01.ECharts的快速上手

```
//步骤1：引入echarts.js文件
//步骤2：准备一个呈现图表的盒子
//步骤3：初始化echarts实例对象
// 步骤4：准备配置项
// 步骤5：将配置项设置给echarts实例对象
<body>
    <!-- 步骤2：准备一个呈现图表的盒子 -->
    <div style="width: 600px;height: 400px"></div>
    <script>
        var mCharts = echarts.init(document.querySelector('div'))
        var option = {
                xAxis: {
                    type: 'category',
                    data: ['小明', '小红', '小王']
                },
                yAxis: {
                    type: 'value'
                },
                series: [{
                    name: '语文',
                    type: 'bar',
                    data: [70, 92, 87]
                }]
            }
        mCharts.setOption(option)
    </script>
```

02.配置项的讲解和使用

```
<!-- 步骤2：准备一个呈现图表的盒子 -->
  <div style="width: 600px;height: 400px"></div>
  <script>
    // 步骤3：初始化echarts实例对象
    // 参数，dom,决定图表最终呈现的位置
    var mCharts = echarts.init(document.querySelector('div'))
    // 步骤4：准备配置项
    var option = {
      title: {
        text: '成绩', // 标题文字
        link: 'http://www.baidu.con', // 标题超链接
        textStyle: { // 标题样式设置
          color: 'red' // 标题文字
        }
      },
      xAxis: {
        type: 'category',  // 指明x轴为 类目轴
        data: ['小明', '小红', '小王'] // 为类目轴提供数据，该数据是一个数组，数组中的每个元素会呈现在
```

```
x轴上
      },
      yAxis: {
        type: 'value'  // 指明x轴为 数值轴，指明数值轴之后，无需指定data
      },
      series: [
        {
          name: '语文', // 为图标起一个名称
          type: 'bar', // 指明该图标类型为 柱状图
          data: [70, 92, 87] // 为x轴的每一个元素，指明呈现在y轴的数值
        }
      ]
    }
    // 步骤5：将配置项设置给echarts实例对象
    mCharts.setOption(option)
  </script>
```

03.柱状图的实现

```
<div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. x轴数据:['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    //3. y轴数据:[88, 92, 63, 77, 94, 80, 72, 86]
    //4. 将type的值设置为bar
    var mCharts = echarts.init(document.querySelector("div")) // 初始化echarts实例对象
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强'] // 准备x轴数据
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86] // 为x轴每一个元素指明数据
    var option = {
      xAxis: {type: 'value'},
      yAxis: {type: 'category',data: xDataArr},
      series: [
        {
          name: '语文',
          type: 'bar',
          markPoint: { // 标记点
            data: [{type: 'max', name: '最大值'},{type: 'min', name: '最小值'}]
          },
          markLine: { // 标记线
            data: [{type: 'average', name: '平均值'}]
          },
          label: { // 柱状图上的文字设置
            show: true, // 是否显示
            rotate: 60, // 旋转角度
            position: 'top' // 显示位置
          },
          barWidth: '30%', // 柱的宽度
          data: yDataArr
        }
      ]
    }

    mCharts.setOption(option)
```

```
  </script>
```

04.通用配置01

```
<div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86]
    var option = {
      title: { // 标题设置
        text: '成绩展示', // 标题文字
        textStyle: { // 标题文字样式设置
          color: 'red'
        },
        borderWidth: 5, // 标题边框宽度
        borderColor: 'blue', // 标题边框颜色
        borderRadius: 5, // 标题边框圆角
        left: 50, // 标题距离左边的距离
        top: 10 // 标题距离顶部的距离
      },
      tooltip: { // 工具提示
        // trigger: 'item' 工具提示的类型 item代表的是每个柱本身, axis代表的是坐标轴
        trigger: 'axis',
        triggerOn: 'click', // 触发时机, click代表点击, mouseOver代表鼠标移过
        // formatter: '{b} 的成绩是 {c}'
        formatter: function(arg){ // 文字格式化
          return arg[0].name + '的分数是:' + arg[0].data
        }
      },
      xAxis: {type: 'category',data: xDataArr},
      yAxis: {type: 'value'},
      series: [
        {name: '语文',type: 'bar',data: yDataArr}
      ]
    }
    mCharts.setOption(option)
  </script>
```

