**Echarts2.2.7技术文档**

下载地址: <https://git.oschina.net/duhongming/echarts2javatag>

# 引用jar包说明

1）静态图表只依赖下面两个包，Echarts2.2.7是Echarts2的最终版本，也是最好用的。Echarts3更改了很多配置项，每时每刻都有bug产生，Echarts2是稳定版本。

<!-- 必须包：Echarts图表依赖包开始 -->

<dependency>

<groupId>com.github.abel533</groupId>

<artifactId>ECharts</artifactId>

<version>2.2.7</version>

</dependency>

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>[2.6.2,)</version>

</dependency>

<!-- 必须包：Echarts图表依赖包开始结束 -->

2）为了演示大数据量，从Sqlite数据库中获取的数据：是近三年中国各个地区每天的温度。

<!-- 演示包：Sqlite数据库API以及ORM框架开始 -->

<dependency>

<groupId>org.xerial</groupId>

<artifactId>sqlite-jdbc</artifactId>

<version>3.8.11.2</version>

</dependency>

<dependency>

<groupId>com.j256.ormlite</groupId>

<artifactId>ormlite-jdbc</artifactId>

<version>4.47</version>

</dependency>

<!-- 演示包：Sqlite数据库API以及ORM框架结束 -->

3）动态图表是后台spring socket推送数据，而不是简单的前台ajax js轮询。

<!-- spring websocket-->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-websocket</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-messaging</artifactId>

<version>${spring.version}</version>

</dependency>

4）Spring Socket推送的json数据，依赖jackson。

<!-- 由于Spring采用对JSON进行了封装的jackson来生成JSON和返回给客户端 开始-->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-core</artifactId>

<version>2.4.4</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.4.4</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-annotations</artifactId>

