

Echarts5.x

1.课程介绍与目标

有句话说的好“一图胜千言”，在我们开发的领域就是说，在对于复杂难懂且体量庞大的数据展示上面而言，图表的信息量要大得多，这也是我们为什么要谈数据可视化。

2.数据可视化介绍

数据可视化这一概念自1987年正式提出，经过30余年的发展，逐渐形成3个分支：科学计算可视化(scientific visualization)、信息可视化(information visualization)和可视分析(visual analytics)。近些年来，这3个子领域出现了逐渐融合的趋势。我们统称为“数据可视化”。

什么是数据可视化？

顾名思义，数据可视化就是将数据转换成图或表等，以一种更直观的方式展现和呈现数据。通过“可视化”的方式，我们看不懂的数据通过图形化的手段进行有效地表达，准确高效、简洁全面地传递某种信息，甚至我们帮助发现某种规律和特征，挖掘数据背后的价值。

同时关于数据可视化的定义有很多，像百度百科的定义是：数据可视化，是关于数据视觉表现形式的科学技术研究。其中，这种数据的视觉表现形式被定义为，一种以某种概要形式抽提出来的信息，包括相应信息单位的各种属性和变量。这种定义可能显得比较晦涩难懂。在大数据分析工具和软件中提到的数据可视化，就是利用运用计算机图形学、图像、人机交互等技术，将采集或模拟的数据映射为可识别的图形、图像。

数据可视化的展现形式

数据可视化有众多展现方式，不同的数据类型要选择适合的展现方法。在数据可视化中除了常用的柱状图、线状图、条形图、面积图、饼图、点图、仪表盘、走势图外，还有和弦图、圈饼图、金字塔、漏斗图、K线图、关系图、网络图、玫瑰图、帕累托图、数学公式图、预测曲线图、正态分布图、迷你图、行政地图、GIS地图等各种展现形式。都可以为我们提供丰富的图表选择，让我们在实际使用过程中有更好的展现方式。



我们可以通过类柱状图

比较类图表显示值与值之间的不同和相似之处。使用图形的长度、宽度、位置、面积、角度和颜色来比较数值的大小，通常用于展示不同分类间的数值对比，不同时间点的数据对比。

柱形图有别于直方图，柱状图无法显示数据在一个区间内的连续变化趋势。柱状图描述的是分类数据，回答的是每一个分类中"有多少？"这个问题。需要注意的是，当柱状图显示的分类很多时会导致分类名重叠等显示问题。

同时可以通过占比类图表显示同一维度上的占比关系。饼图广泛应用在各个领域，用于表示不同分类的占比情况，通过弧度大小来对比各个分类。

饼图通过将一个圆饼按照分类的占比划分成多个区块，整个圆饼代表数据的总量，每个区块（圆弧）表示该分类占总体的比例大小，所有区块（圆弧）的加和等于 100%。

也可以趋势类折线图

趋势类图表显示数据的变化趋势。使用图形的位置表现数据在连续区域上的分布，通常展示数据在连续区域上的大小变化的规律。

折线图用于显示数据在一个连续的时间间隔或者时间跨度上的变化，它的特点是反映事物随时间或有序类别而变化的趋势。

当然，大数据可视化的图表远远不止以上几种，最关键的是如何利用好这些工具及图表，归纳起来，一名数据可视化工程师需要具备三个方面的能力，数据分析能力、交互视觉能力、研发能力。

数据可视化有什么用？

数据可视化的意义是帮助人更好的分析数据，信息的质量很大程度上依赖于其表达方式。对数字罗列所组成的数据中所包含的意义进行分析，使分析结果可视化。其实数据可视化的本质就是视觉对话。数据可视化将技术与艺术完美结合，借助图形化的手段，清晰有效地传达与沟通信息。一方面，数据赋予可视化以价值；另一方面，可视化增加数据的灵性，两者相辅相成，帮助企业从信息中提取知识、从知识中收获价值。精心设计的图形不仅可以提供信息，还可以通过强大的呈现方式增强信息的影响力，吸引人们的注意力并使其保持兴趣，这是表格或电子表格无法做到的。

3.Echarts--商业级数据图表介绍

1.什么是Echarts



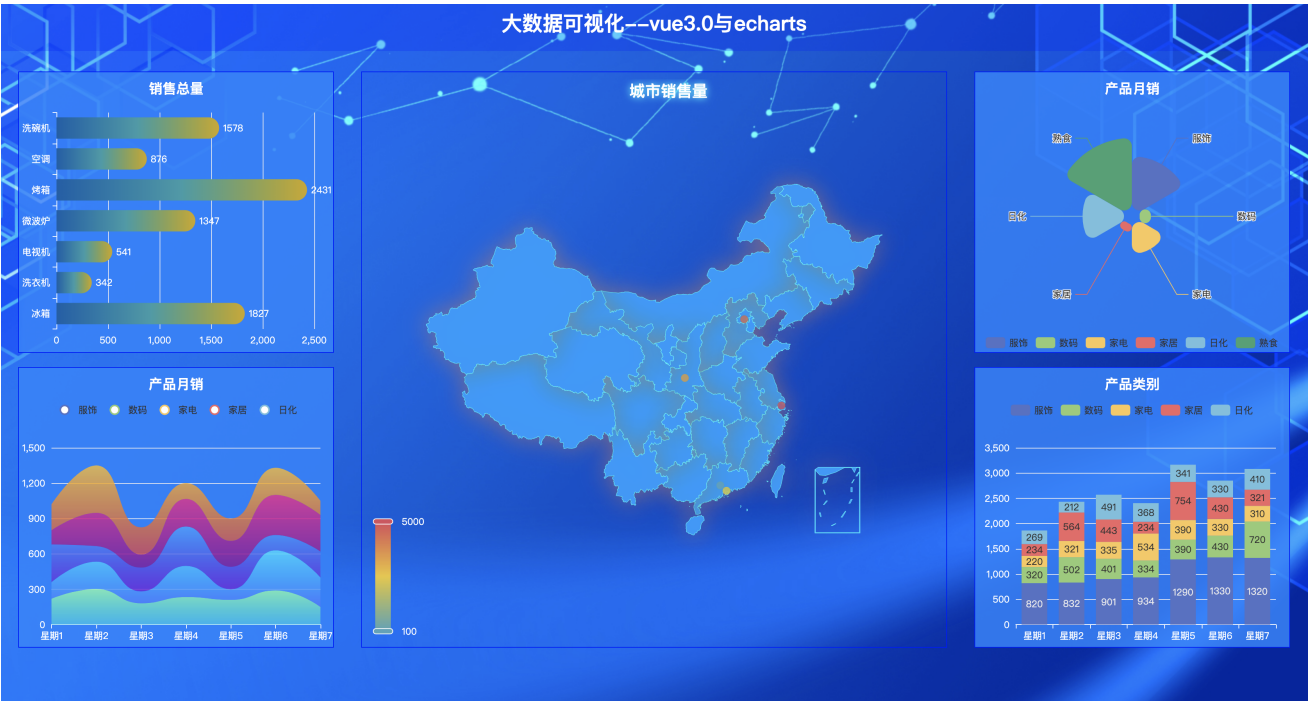
Echarts--商业级数据图表，它是一个纯JavaScript的图标库，可以流畅的运行在PC和移动设备上，兼容当前绝大部分浏览器（IE6/7/8/9/10/11，chrome，firefox，Safari等），底层依赖轻量级的Canvas类库ZRender，提供直观，生动，可交互，可高度个性化定制的数据可视化图表。创新的拖拽重计算、数据视图、值域漫游等特性大大增强了用户体验，赋予了用户对数据进行挖掘、整合的能力。

2. Echarts特点

- 1、丰富的可视化类型:** 提供了常规的折线图、柱状图、散点图、饼图、K线图，用于统计的盒形图，用于地理数据可视化的地图、热力图、线图，用于关系数据可视化的关系图、treemap、旭日图，多维数据可视化的平行坐标，还有用于 BI 的漏斗图，仪表盘，并且支持图与图之间的混搭。
- 2、多种数据格式无需转换直接使用:** 内置的 dataset 属性（4.0+）支持直接传入包括二维表，key-value 等多种格式的数据源，此外还支持输入 TypedArray 格式的数据。
- 3、千万数据的前端展现:** 通过增量渲染技术（4.0+），配合各种细致的优化，ECharts 能够展现千万级的数据量。
- 4、移动端优化:** 针对移动端交互做了细致的优化，例如移动端小屏上适于用手指在坐标系中进行缩放、平移。PC 端也可以用鼠标在图中进行缩放（用鼠标滚轮）、平移等。
- 5、多渲染方案，跨平台使用:** 支持以 Canvas、SVG（4.0+）、VML 的形式渲染图表。
- 6、深度的交互式数据探索:** 提供了 图例、视觉映射、数据区域缩放、tooltip、数据刷选等开箱即用的交互组件，可以对数据进行多维度数据筛取、视图缩放、展示细节等交互操作。
- 7、多维数据的支持以及丰富的视觉编码手段:** 对于传统的散点图等，传入的数据也可以是多个维度的。
- 8、动态数据:** 数据的改变驱动图表展现的改变。

9、**绚丽的特效**: 针对线数据，点数据等地理数据的可视化提供了吸引眼球的特效。

4. 项目演示



1.项目功能演示

2.项目启动与依赖安装

此过程会有引导 引导观众去获取资料

5.HelloWord 初体验

1.echarts获取

前期准备

电脑上面安装node

node下载地址：<http://nodejs.cn/>

淘宝镜像（选按）

淘宝 NPM 镜像站自 2014 年 正式对外服务。于npm命令在国内下载速度很慢。所以淘宝每隔10分钟就会把npm服务器的内容拉取一次放在国内服务器 这样一来我们在下载依赖的时候 速度会快很多

```
1 npm install -g cnpm --registry=https://registry.npm.taobao.org
2
```

