaTextField[]{
                categoryPane,
                valuePane
        };
    }

        //DemoColumnFieldCollectionpopulateField
    @Override
    public void populateBean(DemoColumnFieldCollection dataConf) {
        populateField(categoryPane, dataConf.getCategory());
        populateField(valuePane, dataConf.getValue());
    }

        //DemoColumnFieldCollectionupdateField
    @Override
    public DemoColumnFieldCollection updateBean() {
        DemoColumnFieldCollection dataConfig = new DemoColumnFieldCollection();

        updateField(categoryPane, dataConfig.getCategory());
        updateField(valuePane, dataConfig.getValue());

        return dataConfig;
    }
}
```

UIComboBoxCalculateComboBox

DemoDataSetFieldsPane

```
package com.fr.plugin.demo;

import com.fanruan.api.design.DesignKit;
import com.fanruan.api.design.chart.field.BaseDataSetFieldsPane;
import com.fanruan.api.design.ui.component.UIComboBox;
import com.fanruan.api.design.ui.component.chart.CalculateComboBox;

import java.awt.Component;

public class DemoDataSetFieldsPane extends BaseDataSetFieldsPane<DemoColumnFieldCollection> {

    private UIComboBox categoryPane;
    private UIComboBox valuePane;
    private CalculateComboBox calculateComboBox;

    public void initComponents() {
        categoryPane = new UIComboBox();
        valuePane = new UIComboBox();
        calculateComboBox = new CalculateComboBox();
        super.initComponents();
    }

        //
    @Override
    protected String[] fieldLabels() {
        return new String[]{
                DesignKit.i18nText("Fine-Plugin_Demo_Category"),
                DesignKit.i18nText("Fine-Plugin_Demo_Value"),
                DesignKit.i18nText("Fine-Plugin_Demo_Summary_Method")
        };
    }

        //
    @Override
    protected Component[] fieldComponents() {
        return new Component[]{
                categoryPane,
                valuePane,
                calculateComboBox
        };
    }

    //
    @Override
    protected UIComboBox[] filedComboBoxes() {
        return new UIComboBox[]{
                categoryPane,
                valuePane
        };
    }

        //DemoColumnFieldCollectionpopulateFieldpopulateFunctionField
    @Override
    public void populateBean(DemoColumnFieldCollection dataConf) {
        populateField(categoryPane, dataConf.getCategory());
        populateFunctionField(valuePane, calculateComboBox, dataConf.getValue());
    }

        //DemoColumnFieldCollectionupdateFieldupdateFunctionField
    @Override
    public DemoColumnFieldCollection updateBean() {
        DemoColumnFieldCollection dataConfig = new DemoColumnFieldCollection();

        updateField(categoryPane, dataConfig.getCategory());
        updateFunctionField(valuePane, calculateComboBox, dataConfig.getValue());

        return dataConfig;
    }
}
```

4.BaseChartType

DemoType

```
package com.fr.plugin.demo;

import com.fanruan.api.report.chart.BaseChartType;
import com.fanruan.api.report.chart.BaseChartWithData;


public class DemoType extends BaseChartType {

    //DemoChartpieType
    public BaseChartWithData[] getChartTypes() {
        return new BaseChartWithData[]{
                createDemoChart(PieType.PIE),
                createDemoChart(PieType.RING)
        };
    }


    //webJSecharts.jsdemoWrapper.js
    public String[] getRequiredJS() {
        return new String[]{
                "com/fr/plugin/demo/demoWrapper.js",
                "com/fr/plugin/demo/echarts.js"
        };
    }

    //webCSSdemo.css
    public String[] getRequiredCss() {
        return new String[]{
                "com/fr/plugin/demo/demo.css"
        };
    }

    //JSdemoWrapper.js
    public String getWrapperName() {
        return "demoWrapper";
    }

    private DemoChart createDemoChart(PieType pieType) {
        DemoChart demoChart = new DemoChart();
        demoChart.setPieType(pieType);
        return demoChart;
    }
}
```

5.DefaultTypePane

DemoTypePane

```
package com.fr.plugin.demo;

import com.fanruan.api.design.DesignKit;
import com.fanruan.api.design.chart.DefaultTypePane;
import com.fanruan.api.design.ui.component.UIButtonGroup;
import com.fanruan.api.util.StringKit;

import javax.swing.JPanel;
import java.awt.Component;

public class DemoTypePane extends DefaultTypePane<DemoChart> {

    private UIButtonGroup buttonGroup = new UIButtonGroup(new String[]{DesignKit.i18nText("Fine-
Plugin_Legend_Right"), DesignKit.i18nText("Fine-Plugin_Legend_Left")});

    //
    @Override
    protected String[] getTypeIconPath() {
        return new String[]{
                "com/fr/plugin/demo/pieType.png",
                "com/fr/plugin/demo/ringType.png"
        };
    }

        //chartchartpieType
    @Override
    protected int getSelectIndexInChart(DemoChart chart) {
        return chart.getPieType().ordinal();
    }

        //chart
    @Override
    protected void setSelectIndexInChart(DemoChart chart, int index) {
        chart.setPieType(PieType.parseInt(index));
    }

        //typePane
    @Override
    protected Component[][] getPaneComponents(JPanel typePane) {
        return new Component[][]{
                new Component[]{typePane},
                new Component[]{buttonGroup}
        };
    }

        //
    @Override
    public void populateBean(DemoChart ob) {
        super.populateBean(ob);
        buttonGroup.setSelectedIndex(StringKit.equals("left", ob.getLegendPosition()) ? 0 : 1);
    }

