Learning Everything

Stephen CUI[1]

January 14, 2022

[1]cuixuanStephen@gmail.com

Contents

Chapter 1

Echarts + Vue

1.1 图标相关的基本配置

- `xAxis`: 直角坐标系中的 x 轴, 如果 type 属性的值为 category ,那么需要配置 data 数据, 代表在 x 轴的呈现
- `yAxis`: 直角坐标系中的 y 轴, 如果 type 属性配置为 value , 那么无需配置 data , 此时 y 轴会自动去 series 下找数据进行图表的绘制
- `series`: 系列列表。每个系列通过 type 决定自己的图表类型, data 来设置每个系列的数据

更多配置项可以参考Echarts官方文档。

1.2 基础图标

1.2.1 柱状图

常见的效果：

- 标记（最大值markPoint、最小值markPoint、平均值markLine）
- 显示（数值显示label、柱宽度barWidth、横向柱状图）

1.2.2 通用配置

标题title 用于设置图标的标题，主要包括标题文字、颜色、边框、边框圆角、位置

提示框tooltip 用于配置鼠标或点击图标时的显示框。

1. 触发类型：trigger，主要用两个可选值：item、axis
2. 触发时机：triggerOn
3. 格式化：formatter

工具按钮toolbox Echarts提供的工具栏：内置有导出图片saveAsImage、数据视图dataView、动态类型切换magicType、数据区域缩放dataZoom、重置restore五个工具。

- legend 中的 data 是一个数组
- legend 中的 data 的值需要和 series 数组中某组数据的 name 值一致

图例legend 图例，用于筛选系列，需要配合series使用

1.2.3 折线图

常见效果：

- 标记（最大值markPoint、最小值markPoint、平均值markLine、标注区间markArea）
- 线条控制（平滑smooth、样式lineStyle）
- 填充风格areaStyle、紧挨边缘boundaryGap
- 缩放脱离0值的比例

1.2.4 散点图

数据应该是一个二维数组

常见效果：

- 气泡图效果：散点的大小symbolSize、散点的颜色itemStyle.color
- 涟漪动画效果：`type: effectScatter`, `showEffectOn: "emphasis"`, `rippleEffect`

1.2.5 直接坐标系的常用配置

这叫坐标系的常见图表：柱状图、折线图、散点图。

1. 网格grid: show, borderWidth, borderColor, top, left, right, bottom, height, width;
2. 坐标轴: 坐标轴分为x轴和y轴, 一个 grid 中最多有两种位置的 x 轴和 y 轴.
   - 坐标轴类型 type: value : 数值轴, 自动会从目标数据中读取数据, 如果是category : 类目轴, 该类型必须通过 data 设置类目数据
   - 坐标轴位置: xAxis : [”top”, ”bottom”], yAxis : [”left”, ”right”]
3. 区域缩放dataZoom：用与区域缩放，对数据范围进行过滤，*x*轴和*y*轴都可以拥有，dataZoom接收一个数组，因此可以配置多个区域缩放器。
   - 类型type：slider：滑块，inside，默认，依靠鼠标滚轮或者双指缩放；
   - 指标产生作用的轴：xAxisIndex:设置缩放组件控制的是哪个x轴，一般写0即可，yAxisIndex同理；
   - 指明初始状态的缩放情况：start:数据窗口范围的只是百分比，end:数据窗口范围的结束百分比；

1.2.6 饼图

饼图的数据是由 name 和 value 组成的字典所形成的数组，饼图无须配置 xAxis 和 yAxis。

常见效果：

- 显示数值：`label.formatter`
- 圆环：设置两个半径 `radius: ["50%", "70%"]`
- 南丁格尔图：南丁格尔图指的是每一个扇形的半径随着数据的大小而不同, 数值占比越大, 扇形的半径也就越大 `roseType: "radius"`
- 选中效果:
  - `selectedMode: ["multiple", "single"]`
  - `selectedOffset: 30`

1.2.7 地图

1. ECharts 最基本的代码结构
2. 准备中国的矢量 json 文件, 放到 json/map/ 目录之下
3. 使用 Ajax 获取 china.json
4. 在Ajax的回调函数中, 往 echarts 全局对象注册地图的 json 数据 echarts.registerMap('chinaMap', chinaJson)
5. 获取完数据之后, 需要配置 geo 节点, 再次的 setOption

常见配置：缩放拖动: roam，名称显示: label，初始缩放比例: zoom，地图中心点: center

常见效果：

- 显示某个区域：
  1. 准备安徽省的矢量地图数据
  2. 加载安徽省地图的矢量数据
  3. 在Ajax的回调函数中注册地图矢量数据
  4. 配置 geo 的 type:'map' , map:'anhui'
  5. 通过 zoom 调整缩放比例
  6. 通过 center 调整中心点
- 不同城市显示不同颜色
  1. 显示基本的中国地图
  2. 准备好城市空气质量的数据, 并且将数据设置给 series
  3. 将 series 下的数据和 geo 关联起来
  4. 结合 visualMap 配合使用：visualMap 是视觉映射组件, 和之前区域缩放 dataZoom 很类似, 可以做数据的过滤. 只不过 dataZoom 主要使用在直角坐标系的图表, 而 visualMap 主要使用在地图或者饼图中
- 地图和散点图结合

