* 学习目标

* 能够掌握SSM无配置的搭建

* 能够掌握SSM综合之AOP日志

* 能够理解Echarts的概述

* 能够使用Echarts的饼图展现出员工男女比例

* 能够使用Echarts的柱状图展现出员工男女比例

----

* 能够掌握SSM无配置的搭建

 * 复习SSM集成：[04-springmvc03]()

```
* SSM无配置的搭建
 * 核心配置类
* SpringConfiguration
@Configuration
@ComponentScan(value = "com.lg",
 excludeFilters ={
@ComponentScan.Filter(type = FilterType.ANNOTATION,
                      classes = Controller.class),
@ComponentScan.Filter(type = FilterType.ASSIGNABLE_TYPE,
                      classes = SpringMVCConfig.class)})
@EnableAspectJAutoProxy
@Import(SMConfiguration.class)
public class SpringConfiguration {
}

* SMConfiguration
@Configuration
@EnableTransactionManagement
@MapperScan("com.lg.dao")
@PropertySource("classpath:db.properties")
public class SMConfiguration {
    @Value("${lg.driver}")
```

```java
    private String driver;
    @Value("${lg.url}")
    private String url;
    @Value("${lg.username}")
    private String username;
    @Value("${lg.password}")
    private String password;
    @Bean
    @Scope(ConfigurableBeanFactory.SCOPE_SINGLETON)
    public DataSource dataSource(){
        DruidDataSource ds=new DruidDataSource();
        ds.setDriverClassName(driver);
        ds.setUrl(url);
        ds.setUsername(username);
        ds.setPassword(password);
        return ds;
    }
    @Bean
    public SqlSessionFactoryBean sqlSessionFactoryBean(DataSource dataSource){
        SqlSessionFactoryBean sqlSessionFactoryBean=new SqlSessionFactoryBean(
        sqlSessionFactoryBean.setDataSource(dataSource);
        PathMatchingResourcePatternResolver resolver=new PathMatchingResourcePa
        try {
            Resource[] resources = resolver.getResources("classpath:com/lg/dao/
            sqlSessionFactoryBean.setMapperLocations(resources);
        } catch (IOException e) {
            e.printStackTrace();
        }
        sqlSessionFactoryBean.setTypeAliasesPackage("com.lg.bean");
        return sqlSessionFactoryBean;
    }
    @Bean
    public TransactionManager transactionManager(DataSource dataSource){
        DataSourceTransactionManager transactionManager=new DataSourceTransacti
        transactionManager.setDataSource(dataSource);
        return transactionManager;
    }
}

* SpringMVCConfig
```

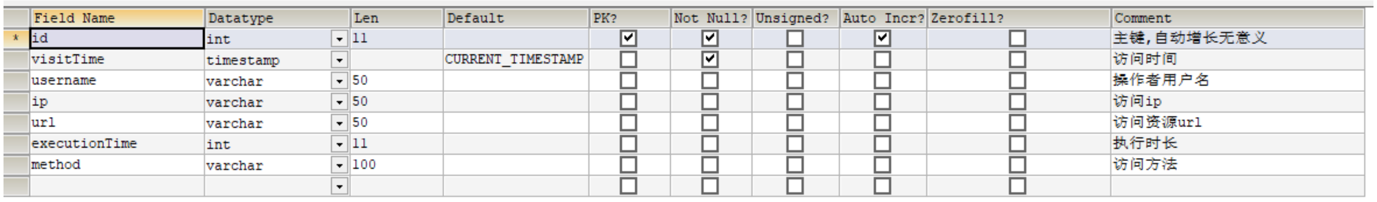
```
@Configuration
@ComponentScan(value = "com.lg", useDefaultFilters = false, includeFilters =
        {@ComponentScan.Filter(type = FilterType.ANNOTATION,classes = Controlle
@EnableAspectJAutoProxy
@EnableWebMvc
public class SpringMVCConfig implements WebMvcConfigurer{
    /**
     * 添加视图解析器
     */
    @Bean
    public InternalResourceViewResolver internalResourceViewResolver() {
        InternalResourceViewResolver internalResourceViewResolver =
            new InternalResourceViewResolver();
        internalResourceViewResolver.setPrefix("/WEB-INF/jsps/");
        internalResourceViewResolver.setSuffix(".jsp");
        return internalResourceViewResolver;
    }

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry.addResourceHandler("/css/**").addResourceLocations("/css/");
        registry.addResourceHandler("/image/**").addResourceLocations("/image/'
        registry.addResourceHandler("/js/**").addResourceLocations("/js/");
    }
}
* WebConfig

public class WebConfig
    extends AbstractAnnotationConfigDispatcherServletInitializer {
    /**
     * @return
     * spring 根容器
     */
    @Override
    protected Class<?>[] getRootConfigClasses() {
        return new Class[]{SpringConfiguration.class};
    }

    /**
     * @return
     * Spring mvc容器
     */
    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[]{SpringMVCConfig.class};
    }

    //DispatcherServlet映射,从"/"开始
    @Override
    protected String[] getServletMappings() {
        return new String[]{"/"};
    }
}
* 控制器
@Controller
@RequestMapping("/emp")
public class TransferEmployeeController {
    @Autowired
    private TransferEmployeeService transferEmployeeService;
    @RequestMapping("/transfer")
    public String transfer(String empno,String job,Integer odid,Integer ndid){
        Map<String,Object> params=new HashMap<>();
        params.put("empno",empno);
        params.put("job",job);
        params.put("odid",odid);
        params.put("ndid",ndid);
        transferEmployeeService.transferEmployee(params);
        return "success";
    }
}

* 单元测试
 @Test
 public void test17() throws Exception {
     String result = mockMvc.perform(delete("/emp/transfer")
                .param("empno","LG008")
                .param("job","前端讲师")
                .param("odid","2")
                .param("ndid","1")).
     andExpect(status().isOk()).
```

```
                andDo(print()).andReturn().getResponse().getContentAsString();
    System.out.println(result);
}
```