        //
    @Override
    public void updateBean(DemoChart ob) {
        super.updateBean(ob);
        ob.setLegendPosition(buttonGroup.getSelectedIndex() == 0 ? "left" : "right");
    }
}
```

6.

BaseOtherPanecreateContentPane

DemoTitlePane

```
package com.fr.plugin.demo;

import com.fanruan.api.design.DesignKit;
import com.fanruan.api.design.chart.BaseOtherPane;
import com.fanruan.api.design.ui.component.formula.UIFormulaTextField;

import javax.swing.JPanel;
import java.awt.BorderLayout;

public class DemoTitlePane extends BaseOtherPane<DemoChart> {

        //
    private UIFormulaTextField title;

        //
    @Override
    public void populate(DemoChart ob) {
        title.populateBean(ob.getTitleFormula().toString());
    }

        //
    @Override
    public void update(DemoChart ob) {
        ob.getTitleFormula().setContent(title.updateBean());
    }

        //
    @Override
    protected JPanel createContentPane() {
        JPanel panel = new JPanel(new BorderLayout(0, 6));
        title = new UIFormulaTextField();
        panel.add(title, BorderLayout.CENTER);
        return panel;
    }

        //
    @Override
    public String title4PopupWindow() {
        return DesignKit.i18nText("Fine-Plugin_Demo_Title");
    }
}
```

FineKitDefaultOtherPane

1

2createContentPanecreateRefreshPanecreateHyperlinkPane

7.IconBaseChartTypeUI

SingleDataPaneBaseDataPanecreateSingleDataPane

DemoUI

```
package com.fr.plugin.demo;

import com.fanruan.api.design.DesignKit;
import com.fanruan.api.design.chart.BaseChartTypeUI;
import com.fanruan.api.design.chart.BaseDataPane;
import com.fanruan.api.design.chart.BaseOtherPane;
import com.fanruan.api.design.chart.DefaultOtherPane;
import com.fanruan.api.design.chart.DefaultTypePane;
import com.fanruan.api.design.chart.SingleDataPane;
import com.fr.design.gui.frpane.AttributeChangeListener;

public class DemoUI extends BaseChartTypeUI {

        //
    @Override
    public DefaultTypePane getPlotTypePane() {
        return new DemoTypePane();
    }

        //
    @Override
    public BaseDataPane getChartDataPane(AttributeChangeListener listener) {
        return new BaseDataPane(listener) {
            @Override
            protected SingleDataPane createSingleDataPane() {
                return new SingleDataPane(new DemoDataSetFieldsPane(), new DemoCellDataFieldsPane());
            }
        };
    }

        //
    @Override
    public BaseOtherPane[] getAttrPaneArray(AttributeChangeListener listener) {
        return new BaseOtherPane[]{new DemoTitlePane(), new DefaultOtherPane()};
    }

        //Icon
    @Override
    public String getIconPath() {
        return "com/fr/plugin/demo/icon.png";
    }

        //
    @Override
    public String getName() {
        return DesignKit.i18nText("Fine-Plugin_Demo_Chart");
    }

        //
    @Override
    public String[] getSubName() {
        return new String[]{
                DesignKit.i18nText("Fine-Plugin_Pie"),
                DesignKit.i18nText("Fine-Plugin_Ring")
        };
    }

        //
    @Override
    public String[] getDemoImagePath() {
        return new String[]{
                "com/fr/plugin/demo/pie.png",
                "com/fr/plugin/demo/ring.png"
        };
    }
}
```

8.JS

Van.FRChartBridge.demoWrapperDemoType.getWrapperName()

demoWrapper.js

```
!(function () {
    Van.FRChartBridge.demoWrapper = Van.FRChartBridge.AbstractChart.extend({
                //echartsecharts
        _init: function (dom, option) {
            var chart = echarts.init(dom);
            //
            chart.on('click', this.getLinkFun());
            chart.setOption(option);
            return chart;
        },

                 //
        _refresh: function (chart, option) {
            chart.setOption(option);
        },

                //options.series.data.length0
        _emptyData: function (options) {
            return options.series.data.length === 0;
        }
    })
})();
```

9.plugin.xml

idjarjarplugin-com.fr.plugin.demoChart-1.0.0.jar

function-recorderDemoChart@FunctionRecorder

ChartTypeProviderChartTypeUIProviderBaseChartTypeBaseChartTypeUIchartIDDemoChartgetID

LocaleFinder8.——

plugin.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><plugin>
 <id>com.fr.plugin.demoChart</id>
 <name><![CDATA[DEMO]]></name>
 <active>yes</active>
 <version>1.0.0</version>
 <env-version>10.0</env-version>
 <jartime>2019-10-09</jartime>
 <vendor>Bjorn</vendor>
 <description><![CDATA[demo]]></description>
 <change-notes><![CDATA[]]></change-notes>

 <function-recorder class="com.fr.plugin.demo.DemoChart"/>

 <extra-chart>
 <ChartTypeProvider class="com.fr.plugin.demo.DemoType" chartID="DEMO_CHART"/>
 </extra-chart>

 <extra-core>
 <LocaleFinder class="com.fr.plugin.demo.DemoLocaleFinder"/>
 </extra-core>

 <extra-chart-designer>
 <ChartTypeUIProvider class="com.fr.plugin.demo.DemoUI" chartID="DEMO_CHART"/>
 </extra-chart-designer>
</plugin>
```