05.通用配置02

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86]
    var option = {
      title: {
        text: '成绩展示',
        textStyle: {
```

```
          color: 'red'
        },
        borderWidth: 5,
        borderColor: 'blue',
        borderRadius: 5,
        left: 50,
        top: 10
      },
      tooltip: {
        // trigger: 'item'
        trigger: 'axis',
        triggerOn: 'click',
        // formatter: '{b} 的成绩是 {c}'
        formatter: function(arg){
          return arg[0].name + '的分数是:' + arg[0].data
        }
      },
      toolbox: { // 工具箱按钮
        feature: {
          saveAsImage: {}, // 导出图片
          dataView: {}, // 数据视图
          restore: {}, // 重置
          dataZoom: {}, // 区域缩放
          magicType: {
            type: ['bar', 'line']
          } // 动态图表类型的切换
        }
      },
      xAxis: {
        type: 'category',
        data: xDataArr
      },
      yAxis: {
        type: 'value'
      },
      series: [
        {
          name: '语文',
          type: 'bar',
          data: yDataArr
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

06.通用配置03

```
<div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))

    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
```

```
var yDataArr1 = [88, 92, 63, 77, 94, 80, 72, 86]
var yDataArr2 = [93, 60, 61, 62, 85, 79, 92, 30]
var option = {
  title: {
    text: '成绩展示',
    textStyle: {
      color: 'red'
    },
    borderWidth: 5,
    borderColor: 'blue',
    borderRadius: 5,
    left: 50,
    top: 10
  },
  tooltip: {
    // trigger: 'item'
    trigger: 'axis',
    triggerOn: 'click',
    // formatter: '{b} 的成绩是 {c}'
    formatter: function(arg){
      return arg[0].name + '的分数是:' + arg[0].data
    }
  },
  toolbox: {
    feature: {
      saveAsImage: {}, // 导出图片
      dataView: {}, // 数据视图
      restore: {}, // 重置
      dataZoom: {}, // 区域缩放
      magicType: {
        type: ['bar', 'line']
      } // 动态图表类型的切换
    }
  },
  legend: { // 图例，图例中的data数据来源于series中每个对象的name，图例可以帮助我们对图表进行筛选
    data: ['语文', '数学']
  },
  xAxis: {
    type: 'category',
    data: xDataArr
  },
  yAxis: {
    type: 'value'
  },
  series: [
    {
      name: '语文',
      type: 'bar',
      data: yDataArr1
    },
    {
      name: '数学',

      type: 'bar',
```

```
          data: yDataArr2
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

07.折线图的实现

```
<div style="width: 600px;height: 400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. x轴数据:['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
'12月']
    //3. y轴数据:[3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    //4. 将type的值设置为line
    var mCharts = echarts.init(document.querySelector('div'))
    var xDataArr = ['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
'12月']
    var yDataArr = [3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    var option = {
      xAxis: {
        type: 'category',
        data: xDataArr
      },
      yAxis: {
        type: 'value'
      },
      series: [
        {
          name: '康师傅',
          data: yDataArr,
          type: 'line' // 设置图表类型为 折线图
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

08.折线图的常见效果

```
<div style="width: 600px;height: 400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. x轴数据:['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
'12月']
    //3. y轴数据:[3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    //4. 将type的值设置为line
    var mCharts = echarts.init(document.querySelector('div'))

    var xDataArr = ['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
```

```
'12月']
    var yDataArr = [3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    var option = {
      xAxis: {
        type: 'category',
        data: xDataArr
      },
      yAxis: {
        type: 'value'
      },
      series: [
        {
          name: '康师傅',
          data: yDataArr,
          type: 'line',
          markPoint: { // 标记点
            data: [
              {
                type: 'max'
              },
              {
                type: 'min'
              }
            ]
          },
          markLine: { // 标记线
            data: [
              {
                type: 'average'
              }
            ]
          },
          markArea: { // 标记区域
            data: [
              [
                {
                  xAxis: '1月'
                },
                {
                  xAxis: '2月'
                }
              ],
              [
                {
                  xAxis: '7月'
                },
                {
                  xAxis: '8月'
                }
              ]
            ]
          },

          smooth: true, // 是否为平滑线
```

```
          lineStyle: { // 线的样式设置
            color: 'green',
            type: 'solid' // dashed dotted solid
          },
          areaStyle: { // 线和x轴形成的区域设置
            color: 'pink'
          }
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

09.折线图的常见效果