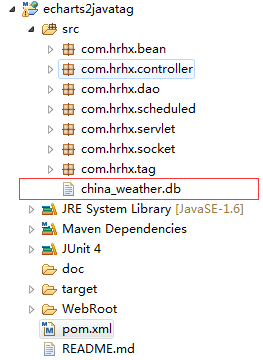
<version>2.4.4</version>

</dependency>

<!-- 由于Spring采用对JSON进行了封装的jackson生成JSON和返回给客户端 结束-->

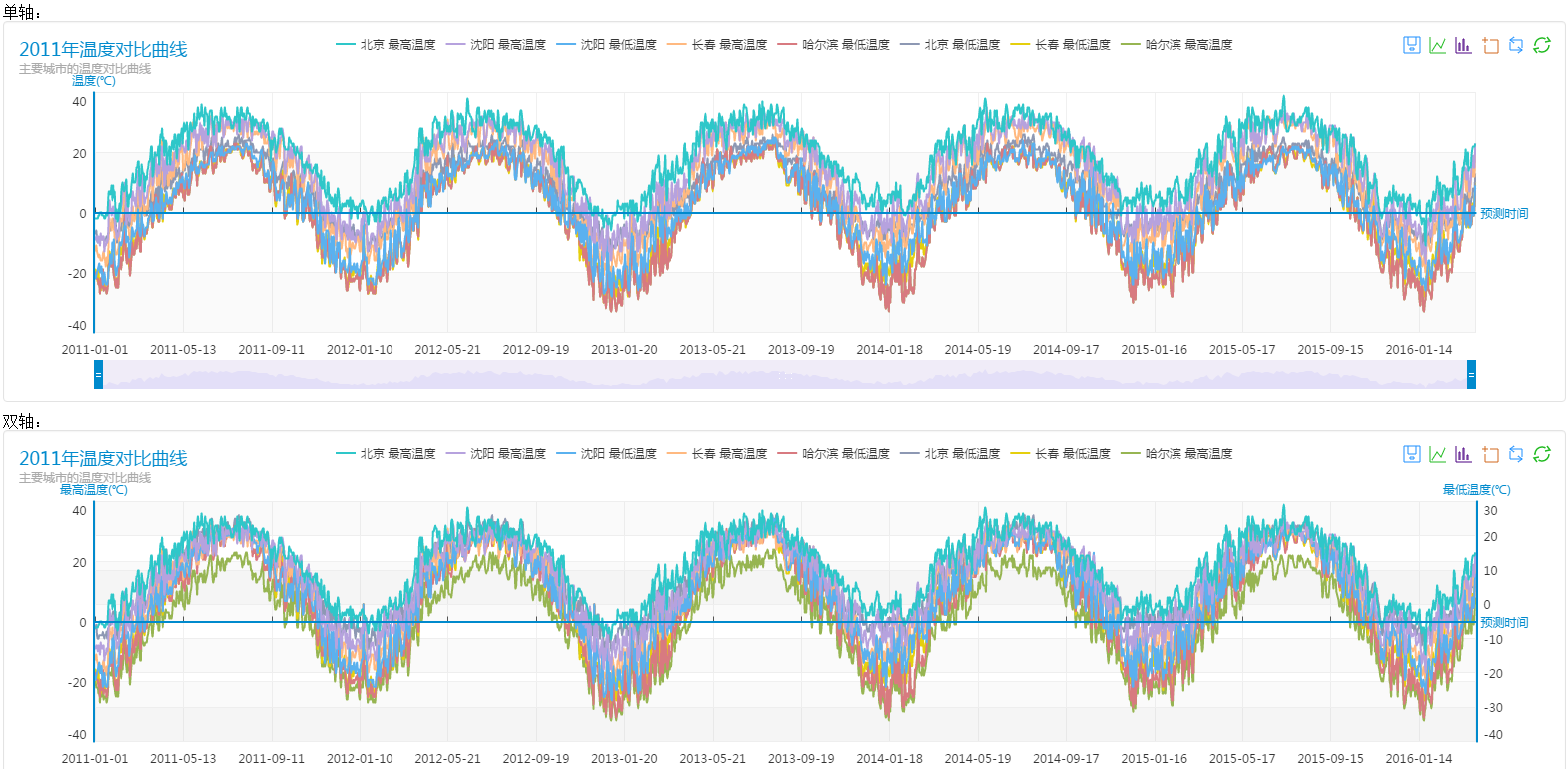
# 2 china\_weather.db数据库

1. 下载地址：<http://git.oschina.net/duhongming/echarts2javatag/attach_files>
2. china\_weather.7z >>>解压>>> 放到项目Src下面就行了.



# 3 基础图表

## 3.1折线图Line



### 3.1.1折线图Line的数据格式

//X轴的数据

List<String> xAxisData;

//Y轴的数据

Map<String,List<Double>> yAxisData;

//Y轴双轴情况下的位置定位

Map<String,Integer> yAxisIndex;

### 3.1.2折线图Line的Tag

唯一需要注意的是div的id和echarts里面的id应该是同一个。

<div id="line\_yAxisIndex" class="main000"></div>

<echarts:line

id="line\_yAxisIndex"

title="2011年温度对比曲线"

subtitle="主要城市的温度对比曲线"

xAxisData="${xAxisData}"

yAxisData="${yAxisData}"

xAxisName="预测时间"

yAxisName="最高温度(℃),最低温度(℃)"

yAxisIndex="${yAxisIndex}"/>

## 3.2柱状图Bar



### 3.2.1柱状图Bar的数据格式

//X轴的数据

private List<String> xAxisData;

//Y轴的数据

private Map<String,List<Double>> yAxisData;

//Y轴双轴情况下的位置定位

private Map<String,Integer> yAxisIndex;

### 3.2.2柱状图Bar的Tag

<div id="line\_yAxisIndex" class="main000"></div>

<echarts:bar

id="line\_yAxisIndex"

title="短期预测数据对比曲线"

subtitle="短期预测数据对比曲线"

xAxisData="${xAxisData}"