地图和散点图结合

1. 给 series 这个数组下增加新的对象
2. 准备好散点数据,设置给新对象的 data
3. 配置新对象的 type：`type:effectScatter`

4. 让散点图使用地图坐标系统`coordinateSystem: "geo"`
5. 让涟漪的效果更加明显

1.2.8 雷达图

实现步骤

1. ECharts 最基本的代码结构: 引入JS文件、DOM容器、初始化对象、设置Option
2. 定义各个维度的最大值：
3. 准备具体产品的数据：
4.

常用配置

- 显示数值：`label.show: true`
- 区域面积：`reaStyle: {}`
- 绘制类型：`shape: "circle"`

1.2.9 仪表盘

仪表盘主要使用在进度把控以及数据范围的监控。

实现步骤

1. ECharts 最基本的代码结构: 引入JS文件、DOM容器、初始化对象、设置Option
2. 准备数据, 设置给 series 下的 data
3. 在 series 下设置 type:gauge

常用效果

- 显示数值：`label.show: true`
- 区域面积：`reaStyle: {}`
- 绘制类型：`shape: "circle"`

1.3 ECharts高级

1.3.1 显示相关

主题

ECharts 中默认内置了两套主题: light、dark，在初始化对象方法 init 中可以指明。

```
var myChart = echarts.init(document.querySelector("div"), "light");
var myChart = echarts.init(document.querySelector("div"), "shine");
```

调色板

它是一组颜色，图形、系列会自动从其中选择颜色, 不断的循环从头取到尾, 再从头取到尾, 如此往复。有主题调色盘、全局调色盘(将覆盖主题调色盘)、局部调色盘(将覆盖全局调色盘)

颜色渐变：线性渐变itemStyle、径向渐变

```
color: {
    type: "radial", 径向渐变// 径向渐变
    都为相对值// x, y, r都为相对值
    x: 0.5,
    y: 0.5,
    r: 1,
    colorStops: [
        { offset: 0, color: "red" },
        { offset: 1, color: "green" },
    ],
},
```

样式

直接样式(itemStyle, textStyle, lineStyle, areaStyle, label)

高亮样式(emphasis中包裹itemStyle, textStyle, lineStyle, areaStyle, label)：图表中, 其实有很多元素都是有两种状态的, 一种是默认状态, 另外一种就是鼠标滑过或者点击形成的高亮状态. 而高亮样式是针对于元素的高亮状态设定的样式。

自适应

当浏览器的窗口大小发生变化的同时，需要图表的大小也随之适配变化。

1. 监听窗口大小变化事件
2. 在事件处理函数中调用 ECharts 实例对象的 resize 即可

```
window.onresize = function () {
myChart.resize();
};
```

1.3.2 动画的使用

加载动画

ECharts 已经内置好了加载数据的动画, 我们只需要在合适的时机显示或者隐藏即可。

一般, 我们会在获取图表数据之前显示加载动画，主要是因为后台获取数据可能会需要一定的时间。

- 显示加载动画`myChart.showLoading()`
- 隐藏加载动画`myChart.hideLoading()`

增量动画

setOption可以设置多次，新的option和旧的option的关系并不是相互壶盖的区别，是相互整合的关系，在设置新的option的时候，只需要考虑到变化的部分就可以了。

所有数据的更新都是通过setOption实现的，不需要考虑数据的具体变化位置，ECharts会找到两组数据之间的差异，然后通过合适的动画去表现数据的变化。

动画的配置

- 开启动画，`animation: true`
- 动画时长（毫秒单位）：`animationDuration: 7000`，也可以接收一个回调参数
- 缓动动画：`animationEasing: 'linear'`
- 动画阈值：`animationThreshold: 10`，单种形式的元素数量大于这个阈值时会关闭动画

1.3.3 交互API

全局echarts对象

全局echarts对象是引入echarts.js文件之后就可以直接使用的。

- init：初始化echarts实例对象，可以用来使用主题
- registerTheme：
- registerMap
- connect: 一个页面中可以有多个独立的图表，每个图表对应一个echarts实例对象，connect可以实现多图关联，传入联动目标为echarts实例对象，支持数组，实现保存图片的自动拼接、刷新按钮、重置按钮、提示框联动、图里选择、数据范围修改等。

echartsInstance对象

echartsInstance对象是通过`echarts.init`方法调用之后得到的。

Chapter 2

后台项目

2.1 Koa2快速上手

1. 检查Node的环境（`cmd node -v`)
2. 安装Koa（`cmd npm init -y`可以用来维护一些第三方包的信息、`cmd npm install koa`）
3. 创建并编写app.js文件
   (a) 创建Koa对象
   (b) 编写响应函数（中间件）
   (c) 监听端口
4. 启动服务器（`cmd node app.js`）

2.1.1 中间件的特点

Koa对象通过use方法加入一个中间件（函数），中间件的执行顺序符合洋葱模型。内层中间件能否执行取决于外层中间件的next函数是否被调用。调用next()函数得到的是Promise对象。

```
app.use(async (ctx, next)=>{
    const msg = await next();
    // codes you want to do
})
```

2.2 后台项目

后台项目的目标包括计算服务器处理请求的总耗时、在响应头上加上相应内容的mime类型、根据URL读取指定目录下的文件内容。

Warnings

本项目没有配置数据库，因此需要读取文件来作为数据的来源。

Part I

从Vue2到Vue3