```
<body>
  <div style="width: 600px;height: 400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector('div'))
    var xDataArr = ['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
'12月']
    // var yDataArr = [3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    var yDataArr = [3005, 3003, 3001, 3002, 3009, 3007, 3003, 3001, 3005, 3004, 3001, 3009]
    var option = {
      xAxis: {
        type: 'category',
        data: xDataArr,
        boundaryGap: false // x轴的第1个元素是否与y轴有距离
      },
      yAxis: {
        type: 'value',
        scale: true
      },
      series: [
        {
          name: '康师傅',
          data: yDataArr,
          type: 'line'
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

10.折线图的常见效果_堆叠图

```
<body>
  <div style="width: 600px;height: 400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['1月', '2月', '3月', '4月', '5月', '6月', '7月', '8月', '9月', '10月', '11月',
```

```
'12月']
    var yDataArr = [3000, 2800, 900, 1000, 800, 700, 1400, 1300, 900, 1000, 800, 600]
    var yDataArr2 = [2000, 3800, 1900, 500, 900, 1700, 2400, 300, 1900, 1500, 1800, 200]
    var option = {
      xAxis: {
        type: 'category',
        data: xDataArr
      },
      yAxis: {
        type: 'value',
      },
      series: [
        {
          type: 'line',
          data: yDataArr,
          stack: 'all', // 堆叠图的设置
          areaStyle: {}
        },
        {
          type: 'line',
          data: yDataArr2,
          stack: 'all', // 堆叠图的设置
          areaStyle: {}
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

11.散点图的实现

```
<div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. x轴和y轴数据 二维数组 [ [身高,体重],...   ]
    //3. 将type的值设置为scatter, x轴和y轴的type都是value

    var axisData = []
    for( var i=0;i<data.length;i++) {
      var height = data[i].height
      var weight = data[i].weight
      var newArr = [height, weight]
      axisData.push(newArr)
    }
    console.log(axisData)
    var mCharts = echarts.init(document.querySelector("div"))
    var option = {
      xAxis: {
        type: 'value',
        scale: true
      },

      yAxis: {
```

```
        type: 'value',
        scale: true
      },
      series: [
        {
          type: 'scatter', // 指明图表的类型为散点图
          data: axisData
        }
      ]
    }
    mCharts.setOption(option)
  </script>
</body>
</html>
```

12.散点图的常见效果

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. x轴和y轴数据 二维数组 [ [身高,体重],...   ]
    //3. 将type的值设置为scatter, x轴和y轴的type都是value

    var axisData = []
    for (var i = 0; i < data.length; i++) {
      var height = data[i].height
      var weight = data[i].weight
      var newArr = [height, weight]
      axisData.push(newArr)
    }
    console.log(axisData)
    var mCharts = echarts.init(document.querySelector("div"))
    var option = {
      xAxis: {
        type: 'value',
        scale: true
      },
      yAxis: {
        type: 'value',
        scale: true
      },
      series: [
        {
          // type: 'scatter',
          type: 'effectScatter', // 指明图表为带涟漪动画的散点图
          showEffectOn: 'emphasis', // 出现涟漪动画的时机 render emphasis
          rippleEffect: {
            scale: 10 // 涟漪动画时, 散点的缩放比例
          },
          data: axisData,

          // symbolSize: 30
```

```
          symbolSize: function (arg) { // 控制散点的大小
            // console.log(arg)
            var height = arg[0] / 100
            var weight = arg[1]
            // bmi = 体重kg / (身高m*身高m)   大于28,就代表肥胖
            var bmi = weight / (height * height)
            if (bmi > 28) {
              return 20
            }
            return 5
          },
          itemStyle: { // 控制散点的样式
            color: function (arg) {
              // console.log(arg)
              var height = arg.data[0] / 100
              var weight = arg.data[1]
              // bmi = 体重kg / (身高m*身高m)   大于28,就代表肥胖
              var bmi = weight / (height * height)
              if (bmi > 28) {
                return 'red'
              }
              return 'green'
            }
          }
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

13.直角坐标系的常用配置_grid

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86]
    var option = {
      grid: { // 坐标轴容器
        show: true, // 是否可见
        borderWidth: 10, // 边框的宽度
        borderColor: 'red', // 边框的颜色
        left: 120, // 边框的位置
        top: 120,
        width: 300, // 边框的大小
        height: 150
      },
      xAxis: {
        type: 'category',
        data: xDataArr

      },
```

```
    yAxis: {
      type: 'value'
    },
    series: [
      {
        name: '语文',
        type: 'bar',
        markPoint: {
          data: [
            {
              type: 'max', name: '最大值'
            },{
              type: 'min', name: '最小值'
            }
          ]
        },
        markLine: {
          data: [
            {
              type: 'average', name: '平均值'
            }
          ]
        },
        label: {
          show: true,
          rotate: 60,
          position: 'top'
        },
        barWidth: '30%',
        data: yDataArr
      }
    ]
  }
  mCharts.setOption(option)
</script>
```

