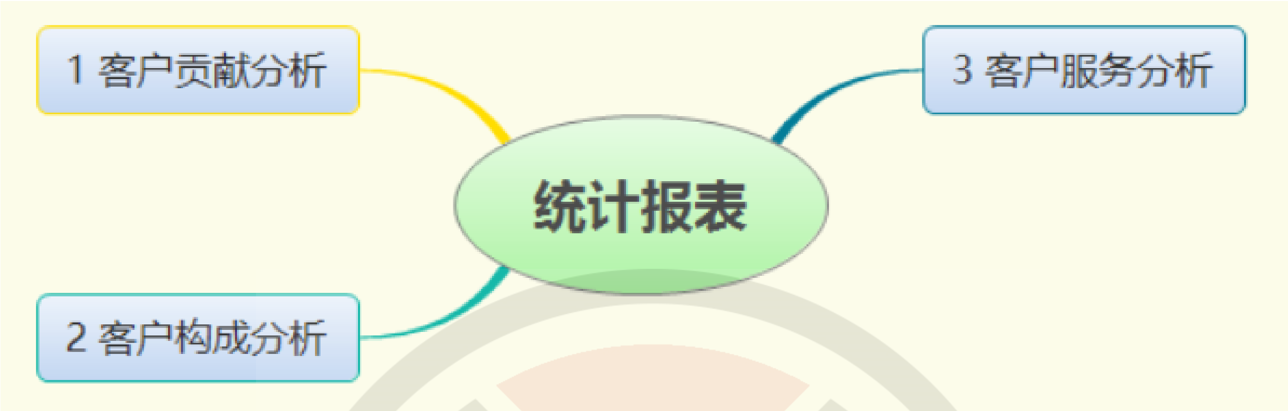
统计报表

一 学习目标

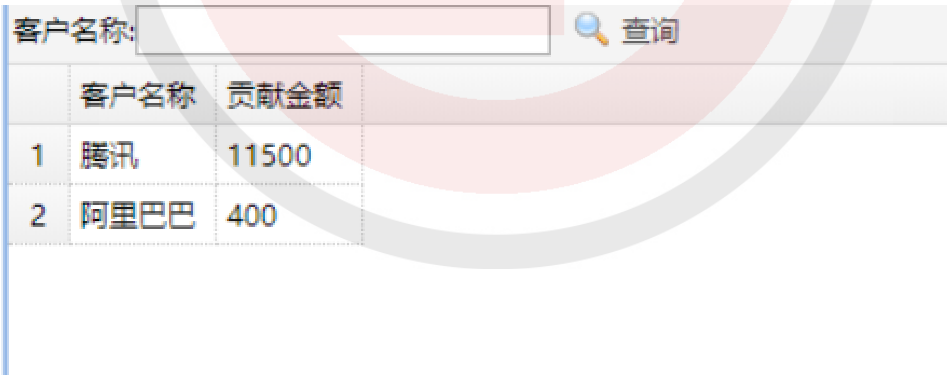


二 客户贡献分析

1 需求分析

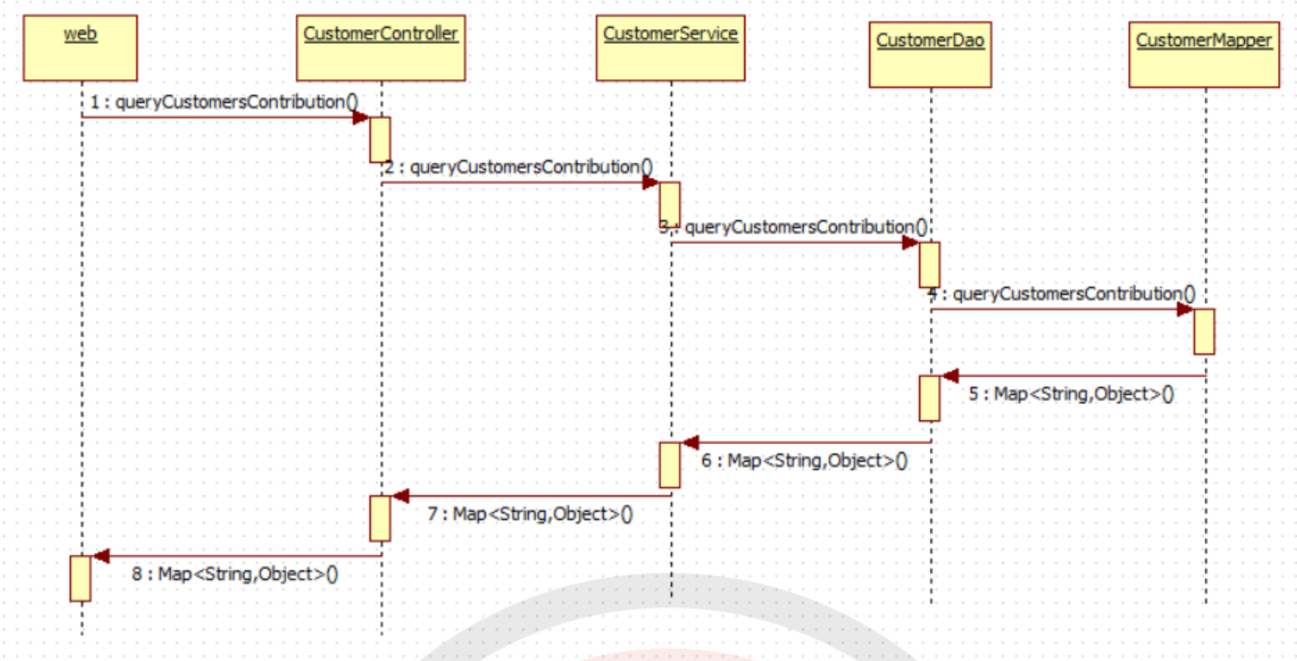
按客户统计客户历史订单金额就代表客户的贡献分析

2 页面原型



	客户名称	贡献金额
1	腾讯	11500
2	阿里巴巴	400

3 流程图



4 编写Controller

```
        @Controller
        @RequestMapping("report")
        public class ReportController extends BaseController{

            @RequestMapping("/{type}")
            public String index(@PathVariable("type") String type){

                switch (type){
                    case "0": return "customer_contribution";
                    case "1": return "customer_gc";
                    case "2": return "customer_serve";
                    case "3": return "customer_loss";
                    default: return "error";
                }
            }
        }

        @RequestMapping("queryCustomersContribution")
        @ResponseBody
        public Map<String,Object> queryCustomersContribution(CustomerGCQuery
customerGCQuery){
            return  customerService.queryCustomersContribution(customerGCQuery);
        }
```

5 编写Service

```
        public Map<String,Object> queryCustomersContribution(CustomerGCQuery
customerGCQuery){
            PageHelper.startPage(customerGCQuery.getPage(),customerGCQuery.getRows());
            List<CustomerDto> list =
customerDao.queryCustomersContribution(customerGCQuery);
            PageInfo<CustomerDto> pageInfo = new PageInfo<>(list);
            Map<String,Object> map = new HashMap<>();
            map.put("rows",pageInfo.getList());
            map.put("total",pageInfo.getTotal());
            return  map;
        }
```

6 编写Dao

```
        public List<CustomerDto> queryCustomersContribution(CustomerGCQuery
customerGCQuery);
```

7 编写Mapper

```
        <select id="queryCustomersContribution"  parameterType="customerGCQuery"
 resultType="customerDto" >

            select c.name,sum(sum) as total

            from t_order_details od
            LEFT JOIN t_customer_order co  on od.order_id = co.order_no
            LEFT JOIN t_customer c on co.cus_id = c.id

            where od.is_valid = 1 and co.is_valid = 1 and c.is_valid = 1

            <if test="customerName != null  and  customerName !='' ">

                and c.name like concat('%',#{customerName},'%')

            </if>

            group by co.cus_id

            ORDER BY total desc


        </select>
```

8 编写Dto

```
        public class CustomerDto {

            private String name;
            private Integer total;
```

```
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getTotal() {
        return total;
    }

    public void setTotal(Integer total) {
        this.total = total;
    }
}
```

9 编写Query

```
public class CustomerGCQuery extends BaseQuery{

    private String customerName;

    public String getCustomerName() {
        return customerName;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }
}
```