安装

1.初始化 npm init -y

2.安装echarts依赖

npm install --save echarts

2.HelloWord

```
1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="main"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        // echarts仅有一个方法init，执行init时传入一个具备大小
12        // （如果没有指定容器的大小将会按照0大小来处理即无法看到图表）的dom节点后即可实例化出图表对象，图表库
        实现为多实例的，
13        // 同一页面可多次init出多个图表。
14        var myChart = echarts.init(document.getElementById("main")); // 绘制图表 // setOption方法
        设置图表实例的配置项 以及数据 所有参数和数据的修改都可以通过 setOption 完成，ECharts 会合并新的参数和
        数据，然后刷新图表。
15        myChart.setOption({
16          title: {
17            //echarts标题
18            text: "ECharts 入门示例",
19          }, //tooltip:提示框组件，用于配置鼠标滑过或点击图表时的显示框。
20          tooltip: {},
21          // 不过我们在Echarts过程中经常会遇到如下问题：图例经常不知道如何调节到我们想要的位置。
22          legend: {}, //横坐标 xAxis配置 直角坐标系x轴
23          xAxis: {
24            data: ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"],
25          },
26          yAxis: {}, //系列 (series)
27          // 系列 (series) 是很常见的名词。在 echarts 里，系列 (series) 是指：一组数值以及他们映射成的
        图。“系列”这个词原本可能来源于“一系列的数据”，而在 echarts 中取其扩展的概念，不仅表示数据，也表示数据映射
        成为的图
28          // charts 里系列类型 (series.type) 就是图表类型。系列类型 (series.type) 至少有：line (折线
        图)、bar (柱状图)、pie (饼图)、scatter (散点图)、graph (关系图)、tree (树图)
29          series: [
30            {
31              name: "销量",
32              type: "bar",
33              data: [5, 20, 36, 10, 10, 20],
34            },
35          ],
36        });
37      },
38    };
39  </script>
40  <style>
41    #main{
42      width: 500px;
43      height: 500px;
44    }
45  </style>
```

6. 配置项--title配置

title 标题组件，包含主标题和副标题。

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="main"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        var myChart = echarts.init(document.getElementById("main"));
12        myChart.setOption({
13          title: {
14            show: true, //显示策略，默认值true,可选为：true（显示） | false（隐藏）
15            text: "1主标题", //主标题文本，'\n'指定换行
16            // link: 'http://www.baidu.com', //主标题文本超链接，默认值true
17            // target: "self", //指定窗口打开主标题超链接，支持'self' | 'blank'，不指定等同
            为'blank'（新窗口）
18            subtext: '副标题', //副标题文本，'\n'指定换行
19            // sublink: '', //副标题文本超链接
20            // subtarget: null, //指定窗口打开副标题超链接，支持'self' | 'blank'，不指定等同
            为'blank'（新窗口）
21            // x: 'center', //水平安放位置，默认为'left'，可选为：'center' | 'left' | 'right' |
            {number}（x坐标，单位px）
22            // y: 'bottom', //垂直安放位置，默认为top，可选为：'top' | 'bottom' | 'center' |
            {number}（y坐标，单位px）
23            // backgroundColor: 'red', //标题背景颜色，默认'rgba(0,0,0,0)'透明
24            // borderWidth: 5, //标题边框线宽，单位px，默认为0（无边框）
25            // borderColor: '#ccffee', //标题边框颜色，默认'#ccc'
26            // padding: 5, //标题内边距，单位px，默认各方向内边距为5，接受数组分别设定上右下左边距
27            // itemGap: 10, //主副标题纵向间隔，单位px，默认为10
28            // textStyle: { //主标题文本样式{"fontSize": 18,"fontWeight": "bolder","color":
            "#333"}
29              // fontFamily: 'Arial, Verdana, sans...',
30              // fontSize: 12,
31              // fontStyle: 'normal',
32              // fontWeight: 'normal', // },
33              // subtextStyle: { //副标题文本样式{"color": "#aaa"}
34              // fontFamily: 'Arial, Verdana, sans...',
35              // fontSize: 12,
36              // fontStyle: 'normal',
37              // fontWeight: 'normal',
38              // },
39              // subtextStyle: {
40              //   color: "#a1b2c3", // 副标题文字的颜色。
41              //   fontStyle: "normal", // 副标题文字字体的风格。 'normal' 'italic' 'oblique'

```



```

42      //   fontWeight: "bold", // 副标题文字字体的粗细。 'normal' 'bold' 'bolder'
    'lighter' 500|600。
43      //   fontSize: 18, // 字体大小
44      //   lineHeight: "130", // 行高
45      //   textBorderColor: "red", // 文字本身的描边颜色。
46      //   textBorderWidth: 5, // 文字本身的描边宽度。
47      //   textShadowColor: "transparent", // 文字本身的阴影颜色。
48      //   textShadowBlur: 0, // 文字本身的阴影长度。
49      //   textShadowOffsetX: 0, // 文字本身的阴影 X 偏移。
50      //   textShadowOffsetY: 0, // 文字本身的阴影 Y 偏移。
51      // },
52    },
53    tooltip: {},
54
55    legend: {},
56    xAxis: {
57      data: ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"],
58    },
59    yAxis: {},
60    series: [
61      {
62        name: "销量",
63        type: "bar",
64        data: [5, 20, 36, 10, 10, 20],
65      },
66    ],
67  });
68  },
69  };
70 </script>
71 <style>
72 #main {
73   width: 500px;
74   height: 500px;
75 }
76 </style>

```

7. 配置项--tooltip

提示框组件，用于配置鼠标滑过或点击图表时的显示框

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="main"></div>
5    </div>
6  </template>
7  <script>
8  import * as echarts from "echarts";
9  export default {
10    mounted() {

```

```

11     var myChart = echarts.init(document.getElementById("main"));
12     myChart.setOption({
13         title: {
14             text: '主标题'
15         },
16         tooltip: { //提示框组件，用于配置鼠标滑过或点击图表时的显示框。
17             show: true, // 是否显示
18             trigger: 'axis', // 触发类型 'item' 图形触发：散点图，饼图等无类目轴的图表中使用； 'axis' 坐
                标轴触发； 'none'：什么都不触发。
19             axisPointer: { // 坐标轴指示器配置项。
20                 type: 'cross', // 'line' 直线指示器 'shadow' 阴影指示器 'none' 无指示器 'cross'
                十字准星指示器。
21             },
22             // showContent: true, //是否显示提示框浮层，默认显示。
23             // triggerOn: 'mouseover', // 触发时机 'click' 鼠标点击时触发。
24             backgroundColor: 'rgba(50,50,50,0.7)', // 提示框浮层的背景颜色。
25             borderColor: '#333', // 提示框浮层的边框颜色。
26             borderWidth: 0, // 提示框浮层的边框宽。
27             padding: 5, // 提示框浮层内边距，
28             textStyle: { // 提示框浮层的文本样式。
29                 color: '#fff',
30                 fontStyle: 'normal',
31                 fontWeight: 'normal',
32                 fontFamily: 'sans-serif',
33                 fontSize: 14,
34             },
35             // 提示框浮层内容格式器，支持字符串模板和回调函数两种形式。
36             // 模板变量有 {a}, {b}, {c}，分别表示系列名，数据名，数据值等
37             // formatter: '{a}--{b} 的成绩是 {c}'
38             formatter: function(arg) {
39                 return arg[0].name + '的分数是:' + arg[0].data
40             }
41         },
42
43         legend: {},
44         xAxis: {
45             data: ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"],
46         },
47         yAxis: {},
48         series: [
49             {
50                 name: "销量",
51                 type: "bar",
52                 data: [5, 20, 36, 10, 10, 20],
53             },
54         ],
55     });
56 },
57 };
58 </script>
59 <style>
60 #main {
61     width: 500px;

```



```

62     height: 500px;
63   }
64 </style>

```

8.配置项--legend

图例组件展现了不同系列的标记，颜色和名字。可以通过点击图例控制哪些系列不显示。

```

1   <template>
2     <div class="about">
3       <h1>This is an about page</h1>
4       <div id="main"></div>
5     </div>
6   </template>
7   <script>
8     import * as echarts from "echarts";
9     export default {
10      mounted() {
11        var myChart = echarts.init(document.getElementById("main"));
12        myChart.setOption({
13          title: {
14            text: '主标题'
15          },
16          tooltip: {
17
18          },
19
20          legend: {
21            show: true, //是否显示
22            icon: "circle", //图例样式
23            // top: "55%", // 组件离容器的距离
24            // bottom:"20%", // 组件离容器的距离
25            // left 的值可以是像 20 这样的具体像素值，可以是像 '20%' 这样相对于容器高宽的百分比，也可以是
26            'left', 'center', 'right'
27            // right: "5%",
28            // left:"10%" // // 组件离容器的距离
29            // padding: 5, // 图例内边距
30            // itemWidth: 6, // 图例标记的图形宽度。
31            // itemGap: 20, // 图例每项之间的间隔。
32            // itemHeight: 14, // 图例标记的图形高度。
33            // selectedMode: false, // 图例选择的模式，控制是否可以通过点击图例改变系列的显示状态。默认开
34            启图例选择，可以设成 false 关闭。
35            inactiveColor: "#fffddd", // 图例关闭时的颜色。
36            textStyle: { //图例的公用文本样式。
37              // color: "#aabbcc", // 文字的颜色。
38              // fontStyle: "normal", // 文字字体的风格。'italic'
39              // fontWeight: "normal", // 文字字体的粗细。 'normal' 'bold' 'bolder' 'lighter' 100
40              | 200 | 300 | 400...
41              // fontFamily: "sans-serif", // 文字的字体系列。
42              // fontSize: 12, // 文字的字体大小。
43              // lineHeight: 20, // 行高。

```

```

41      // backgroundColor: "transparent", // 文字块背景色。
42      // borderColor: "transparent", // 文字块边框颜色。
43      // borderWidth: 0, // 文字块边框宽度。
44      // borderRadius: 0, // 文字块的圆角。
45      // padding: 0, // 文字块的内边距
46      // shadowColor: "transparent", // 文字块的背景阴影颜色
47      // shadowBlur: 0, // 文字块的背景阴影长度。
48      // shadowOffsetX: 0, // 文字块的背景阴影 X 偏移。
49      // shadowOffsetY: 0, // 文字块的背景阴影 Y 偏移。
50      // // width: 50, // 文字块的宽度。 默认
51      // // height: 40, // 文字块的高度 默认
52      // textBorderColor: "transparent", // 文字本身的描边颜色。
53      // textBorderWidth: 0, // 文字本身的描边宽度。
54      // textShadowColor: "transparent", // 文字本身的阴影颜色。
55      // textShadowBlur: 0, // 文字本身的阴影长度。
56      // textShadowOffsetX: 0, // 文字本身的阴影 X 偏移。
57      // textShadowOffsetY: 0, // 文字本身的阴影 Y 偏移。
58    }
59  },
60  xAxis: {
61    data: ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"],
62  },
63  yAxis: {},
64  series: [
65    {
66      name: "销量",
67      type: "bar",
68      data: [5, 20, 36, 10, 10, 20],
69    },
70  ],
71  });
72  },
73  };
74  </script>
75  <style>
76  #main {
77    width: 500px;
78    height: 500px;
79  }
80  </style>

```

9.柱状图基本设置

柱状图：一种图表类型,因为构成是由一根一根类似柱子的数据条组合而成的坐标平面,所以命名为柱状图。主要是用来反应对比数据之间的关系,也可以用来反应数据的变化趋势等等。

