《01(JavaEE综合练习)前台后台概念_》

前台和后台(对于开发者而言都一样只是功能不同而已)

数据库(数据都来源与数据库)--java(操作数据库)--网页(把数据放在网页上显示)

前台：

- 开发的技术,java,网页,数据库
- 用户角度
  - 面向所有用户
  - 功能:查询,新增

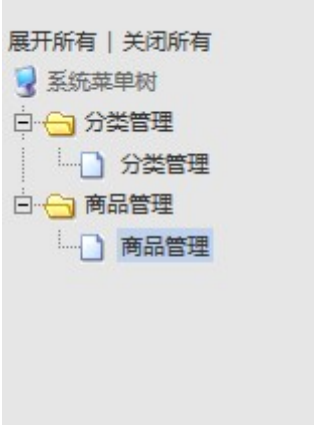
后台

- 开发的技术,java,网页,数据库
- 用户角度
  - 面向管理者
  - 功能:全面的

《02(JavaEE综合练习)页面介绍_》

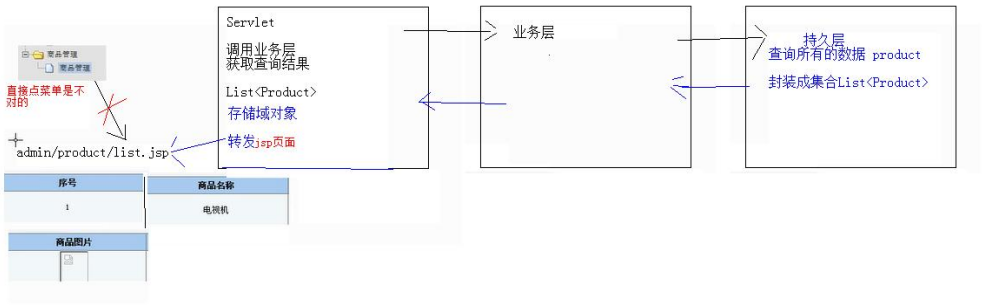
后台页面是framset框架做的

```
    <script type="text/javascript">

        d = new dTree('d');
        d.add('01',-1,'系统菜单树');
        d.add('0102','01','分类管理','','','mainFrame');
        d.add('010201','0102','分类管
理','${pageContext.request.contextPath}/admin/category/list.jsp','','mainFrame')
;
        d.add('0104','01','商品管理');
        d.add('010401','0104','商品管
理','${pageContext.request.contextPath}/admin/product/list.jsp','','mainFrame');
        document.write(d);

    </script>
```



《03(JavaEE综合练习)展示所有数据分层_》

展示全部商品数据



《04(JavaEE综合练习)展示所有数据服务器_》

```
//修改商品管理菜单,超链接改变，到达获取所有数据的Servlet
        d.add('010401','0104','商品管
理','${pageContext.request.contextPath}/findAll','','mainFrame');
```

dao层

```
/**
     * 方法findAll()
     * 查询所有商品数据
     * 返回集合 List<Product>
     */
    public List<Product> findAll() throws SQLException {//抛异常
        //编写sql语句
        String sql = "select * from product";
        //执行SQL语句 返回结果集
        return qr.query(sql, new BeanListHandler<Product>(Product.class));
    }
```

service层

```
/**
     * 方法findAll()
     * 查询所有商品数据
     * 返回集合
     */
    public List<Product> findAll() {
```

```
        List<Product> list = null;//提升作用域
        try {
            list = dao.findAll();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return list;
    }
```

web层

```
    /**
         * 调用业务层方法findAll()
         * 获取集合，存储域对象
         * 转发页面
         */
        //创建service对象调用方法接收返回值
        ProductService service = new ProductService();
        List<Product> list = service.findAll();
        //把就收的值存放request域中
        request.setAttribute("list", list);
        //获取转发器 转发属于服务器内部行为不出服务器 所以不需要写WEB应用名称 相对路径
        RequestDispatcher dispatcher =
request.getRequestDispatcher("/admin/product/list.jsp");
        //转发方法
        dispatcher.forward(request, response);
```