* 能够掌握SSM综合之AOP日志

 * 复习AOP:[03-spring03]

  * 创建表syslog



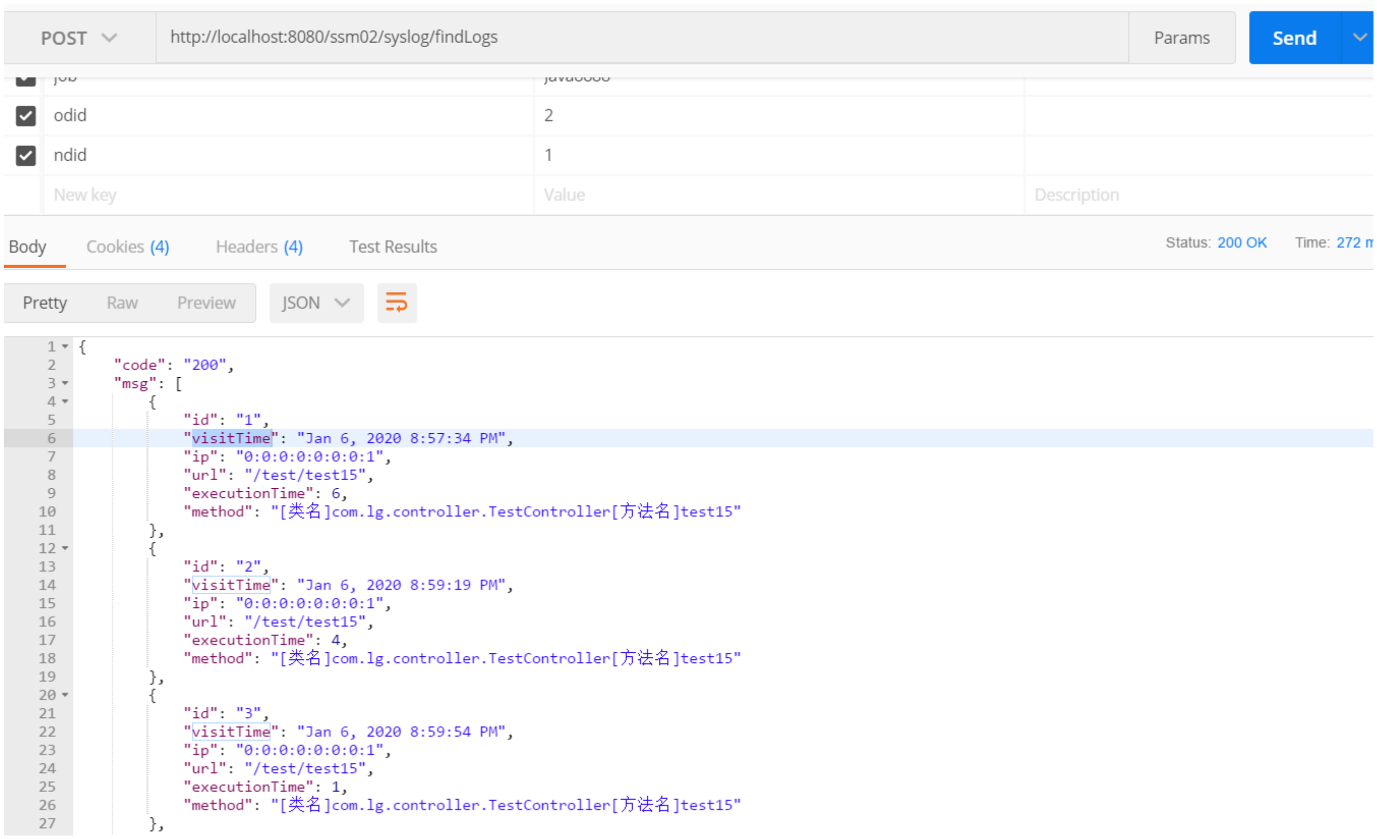
	Field Name	Datatype	Len	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
*	id	int	11		✓	✓	☐	✓	☐	主键,自动增长无意义
	visitTime	timestamp		CURRENT_TIMESTAMP	☐	✓	☐	☐	☐	访问时间
	username	varchar	50		☐	☐	☐	☐	☐	操作者用户名
	ip	varchar	50		☐	☐	☐	☐	☐	访问ip
	url	varchar	50		☐	☐	☐	☐	☐	访问资源url
	executionTime	int	11		☐	☐	☐	☐	☐	执行时长
	method	varchar	100		☐	☐	☐	☐	☐	访问方法
					☐	☐	☐	☐	☐	

```
* 案例:
 * 表syslog
 CREATE TABLE syslog (
  id int(11) NOT NULL AUTO_INCREMENT COMMENT '主键,自动增长无意义',
  visitTime timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
  ON UPDATE CURRENT_TIMESTAMP COMMENT '访问时间',
  username varchar(50) DEFAULT NULL COMMENT '操作者用户名',
  ip  varchar(50) DEFAULT NULL COMMENT '访问ip',
  url varchar(50) DEFAULT NULL COMMENT '访问资源url',
  executionTime int(11) DEFAULT NULL COMMENT '执行时长',
  method varchar(100) DEFAULT NULL COMMENT '访问方法',
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
 * 实体类
@Data
@AllArgsConstructor
@NoArgsConstructor
public class SysLog {
    private String id;
    private Date visitTime;
    private String visitTimeStr;
    private String username;
```

```java
    private String ip;
    private String url;
    private Long executionTime;
    private String method;
}
* dao
public interface SysLogDao {
    @Select("select * from syslog")
    @Results({ @Result(id=true,column="id",property="id"),
            @Result(column="visitTime",property="visitTime"),
            @Result(column="ip",property="ip"),
            @Result(column="url",property="url"),
            @Result(column="executionTime",property="executionTime"),
            @Result(column="method",property="method"),
            @Result(column="username",property="username") })
     List<SysLog> findAll() throws Exception;

    @Insert("insert into syslog(visitTime,username,ip,url,executionTime,method)
            values(#{visitTime},#{username},#{ip},#{url},#{executionTime},#{met
     void save(SysLog log) throws Exception;
}
* service
public interface SysLogService {
    List<SysLog> findAll() throws Exception;
    void save(SysLog log) throws Exception;
}
@Service
public class SysLogServiceImpl implements SysLogService {
    @Autowired
    private SysLogDao sysLogDao;
    @Override
    public List<SysLog> findAll() throws Exception {
        return sysLogDao.findAll();
    }
    @Override
    public void save(SysLog log) throws Exception {
        sysLogDao.save(log);
    }
}