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <!-- 2.echarts根结点根容器如果我们没有去指定当前容器的大小 echarts会按照0来进行处理 -->
5      <div id="myecharts" ref="demoh"></div>

```

```

6     </div>
7 </template>
8 <script>
9 import * as echarts from "echarts";
10 export default {
11   mounted() {
12     let myChart = echarts.init(this.$refs.demoh);
13     let xData = ["美食", "数码", "日化", "蔬菜", "熟食"]; //x轴数据
14     let yData = [88, 75, 20, 210, 35]; //y轴数据
15     let option = {
16       xAxis: {
17         //配置x轴坐标参数
18         data: xData,
19         type: "category", //坐标轴类型。'value' 数值轴，适用于连续数据。
20         // 'category' 类目轴，适用于离散的类目数据。为该类型时类目数据可自动从 series.data 或
dataset.source 中取，或者可通过 xAxis.data 设置类目数据。
21         // 'time' 时间轴，适用于连续的时序数据，与数值轴相比时间轴带有时间的格式化，在刻度计算上也有所不
同，例如会根据跨度的范围来决定使用月，星期，日还是小时范围的刻度。
22         // 'log' 对数轴。适用于对数数据。
23       },
24       yAxis: {
25         //配置y轴坐标参数
26         type: "value", //同x轴的参数
27       },
28       series: [
29         //系列 配置图表的类型
30         {
31           type: "bar",
32           name: "销量", //系列名称，用于提示框组件的显示，
33           data: yData,
34         },
35       ],
36     };
37     // 绘制图表 setOption 配置图表的配置项
38     myChart.setOption(option);
39   },
40 };
41 </script>
42 <style scoped>
43 #myecharts {
44   width: 600px;
45   height: 600px;
46   border: 2px solid red;
47 }
48 </style>

```

10.柱状图效果实现

当基本的柱状图设置完之后我们来看一下 柱状图的更多设置 柱状图标记效果

最大值最小值平均值 通过markPoint进行设置

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="myecharts" ref="demoh"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        let myChart = echarts.init(this.$refs.demoh);
12        let xData = ["美食", "数码", "日化", "蔬菜", "熟食"]; //x轴数据
13        let yData = [88, 75, 20, 210, 35]; //y轴数据
14        let option = {
15          xAxis: {
16            //配置x轴坐标参数
17            data: xData,
18            type: "category", //坐标轴类型。'value' 数值轴，适用于连续数据。
19            // 'category' 类目轴，适用于离散的类目数据。为该类型时类目数据可自动从 series.data 或
            // dataset.source 中取，或者可通过 xAxis.data 设置类目数据。
20            // 'time' 时间轴，适用于连续的时序数据，与数值轴相比时间轴带有时间的格式化，在刻度计算上也有所不
            // 同，例如会根据跨度的范围来决定使用月，星期，日还是小时范围的刻度。
21            // 'log' 对数轴。适用于对数数据。
22          },
23          yAxis: {
24            //配置y轴坐标参数
25            type: "value", //同x轴的参数
26          },
27          series: [
28            //系列 配置图表的类型
29            {
30              type: "bar",
31              name: "销量", //系列名称，用于提示框组件的显示，
32              data: yData,
33              ///////////////最大值最小值////////////////////
34              markPoint: {
35                //图表标注。
36                data: [
37                  //标注的数据数组。每个数组项是一个对象
38                  {
39                    type: "max", //直接用 type 属性标注系列中的最大值，最小值。
40                    name: "最大值",
41                  },
42                  {
43                    type: "min",
44                    name: "最小值",
45                  },
46                ],
47              },
48              ///////////////最大值最小值////////////////////
49              ///////////////平均值////////////////////
50              markLine: {
51                //图表标线

```

```

52         data: [
53             //标线的数据数组。
54             {
55                 type: "average",
56                 name: "平均值",
57             },
58         ],
59     },
60     //////////////平均值////////////////////
61 },
62 ],
63 };
64 // 绘制图表 setOption 配置图表的配置项
65 myChart.setOption(option);
66 },
67 };
68 </script>
69 <style scoped>
70 #myecharts {
71     width: 600px;
72     height: 600px;
73     border: 2px solid red;
74 }
75 </style>

```

11.柱状图效果实现2--xAxis , yAxis

水平柱状图

通过设置xAxis yAxis中的type属性值来进行设置

barWidth : xx,设置柱图宽度

设置单独柱子的颜色

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="myecharts" ref="demoh"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        let myChart = echarts.init(this.$refs.demoh);
12        let xData = ["美食", "数码", "日化", "蔬菜", "熟食"]; //x轴数据
13        let yData = [88, 75, 20, 210, 35]; //y轴数据
14        let option = {
15          xAxis: {
16            type: "value", //数值轴
17            //坐标轴类型。'value' 数值轴，适用于连续数据。

```

```
18      // 'category' 类目轴，适用于离散的类目数据。为该类型时类目数据可自动从 series.data 或
dataset.source 中取，或者可通过 xAxis.data 设置类目数据。
19      // 'time' 时间轴，适用于连续的时序数据，与数值轴相比时间轴带有时间的格式化，在刻度计算上也有所不
同，例如会根据跨度的范围来决定使用月，星期，日还是小时范围的刻度。
20      // 'log' 对数轴。适用于对数数据。
21  },
22  yAxis: {
23    data: xData,
24    type: "category", //设置y为类目轴
25  },
26  series: [
27    {
28      type: "bar",
29      name: "销量",
30      data: yData,
31      barWidth: 50, //设置宽度
32      // color:"red", //设置颜色
33      // 单独设置每个柱子的颜色
34      itemStyle: {
35        normal: {
36          //每根柱子颜色设置
37          color: function (params) {
38            let colorList = [
39              "#c23531",
40              "#2f4554",
41              "#61a0a8",
42              "#d48265",
43              "#91c7ae",
44            ];
45            return colorList[params.dataIndex];
46          },
47        },
48      },
49      markPoint: {
50        data: [
51          {
52            type: "max",
53            name: "最大值",
54          },
55          {
56            type: "min",
57            name: "最小值",
58          },
59        ],
60      },
61      markLine: {
62        data: [
63          {
64            type: "average",
65            name: "平均值",
66          },
67        ],
68      },

```

```

69         },
70     },
71 ],
72 };
73 // 绘制图表 setOption 配置图表的配置项
74 myChart.setOption(option);
75 },
76 };
77 </script>
78 <style scoped>
79 #myecharts {
80     width: 600px;
81     height: 600px;
82     border: 2px solid red;
83 }
84 </style>

```

12. 饼状图基本设置

饼状图是用整个圆表示总体的数量或整体值“1”，用圆内各个扇形的大小表示各部分数量或该部分占总体的百分比。一般由标题（包括单位）、图例和数据等组成。1.主要运用在对数据进行比较分析的时候，既可以表示绝对量，又可以表示相对量。2.比柱形图等好在：数据更为清晰，各部分占总体的比重大小更为直观，可谓一目了然

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6  import * as echarts from "echarts"
7  export default {
8    mounted(){
9      // 1.初始化
10     let myChart=echarts.init(this.$refs.myChart)
11     // 2.设置echarts数据
12     let data=[
13       { value: 67, name: '美食' },
14       { value: 85, name: '日化' },
15       { value: 45, name: '数码' },
16       { value: 98, name: '家电' }
17     ]
18
19     // 3.设置配置项
20     let option={
21       title: {
22         text: '饼状图',
23         subtext: '基本设置',
24         left: 'center' //设置位置居中
25       },
26       tooltip: {
27         trigger: 'item' //触发类型item数据项图形触发
28       },

```



```

29   legend: {
30     orient: 'vertical', //图例列表的布局朝向vertical纵向
31     left: 'left'
32   },
33   series: [
34     {
35       name: '销售量',
36       type: 'pie', //饼图主要用于表现不同类目的数据在总和中的占比。每个的弧度表示数据数量的比例。
37       data
38     }
39   ]
40   }
41   // 4.设置图表绘制图表
42   myChart.setOption(option)
43   }
44 }
45 </script>
46
47 <style>
48 #myChart{
49   width: 500px;
50   height: 500px;
51   border: 1px solid red;
52 }
53 </style>

```

13.饼状图效果实现

但是饼状图还有更多的效果

环形图 样式等内容设置

```

1   <template>
2     <div ref="myChart" id="myChart"></div>
3   </template>
4
5   <script>
6     import * as echarts from "echarts"
7     export default {
8       mounted(){
9         // 1.初始化
10        let myChart=echarts.init(this.$refs.myChart)
11        // 2.设置echarts数据
12        // let data=[
13        //   { value: 67, name: '美食' },
14        //   { value: 85, name: '日化' },
15        //   { value: 45, name: '数码' },
16        //   { value: 98, name: '家电' }
17        // ]
18        let data=[
19          {

```

```

20         value: 67,
21         name: '美食',
22         itemStyle:{
23             normal:{
24                 color:'rgb(1,175,80)'
25             }
26         },
27     },
28     {
29         value: 85,
30         name: '日化',
31         itemStyle:{
32             normal:{
33                 color:'rgb(255,175,80)'
34             }
35         },
36     },
37 },
38 {
39     value: 45,
40     name: '数码',
41     itemStyle:{
42         normal:{
43             color:'rgb(1,0,80)'
44         }
45     },
46 },
47 {
48     value: 98,
49     name: '家电',
50     itemStyle:{
51         normal:{
52             color:'rgb(30,50,70)'
53         }
54     },
55 }
56 ]
57 // 单独设置每个颜色
58
59 // 3.设置配置项
60 let option={
61     title: {
62         text: '饼状图',
63         subtext: '基本设置',
64         left: 'center' //设置位置居中
65     },
66     tooltip: {
67         trigger: 'item' //触发类型item数据项图形触发
68     },
69     legend: {
70         orient: 'vertical', //图例列表的布局朝向vertical纵向
71         left: 'left'
72     },

```

```

73     series: [
74         {
75             name: '销售量',
76             type: 'pie',
77             // 设置环形图
78             radius: ['40%', '70%'], //饼图的半径。数组的第一项是内半径，第二项是外半径。
79             // 设置环形图
80             label: { //饼图图形上的文本标签
81                 show: true,
82                 position: "inside", //outside饼图扇区外侧inside饼图扇区内部center在饼图中心位置
83                 color: "yellow"
84             },
85             labelLine: { //标签的视觉引导线配置
86                 show: false
87             },
88             roseType: 'area', //是否展示成南丁格尔图，通过半径区分数据大小
89             itemStyle: { //设置内容样式
90                 color: '#c23531',
91                 shadowBlur: 200,
92                 shadowColor: 'rgba(0, 0, 0, 0.5)'
93             },
94             data
95         }
96     ]
97 }
98 // 4.设置图表绘制图表
99 myChart.setOption(option)
100 }
101 }
102 </script>
103
104 <style>
105 #myChart{
106     width: 500px;
107     height: 500px;
108     border: 1px solid red;
109 }
110 </style>