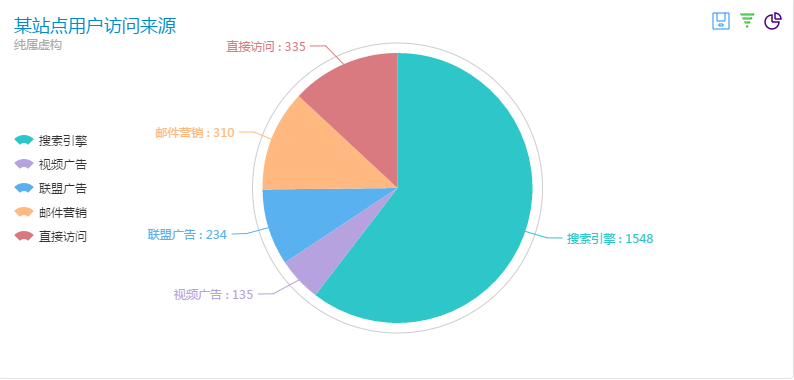
yAxisData="${yAxisData}"

xAxisName="预测时间"

yAxisName="实际电量(MW),实际总辐射(w/㎡)"

yAxisIndex="${yAxisIndex}"/>

## 3.3饼状图Pie



### 3.3.1饼图Pie的数据格式

//key-value数据

private Map<String,Object> orientData;

### 3.3.2饼状图Pie的tag

<div id="pie" class="main000"></div>

<echarts:pie

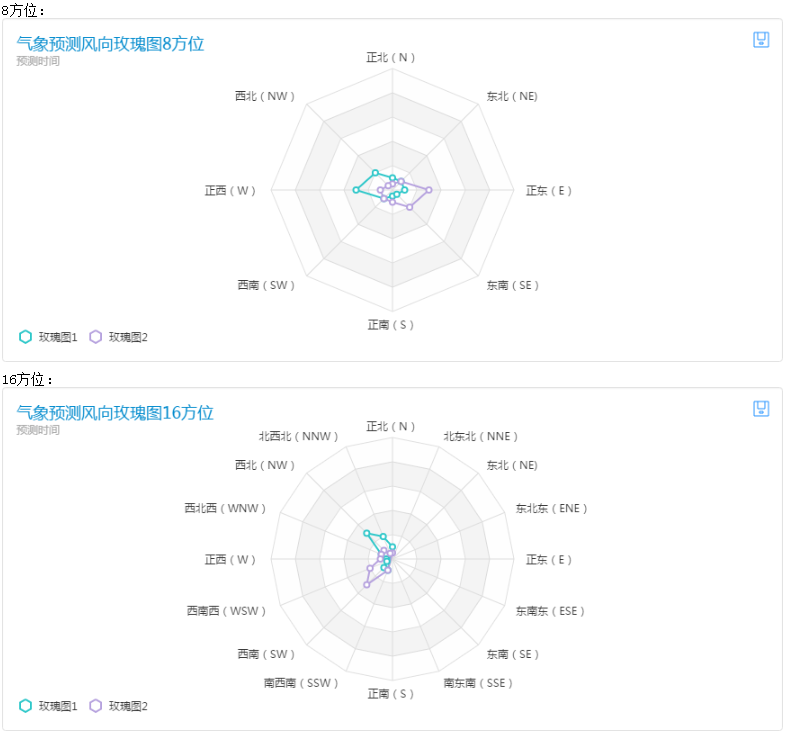
id="pie"

title="某站点用户访问来源"

subtitle="纯属虚构"

orientData="${orientData}"/>

## 3.4玫瑰图Radar



### 3.4.1玫瑰图Radar的数据格式

//二维表结构数据

private List<Map<String,Object>> orientData;

### 3.4.2玫瑰图Radar的Tag

<div id="radar16" class="main000"></div>

<echarts:radar

id="radar16"

title="气象预测风向玫瑰图16方位"