```

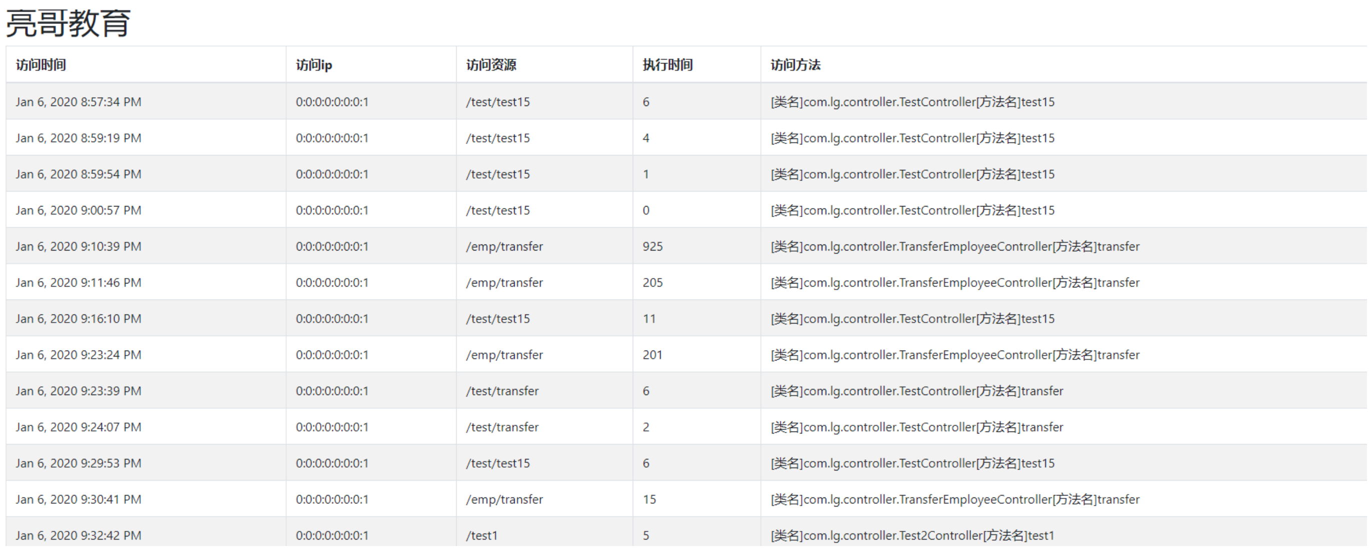
```java
* 忽略LogAop注解
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
public @interface IgnoreLog {
}

* Controller
@RestController
@RequestMapping("/syslog")
public class SysLogController {
    @Autowired
    private SysLogService sysLogService;
    @RequestMapping("/findLogs")
    public SysLogVo findSysLog(){
        List<SysLog> sysLogs = null;
        SysLogVo sysLogVo=new SysLogVo();
        try {
            sysLogs = sysLogService.findAll();
            sysLogVo.setCode("200");
            sysLogVo.setMsg(sysLogs);
        } catch (Exception e) {
            sysLogVo.setCode("500");
            sysLogVo.setMsg("服务器错误");
        }
        return sysLogVo;
    }
}

* 切面
@Controller
@Aspect
public class LogAop {
    @Autowired
    private HttpServletRequest request;
    @Autowired
    private SysLogService sysLogService;

    private Long startTime;// 访问时间
    @Pointcut("execution(* com.lg.controller.*.*(..))")
    public void pointcut(){
    }
    @Around("pointcut()")
    public Object recordSysLog(ProceedingJoinPoint joinPoint) throws Throwable
        startTime=System.currentTimeMillis();
       // 执行业务方法
        Object result = joinPoint.proceed();
        try{
            handle(joinPoint);
        }catch (Exception e){
            e.printStackTrace();
        }
        return result;
    }

    private void handle(ProceedingJoinPoint jp) throws Exception {
        // 获取访问的类
        Class executionClass=jp.getTarget().getClass();
        // 获取访问的方法的名称
        MethodSignature msig = (MethodSignature) jp.getSignature();
        Method executionMethod=executionClass.getMethod(msig.getName(),msig.get
        IgnoreLog ignoreLogAnnotation = executionMethod.getAnnotation(IgnoreLog
        if(ignoreLogAnnotation!=null){
            return;
        }
        //获取类上的@RequestMapping对象
        Annotation classAnnotation =  executionClass.getAnnotation(RequestMappi
        RequestMapping methodAnnotation = executionMethod.getAnnotation(Request
        if(classAnnotation==null && methodAnnotation==null){
            return;
        }
        String url = null;
        if(classAnnotation!=null){
            RequestMapping classAnnotation1=(RequestMapping)classAnnotation;
            if(methodAnnotation!=null){
                // 它的值应该是类上的@RequestMapping的value+方法上的 @RequestMapp
                url = classAnnotation1.value()[0] + methodAnnotation.value()[0
            }
        }else{
            if(methodAnnotation!=null){
                // 方法上的 @RequestMapping的value
                url = methodAnnotation.value()[0];
            }
        }
        SysLog sysLog = new SysLog();
        // 获取访问时长
        Long executionTime = System.currentTimeMillis() - startTime;
        // 将sysLog对象属性封装
        sysLog.setExecutionTime(executionTime);
        sysLog.setUrl(url);
        // 获取ip
        String ip = request.getRemoteAddr();
        sysLog.setIp(ip);
        // 用户名从session中获取
        Object username=request.getSession().getAttribute("username");
        if(username!=null){
            sysLog.setUsername((String) username);
        }
        sysLog.setMethod("[类名]" + executionClass.getName() + "[方法名]" + exec
        sysLog.setVisitTime(new Date(startTime));
        // 调用Service，调用dao将sysLog insert数据库
        sysLogService.save(sysLog);
    }

}


* 测试
@RequestMapping("/test15")
 public String test15(){
        System.out.println("hello test15");
        return "success";
}
* http://localhost:8080/ssm02/test/test15
* 界面
<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="fal
<html>
<head>
    <meta charset="UTF-8">
    <title>日志列表</title>
    <link rel="stylesheet" href="${pageContext.request.contextPath}/css/bootstr
    <script src="${pageContext.request.contextPath}/js/jquery-3.3.1.js"></scrip
    <script src="${pageContext.request.contextPath}/js/popper.min.js"></script
    <script src="${pageContext.request.contextPath}/js/bootstrap.js"></script>
    <script>
        var url="/ssm02/syslog/findLogs";
        $.ajax({
            type:"POST",
            url:url,
            contentType:"application/json;charset=utf-8",
            success:function(result,status){
                var content = JSON.parse(JSON.stringify(result));
                if(content.code==200){
                    var tbody=document.getElementsByTagName("tbody")[0];
                    tbody.innerHTML="";
                    for(var index in content.msg){
                        var syslog=content.msg[index];
                        tbody.innerHTML+="<tr>\n" +
                            "            <td>"+syslog.visitTime+"</td>\n" +
                            "            <td>"+syslog.ip+"</td>\n" +
                            "            <td>"+syslog.url+"</td>\n" +
                            "            <td>"+syslog.executionTime+"</td>\n" +
                            "            <td>"+syslog.method+"</td>\n" +
                            "        </tr>";
                    }
                }
            },
            error:function(result){
                alert("请求数据失败")
            }
        });

    </script>
</head>
<body>
<div class="container-fluid">
    <h1>亮哥教育</h1>
    <table class="table table-bordered table-striped">
        <thead>
        <tr>
            <th>访问时间</th>
```

```html
                <th>访问ip</th>
                <th>访问资源</th>
                <th>执行时间</th>
                <th>访问方法</th>
            </tr>
            </thead>
            <tbody>
            <tr>
                <td>Jan 6, 2020 8:57:34 PM</td>
                <td>0:0:0:0:0:0:0:1</td>
                <td>/test/test15</td>
                <td>6</td>
                <td>[类名]com.lg.controller.TestController[方法名]test15</td>
            </tr>
            </tbody>
        </table>
</div>
</body>
</html>
```