```

14.折线图基本设置

折线图是用折线将各个数据点标志连接起来的图表，用于展现数据的变化趋势。

不仅可以表示数量的多少，而且可以反映同一事物在不同时间里的发展变化的情况。易于显示数据变化趋势，可以直观地反映这种变化以及各组之间的差别。

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>

```

```

4
5 <script>
6 import * as echarts from "echarts"
7 export default {
8   mounted(){
9     // 1.初始化
10    let myChart=echarts.init(this.$refs.myChart)
11    // 2.设置数据
12    let xData=['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
13    let data=[150, 230, 224, 218, 135, 147, 260]
14    // 3.设置配置项
15    let option = {
16      xAxis: {
17        type: 'category',
18        data: xData
19      },
20      yAxis: {
21        type: 'value'
22      },
23      series: [
24        {
25          data,
26          type: 'line' //设置系列为折线图
27        }
28      ]
29    };
30    // 4.设置图表绘制图表
31    myChart.setOption(option)
32  }
33 }
34 </script>
35
36 <style>
37 #myChart{
38   width: 500px;
39   height: 500px;
40   border: 1px solid red;
41 }
42 </style>

```

15.折线图效果实现

设置平滑过渡样式 并且可以对内容进行颜色的填充 加上对应的标记点

```

1 <template>
2   <div ref="myChart" id="myChart"></div>
3 </template>
4
5 <script>
6 import * as echarts from "echarts"
7 export default {

```

```

8     mounted(){
9         // 1.初始化
10        let myChart=echarts.init(this.$refs.myChart)
11        // 2.设置数据
12        let xData=['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
13        let data=[150, 230, 224, 218, 135, 147, 260]
14        // 3.设置配置项
15        let option = {
16            xAxis: {
17                type: 'category',
18                data: xData
19            },
20            yAxis: {
21                type: 'value'
22            },
23            series: [
24                {
25                    data,
26                    type: 'line',//设置系列为折线图
27                    smooth: true,//是否平滑曲线显示如果是 number 类型（取值范围 0 到 1），表示平滑程度，越小表示越接近折线段，反之则反。设为 true 时相当于设为 0.5
28                    areaStyle: {},//区域填充样式。设置后显示成区域面积图。
29                    markPoint: {//图表标注。
30                        data: [
31                            { type: 'max', name: 'Max' },
32                            { type: 'min', name: 'Min' }
33                        ]
34                    },
35                    markLine: {//图表标线。
36                        data: [{ type: 'average', name: 'Avg' }]
37                    }
38                }
39            ]
40        };
41        // 4.设置图表绘制图表
42        myChart.setOption(option)
43    }
44 }
45 </script>
46
47 <style>
48 #myChart{
49     width: 500px;
50     height: 500px;
51     border: 1px solid red;
52 }
53 </style>

```

16.折线图堆叠效果

设置多折折线效果

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置数据
12       let xData = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"];
13       // 3.设置配置项
14       let option = {
15         xAxis: {
16           type: "category",
17           data: xData,
18         },
19         yAxis: {
20           type: "value",
21         },
22
23         series: [
24           {
25             name: "美食",
26             type: "line",
27             stack: "num", //数据堆叠，同个类目轴上系列配置相同的stack值后，后一个系列的值会在前一个系列
14         的值上相加。
28             data: [120, 132, 101, 134, 90, 230, 210],
29             areaStyle: {}, //区域填充样式。设置后显示成区域面积图。
30             emphasis: {
31               //折线图的高亮状态。
32               focus: "series", //聚焦当前高亮的数据所在的系列的所有图形。
33             },
34           },
35           {
36             name: "日化",
37             type: "line",
38             stack: "num",
39             data: [220, 182, 191, 234, 290, 330, 310],
40             areaStyle: {}, //区域填充样式。设置后显示成区域面积图。
41             emphasis: {
42               //折线图的高亮状态。
43               focus: "series", //聚焦当前高亮的数据所在的系列的所有图形。
44             },
45           },
46           {
47             name: "熟食",
48             type: "line",
49             stack: "num",
50             data: [150, 232, 201, 154, 190, 330, 410],
51             areaStyle: {}, //区域填充样式。设置后显示成区域面积图。
52             emphasis: {

```

```

53         //折线图的高亮状态。
54         focus: "series", //聚焦当前高亮的数据所在的系列的所有图形。
55     },
56 },
57 ],
58 };
59 // 4.设置图表绘制图表
60 myChart.setOption(option);
61 },
62 };
63 </script>
64
65 <style>
66 #myChart {
67     width: 500px;
68     height: 500px;
69     border: 1px solid red;
70 }
71 </style>

```

17.散点图基本效果设置

当存在大量数据点时，散点图的作用尤为明显。散点图与折线图相似，而不同之处在于折线图通过将点或数据点相连来显示每一个变化。

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {
13         xAxis: {},
14         yAxis: {},
15         series: [
16           {
17             symbolSize: 20,
18             data: [
19               [9.0, 7.04],
20               [18.07, 4.33],
21               [3.0, 9.65],
22               [9.05, 8.23],
23               [18.0, 9.76],
24               [15.0, 7.56],
25               [23.4, 5.31],
26               [10.1, 7.47],
27               [16.0, 8.26],
28               [12.7, 3.53],

```



```

29         [9.35, 7.2],
30         [7.4, 8.2],
31         [3.07, 4.82],
32         [18.2, 6.83],
33         [2.02, 4.47],
34         [1.05, 3.33],
35         [4.05, 4.96],
36         [6.03, 7.24],
37         [17.0, 6.55],
38         [12.0, 8.84],
39         [8.18, 5.82],
40         [6.32, 5.68],
41     ],
42     type: "scatter", //散点图
43
44     },
45 ],
46 };
47 // 3.设置图表绘制图表
48 myChart.setOption(option);
49 },
50 };
51 </script>
52
53 <style>
54 #myChart {
55     width: 500px;
56     height: 500px;
57     border: 1px solid red;
58 }
59 </style>

```

18.散点图效果实现

样式相关设置

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {
13         xAxis: {},

```

```

14     yAxis: {
15
16     },
17     tooltip: {}, //提示框组件
18     series: [
19         {
20             symbolSize: 20,
21             data: [
22                 [9.0, 7.04],
23                 [18.07, 4.33],
24                 [3.0, 9.65],
25                 [9.05, 8.23],
26                 [18.0, 9.76],
27                 [15.0, 7.56],
28                 [23.4, 5.31],
29                 [10.1, 7.47],
30                 [16.0, 8.26],
31                 [12.7, 3.53],
32                 [9.35, 7.2],
33                 [7.4, 8.2],
34                 [3.07, 4.82],
35                 [18.2, 6.83],
36                 [2.02, 4.47],
37                 [1.05, 3.33],
38                 [4.05, 4.96],
39                 [6.03, 7.24],
40                 [17.0, 6.55],
41                 [12.0, 8.84],
42                 [8.18, 5.82],
43                 [6.32, 5.68],
44             ],
45             type: "scatter", //散点图
46             // 圆形样式
47             color: { //线性渐变,前四个参数分别是 x0, y0, x2, y2, 范围从 0 - 1,相当于在图形包围盒中的
百分比
48                 type: "linear",
49                 x: 0,
50                 y: 0,
51                 x2: 1,
52                 y2: 0,
53                 colorStops: [
54                     {
55                         offset: 0,
56                         color: "#00CCFF", // 0% 处的颜色
57                     },
58                     {
59                         offset: 1,
60                         color: "rgba(255, 173, 119, 1)", // 100% 处的颜色
61                     },
62                 ],
63                 globalCoord: true, // 如果 globalCoord 为 `true`, 则该四个值是绝对的像素位置
64             },
65             emphasis: { //高亮的图形和标签样式

```

```

66         itemStyle: {
67             borderColor: "rgba(102,205,46,0.30)",
68             borderWidth: 30,
69         },
70     },
71 },
72 },
73 ],
74 };
75 // 3.设置图表绘制图表
76 myChart.setOption(option);
77 },
78 };
79 </script>
80
81 <style>
82 #myChart {
83     width: 500px;
84     height: 500px;
85     border: 1px solid red;
86 }
87 </style>

```

19.配置项--grid

grid 为直角坐标系内绘图网格。可以在网格上绘制折线图，柱状图 散点图（气泡图）也就是设置图标离容器的距离样式

```

1  <template>
2    <div class="about">
3      <!-- 2.echarts根结点根容器如果我们没有去指定当前容器的大小 echarts会按照0来进行处理 -->
4      <div id="myecharts" ref="demoh"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        let myChart = echarts.init(this.$refs.demoh);
12        let xData = ["美食", "数码", "日化", "蔬菜", "熟食"];
13        let yData = [88, 75, 20, 210, 35];
14        let option = {
15          xAxis: {
16            data: xData,
17            type: "category",
18          },
19          yAxis: {
20            type: "value",
21          },
22          // grid配置项：图标离容器的距离
23          // show:是否显示直角坐标系网格-----值:true?false

```

```

24      // left:图表离容器左侧的距离-----值:number?百分比
25      // top:图表离容器顶部的距离-----值:number?百分比
26      // right:图表离容器右侧的距离-----值:number?百分比
27      // bottom:图表离容器底部的距离-----值:number?百分比
28      // backgroundColor:网格背景色-----值:rgba或#000000
29      // borderColor:网格的边框颜色-----值:rgba或#000000
30      // borderWidth:网格的边框线宽-----值:number
31      grid: {
32          show: true,
33          left: "5%",
34          top: "5%",
35          right: "5%",
36          bottom: "5%",
37          backgroundColor: "rgba(224, 17, 17, 1)",
38          borderColor: "rgba(96, 67, 67, 1)",
39      },
40      series: [
41          {
42              type: "bar",
43              name: "销量",
44              data: yData,
45          },
46      ],
47      };
48      myChart.setOption(option);
49      },
50      };
51  </script>
52  <style scoped>
53  #myecharts {
54      width: 600px;
55      height: 600px;
56      border: 2px solid red;
57  }
58  </style>