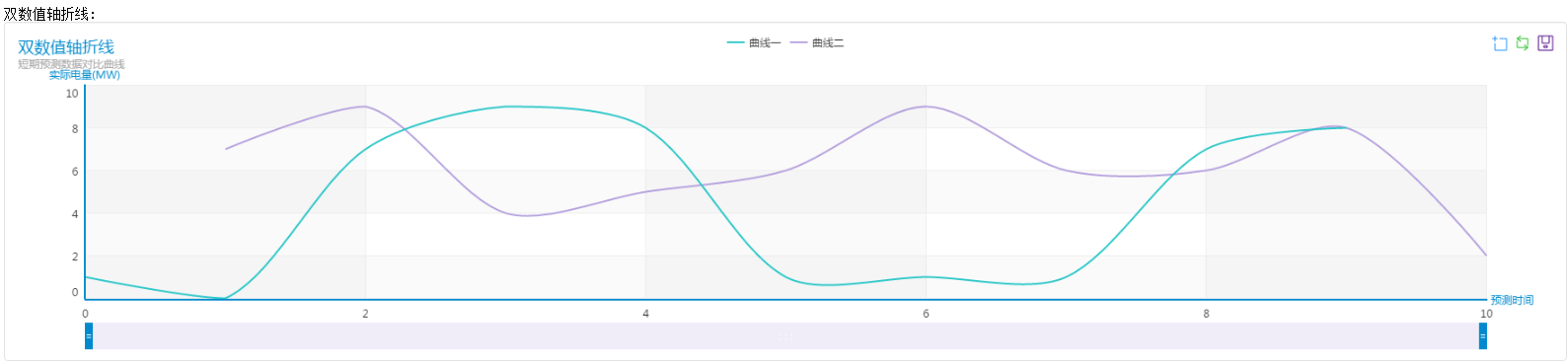
subtitle="预测时间"

orientData="${orientData}"

polarType="16"/>

# 4高级图表

## 4.1双数值轴折线图lineDoubleNum



### 4.1.1双数值轴折线图lineDoubleNum的数据格式

//每种类型数据是Double数组

private Map<String,Double[][]> axisDataArr;

### 4.1.2双数值轴折线图lineDoubleNum的Tag

<div id="line\_doubleNum" class="main000"></div>

<echarts:lineDoubleNum

id="line\_doubleNum"

title="双数值轴折线"