14.直角坐标系的常用配置_axis

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86]
    var option = {
      grid: {
        show: true,
        borderColor: 'red',
      },
      xAxis: {
        type: 'category',

        data: xDataArr,
```

```
        position: 'top' // 控制坐标轴的位置
      },
      yAxis: {
        type: 'value',
        position: 'right' // 控制坐标轴的位置
      },
      series: [
        {
          name: '语文',
          type: 'bar',
          markPoint: {
            data: [
              {
                type: 'max', name: '最大值'
              },{
                type: 'min', name: '最小值'
              }
            ]
          },
          markLine: {
            data: [
              {
                type: 'average', name: '平均值'
              }
            ]
          },
          label: {
            show: true,
            rotate: 60,
            position: 'top'
          },
          barWidth: '30%',
          data: yDataArr
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

15.直角坐标系的常用配置dataZoom

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    var mCharts = echarts.init(document.querySelector("div"))
    var xDataArr = ['张三', '李四', '王五', '闰土', '小明', '茅台', '二妞', '大强']
    var yDataArr = [88, 92, 63, 77, 94, 80, 72, 86]
    var option = {
      dataZoom: [ // 控制区域缩放效果的实现
        {
          type: 'slider', // 缩放的类型  slide代表滑块  inside代表依靠鼠标滚轮

          // type: 'inside'
```

```
      xAxisIndex: 0
    },
    {
      type: 'slider',
      yAxisIndex: 0,
      start: 0, // 渲染完成后，数据筛选的初始值，百分比
      end: 80 // 渲染完成后，数据筛选的结束值，百分比
    }
  ],
  toolbox: {
    feature: {
      dataZoom: {}
    }
  },
  grid: {
    show: true,
    borderColor: 'red',
  },
  xAxis: {
    type: 'category',
    data: xDataArr
  },
  yAxis: {
    type: 'value'
  },
  series: [
    {
      name: '语文',
      type: 'bar',
      markPoint: {
        data: [
          {
            type: 'max', name: '最大值'
          },{
            type: 'min', name: '最小值'
          }
        ]
      },
      markLine: {
        data: [
          {
            type: 'average', name: '平均值'
          }
        ]
      },
      label: {
        show: true,
        rotate: 60,
        position: 'top'
      },
      barWidth: '30%',
      data: yDataArr

    }
```

```
      ]
    }
    mCharts.setOption(option)
  </script>
```

16.饼图的实现

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. 准备数据[{name:???, value:??? },{}]
    //   淘宝: 11231  京东: 22673  唯品会: 6123  1号店: 8989   聚美优品: 6700
    //3. 将type的值设置为pie
    var mCharts = echarts.init(document.querySelector("div"))
    // pieData就是需要设置给饼图的数据，数组,数组中包含一个又一个的对象，每一个对象中，需要有name和
value
    var pieData = [
      {
        name: '淘宝',
        value: 11231
      },
      {
        name: '京东',
        value: 22673
      },
      {
        name: '唯品会',
        value: 6123
      },
      {
        name: '1号店',
        value: 8989
      },
      {
        name: '聚美优品',
        value: 6700
      }
    ]
    var option = {
      series: [
        {
          type: 'pie',
          data: pieData
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

17.饼图的常见效果