```

20.K 线图

K线图可以查看k线历史走势，近期趋势，是上涨还是下跌，是调整还是震荡。分析k线的高低点和相对高低点。方便对于数据的走势进行查看

基本设置

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";

```

```

7   export default {
8     mounted() {
9       // 1.初始化
10      let myChart = echarts.init(this.$refs.myChart);
11      // 2.设置配置项
12      let option = {
13        xAxis: {
14          data: ["蔬菜", "水果", "熟食", "便捷食品"],
15        },
16        yAxis: {},
17        series: [
18          {
19            type: "candlestick", //k线图
20            data: [
21              [20, 34, 10, 38],
22              [40, 35, 30, 50],
23              [31, 38, 33, 44],
24              [38, 15, 5, 42],
25            ],
26          },
27        ],
28      };
29      // 3.设置图表绘制图表
30      myChart.setOption(option);
31    },
32  };
33 </script>
34
35 <style>
36 #myChart {
37   width: 500px;
38   height: 500px;
39   border: 1px solid red;
40 }
41 </style>

```

21 k线图效果优化

```

1   <template>
2     <div ref="myChart" id="myChart"></div>
3   </template>
4
5   <script>
6     import * as echarts from "echarts";
7     export default {
8       data(){
9         return {
10           data:[
11             [20, 34, 10, 38],
12             [40, 35, 30, 50],
13             [31, 38, 33, 44],
14             [38, 15, 5, 42],

```

```

15     ]
16   }
17 },
18 computed:{
19   newarr(){
20     let linstdata= this.data.map((v)=>{
21
22       return v[0]
23     })
24     return linstdata
25   }
26 },
27 mounted() {
28   // 1.初始化
29   let myChart = echarts.init(this.$refs.myChart);
30   // 2.设置配置项
31   let option = {
32     xAxis: {
33       data: ["蔬菜", "水果", "熟食", "便捷食品"],
34     },
35     yAxis: {},
36     tooltip: {
37       //设置提示框
38       trigger: "axis",
39       axisPointer: {
40         type: "cross",
41       },
42     },
43     series: [
44       {
45         type: "candlestick", //k线图
46         data: this.data,
47         itemStyle: {
48           color: "#ec0000", //上涨的颜色
49           color0: "#00da3c", //下跌的颜色
50           borderColor: "#8A0000", //上涨的边框色
51           borderColor0: "#008F28", //下跌的边框色
52         },
53         markPoint: {
54           data: [
55             {
56               name: "最大值",
57               type: "max",
58               valueDim: "highest", //valueDim 指定是在哪个维度上的最大值、最小值、平均值
59             },
60             {
61               name: "最小值",
62               type: "min",
63               valueDim: "lowest",
64             },
65             {
66               name: "平均值",
67               type: "average",

```

```

68         valueDim: "close",
69     },
70 ],
71 },
72 },
73 {
74     name: "MA20",
75     type: "line",
76     data: this.newarr,
77     smooth: true,
78     lineStyle: {
79         opacity: 0.5,
80     },
81 },
82 ],
83 };
84 // 3.设置图表绘制图表
85 myChart.setOption(option);
86 },
87 };
88 </script>
89
90 <style>
91 #myChart {
92     width: 500px;
93     height: 500px;
94     border: 1px solid red;
95 }
96 </style>

```

22.雷达图

基本设置

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {
13         title: {
14           text: "雷达图",
15         },
16         radar: [//雷达图坐标系组件，只适用于雷达图
17           {

```



```

18         // shape: 'circle', //设置及雷达图效果
19         indicator: [ //雷达图的指示器，用来指定雷达图中的多个变量（维度）
20             { name: "蔬菜", max: 6500 },
21             { name: "水果", max: 16000 },
22             { name: "熟食", max: 30000 },
23             { name: "数码", max: 38000 },
24             { name: "家电", max: 52000 },
25             { name: "日化", max: 25000 },
26         ],
27     },
28 ],
29
30     series: [
31         {
32             type: "radar", //雷达图
33             data: [
34                 {
35                     value: [4200, 3000, 20000, 35000, 50000, 18000],
36                     name: "销量",
37                 },
38             ],
39         },
40     ],
41 };
42 // 3.设置图表绘制图表
43 myChart.setOption(option);
44 },
45 };
46 </script>
47
48 <style>
49 #myChart {
50     width: 500px;
51     height: 500px;
52     border: 1px solid red;
53 }
54 </style>

```

23.雷达图效果优化

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {

```

```

13     title: {
14         text: "雷达图",
15     },
16     radar: [//雷达图坐标系组件，只适用于雷达图
17         {
18             // shape: 'circle', //设置及雷达图效果
19             indicator: [//雷达图的指示器，用来指定雷达图中的多个变量（维度）
20                 { name: "蔬菜", max: 6500 },
21                 { name: "水果", max: 16000 },
22                 { name: "熟食", max: 30000 },
23                 { name: "数码", max: 38000 },
24                 { name: "家电", max: 52000 },
25                 { name: "日化", max: 25000 },
26             ],
27             radius: 120, //半径
28             startAngle: 90, //坐标系起始角度，也就是第一个指示器轴的角度(可以让内容旋转)
29             splitNumber: 10, //指示器轴的分割段数(内部的分割数量)。
30             shape: "circle", //雷达图绘制类型
31             axisName: { //雷达图每个指示器名称的配置项
32                 formatter: "{value}", //使用字符串模板，模板变量为指示器名称 {value}
33                 color: "#428BD4",
34             },
35             splitArea: { //坐标轴在 grid 区域中的分隔区域，默认不显示。
36                 areaStyle: { //分隔区域的样式设置。
37                     color: ["#77EADF", "#26C3BE", "#64AFE9", "#428BD4"],
38                     shadowColor: "rgba(0, 0, 0, 0.2)",
39                     shadowBlur: 10,
40                 },
41             },
42         },
43     ],
44     series: [
45         {
46             type: "radar", //雷达图
47             symbol: "rect", //标记的图形。
48             symbolSize: 12, //标记大小
49             lineStyle: {
50                 type: "dashed",
51             },
52             data: [
53                 {
54                     value: [4200, 3000, 20000, 35000, 50000, 18000],
55                     name: "销量",
56                     areaStyle: {
57                         //设置填充
58                         color: "rgba(255, 228, 52, 0.6)",
59                     },
60                 },
61             ],
62         },
63     ],
64 };
65 // 3.设置图表绘制图表

```

```
66     myChart.setOption(option);
67   },
68 };
69 </script>
70
71 <style>
72 #myChart {
73   width: 500px;
74   height: 500px;
75   border: 1px solid red;
76 }
77 </style>
```

24.漏斗图 基本设置

```
1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {
13         title: {
14           text: "漏斗图",
15         },
16         tooltip: { //设置弹框
17           trigger: "item",
18           formatter: "{a} <br/>{b} : {c}%",
19         },
20
21
22         series: [
23           {
24
25             type: "funnel", //设置漏斗图
26             data: [
27               { value: 60, name: "美食" },
28               { value: 40, name: "日化" },
29               { value: 20, name: "数码" },
30               { value: 80, name: "家电" },
31               { value: 100, name: "蔬菜" },
32             ],
33           },
34         ],
35       };
36       // 3.设置图表绘制图表
```

```

37     myChart.setOption(option);
38   },
39 };
40 </script>
41
42 <style>
43 #myChart {
44   width: 500px;
45   height: 500px;
46   border: 1px solid red;
47 }
48 </style>

```

25 漏斗图效果实现

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        // 1.初始化
10       let myChart = echarts.init(this.$refs.myChart);
11       // 2.设置配置项
12       let option = {
13         title: {
14           text: "漏斗图",
15         },
16         tooltip: { //设置弹框
17           trigger: "item",
18           formatter: "{a} <br/>{b} : {c}%",
19         },
20
21
22         series: [
23           {
24
25             type: "funnel", //设置漏斗图
26             left: "10%", //漏斗图组件离容器左侧的距离
27             top: 60, //顶部距离
28             bottom: 60, //底部距离
29             // width: "80%",
30             min: 0, //指定的数据最小值。
31             max: 100,
32             minSize: "0%", //数据最小值 min 映射的宽度。
33             maxSize: "100%",
34             sort: "ascending", //数据排序递减的    ascending递增 none根据数据
35             gap: 2, //数据图形间距。
36             label: { //提示信息位置
37               show: true,

```

```

38         position: "inside",
39     },
40
41     itemStyle: { //漏斗图样式
42         borderColor: "red",
43         borderWidth: 2,
44     },
45     emphasis: { //选中高亮的标签和图形样式。
46         label: {
47             fontSize: 30,
48         },
49     },
50     data: [
51         { value: 60, name: "美食" },
52         { value: 40, name: "日化" },
53         { value: 20, name: "数码" },
54         { value: 80, name: "家电" },
55         { value: 100, name: "蔬菜" },
56     ],
57 },
58 ],
59 };
60 // 3.设置图表绘制图表
61 myChart.setOption(option);
62 },
63 };
64 </script>
65
66 <style>
67 #myChart {
68     width: 500px;
69     height: 500px;
70     border: 1px solid red;
71 }
72 </style>

```

26.仪表盘

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts"
7    export default {
8      mounted(){
9        let myEcharts=echarts.init(this.$refs.myChart)
10       let options={
11         series:[

```

```

12      {
13        type: "gauge",
14        data: [
15          {
16            value: 45,
17            name: "提示信息"
18          }
19        ],
20        detail: {
21          valueAnimation: true
22        },
23        progress: {
24          show: true
25        }
26      }
27    ]
28  }
29
30  myEcharts.setOption(options)
31  }
32
33  }
34  </script>
35
36  <style>
37  #myChart{
38    width: 500px;
39    height: 500px;
40    border: 1px solid red;
41  }
42  </style>

```

27. 关系图

创建节点

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6  import * as echarts from "echarts"
7  export default {
8    data(){
9      return {
10        list: [ //创建节点数据
11          {
12            name: "韦小宝",
13            id: "1",
14          },
15          {
16            name: "方怡",

```

```

17         id: "2",
18     },
19     {
20         name: "双儿",
21         id: "3",
22     },
23     {
24         name: "茅十八",
25         id: "4",
26     },
27     {
28         name: "吴六奇",
29         id: "5",
30     },
31 ]
32 }
33 },
34 mounted(){
35     let myEcharts=echarts.init(this.$refs.myChart)
36
37     let options = {
38         series:[
39             {
40                 type: 'graph', //图标类型为关系图用于展现节点以及节点之间的关系数据
41                 layout: 'force', //图的布局 引导布局
42                 data: this.list
43             }
44         ]
45     }
46
47     myEcharts.setOption(options)
48 }
49
50 }
51 </script>
52
53 <style>
54 #myChart{
55     width: 500px;
56     height: 500px;
57     border: 1px solid red;
58 }
59 </style>

```

增加节点样式

```

1 <template>
2   <div ref="myChart" id="myChart"></div>
3 </template>
4
5 <script>
6 import * as echarts from "echarts"
7 export default {