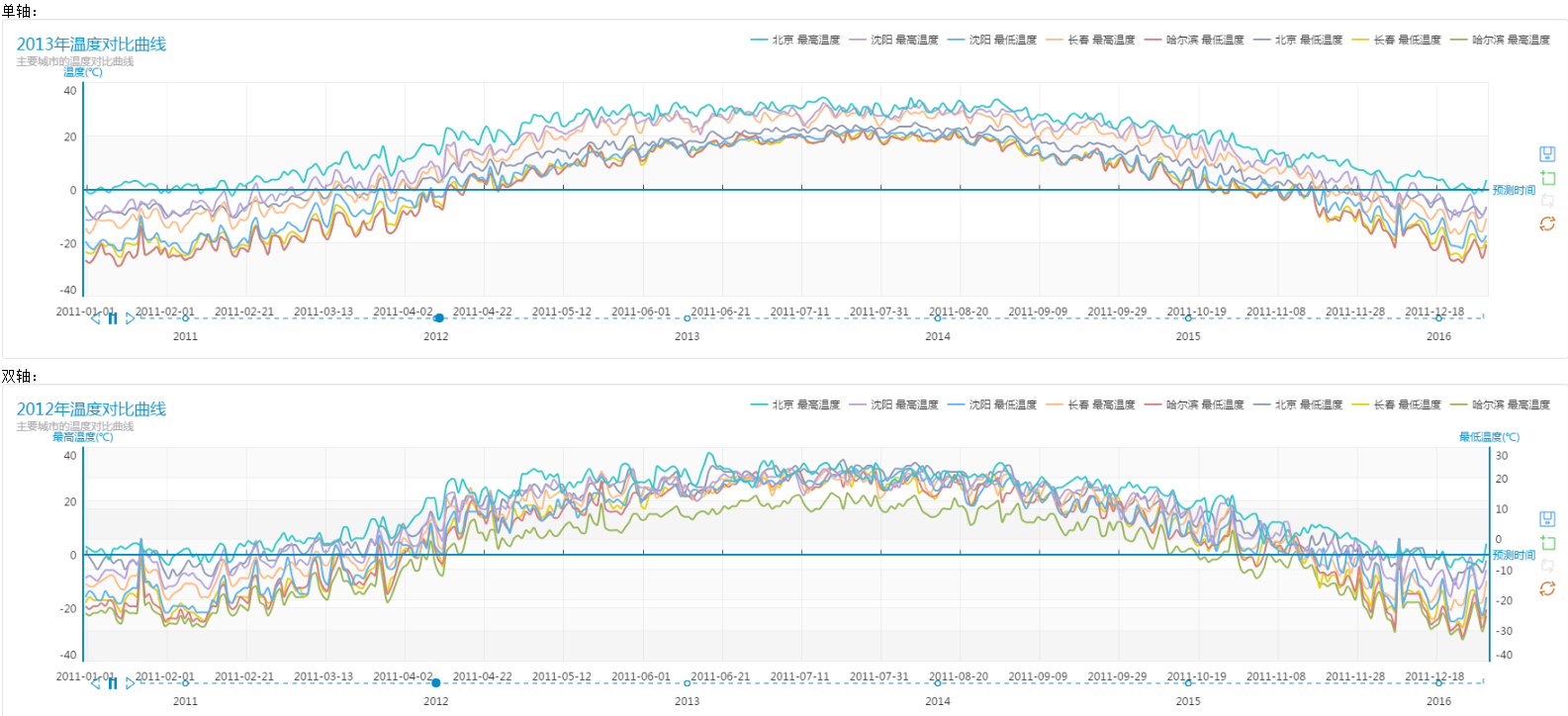
subtitle="短期预测数据对比曲线"

xAxisName="预测时间"

yAxisName="实际电量(MW)"

axisDataArr="${axisDataArr}"/>

## 4.2搭配时间轴折线图lineTimeLine



### 4.2.1搭配时间轴折线图lineTimeLine的数据格式

//X轴的数据

List<String> xAxisData;

//Y轴的数据

Map<String,List<Double>> yAxisData;

//Y轴双轴情况下的位置定位

Map<String,Integer> yAxisIndex;

//TimeLine的Y轴数据

List<Map<String,List<Double>>> timelineAxisData;

//TimeLine的X轴数据

List<String> timelineData;

### 4.2.2搭配时间轴折线图lineTimeLine的Tag

<div id="line\_yAxisIndex" class="main000"></div>

<echarts:lineTimeLine

id="line\_yAxisIndex"

title="2011年温度对比曲线"

subtitle="主要城市的温度对比曲线"

xAxisName="预测时间"

yAxisName="最高温度(℃),最低温度(℃)"

xAxisData="${xAxisData}"