odid	2	
ndid	1	
New key	Value	Description

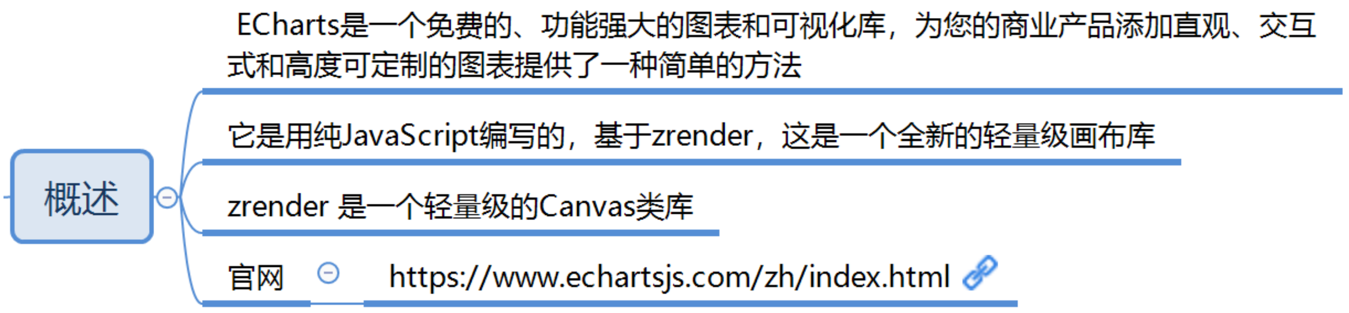
```json
{
    "code": "200",
    "msg": [
        {
            "id": "1",
            "visitTime": "Jan 6, 2020 8:57:34 PM",
            "ip": "0:0:0:0:0:0:0:1",
            "url": "/test/test15",
            "executionTime": 6,
            "method": "[类名]com.lg.controller.TestController[方法名]test15"
        },
        {
            "id": "2",
            "visitTime": "Jan 6, 2020 8:59:19 PM",
            "ip": "0:0:0:0:0:0:0:1",
            "url": "/test/test15",
            "executionTime": 4,
            "method": "[类名]com.lg.controller.TestController[方法名]test15"
        },
        {
            "id": "3",
            "visitTime": "Jan 6, 2020 8:59:54 PM",
            "ip": "0:0:0:0:0:0:0:1",
            "url": "/test/test15",
            "executionTime": 1,
            "method": "[类名]com.lg.controller.TestController[方法名]test15"
        },
```