```



```

8     data(){
9         return {
10             list:[//创建节点数据
11                 {
12                     name: "韦小宝",
13                     id: "1",
14                     symbolSize: 30,//节点大小
15                     symbol:'circle',//节点形状,
16                 },
17                 {
18                     name: "方怡",
19                     id: "2",
20                     symbolSize: 30,//节点大小
21                     symbol:'circle',//节点形状,
22                 },
23                 {
24                     name: "双儿",
25                     id: "3",
26                     symbolSize: 30,//节点大小
27                     symbol:'circle',//节点形状,
28                 },
29                 {
30                     name: "茅十八",
31                     id: "4",
32                     symbolSize: 30,//节点大小
33                     symbol:'circle',//节点形状,
34                 },
35                 {
36                     name: "吴六奇",
37                     id: "5",
38                     symbolSize: 30,//节点大小
39                     symbol:'circle',//节点形状,
40                 },
41             ]
42         }
43     },
44     mounted(){
45         let myEcharts=echarts.init(this.$refs.myChart)
46
47         let options = {
48             series:[
49                 {
50                     type: 'graph',//图标类型为关系图用于展现节点以及节点之间的关系数据
51                     layout: 'force',//图的布局 引导布局
52                     data:this.list,
53                     itemStyle: {//节点的样式
54                         color: "#95dcb2"
55                     },
56                     label: {//图形上的文本标签
57                         show: true,
58                         position: "bottom",//位置底部
59                         distance: 5,//距离图形元素的距离
60                         fontSize: 18,

```

```

61         align: "center", //文字水平对齐方式
62     },
63 }
64 ]
65 }
66
67 myEcharts.setOption(options)
68 }
69
70 }
71 </script>
72
73 <style>
74 #myChart{
75     width: 500px;
76     height: 500px;
77     border: 1px solid red;
78 }
79 </style>

```

创建关系数据与图

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      data() {
9        return {
10          list: [
11            //创建节点数据
12            {
13              name: "韦小宝",
14              id: "1",
15              symbolSize: 30, //节点大小
16              symbol: "circle", //节点形状,
17            },
18            {
19              name: "方怡",
20              id: "2",
21              symbolSize: 30, //节点大小
22              symbol: "circle", //节点形状,
23            },
24            {
25              name: "双儿",
26              id: "3",
27              symbolSize: 30, //节点大小
28              symbol: "circle", //节点形状,
29            },
30            {
31              name: "茅十八",

```

```
32         id: "4",
33         symbolSize: 30, //节点大小
34         symbol: "circle", //节点形状,
35     },
36     {
37         name: "吴六奇",
38         id: "5",
39         symbolSize: 30, //节点大小
40         symbol: "circle", //节点形状,
41     },
42 ],
43 num: [
44     //关系数据
45     {
46         source: "1", //边的源节点名称的字符串
47         target: "2", //边的目标节点名称的字符串
48         relation: {
49             name: "老婆",
50             id: "1",
51         },
52     },
53     {
54         source: "1",
55         target: "3",
56         relation: {
57             name: "老婆",
58             id: "1",
59         },
60     },
61     {
62         source: "1",
63         target: "4",
64         relation: {
65             name: "兄弟",
66             id: "1",
67         },
68     },
69     {
70         source: "4",
71         target: "1",
72         relation: {
73             name: "兄弟",
74             id: "1",
75         },
76     },
77     {
78         source: "3",
79         target: "5",
80         relation: {
81             name: "义妹",
82             id: "1",
83         },
84     },
85 ],
```

```

85     ],
86     };
87 },
88 mounted() {
89     let myEcharts = echarts.init(this.$refs.myChart);
90
91     let options = {
92         series: [
93             {
94                 type: "graph", //图标类型为关系图用于展现节点以及节点之间的关系数据
95                 layout: "force", //图的布局 引导布局
96                 data: this.list,
97                 itemStyle: {
98                     //节点的样式
99                     color: "#95dcb2",
100                 },
101                 label: {
102                     //图形上的文本标签
103                     show: true,
104                     position: "bottom", //位置底部
105                     distance: 5, //距离图形元素的距离
106                     fontSize: 18,
107                     align: "center", //文字水平对齐方式
108                 },
109                 force: {
110                     //设置间距
111                     repulsion: 100, //点之间的距离
112                     gravity: 0.01, //设置距离中心点位置
113                     edgeLength: 200, //边的两个节点之间的距离
114                 },
115                 links: this.num, //节点间的关系数据
116
117                 edgeLabel: {
118                     //标签
119                     show: true,
120                     position: "middle", //标签位置线的中点
121                     fontSize: 12,
122                     formatter: (params) => {
123                         //标签内容格式设置内容
124                         return params.data.relation.name;
125                     },
126                 },
127                 edgeSymbol: ["circle", "arrow"], //边两边的类型
128
129                 autoCurveness: 0.01, //针对节点之间存在多边的情况，自动计算各边曲率
130             },
131         ],
132     };
133
134     myEcharts.setOption(options);
135 },
136 };
137 </script>

```

```
138
139 <style>
140 #myChart {
141   width: 500px;
142   height: 500px;
143   border: 1px solid red;
144 }
145 </style>
```

28.数据区域缩放

用于区域缩放，从而能自由关注细节的数据信息，或者概览数据整体，或者去除离群点的影响。

```
1 <template>
2   <div ref="myChart" id="myChart"></div>
3 </template>
4
5 <script>
6 import * as echarts from "echarts"
7 export default {
8   mounted(){
9     let myChart=echarts.init(this.$refs.myChart)
10    let xData=['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
11    let data=[150, 230, 224, 218, 135, 147, 260]
12    let option = {
13      xAxis: {
14        type: 'category',
15        data: xData
16      },
17      yAxis: {
18        type: 'value'
19      },
20      series: [
21        {
22          data,
23          type: 'line',
24          smooth: true,
25          markPoint: {
26            data: [
27              { type: 'max', name: 'Max' },
28              { type: 'min', name: 'Min' }
29            ]
30          },
31          markLine: {
32            data: [{ type: 'average', name: 'Avg' }]
33          }
34        },
35      ],
36      dataZoom: [//用于区域缩放，从而能自由关注细节的数据信息，或者概览数据整体，或者去除离群点的影响。
37        {
38          type: 'slider', //滑动条型数据区域缩放组件
39          xAxisIndex: 0, //x轴设置
```

```

41     filterMode: 'none' //设置-----数据过滤不过滤数据，只改变数轴范围
42   },
43   {
44     type: 'slider',
45     yAxisIndex: 0, //y轴设置
46     filterMode: 'none'
47   },
48 ],
49 ];
50 };
51 // 4.设置图表绘制图表
52 myChart.setOption(option)
53 }
54 }
55 </script>
56
57 <style>
58 #myChart{
59   width: 500px;
60   height: 500px;
61   border: 1px solid red;
62 }
63 </style>

```

29.基本树形图

树图主要用来可视化树形数据结构，是一种特殊的层次类型，具有唯一的根节点，左子树，和右子树。

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {
8      mounted() {
9        let myEcharts = echarts.init(this.$refs.myChart);
10       let data = { // 注意，最外层是一个对象，代表树的根节点
11         name: "层级1", // 节点的名称，当前节点 label 对应的文本
12         children: [
13           {
14             name: "层级2",
15             children: [
16               {
17                 name: "层级3-1",
18                 children: [ //子节点
19                   { name: "数据1", value: 3938 }, // value 值，只在 tooltip 中显示
20                   { name: "数据2", value: 3812 },
21                   { name: "数据3", value: 6714 },
22                   { name: "数据4", value: 743 },
23                 ],
24               },
25             ],
26           },
27         ],
28       },
29     },
30   },
31 }

```

```

26         name: "层级3-2",
27         children: [
28             { name: "数据1", value: 3534 },
29             { name: "数据2", value: 5731 },
30             { name: "数据3", value: 7840 },
31             { name: "数据4", value: 5914 },
32             { name: "数据5", value: 3416 },
33         ],
34     },
35 ],
36 },
37 ],
38 };
39
40 let options = {
41     tooltip: { //提示框
42         trigger: "item", //触发时机
43     },
44     series: [
45         {
46             type: "tree", //树图
47             data: [data],
48             top: "1%", //tree组件离容器顶部的距离
49             left: "7%",
50             bottom: "1%",
51             right: "20%",
52             symbolSize: 10, //标记的大小
53             label: { //描述了每个节点所对应的文本标签的样式。
54                 position: "left", //标签的位置。
55                 verticalAlign: "middle", //文字垂直对齐方式
56                 align: "right", //文字水平对齐方式
57                 fontSize: 9,
58             },
59             leaves: { //叶子节点的特殊配置
60                 label: { //了叶子节点所对应的文本标签的样式
61                     position: "right",
62                     verticalAlign: "middle",
63                     align: "left",
64                 },
65             },
66             emphasis: { //树图中个图形和标签高亮的样式。
67                 focus: "descendant", //聚焦所有子孙节点
68             },
69             expandAndCollapse: true, //子树折叠和展开的交互由于绘图区域是有限的，而通常一个树图的节点可能
             // 会出现节点之间相互遮盖的问题。为了避免这一问题，可以将暂时无关的子树折叠收起，
             // 等到需要时再将其展开。如上面径向布局树图示例，节点中心用蓝色填充的就是折叠收起的子树，
             // 可以点击将其展开。
70             animationDuration: 550,
71             animationDurationUpdate: 750,
72         },
73     ],
74 };

```

```

78
79     myEcharts.setOption(options);
80   },
81 };
82 </script>
83
84 <style>
85 #myChart {
86   width: 500px;
87   height: 500px;
88   border: 1px solid red;
89 }
90 </style>

```

30方向切换树形图

```

1
2
3 <template>
4   <div ref="myChart" id="myChart"></div>
5 </template>
6
7 <script>
8 import * as echarts from "echarts";
9 export default {
10   mounted() {
11     let myEcharts = echarts.init(this.$refs.myChart);
12     let data = { // 注意，最外层是一个对象，代表树的根节点
13       name: "层级1", // 节点的名称，当前节点 label 对应的文本
14       children: [
15         {
16           name: "层级2",
17           children: [
18             {
19               name: "层级3-1",
20               children: [ // 子节点
21                 { name: "数据1", value: 3938 }, // value 值，只在 tooltip 中显示
22                 { name: "数据2", value: 3812 },
23                 { name: "数据3", value: 6714 },
24                 { name: "数据4", value: 743 },
25               ],
26             },
27             {
28               name: "层级3-2",
29               children: [
30                 { name: "数据1", value: 3534 },
31                 { name: "数据2", value: 5731 },
32                 { name: "数据3", value: 7840 },
33                 { name: "数据4", value: 5914 },
34                 { name: "数据5", value: 3416 },
35               ],
36             },