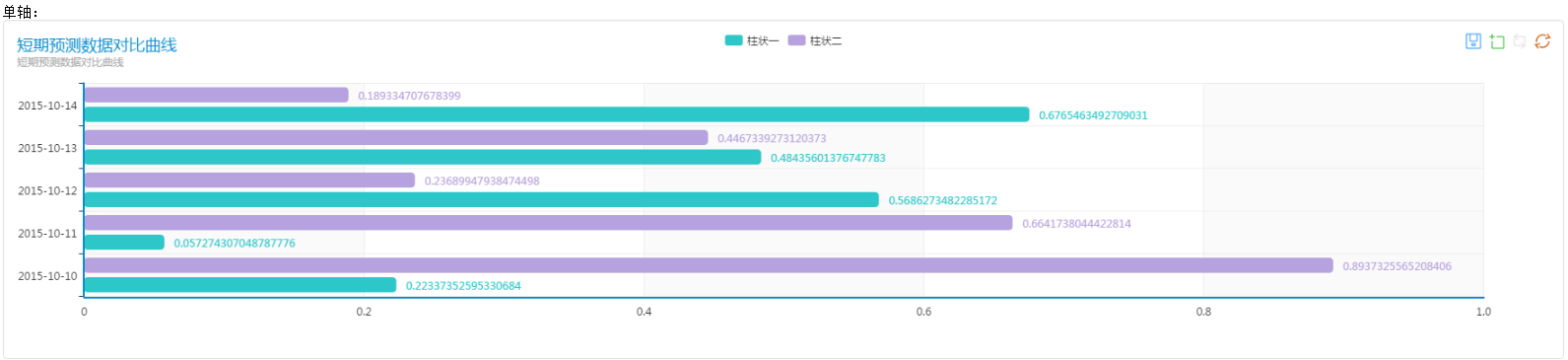
yAxisData="${yAxisData}"

timelineData="${timelineData}"

timelineAxisData="${timelineAxisData}"

yAxisIndex="${yAxisIndex}"/>

## 4.3反转条形图Bar



### 4.3.1反转条形图Bar的数据格式

//X轴的数据

private List<String> xAxisData;

//Y轴的数据

private Map<String,List<Double>> yAxisData;

//Y轴双轴情况下的位置定位

private Map<String,Integer> yAxisIndex;

### 4.3.2反转条形图Bar的Tag

<div id="line\_yAxisIndex" class="main000"></div>

<echarts:bar

id="line\_yAxisIndex"

title="短期预测数据对比曲线"

subtitle="短期预测数据对比曲线"

xAxisData="${xAxisData}"

yAxisData="${yAxisData}"

xAxisName="预测时间"

yAxisName="实际电量(MW),实际总辐射(w/㎡)"

yAxisIndex="${yAxisIndex}"/>

# 5动态图表

了解Spring Socket推送信息：

1. WebSocketConfig中注册一个SocketURL：/webSocket/data
2. 设置定时推送数据：

@Scheduled(cron="0/2 \* \* \* \* ? ") //每2秒执行一次

infoHandlerBF().sendMessageToUsers(new TextMessage(jsonStr));

1. 给所有连接用户推送消息：

public void sendMessageToUsers(TextMessage message) {

for (WebSocketSession user : users) {

try {

if (user.isOpen()) {

user.sendMessage(message);

}

} catch (IOException e) {

e.printStackTrace();

}

}

}

1. 前台接受数据：

var ws = new SockJS("/echarts2javatag /webSocket/data");

ws.onopen = function () {

console.log('Info: connection opened.');

};

ws.onmessage = function (event) {

var json=eval("("+event.data+")");//将数据转成json格式

//业务代码

};