10 前台js

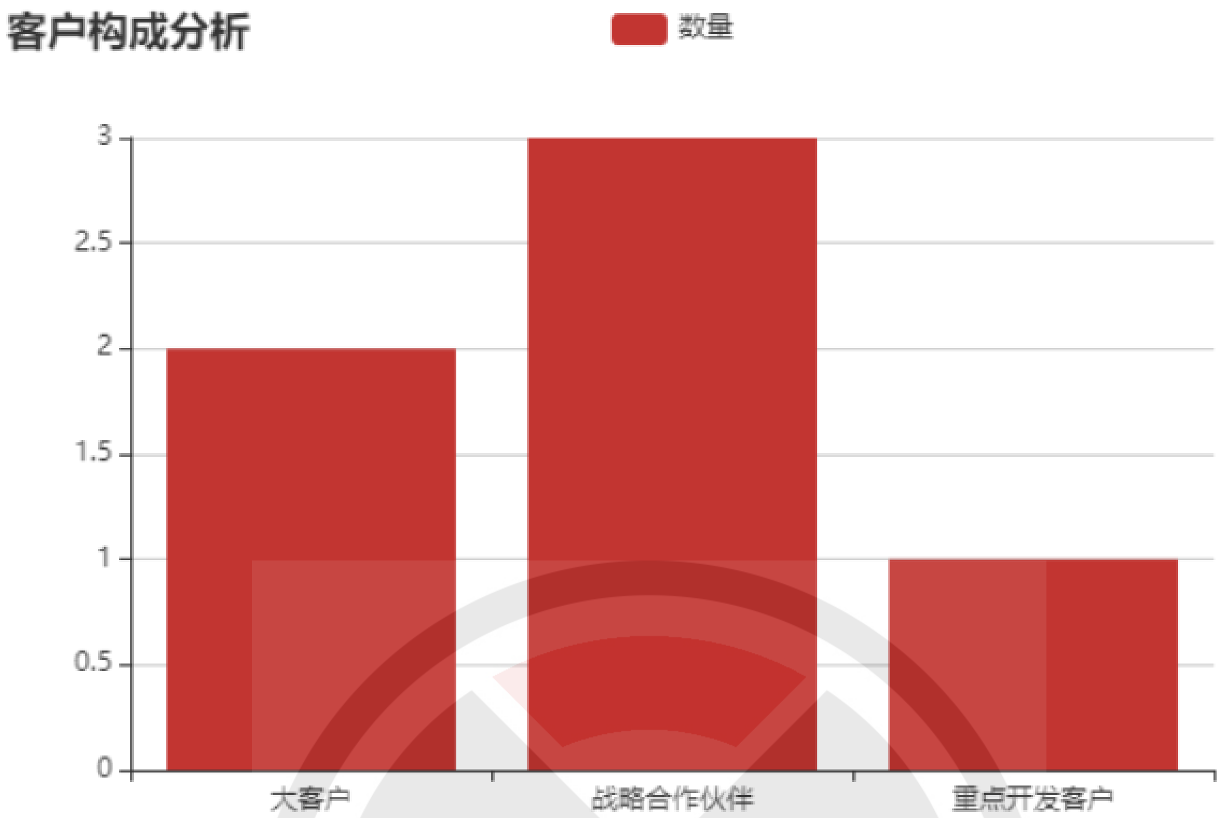
```
function searchCustomerContributions() {
    $("#dg").datagrid('load',{
        customerName:$("#customerName").val()
    }) ;
}
```