```
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. 准备数据[{name:???, value:??? },{}]
    //   淘宝: 11231  京东: 22673  唯品会: 6123  1号店: 8989   聚美优品: 6700
    //3. 将type的值设置为pie
    var mCharts = echarts.init(document.querySelector("div"))
    // pieData就是需要设置给饼图的数据，数组,数组中包含一个又一个的对象，每一个对象中，需要有name和
value
    var pieData = [
      {
        name: '淘宝',
        value: 11231
      },
      {
        name: '京东',
        value: 22673
      },
      {
        name: '唯品会',
        value: 6123
      },
      {
        name: '1号店',
        value: 8989
      },
      {
        name: '聚美优品',
        value: 6700
      }
    ]
    var option = {
      series: [
        {
          type: 'pie',
          data: pieData,
          label: { // 饼图文字的显示
            show: true, // 显示文字
            //formatter: 'hehe' // 决定文字显示的内容
            formatter: function(arg){
              // console.log(arg)
              return arg.name + '平台' + arg.value + '元\n' + arg.percent + '%'
            }
          },
          // radius: 20 // 饼图的半径
          // radius: '20%' // 百分比参照的是宽度和高度中较小的那一部分的一半来进行百分比设置
          // radius: ['50%', '75%'] // 第0个元素代表的是内圆的半径 第1个元素外圆的半径
          roseType: 'radius', // 南丁格尔图 饼图的每一个区域的半径是不同的
          // selectedMode: 'single' // 选中的效果,能够将选中的区域偏离圆点一小段距离
          selectedMode: 'multiple',
          selectedOffset: 30

        }
```

```
      ]
    }
    mCharts.setOption(option)
  </script>
```

18.地图的实现

```
  <script src="lib/echarts.min.js"></script>
  <script src="lib/jquery.min.js"></script>
</head>

<body>
  <div style="width: 600px;height:400px;border: 1px solid #f00"></div>

  <script>
    //1. ECharts最基本的代码结构
    //2. 准备中国地图的矢量数据
    //3. 使用Ajax获取矢量地图数据
    //4. 在Ajax的回调函数中注册地图矢量数据 echarts.registerMap('chinaMap', 矢量地图数据)
    //5. 配置geo的type为'map', map为'chinaMap'
    var mCharts = echarts.init(document.querySelector("div"))
    $.get('json/map/china.json', function (ret) {
      // ret 就是中国的各个省份的矢量地图数据
      // console.log(ret)
      echarts.registerMap('chinaMap', ret)
      var option = {
        geo: {
          type: 'map',
          map: 'chinaMap', // chinaMap需要和registerMap中的第一个参数保持一致
          roam: true, // 设置允许缩放以及拖动的效果
          label: {
            show: true // 展示标签
          },
          zoom: 1, // 设置初始化的缩放比例
          center: [87.617733, 43.792818] // 设置地图中心点的坐标
        }
      }
      mCharts.setOption(option)
    })
  </script>
```

19.地图的常见效果_显示某个省份的数据

```
<body>
  <div style="width: 600px;height:400px;border: 1px solid #f00"></div>

  <script>
    //1. 加载安徽省地图的矢量数据
    //2. 在Ajax的回调函数中注册地图矢量数据 echarts.registerMap('anhui', 矢量地图数据)
    //3. 配置geo的type为'map', map为'anhui'
```

```
    //4. 通过zoom调整缩放比例
    //5. 通过center调整中心点
    var mCharts = echarts.init(document.querySelector("div"))
    $.get('json/map/anhui.json', function (ret) {
      console.log(ret)
      echarts.registerMap('anhui', ret)
      var option = {
        geo: {
          type: 'map',
          map: 'anhui',
          zoom: 1.2,
          label: {
            show: true
          },
          center: [116.507676, 31.752889] // 这个坐标值，我们是可以通过地图矢量数据获取到的
        }
      }
      mCharts.setOption(option)
    })
  </script>
```

20.地图的常见效果_不同城市的颜色不同