亮哥教育

访问时间	访问ip	访问资源	执行时间	访问方法
Jan 6, 2020 8:57:34 PM	0:0:0:0:0:0:0:1	/test/test15	6	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 8:59:19 PM	0:0:0:0:0:0:0:1	/test/test15	4	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 8:59:54 PM	0:0:0:0:0:0:0:1	/test/test15	1	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 9:00:57 PM	0:0:0:0:0:0:0:1	/test/test15	0	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 9:10:39 PM	0:0:0:0:0:0:0:1	/emp/transfer	925	[类名]com.lg.controller.TransferEmployeeController[方法名]transfer
Jan 6, 2020 9:11:46 PM	0:0:0:0:0:0:0:1	/emp/transfer	205	[类名]com.lg.controller.TransferEmployeeController[方法名]transfer
Jan 6, 2020 9:16:10 PM	0:0:0:0:0:0:0:1	/test/test15	11	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 9:23:24 PM	0:0:0:0:0:0:0:1	/emp/transfer	201	[类名]com.lg.controller.TransferEmployeeController[方法名]transfer
Jan 6, 2020 9:23:39 PM	0:0:0:0:0:0:0:1	/test/transfer	6	[类名]com.lg.controller.TestController[方法名]transfer
Jan 6, 2020 9:24:07 PM	0:0:0:0:0:0:0:1	/test/transfer	2	[类名]com.lg.controller.TestController[方法名]transfer
Jan 6, 2020 9:29:53 PM	0:0:0:0:0:0:0:1	/test/test15	6	[类名]com.lg.controller.TestController[方法名]test15
Jan 6, 2020 9:30:41 PM	0:0:0:0:0:0:0:1	/emp/transfer	15	[类名]com.lg.controller.TransferEmployeeController[方法名]transfer
Jan 6, 2020 9:32:42 PM	0:0:0:0:0:0:0:1	/test1	5	[类名]com.lg.controller.Test2Controller[方法名]test1

* 能够理解Echarts的概述



ECharts是一个免费的、功能强大的图表和可视化库，为您的商业产品添加直观、交互式和高度可定制的图表提供了一种简单的方法

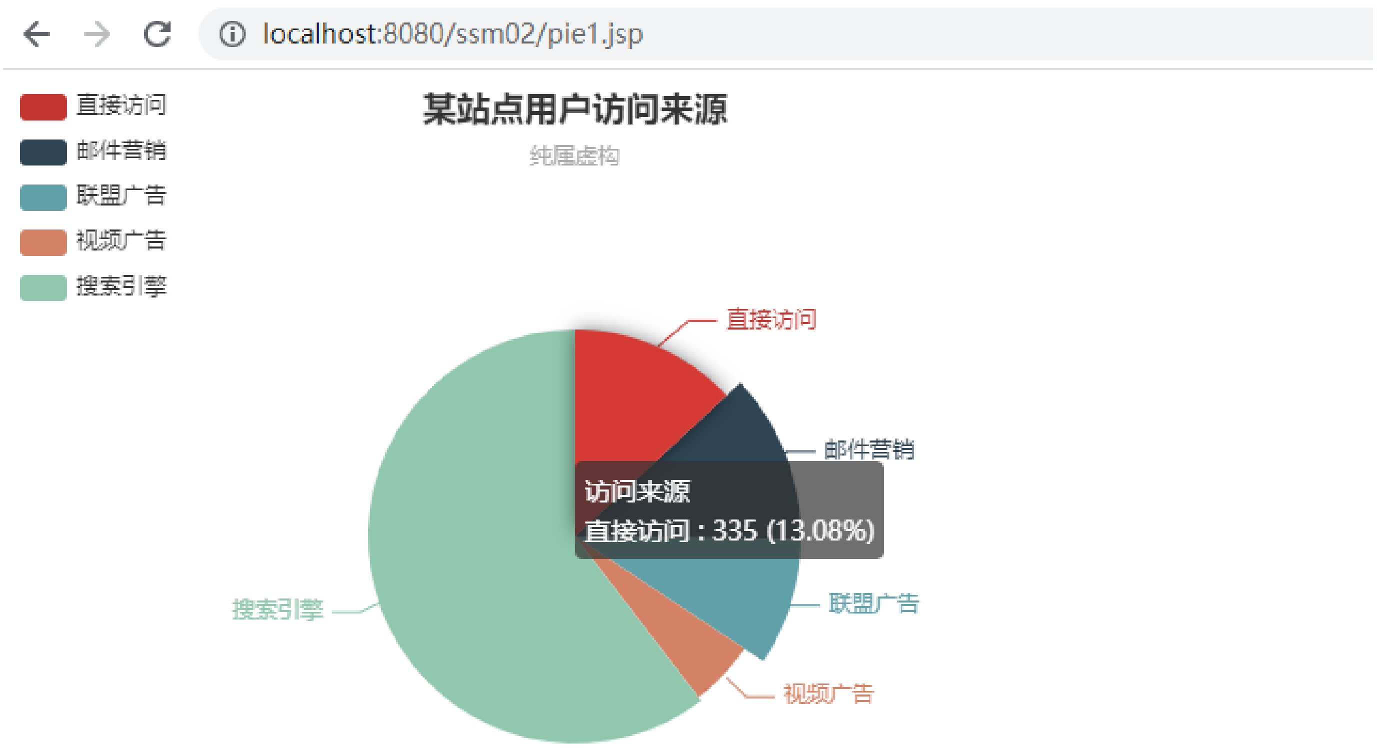
它是用纯JavaScript编写的，基于zrender，这是一个全新的轻量级画布库