三 客户构成分析

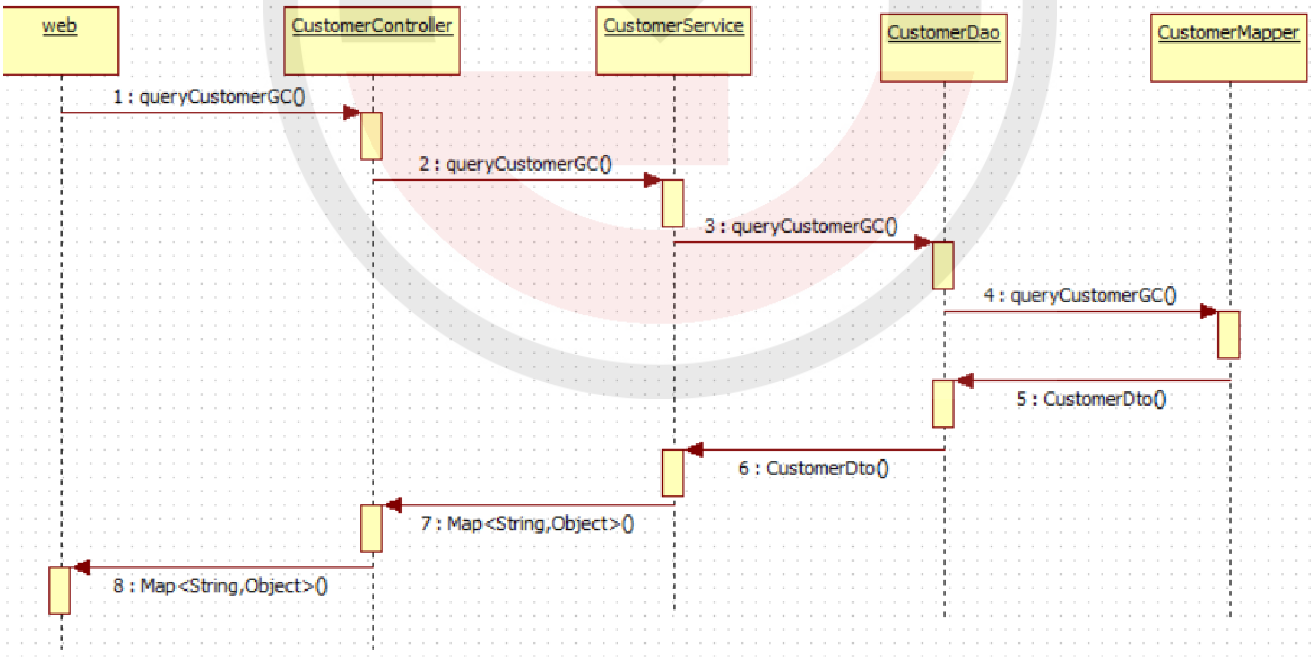
1 需求分析

将客户的构成情况以柱状图的形式来显示出来

2 页面原型



3 流程图



4 编写Controller

```
/**
     * 客户构成分析
     * @return
     */
@RequestMapping("queryCustomerGC")
@ResponseBody
public Map<String,Object> queryCustomerGC(){
    return  customerService.queryCustomerGC();
}
```

5 编写Service

```
public Map<String,Object> queryCustomerGC(){
    List<CustomerDto>  customerDtos = customerDao.queryCustomerGC();

    /**
         * 准备 X坐标的数组 和 Y坐标的数组
         *
         */
    String[] levels = null;

    Integer[] counts = null;

    Map<String,Object> map  = new HashMap<>();

    map.put("code",300);

    if(customerDtos!=null&&customerDtos.size()>0){

        levels=new String[customerDtos.size()];
        counts=new Integer[customerDtos.size()];


        for (int i = 0; i <customerDtos.size() ; i++) {

            CustomerDto customerDto  = customerDtos.get(i);

            levels[i] = customerDto.getLevel();

            counts[i] = customerDto.getCount();
        }
        map.put("code",200);
    }

    map.put("levels",levels);
    map.put("counts",counts);

    return map;

}
```

6 编写Dao

```
        public List<CustomerDto> queryCustomerGC();
```

7 编写Mapper

```
<select id="queryCustomerGC" resultType="customerDto">
    select level,count(*) as count from
    t_customer
    WHERE is_valid = 1
    GROUP BY level
</select>
```

8 编写Dto

```
public class CustomerDto {

    private String name;
    private Integer total;

    private String level;
    private Integer count;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getTotal() {
        return total;
    }

    public void setTotal(Integer total) {
        this.total = total;
    }

    public String getLevel() {
        return level;
    }

    public void setLevel(String level) {
        this.level = level;
    }

    public Integer getCount() {
        return count;
    }

    public void setCount(Integer count) {
        this.count = count;
    }
}
```