```
<div style="width: 600px;height:400px;border: 1px solid #f00"></div>
  <script>
    //1. 显示基本的中国地图
    //2. 将空气质量的数据设置给series下的对象
    //3. 将series下的数据和geo关联起来
    //4. 配置visualMap
    var airData = [
        { name: '北京', value: 39.92 },
        { name: '天津', value: 39.13 },
        { name: '上海', value: 31.22 },
        { name: '重庆', value: 66 },
        { name: '河北', value: 147 },
        { name: '河南', value: 113 },
        { name: '云南', value: 25.04 },
        { name: '辽宁', value: 50 },
        { name: '黑龙江', value: 114 },
        { name: '湖南', value: 175 },
        { name: '安徽', value: 117 },
        { name: '山东', value: 92 },
        { name: '新疆', value: 84 },
        { name: '江苏', value: 67 },
        { name: '浙江', value: 84 },
        { name: '江西', value: 96 },
        { name: '湖北', value: 273 },
        { name: '广西', value: 59 },
        { name: '甘肃', value: 99 },
        { name: '山西', value: 39 },
        { name: '内蒙古', value: 58 },

        { name: '陕西', value: 61 },
```

```
        { name: '吉林', value: 51 },
        { name: '福建', value: 29 },
        { name: '贵州', value: 71 },
        { name: '广东', value: 38 },
        { name: '青海', value: 57 },
        { name: '西藏', value: 24 },
        { name: '四川', value: 58 },
        { name: '宁夏', value: 52 },
        { name: '海南', value: 54 },
        { name: '台湾', value: 88 },
        { name: '香港', value: 66 },
        { name: '澳门', value: 77 },
        { name: '南海诸岛', value: 55 }
    ]
    var mCharts = echarts.init(document.querySelector("div"))
    $.get('json/map/china.json', function (ret) {
      // ret 就是中国的各个省份的矢量地图数据
      console.log(ret)
      echarts.registerMap('chinaMap', ret)
      var option = {
        geo: {
          type: 'map',
          map: 'chinaMap', // chinaMap需要和registerMap中的第一个参数保持一致
          roam: true, // 设置允许缩放以及拖动的效果
          label: {
            show: true // 展示标签
          }
        },
        series: [
          {
            data: airData,
            geoIndex: 0, // 将空气质量的数据和第0个geo配置关联在一起
            type: 'map'
          }
        ],
        visualMap: {
          min: 0,
          max: 300,
          inRange: {
            color: ['white', 'red'] // 控制颜色渐变的范围
          },
          calculable: true // 出现滑块
        }
      }
      mCharts.setOption(option)
    })

  </script>
```

21.地图常见效果_地图和散点图结合

```
<body>
```

```
<div style="width: 600px;height:400px;border: 1px solid #f00"></div>
<script>
  //1. 给series下增加一个新的对象
  //2. 准备数据散点数据  , 配置给series下的另外一个对象
  //3. 配置series下的新对象的type值为effectScatter
  //4. 指明散点图的坐标系统为geo
  //5. 调整涟漪动画效果
  var airData = [
    { name: '北京', value: 39.92 },
    { name: '天津', value: 39.13 },
    { name: '上海', value: 31.22 },
    { name: '重庆', value: 66 },
    { name: '河北', value: 147 },
    { name: '河南', value: 113 },
    { name: '云南', value: 25.04 },
    { name: '辽宁', value: 50 },
    { name: '黑龙江', value: 114 },
    { name: '湖南', value: 175 },
    { name: '安徽', value: 117 },
    { name: '山东', value: 92 },
    { name: '新疆', value: 84 },
    { name: '江苏', value: 67 },
    { name: '浙江', value: 84 },
    { name: '江西', value: 96 },
    { name: '湖北', value: 273 },
    { name: '广西', value: 59 },
    { name: '甘肃', value: 99 },
    { name: '山西', value: 39 },
    { name: '内蒙古', value: 58 },
    { name: '陕西', value: 61 },
    { name: '吉林', value: 51 },
    { name: '福建', value: 29 },
    { name: '贵州', value: 71 },
    { name: '广东', value: 38 },
    { name: '青海', value: 57 },
    { name: '西藏', value: 24 },
    { name: '四川', value: 58 },
    { name: '宁夏', value: 52 },
    { name: '海南', value: 54 },
    { name: '台湾', value: 88 },
    { name: '香港', value: 66 },
    { name: '澳门', value: 77 },
    { name: '南海诸岛', value: 55 }
  ]
  var scatterData = [
    {
      value: [117.283042, 31.86119]
    }
  ]
  var mCharts = echarts.init(document.querySelector("div"))
  $.get('json/map/china.json', function (ret) {
    // ret 就是中国的各个省份的矢量地图数据