```



```

37         ],
38     },
39 ],
40 };
41
42 let options = {
43     tooltip: { //提示框
44         trigger: "item", //触发时机
45     },
46     series: [
47         {
48             type: "tree",
49             data: [data],
50             top: "1%",
51             left: "7%",
52             bottom: "1%",
53             right: "20%",
54             symbolSize: 10,
55             //树图中 正交布局 的方向
56             // 水平 方向的 从左到右'LR', 从右到左'RL' ;
57             // 以及垂直方向的 从上到下'TB', 从下到上'BT'
58             orient: 'BT',
59             label: {
60                 position: "bottom",
61                 rotate: 90, //文字旋转
62                 verticalAlign: "middle",
63                 align: "right",
64                 fontSize: 9,
65             },
66             leaves: {
67                 label: {
68                     position: "right",
69                     verticalAlign: "middle",
70                     align: "left",
71                 },
72             },
73             emphasis: {
74                 focus: "descendant",
75             },
76             expandAndCollapse: true,
77
78             animationDuration: 550,
79             animationDurationUpdate: 750,
80         },
81     ],
82 };
83
84 myEcharts.setOption(options);
85 },
86 };
87 </script>
88
89 <style>

```

```

90 #myChart {
91   width: 500px;
92   height: 500px;
93   border: 1px solid red;
94 }
95 </style>

```

31.数据排序

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <!-- 2.echarts根结点根容器如果我们没有去指定当前容器的大小 echarts会按照0来进行处理 -->
5      <div id="myecharts" ref="demoh"></div>
6    </div>
7  </template>
8  <script>
9    import * as echarts from "echarts";
10   export default {
11     mounted() {
12       let myChart = echarts.init(this.$refs.demoh);
13
14       let option = {
15         dataset: [
16           //数据集组件用于单独的数据集声明，从而数据可以单独管理，被多个组件复用，并且可以自由指定数据到视觉
17           //的映射。这在不少场景下能带来使用上的方便。
18           {
19             dimensions: ["分类", "数量"], //设置分类数据
20             source: [
21               //原始数据。一般来说，原始数据表达的是二维表。
22               ["Hannah Krause", 41],
23               ["Zhao Qian", 20],
24               ["Jasmin Krause ", 52],
25               ["Li Lei", 37],
26               ["Karle Neumann", 25],
27               ["Adrian Groß", 19],
28               ["Mia Neumann", 71],
29             ],
30             transform: {
31               //数据改变
32               type: "sort", //按照大小排序
33               config: { dimension: "数量", order: "desc" }, // "sort" 数据转换器的“条件”
34             },
35           },
36         ],
37         xAxis: {
38           type: "category",

```

```

40     axisLabel: {
41         //坐标轴刻度标签的相关设置。
42         interval: 0, //坐标轴刻度标签的显示间隔，在类目轴中有效。
43         rotate: 30, //刻度标签旋转的角度
44     },
45 },
46 yAxis: {},
47 series: [
48     //系列 配置图表的类型
49     {
50         type: "bar",
51         encode: {
52             //可以定义 data 的哪个维度被编码成什么。
53             x: "分类", //x映射内容
54             y: "数量",
55         },
56         datasetIndex: 1,
57     },
58 ],
59 };
60 // 绘制图表 setOption 配置图表的配置项
61 myChart.setOption(option);
62 },
63 };
64 </script>
65 <style scoped>
66 #myecharts {
67     width: 600px;
68     height: 600px;
69     border: 2px solid red;
70 }
71 </style>

```

32.内置主题

echarts中默认主题有两个:light、dark

echarts.init(选取容器dom,'主题')

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4
5      <div id="myecharts" ref="demoh"></div>
6    </div>
7  </template>
8  <script>
9    import * as echarts from "echarts";
10   export default {
11     mounted() {
12       //echarts中默认主题有两个:light、dark

```

```

13     let myChart = echarts.init(this.$refs.demoh, "dark");
14     let xData = ["美食", "数码", "日化", "蔬菜", "熟食"];
15     let yData = [88, 75, 20, 210, 35];
16     let option = {
17       xAxis: {
18         data: xData,
19         type: "category",
20       },
21       yAxis: {
22         type: "value",
23       },
24       series: [
25         {
26           type: "bar",
27           name: "销量",
28           data: yData,
29         },
30       ],
31     };
32
33     myChart.setOption(option);
34   },
35 };
36 </script>
37 <style scoped>
38 #myecharts {
39   width: 600px;
40   height: 600px;
41   border: 2px solid red;
42 }
43 </style>

```

33.自定义主题

1.在主题编辑器中编辑主题

主题编辑器地址：<https://echarts.apache.org/zh/theme-builder.html>

2.下载对应主题json格式

3.创建js文件把刚才下载的文件写入并且暴露

```

1   let roma=你的主题json
2
3   export default roma

```

4.引用主题文件

```

1   import roma from "../assets/roma"

```

5.在init方法中使用主题

34.中国地图展示效果

1.准备echarts基本结构

2.设置中国地图的矢量数据创建js文件（在其中创建变量接受json数据 并且暴露）

地图数据下载地址：https://datav.aliyun.com/portal/school/atlas/area_selector

3.在组件中获取地图矢量数据（引用数据json）

4.使用地图数据创建地图

```
1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4
5      <div id="myecharts" ref="demoh"></div>
6    </div>
7  </template>
8  <script>
9    import * as echarts from "echarts";
10   // 引用的就是中国各省份的矢量数据
11   import cmap from "../assets/roma"
12   export default {
13     mounted() {
14
15       let myChart = echarts.init(this.$refs.demoh);
16       echarts.registerMap("chinaMap",cmap)//使用 registerMap 注册的地图名称。
17       let option = {
18         geo:{//地理坐标系组件。地理坐标系组件用于地图的绘制
19           type:"map",
20           map:"chinaMap",//使用 registerMap 注册的地图名称
21           // 默认设置完地图是固定死的不能拖动
22           roam:true,//否开启鼠标缩放和平移漫游。默认不开启。
23           zoom :10,//当前视角的缩放比例。越大比例越大
24           center:[108.956239,34.268309],//当前视角的中心点，用经纬度表示108.956239,34.268309
25           label:{//地图上显示文字提示信息
26             show:true,
27             color:"#ff6600",
28             fontSize:10//字体大小
29           },
30           itemStyle:{//地图区域的多边形 图形样式。
31             areaColor:"#ff6600"//地图区域的颜色。
32           }
33         }
34       };
35
36       myChart.setOption(option);
37     },
38   };
39 </script>
```

```

40 <style scoped>
41 #myecharts {
42   width: 600px;
43   height: 600px;
44   border: 2px solid red;
45 }
46 </style>

```

35.省份地图显示

同中国地图使用方式 就是切换地图数据即可

36.地图标记设置与效果

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4
5      <div id="myecharts" ref="demoh"></div>
6    </div>
7  </template>
8  <script>
9    import * as echarts from "echarts";
10
11    import cmap from "../assets/roma";
12    export default {
13      mounted() {
14        // 设置气泡点数据
15        let data = [
16          {
17            value: [108.956239, 34.268309],
18          },
19        ];
20
21        let myChart = echarts.init(this.$refs.demoh);
22        echarts.registerMap("chinaMap", cmap);
23        let option = {
24          geo: {
25            type: "map",
26            map: "chinaMap",
27
28            roam: true,
29          },
30          series: [
31            {
32              type: "effectScatter", //带有涟漪特效动画的散点（气泡）图。利用动画特效可以将某些想要突出的
数据点进行视觉突出。
33              coordinateSystem: "geo", //使用什么坐标系geo使用地理坐标系
34              data,
35              // 这个时候地图上就会有有点的涟漪效果
36              rippleEffect: {
37                //涟漪特效相关配置。

```

```

38         number: 2, //波纹的数量。
39         scale: 4, //动画中波纹的最大缩放比例
40     },
41     // label:{
42     //     show:true
43     // },
44     itemStyle: {
45         color: "red",
46     },
47 },
48 // 也可以绘制点效果
49 {
50     symbolSize: 20,
51     data: [
52         {
53             name: "北京市", // 数据项名称, 在这里指地区名称
54             value: [
55                 // 数据项值
56                 116.46, // 地理坐标, 经度
57                 39.92, // 地理坐标, 纬度
58                 340, // 北京地区的数值
59             ],
60         },
61     ],
62     type: "scatter",
63     coordinateSystem: "geo",//// series坐标系类型
64 },
65 ],
66 };
67
68 myChart.setOption(option);
69 },
70 };
71 </script>
72 <style scoped>
73 #myecharts {
74     width: 600px;
75     height: 600px;
76     border: 2px solid red;
77 }
78 </style>

```

37.图表自适应大小

当浏览器大小改变的时候 我们需要让图表一同改变 这个时候就会用到图表自适应大小

```

1  <template>
2    <div ref="myChart" id="myChart"></div>
3  </template>
4
5  <script>
6    import * as echarts from "echarts";
7    export default {

```

```
8   mounted() {
9     let myChart = echarts.init(this.$refs.myChart);
10    let xData = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"];
11    let option = {
12      xAxis: {
13        type: "category",
14        data: xData,
15      },
16      yAxis: {
17        type: "value",
18      },
19
20      series: [
21        {
22          name: "美食",
23          type: "line",
24          stack: "num",
25          data: [120, 132, 101, 134, 90, 230, 210],
26          areaStyle: {},
27          emphasis: {
28            focus: "series",
29          },
30        },
31        {
32          name: "日化",
33          type: "line",
34          stack: "num",
35          data: [220, 182, 191, 234, 290, 330, 310],
36          areaStyle: {},
37          emphasis: {
38            focus: "series",
39          },
40        },
41        {
42          name: "熟食",
43          type: "line",
44          stack: "num",
45          data: [150, 232, 201, 154, 190, 330, 410],
46          areaStyle: {},
47          emphasis: {
48            focus: "series",
49          },
50        },
51      ],
52    };
53    myChart.setOption(option);
54
55    // 监听页面的大小
56    window.addEventListener("resize", () => {
57      myChart.resize()
58    });
59  },
60  };
```



```

61   </script>
62
63   <style>
64   #myChart {
65     width: 100%;
66     height: 500px;
67     border: 1px solid red;
68   }
69   </style>

```

38.图表加载动画效果

myChart.showLoading();开始等待

myChart.hideLoading();关闭等待

1.设置json-server模拟数据

(1) 全局下载 npm install -g json-server

(2) 新建mock文件夹 并且在其中创建json文件 设置数据

(3) 终端cd到mock文件夹下 启动 json-server --watch xx.json --port 端口号

2.页面请求数据并且设置等待效果