9 前台js

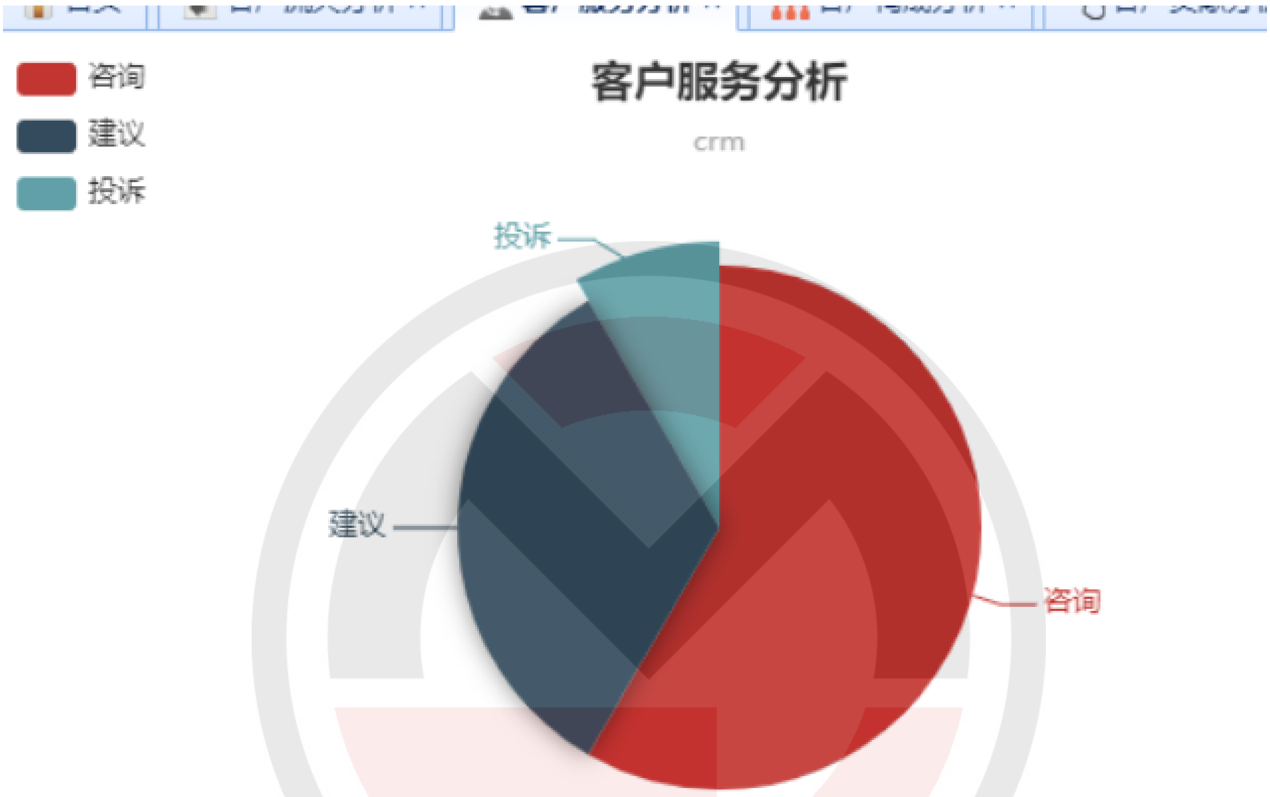
```
$(function () {
    loadCustomerGC()
})

function loadCustomerGC() {
    $.ajax({
        type: "post",
        url: ctx + "/customer/queryCustomerGC",
        dataType: "json",
        success: function (data) {
            if (data.code == 200) {
                var levels= data.levels;
                var counts= data.counts;
                var myChart = echarts.init(document.getElementById('main'));
                var option = {
                    title: {
                        text: '客户构成分析 '
                    },
                    tooltip: {},
                    legend: {
                        data: ['数量']
                    },
                    xAxis: {
                        data: levels
                    },
                    yAxis: {},
                    series: [{
                        name: '数量',
                        type: 'bar',
                        data: counts
                    }]
                };
                myChart.setOption(option);
            }
        }
    })
}
```