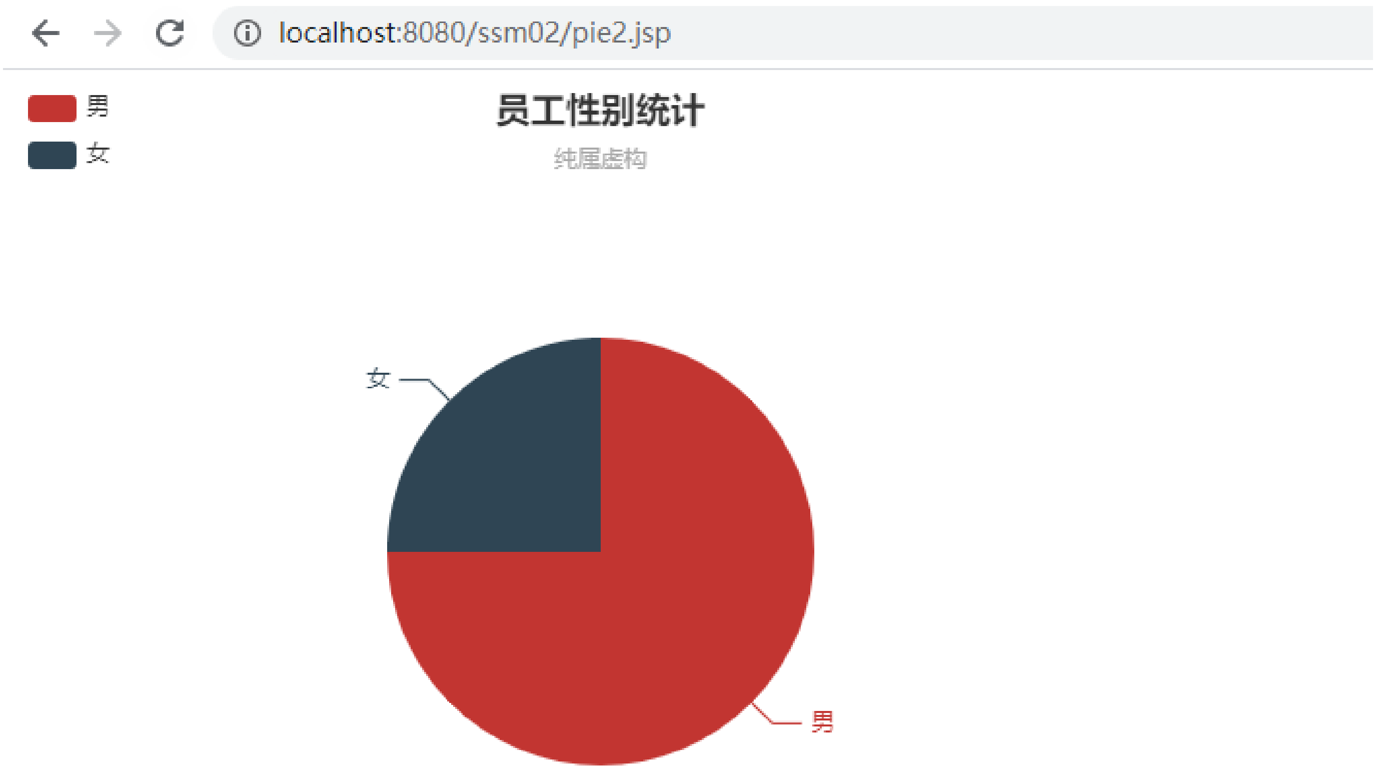
zrender 是一个轻量级的Canvas类库

官网 https://www.echartsjs.com/zh/index.html

* https://www.echartsjs.com/zh/index.html

* 能够使用Echarts的饼图展现出员工男女比例





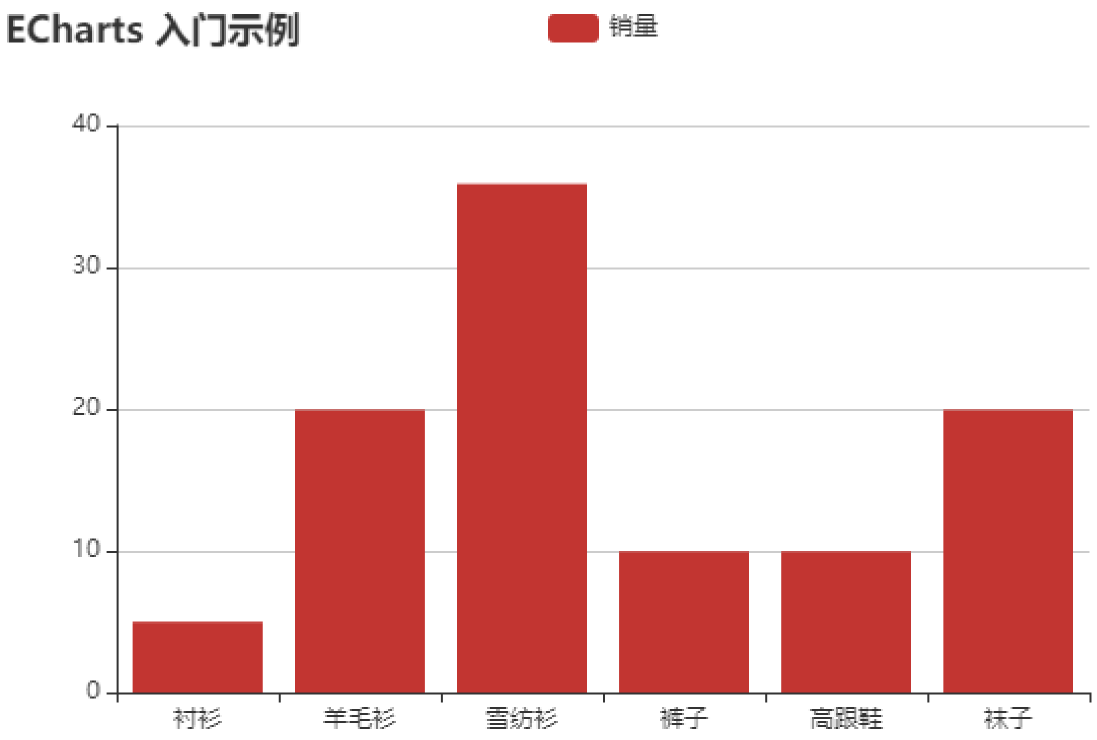
```
* 案例一（官网例子）
* 下载echart依赖库
* 代码
```

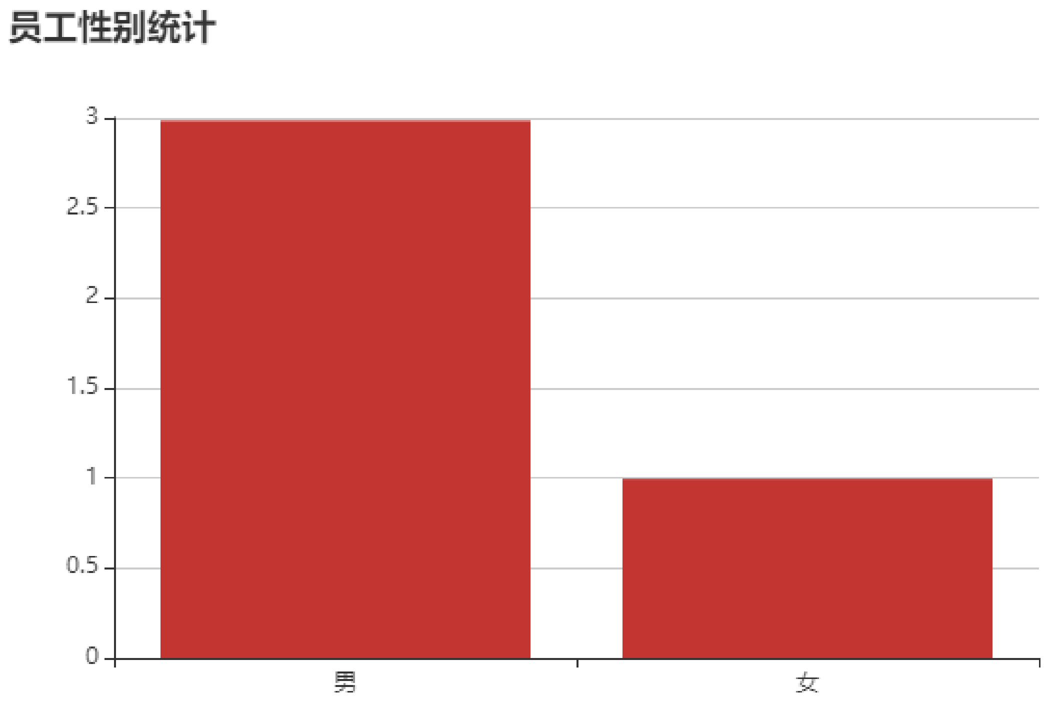
```
<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="fal
<html>
<head>
    <title>pie图演示</title>
    <meta charset="utf-8">
    <script src="${pageContext.request.contextPath}/js/jquery-3.3.1.js"></scrip
    <!-- 引入 ECharts 文件 -->
    <script src="${pageContext.request.contextPath}/js/echarts.js"></script>
    <script>
        $(function () {
            // 基于准备好的dom，初始化echarts实例
            var myChart = echarts.init(document.getElementById('main'));
            // 指定图表的配置项和数据
            var option = {
                title: {
                    text: '某站点用户访问来源',
                    subtext: '纯属虚构',
                    left: 'center'
                },
                tooltip: {
                    trigger: 'item',
                    formatter: '{a} <br/>{b} : {c} ({d}%)'
                },
                legend: {
                    orient: 'vertical',
                    left: 'left',
                    data: ['直接访问', '邮件营销', '联盟广告', '视频广告', '搜索引
                },
                series: [
                    {
                        name: '访问来源',
                        type: 'pie',
                        radius: '55%',
                        center: ['50%', '60%'],
                        data: [
                            {value: 335, name: '直接访问'},
                            {value: 310, name: '邮件营销'},
                            {value: 234, name: '联盟广告'},
                            {value: 135, name: '视频广告'},
                            {value: 1548, name: '搜索引擎'}
                        ],
                        emphasis: {
                            itemStyle: {
                                shadowBlur: 10,
                                shadowOffsetX: 0,
                                shadowColor: 'rgba(0, 0, 0, 0.5)'
                            }
                        }
                    }
                ]
            };
            // 使用刚指定的配置项和数据显示图表。
            myChart.setOption(option);
        });
    </script>
</head>
<body>
    <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
    <div id="main" style="width: 600px;height:400px;"></div>
</body>
</html>

* 案例二：（统计员工男女性别）
* 实体
@Data
@AllArgsConstructor
@NoArgsConstructor
public class ResResult {
    private String code;
    private Object msg;
}

* dao
public interface EmployeeDao {
    @Select("SELECT COUNT(*) FROM employee WHERE sex=#{sex}")
    Integer selectEmpCountBySex(String sex);
}
* service
public interface EmployeeService {
    List<Integer> queryEmployeeSexCount();
```

```
}
@Service
public class EmployeeServiceImpl implements EmployeeService {
    @Autowired
    private EmployeeDao employeeDao;
    @Transactional
    @Override
    public List<Integer> queryEmployeeSexCount() {
        List<Integer> list=new ArrayList<>();
        list.add(employeeDao.selectEmpCountBySex("男"));
        list.add(employeeDao.selectEmpCountBySex("女"));
        return list;
    }
}

* Controller
@RestController
@RequestMapping("/emp")
public class EmployeeController {
    @Autowired
    private EmployeeService employeeService;
    @RequestMapping("/sexCount")