```

1   <template>
2     <div ref="myChart" id="myChart"></div>
3   </template>
4
5   <script>
6     import * as echarts from "echarts";
7     import axios from "axios";
8     // import {mapData} from "../assets/mapData.js"
9     export default {
10      data() {
11        return {
12          echartsData: {},
13        };
14      },
15      methods: {
16        // 获取json-server数据
17        async linkData() {
18          let mapnum = await axios({ url: "http://localhost:3000/one" });
19          console.log(mapnum.data);
20          this.echartsData = mapnum.data;
21        },
22      },
23
24      mounted() {
25        // 1.初始化
26        let myChart = echarts.init(this.$refs.myChart);
27        // 设置开始等待
28        myChart.showLoading();
29

```

```

30 // 调用数据请求方法
31 this.linkData().then(() => {
32     myChart.hideLoading();
33     // 2.设置echarts数据
34
35     let option = {
36         title: {
37             text: "饼状图",
38             subtext: "基本设置",
39             left: "center", //设置位置居中
40         },
41         tooltip: {
42             trigger: "item", //触发类型item数据项图形触发
43         },
44         legend: {
45             orient: "vertical", //图例列表的布局朝向vertical纵向
46             left: "left",
47         },
48         series: [
49             {
50                 name: "销售量",
51                 type: "pie",
52                 // 设置环形图
53                 radius: ["40%", "70%"], //饼图的半径。数组的第一项是内半径，第二项是外半径。
54                 // 设置环形图
55                 label: {
56                     //饼图图形上的文本标签
57                     show: true,
58                     position: "inside", //outside饼图扇区外侧inside饼图扇区内部center在饼图中心位置
59                     color: "yellow",
60                 },
61                 labelLine: {
62                     //标签的视觉引导线配置
63                     show: false,
64                 },
65                 roseType: "area", //是否展示成南丁格尔图，通过半径区分数据大小
66                 itemStyle: {
67                     //设置内容样式
68                     color: "#c23531",
69                     shadowBlur: 200,
70                     shadowColor: "rgba(0, 0, 0, 0.5)",
71                 },
72                 data: this.echartsData,
73             },
74         ],
75     };
76     // 4.设置图表绘制图表
77     myChart.setOption(option);
78 });
79 },
80 };
81 </script>
82

```

```

83 <style>
84 #myChart {
85   width: 500px;
86   height: 500px;
87   border: 1px solid red;
88 }
89 </style>

```

39.图表动画配置

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="myecharts" ref="demoh"></div>
5    </div>
6  </template>
7  <script>
8    import * as echarts from "echarts";
9    export default {
10      mounted() {
11        let myChart = echarts.init(this.$refs.demoh);
12        let xData = ["美食", "数码", "日化", "蔬菜", "熟食"];
13        let yData = [88, 75, 20, 210, 35];
14        let option = {
15          animation: true, //是否开启动画。
16          animationThreshold: 5, //是否开启动画的阈值，当单个系列显示的图形数量大于这个阈值时会关闭动画。
17          animationDuration: 2000, //初始动画的时长
18          animationEasing: "linear", //初始动画的缓动效果。官方更多解释：
19            https://echarts.apache.org/examples/zh/editor.html?c=line-easing
20          animationDelay: 1000, //初始动画的延迟
21          xAxis: {
22            type: "value",
23          },
24          yAxis: {
25            data: xData,
26            type: "category",
27          },
28          series: [
29            {
30              type: "bar",
31              name: "销量",
32              data: yData,
33              barWidth: 50,
34              itemStyle: {
35                normal: {
36                  color: function (params) {
37                    let colorList = [
38                      "#c23531",
39                      "#2f4554",
40                      "#61a0a8",
41                      "#d48265",
42                      "#91c7ae",

```

```

42         ];
43         return colorList[params.dataIndex];
44     },
45     },
46 },
47 markPoint: {
48     data: [
49         {
50             type: "max",
51             name: "最大值",
52         },
53         {
54             type: "min",
55             name: "最小值",
56         },
57     ],
58 },
59
60 markLine: {
61     data: [
62         {
63             type: "average",
64             name: "平均值",
65         },
66     ],
67 },
68 },
69 ],
70 };
71 myChart.setOption(option);
72 },
73 };
74 </script>
75 <style scoped>
76 #myecharts {
77     width: 600px;
78     height: 600px;
79     border: 2px solid red;
80 }
81 </style>

```

40.echarts 事件

ECharts 中我们可以通过监听用户的操作行为来回调对应的函数。

ECharts 通过 `on` 方法来监听用户的行为，例如监控用户的点击行为。

```

1  <template>
2    <div class="about">
3      <h1>This is an about page</h1>
4      <div id="myecharts" ref="demoh"></div>

```

```

5     </div>
6 </template>
7 <script>
8 import * as echarts from "echarts";
9 export default {
10   mounted() {
11     let myChart = echarts.init(this.$refs.demoh);
12     // 事件
13     // ECharts 中我们可以通过监听用户的操作行为来回调对应的函数。
14
15     // ECharts 通过 on 方法来监听用户的行为，例如监控用户的点击行为。
16     myChart.on("click", function (params) {
17       // 在用户点击后控制台打印数据的名称
18       // params对象的属性
19       // componentType 当前点击的图形元素所属的组件名称
20       // seriesType 系列类型
21
22       // seriesName 系列名称。
23       // name数据名，类目名
24       // data传入的原始数据项
25       // value传入的数据值
26
27       console.log(params);
28     });
29
30     let xData = ["美食", "数码", "日化", "蔬菜", "熟食"];
31     let yData = [88, 75, 20, 210, 35];
32     let option = {
33       xAxis: {
34         type: "value",
35       },
36       yAxis: {
37         data: xData,
38         type: "category",
39       },
40       series: [
41         {
42           type: "bar",
43           name: "销量",
44           data: yData,
45           barWidth: 50,
46           itemStyle: {
47             normal: {
48               color: function (params) {
49                 let colorList = [
50                   "#c23531",
51                   "#2f4554",
52                   "#61a0a8",
53                   "#d48265",
54                   "#91c7ae",
55                 ];
56                 return colorList[params.dataIndex];
57               },

```

```

58         },
59     },
60     markPoint: {
61         data: [
62             {
63                 type: "max",
64                 name: "最大值",
65             },
66             {
67                 type: "min",
68                 name: "最小值",
69             },
70         ],
71     },
72
73     markLine: {
74         data: [
75             {
76                 type: "average",
77                 name: "平均值",
78             },
79         ],
80     },
81 },
82
83 ],
84 };
85 myChart.setOption(option);
86 },
87 };
88 </script>
89 <style scoped>
90 #myecharts {
91     width: 600px;
92     height: 600px;
93     border: 2px solid red;
94 }
95 </style>

```

有多个图形怎么监听呢？

使用 query 只对指定的组件的图形元素的触发回调：

```

1 chart.on(eventName, query, handler);

```

```

1 chart.on('click', 'series', function () {...});
2 chart.on('click', 'series.line', function () {...});
3 chart.on('click', 'dataZoom', function () {...});
4 chart.on('click', 'xAxis.category', function () {...});

```

下面就在添加一个折线图

```

1 <template>

```

```

2     <div class="about">
3         <h1>This is an about page</h1>
4         <div id="myecharts" ref="demoh"></div>
5     </div>
6 </template>
7 <script>
8 import * as echarts from "echarts";
9 export default {
10     mounted() {
11         let myChart = echarts.init(this.$refs.demoh);
12         // 事件
13         // ECharts 中我们可以通过监听用户的操作行为来回调对应的函数。
14
15         // ECharts 通过 on 方法来监听用户的行为，例如监控用户的点击行为。
16         // myChart.on("click", function (params) {
17             // 在用户点击后控制台打印数据的名称
18             // params对象的属性
19             // componentType 当前点击的图形元素所属的组件名称
20             // seriesType 系列类型
21
22             // seriesName 系列名称。
23             // name数据名，类目名
24             // data传入的原始数据项
25             // value传入的数据值
26
27             // console.log(params);
28             // });
29
30
31         // 只对折线图做出反应
32         // myChart.on("click", 'series.line',function (params) {
33             // console.log(params);
34             // });
35         // 只对某一项最做出反应
36         // 比如对折线图的数码项点击做出反应
37         // myChart.on("click",{name:"数码"},function (params) {
38             // console.log(params);
39             // });
40         // 但是发现折线图柱状图都可以
41         // 只对折线图生效
42         myChart.on("click",{seriesIndex: 1,name:"数码"},function (params) {
43             console.log(params);
44         });
45
46         let xData = ["美食", "数码", "日化", "蔬菜", "熟食"];
47         let yData = [88, 75, 20, 210, 35];
48         let option = {
49             xAxis: {
50                 type: "value",
51             },
52             yAxis: {
53                 data: xData,
54                 type: "category",

```

```

55     },
56     series: [
57         {
58             type: "bar",
59             name: "销量",
60             data: yData,
61             barWidth: 50,
62             itemStyle: {
63                 normal: {
64                     color: function (params) {
65                         let colorList = [
66                             "#c23531",
67                             "#2f4554",
68                             "#61a0a8",
69                             "#d48265",
70                             "#91c7ae",
71                         ];
72                         return colorList[params.dataIndex];
73                     },
74                 },
75             },
76             markPoint: {
77                 data: [
78                     {
79                         type: "max",
80                         name: "最大值",
81                     },
82                     {
83                         type: "min",
84                         name: "最小值",
85                     },
86                 ],
87             },
88
89             markLine: {
90                 data: [
91                     {
92                         type: "average",
93                         name: "平均值",
94                     },
95                 ],
96             },
97         },
98         // 在添加一个折线图
99         {
100             data: [150, 230, 224, 218, 135],
101             type: "line", //设置系列为折线图
102             smooth: true, //是否平滑曲线显示如果是 number 类型（取值范围 0 到 1），表示平滑程度，越小
103             //表示越接近折线段，反之则反。设为 true 时相当于设为 0.5
104
105             markPoint: {
106                 //图表标注。
107                 data: [

```



```
107         { type: "max", name: "Max" },
108         { type: "min", name: "Min" },
109     ],
110 },
111     markLine: {
112         //图表标线。
113         data: [{ type: "average", name: "Avg" }],
114     },
115 },
116 ],
117 };
118 myChart.setOption(option);
119 },
120 };
121 </script>
122 <style scoped>
123 #myecharts {
124     width: 600px;
125     height: 600px;
126     border: 2px solid red;
127 }
128 </style>
```