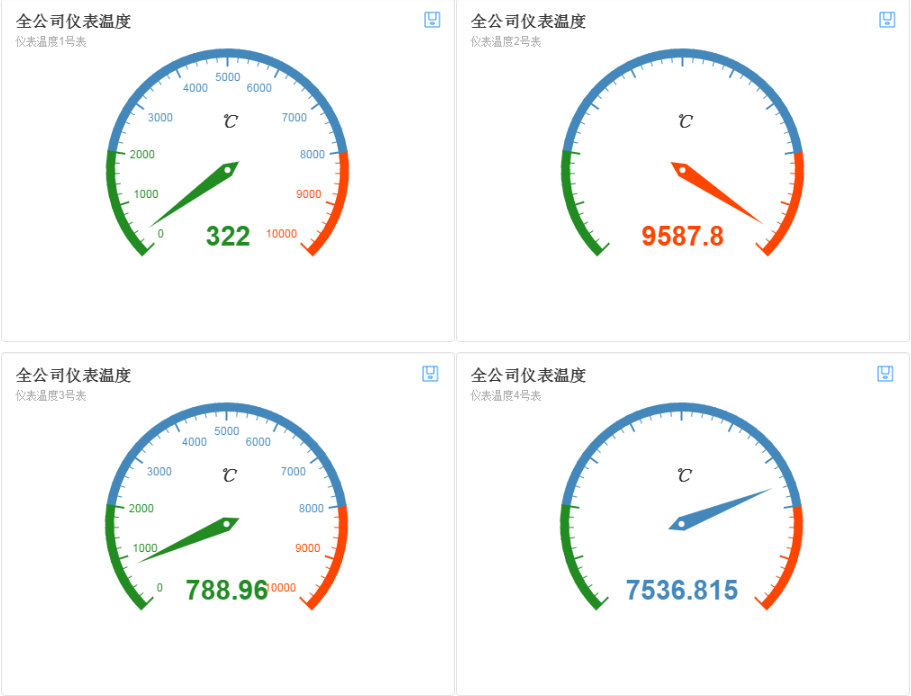
ws.onclose = function (event) {

console.log('Info: connection closed.');

console.log(event);

};

## 5.1动态仪表盘Gauge



### 5.1.1动态仪表盘Gauge的数据格式

public class SocketDataGauge {

private String tag;

private Double value;

}

private List<SocketDataGauge>

Json数据格式：

[{"tag":"temperature1","value":178.03634028465075},{"tag":"temperature2","value":6482.322904395684},{"tag":"temperature3","value":2495.388315562964},{"tag":"temperature4","value":1358.7228569841902}]

### 5.1.2动态仪表盘Gauge的Tag

<ec:gauge id="temperature1"

height="360px"

uri="/echarts2javatag/webSocket/data"

unitName="℃"

title="全公司仪表温度"

subtitle="仪表温度1号表"

measureRange="10000"