    public ResResult queryEmployeeSexCount(){
        ResResult resResult=new ResResult();
        List<Integer> list = employeeService.queryEmployeeSexCount();
        resResult.setCode("200");
        resResult.setMsg(list);
        return resResult;
    }
}
* 测试
 @Test
public void test18() throws Exception {
  String result = mockMvc.perform(delete("/emp/sexCount")).
                andExpect(status().isOk()).
                andDo(print()).andReturn().getResponse().getContentAsString();
 System.out.println(result);
}
* 页面
<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="fal
```

```
<html>
<head>
    <title>pie图演示</title>
    <meta charset="utf-8">
    <script src="${pageContext.request.contextPath}/js/jquery-3.3.1.js"></scrip
    <!-- 引入 ECharts 文件 -->
    <script src="${pageContext.request.contextPath}/js/echarts.js"></script>
    <script>
        $(function () {
            // 基于准备好的dom，初始化echarts实例
            var myChart = echarts.init(document.getElementById('main'));
            // 指定图表的配置项和数据
            var option = {
                title: {
                    text: '员工性别统计',
                    subtext: '纯属虚构',
                    left: 'center'
                },
                tooltip: {
                    trigger: 'item',
                    formatter: '{a} <br/>{b} : {c} ({d}%)'
                },
                legend: {
                    orient: 'vertical',
                    left: 'left',
                    data: ['男', '女']
                },
                series: [
                    {
                        name: '性别',
                        type: 'pie',
                        radius: '55%',
                        center: ['50%', '60%'],
                        data: [
                        ],
                        emphasis: {
                            itemStyle: {
                                shadowBlur: 10,
                                shadowOffsetX: 0,
                                shadowColor: 'rgba(0, 0, 0, 0.5)'
                                }
                            }
                        }
                    ]
                };
                // 使用刚指定的配置项和数据显示图表。
                myChart.setOption(option);
                var url="/ssm02/emp/sexCount";
                $.ajax({
                    type:"POST",
                    url:url,
                    contentType:"application/json;charset=utf-8",
                    success:function(result,status){
                        var content = JSON.parse(JSON.stringify(result));
                        if(content.code==200){
                            // 填入数据
                            myChart.setOption({
                                series: [{
                                    data: [
                                        {value: content.msg[0], name: '男'},
                                        {value: content.msg[1], name: '女'},
                                    ]
                                }]
                            });
                        }
                    },
                    error:function(result){
                        alert("请求数据失败")
                    }
                });
            });
        </script>
</head>
<body>
    <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
    <div id="main" style="width: 600px;height:400px;"></div>
</body>
</html>