    console.log(ret)
```

```
      echarts.registerMap('chinaMap', ret)
      var option = {
        geo: {
          type: 'map',
          map: 'chinaMap', // chinaMap需要和registerMap中的第一个参数保持一致
          roam: true, // 设置允许缩放以及拖动的效果
          label: {
            show: true // 展示标签
          }
        },
        series: [
          {
            data: airData,
            geoIndex: 0, // 将空气质量的数据和第0个geo配置关联在一起
            type: 'map'
          },
          {
            data: scatterData, // 配置散点的坐标数据
            type: 'effectScatter',
            coordinateSystem: 'geo', // 指明散点使用的坐标系统  geo的坐标系统
            rippleEffect: {
              scale: 10 // 设置涟漪动画的缩放比例
            }
          }
        ],
        visualMap: {
          min: 0,
          max: 300,
          inRange: {
            color: ['white', 'red'] // 控制颜色渐变的范围
          },
          calculable: true // 出现滑块
        }
      }
      mCharts.setOption(option)
    })
  </script>
```

22.雷达图的实现

```
  <script src="lib/echarts.min.js"></script>
</head>
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. 定义各个维度的最大值，通过radar属性配置
    //   易用性,功能,拍照,跑分,续航，每个维度的最大值都是100
    //3. 准备产品数据，设置给series下的data
    //   华为手机1: 80, 90, 80, 82, 90
    //   中兴手机1: 70, 82, 75, 70, 78
```

```
    //4. 将type的值设置为radar
    var mCharts = echarts.init(document.querySelector("div"))
    // 各个维度的最大值
    var dataMax = [
      {
        name: '易用性',
        max: 100
      },
      {
        name: '功能',
        max: 100
      },
      {
        name: '拍照',
        max: 100
      },
      {
        name: '跑分',
        max: 100
      },
      {
        name: '续航',
        max: 100
      }
    ]
    var option = {
      radar: {
        indicator: dataMax, // 配置各个维度的最大值
        shape: 'polygon' // 配置雷达图最外层的图形 circle polygon
      },
      series: [
        {
          type: 'radar', // radar 此图表时一个雷达图
          label: { // 设置标签的样式
            show: true // 显示数值
          },
          areaStyle: {}, // 将每一个产品的雷达图形成阴影的面积
          data: [
            {
              name: '华为手机1',
              value: [80, 90, 80, 82, 90]
            },
            {
              name: '中兴手机1',
              value: [70, 82, 75, 70, 78]
            }
          ]
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```

23.仪表盘的实现

```
<script src="lib/echarts.min.js"></script>
</head>
<body>
  <div style="width: 600px;height:400px"></div>
  <script>
    //1. ECharts最基本的代码结构
    //2. 准备数据，设置给series下的data
    //3. 将type的值设置为gauge
    var mCharts = echarts.init(document.querySelector("div"))
    var option = {
      series: [
        {
          type: 'gauge',
          data: [
            {
              value: 97,
              itemStyle: { // 指针的样式
                color: 'pink' // 指针的颜色
              }
            }, // 每一个对象就代表一个指针
            {
              value: 85,
              itemStyle: {
                color: 'green'
              }
            }
          ],
          min: 50 // min max 控制仪表盘数值范围
        }
      ]
    }
    mCharts.setOption(option)
  </script>
```