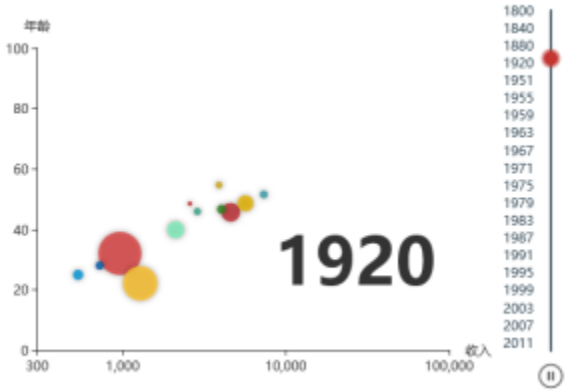
beforeRenderer示例说明

beforeRenderer

1、beforeRenderer：图形渲染之前执行，主要用于修改图形样式（options），只能使用图形控件开放的options配置，宏代码执行完成后才刷新图形。

> *beforeRenderer示例代码：以官网示例demo“图形主题”页面中动态散点图为例，说明如何修改图形options。demo效果*



获取图形对象

```
function main(chartView) {
//        1smartbioptions,json
//        jsonEcharts http://echarts.baidu.com/option.html#title
//        optionsjsonEcharts
         var chart = chartView.getChartObject();
    var option = chart.getOptions();

//        2optionoptionsoptions http://echarts.baidu.com/option.html#title
//
    var series = option.series;
    for (var i in series) {
        var seriesItem = series[i];
        data = seriesItem.data;
        delete seriesItem.data; //optiondataoptions
    }
    var dataOptions = [];
//    ...................
//
        option.grid = {
                right: '110' //timeline
        };

//        options

//        demooptions2
//  ...................
//  timelineoptions
    var newOptions = {
        baseOption: option,
        options: dataOptions
    };
//        3options
    chart.setOptions(newOptions);
}
```

详细代码示例，仅供参考：

```
function main(chartView) {
    var chart = chartView.getChartObject();
    var option = chart.getOptions();
    var data = [];
    var series = option.series;

    option.chartex = {
        timelineFieldIndex: 3
    };
    if (!option.chartex || !option.chartex.timelineFieldIndex) {
        return;
    }

    for (var i in series) {
        var seriesItem = series[i];
        data = seriesItem.data;
        delete seriesItem.data; //optiondataoptions
    }

    var timelineData = [],
        chartData = chartView.getGridData().data, //
        oldValue = "",
        dataSlice = []; //
    for (var i = 0; i < chartData.length; i++) {
        var row = chartData[i];
        //
        var cellValue = row[option.chartex.timelineFieldIndex].value;
        if (oldValue != cellValue) {
            timelineData.push(cellValue);
            dataSlice.push(i);
            oldValue = cellValue;
        }
    }

    // timeline,
    option.timeline = {
        data: timelineData, ////timeline
        axisType: 'category',
        orient: 'vertical',
        autoPlay: true,
        inverse: true,
        playInterval: 100,
        left: null,
        right: 0,
        top: 20,
        bottom: 20,
        width: 55,
        height: null,
        symbol: 'none',
        controlStyle: {
            showNextBtn: false,
            showPrevBtn: false,
            normal: {
                color: '#666',
                borderColor: '#666'
            }
        }
    };
    option.grid = {
        right: '110' //timeline
    };
    //
    option.title = {
        text: timelineData[0] + '',
        textAlign: 'center',
        left: '63%',
        top: '55%',
```

```
    textStyle: {
        fontSize: 60
    }
};
        // color: 'rgba(255, 255, 255, 0.7)'

var itemStyle = {
    normal: {
        opacity: 0.8,
        shadowBlur: 5,
        shadowOffsetX: 0,
        shadowOffsetY: 0,
        shadowColor: 'rgba(25, 0, 0, 0.5)'
    }
};
// timeline
var newdata = data;
var dataOptions = [];
for (i = 1; i < dataSlice.length; i++) {
    var data1 = newdata.slice(dataSlice[i - 1], dataSlice[i]);
    dataOptions.push({
        series: {
            data: data1,
            itemStyle: itemStyle,
            symbolSize: function(val) {
                var x = val[2];
                var y = Math.sqrt(x / 5e8) + 0.1;
                return y * 40;
            }
        },
        title: {
            show: true,
            text: timelineData[i - 1] + ''
        }
    });
    if (i == dataSlice.length - 1) {
        data1 = newdata.slice(dataSlice[i]);
        dataOptions.push({
            series: {
                data: data1,
                itemStyle: itemStyle,
                symbolSize: function(val) {
                    var x = val[2];
                    var y = Math.sqrt(x / 5e8) + 0.1;
                    return y * 40;
                }
            },
            title: {
                show: true,
                text: timelineData[i] + ''
            }
        });
    }
}
//
var newOptions = {
    baseOption: option,
    options: dataOptions
};
//visualMap,,
var visualMapCategories = [];
for (var i = 0; i < data1.length; i++) {
    var name = data1[i].value[3];
    visualMapCategories.push(name);
}
// visualMapCategories = ["China","United States","United Kingdom","Russia",
// "India","France","Germany","Australia","Canada","Cuba","Finland","Iceland","Japan",
// "North Korea","South Korea","New Zealand","Norway","Poland","Turkey"];
var visualMap2 = {
    show: false,
    dimension: 3,
```

```
        categories: visualMapCategories,
        calculable: true,
        precision: 0.2,
        inRange: {
            color: [ '#dd4444', '#fec42c', '#80F1BE','#c12e34', '#e6b600', '#0098d9', '#2b821d', '#005eaa',
'#339ca8', '#cda819', '#32a487']
        }
    };
    option.visualMap = visualMap2;
    chart.setOptions(newOptions);
}
```