```

* 能够使用Echarts的柱状图展现出员工男女比例





```
* 案例一
<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="fal
<html>
<head>
    <title>柱状图演示</title>
    <meta charset="utf-8">
    <script src="${pageContext.request.contextPath}/js/jquery-3.3.1.js"></scrip
    <!-- 引入 ECharts 文件 -->
    <script src="${pageContext.request.contextPath}/js/echarts.js"></script>
    <script>
        $(function () {
            // 基于准备好的dom，初始化echarts实例
            var myChart = echarts.init(document.getElementById('main'));
            // 指定图表的配置项和数据
            var option = {
                title: {
                    text: 'ECharts 入门示例'
                },
                tooltip: {},
                legend: {
                    data:['销量']
                },
                xAxis: {
                    data: ["衬衫","羊毛衫","雪纺衫","裤子","高跟鞋","袜子"]
                },
                yAxis: {},
                series: [{
                    name: '销量',
                    type: 'bar',
                    data: [5, 20, 36, 10, 10, 20]
                }]
            };
            // 使用刚指定的配置项和数据显示图表。
            myChart.setOption(option);
        });
    </script>
</head>
<body>
    <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
    <div id="main" style="width: 600px;height:400px;"></div>
</body>
</html>

* 案例二
<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="fa
```

```
<html>
<head>
    <title>柱状图演示</title>
    <meta charset="utf-8">
    <script src="${pageContext.request.contextPath}/js/jquery-3.3.1.js"></scrip
    <!-- 引入 ECharts 文件 -->
    <script src="${pageContext.request.contextPath}/js/echarts.js"></script>
    <script>
        $(function () {
            // 基于准备好的dom，初始化echarts实例
            var myChart = echarts.init(document.getElementById('main'));
            // 指定图表的配置项和数据
            var option = {
                title: {
                    text: '员工性别统计'
                },
                tooltip: {},
                legend: {
                    data:['性别统计']
                },
                xAxis: {
                    data: ["男","女"]
                },
                yAxis: {},
                series: [{
                    name: '性别',
                    type: 'bar',
                    data: []
                }]
            };
            // 使用刚指定的配置项和数据显示图表。
            myChart.setOption(option);
            var url="/ssm02/emp/sexCount";
            $.ajax({
                type:"POST",
                url:url,
                contentType:"application/json;charset=utf-8",
                success:function(result,status){
                    var content = JSON.parse(JSON.stringify(result));
                    if(content.code==200){
                        // 填入数据
                        myChart.setOption({
                            series: [{
                                // 根据名字对应到相应的系列
                                name: '性别',
                                data: content.msg
                            }]
                        });
                    }
                },
                error:function(result){
                    alert("请求数据失败")
                }
            });
        });
    </script>
</head>
<body>
    <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
    <div id="main" style="width: 600px;height:400px;"></div>
</body>
</html>
```