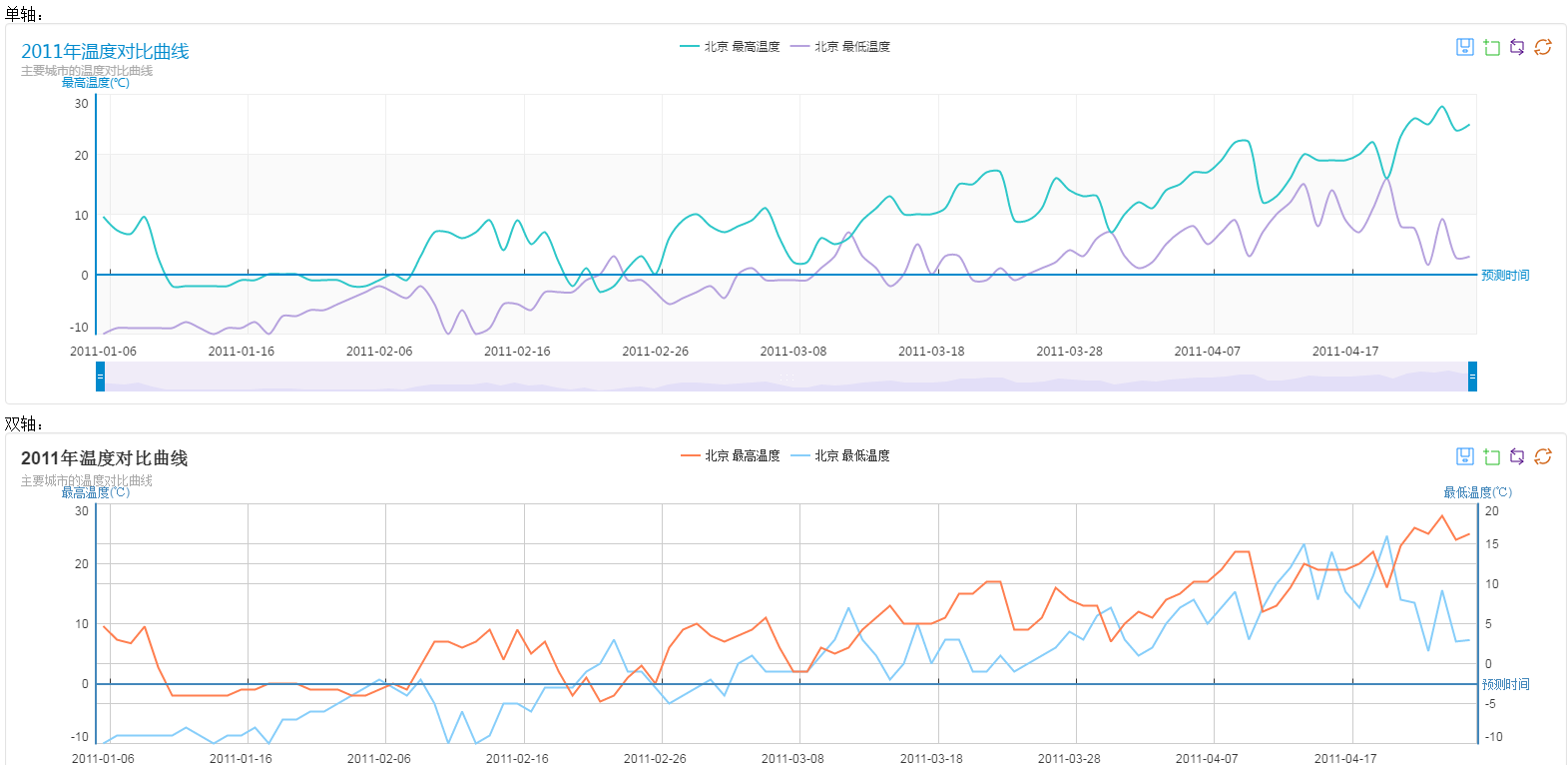
splitNumber="10"

axisLabelShow="true"

toFixed="0">

</ec:gauge>

## 5.2动态折线图Line



### 5.2.1动态折线图Line的数据格式

静态部分：

//X轴的数据

List<String> xAxisData;

//Y轴的数据

Map<String,List<Double>> yAxisData;

//Y轴双轴情况下的位置定位

Map<String,Integer> yAxisIndex;

动态部分：

//X轴数据

private String tag;

//Y轴的1数据

private Double value1;

//Y轴的2数据

private Double value2;

### 5.2.2动态折线图Line的Tag

静态部分和以前一样：

var option =<echarts:dynamicLine

id="line\_yAxisIndex"

title="2011年温度对比曲线"

subtitle="主要城市的温度对比曲线"

xAxisData="${xAxisData}"

yAxisData="${yAxisData}"

xAxisName="预测时间"

yAxisName="最高温度(℃),最低温度(℃)"

yAxisIndex="${yAxisIndex}"/>;

动态部分：

var uri = "/echarts2javatag/webSocket/line/data";

var ws = new SockJS(uri);

ws.onopen = function () {

//console.log('Info: connection opened.');

};

ws.onmessage = function (event) {

var json=eval("("+event.data+")");//将数据转成json格式

// 动态数据接口 addData

myChart.addData([

[

0, // 系列索引

json.value1, // 新增数据

true, // 新增数据是否从队列头部插入

false // 是否增加队列长度，false则自定删除原有数据，队头插入删队尾，队尾插入删队头

],

[

1, // 系列索引

json.value2, // 新增数据

false, // 新增数据是否从队列头部插入

false, // 是否增加队列长度，false则自定删除原有数据，队头插入删队尾，队尾插入删队头

json.tag // 坐标轴标签

]

]);

console.log('Received: ' + json);

};