四 客户服务分析

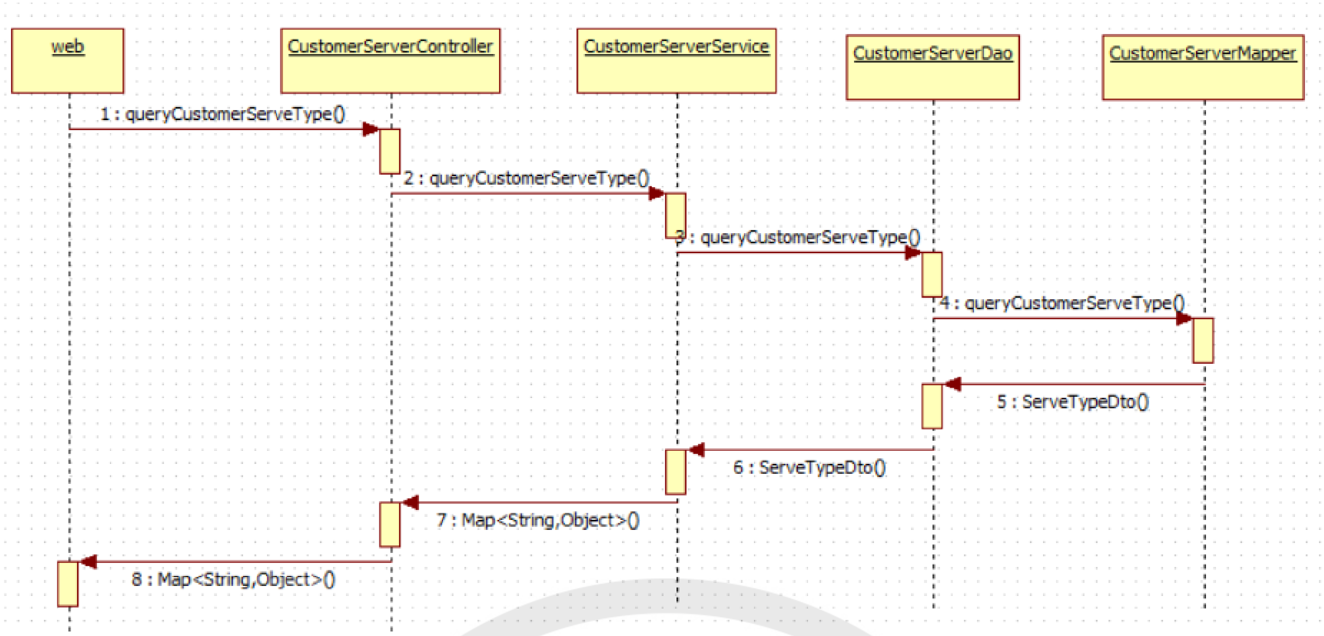
1 需求分析

将客户的服务情况以柱状图的形式来显示出来

2 页面原型



3 流程图



4 编写Controller

```
        @ResponseBody
        @RequestMapping("queryCustomerServeType")
        public Map<String,Object> queryCustomerServeType(){
            return customerServeService.queryCustomerServeType();
        }
```

5 编写Service

```
        public Map<String,Object> queryCustomerServeType(){
            List<ServeTypeDto> list = customerServeDao.queryCustomerServeType();
            String[] types = null;
            ServeTypeDto[] serveTypeDtos = null;
            Map<String,Object> map = new HashMap<>();
            map.put("code",300);

            if(list!=null&& list.size()>0){
                types = new String[list.size()];
                serveTypeDtos = new ServeTypeDto[list.size()];
                for (int i = 0; i <list.size() ; i++) {
                    ServeTypeDto serveTypeDto = list.get(i);
                    types[i] = serveTypeDto.getName();
                    serveTypeDtos[i] = serveTypeDto;
                }
                map.put("code",200);
            }
            map.put("types",types);
            map.put("datas",serveTypeDtos);
            System.out.println(JSON.toJSON(map));
            return  map;
        }
```

6 编写Dao

```
        public List<ServeTypeDto> queryCustomerServeType();
```

7 编写Mapper

```
<select id="queryCustomerServeType" resultType="serveTypeDto">

    select serve_type as name , count(*) as value
    from t_customer_serve
    where is_valid = 1
    group by serve_type
</select>
```

8 编写Dto

```
public class ServeTypeDto {


    private String name;
    private Integer value;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getValue() {
        return value;
    }

    public void setValue(Integer value) {
        this.value = value;
    }
}
```

9 前台js

```
$(function () {
    loadCustomerServeType()
})
```

```
function loadCustomerServeType() {
    $.ajax({
        type:"post",
        url:ctx + "/customer_serve/queryCustomerServeType",
        dataType:"json",
        success:function (data) {

            if(data.code==200){
                //types={退款，售后支持}
                var types = data.types;
                //datas = {{name: 退款, value : 1},{{name: 售后支持, value : 3}}
                var datas = data.datas;
                //基于准备好的dom，初始化echarts实例
                var chart = echarts.init(document.getElementById("main"));
                var option = {
                    title : {
                        text: '客户服务分析',
                        subtext: 'crm',
                        x:'center'
                    },
                    tooltip : {
                        trigger: 'item',
                        formatter: "{a} <br/>{b} : {c} ({d}%)"
                    },
                    legend: {
                        orient: 'vertical',
                        left:'left',
                        data: types

                    },
                    series : [
                        {
                            name: '服务类型',
                            type: 'pie',
                            radius : '55%',
                            center: ['50%', '50%'],
                            data: datas,
                            itemStyle: {
                                emphasis: {
                                    shadowBlur: 10,
                                    shadowOffsetX: 0,
                                    shadowColor: 'rgba(0, 0, 0, 0.5)'
                                }
                            }
                        }
                    ]
                };
                //设置图表的配置项和数据项
                chart.setOption(option);
            }else{
                $.messager.alert("crm","暂无数据","info");
            }
```

```
                }
            })
        }
```