《05(JavaEE综合练习)展示所有数据页面_》

jsp页面 导入标签库jstl 用标签遍历形式配合EL取出数据

```
<!--商品展示
                        forEach标签属性 varStatus循环变量状态（自动存在page域中）
                                        count循环次数
                    -->
                    <c:forEach items="${requestScope.list}" var="product"
varStatus="vs">
                        <tr onmouseover="this.style.backgroundColor = '#CCCCCC'"
                            onmouseout="this.style.backgroundColor =
'#F5FAFE';">
                            <td style="CURSOR: hand; HEIGHT: 22px"
align="center"
                                width="18%">${vs.count}</td>
                            <td style="CURSOR: hand; HEIGHT: 22px"
align="center"
                                width="17%"><img width="40" height="45"
src="${product.pimage}"></td>
                            <td style="CURSOR: hand; HEIGHT: 22px"
align="center"
                                width="17%">${product.pname}</td>
                            <!-- 数据表 1表示热门，0表示非热门-->
                            <td style="CURSOR: hand; HEIGHT: 22px"
align="center"
                                width="17%">${product.shop_price}</td>
```

```
                            <td style="CURSOR: hand; HEIGHT: 22px"
align="center"
                                width="17%">${product.is_hot==1?"是":"不是"}</td>
                            <td align="center" style="HEIGHT: 22px"><a
                                    href="${ pageContext.request.contextPath
}/admin/product/edit.jsp">
                                <img

 src="${pageContext.request.contextPath}/images/i_edit.gif"
                                        border="0" style="CURSOR: hand">
                            </a></td>

                            <td align="center" style="HEIGHT: 22px"><a href="#">
<img

 src="${pageContext.request.contextPath}/images/i_del.gif"
                                    width="16" height="16" border="0"
style="CURSOR: hand">
                            </a></td>
                        </tr>
                    </c:forEach>
                    <!--商品展示结束-->
```

《06(JavaEE综合练习)删除商品的步骤_》

删除商品

实现步骤

- 点击删除图片,出现提示框(JS函数 confirm)
- 获取点击商品的**主键**数据
- 主键提交服务器Servlet
- 主键传递业务层
- 业务层将主键传递dao
- 执行SQL语句删除
- Servlet客户端响应，看到所有数据，转发/findAll

给删除图片添加事件

```
 function delProduct(){
            confirm("是否要删除该商品？");
        }
```

《07(JavaEE综合练习)删除商品页面修改_》

```
function delProduct(pid) {
            confirm("是否要删除该商品？" + pid);
            //获取点击的商品的主键 (参数传递)
            //主键数据提交服务器Servlet 数据带参数传递给服务器
            window.location.href="${pageContext.request.contextPath}/delProduct?
pid="+pid;
        }
```

```
 <td align="center" style="HEIGHT: 22px"><a href="#"> <img

 src="${pageContext.request.contextPath}/images/i_del.gif"
                                     width="16" height="16" border="0"
style="CURSOR: hand"
                                 <%--加引号避免字符串报错--%>
                                     onclick="delProduct('${product.pid}')">
                                </a></td>
```

《08(JavaEE综合练习)删除商品服务器_》

web层

```
@WebServlet(urlPatterns = "/delProduct")
public class DelProductServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        /**
         * 获取客户端提交的商品主键
         * 调用业务层方法传递主键
         * 响应不要直接回到页面(直接回到页面什么都看不到)
         * 转发到查询所有数据的Servlet
         */
        String pid = request.getParameter("pid");
        ProductService service = new ProductService();
        service.delProduct(pid);
        //获取转发器
        RequestDispatcher dispatcher = request.getRequestDispatcher("/findAll");
        dispatcher.forward(request, response);
    }
```

service层

```
  /**
     * 方法delProduct()
     * 删除指定商品数据
     * WEB层调用方法,传递主键
     * 主键传递到dao
     */
   public void delProduct(String pid){
       try {
           dao.delProduct(pid);
       } catch (SQLException e) {
           e.printStackTrace();
       }
   }
```

dao层

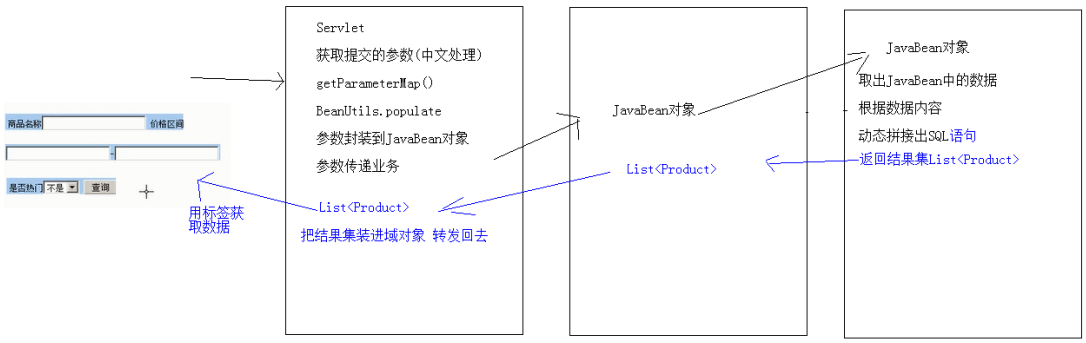
```java
/**
     * 方法 delProduct()
     * 删除指定的商品数据
     * 业务层传递商品主键
     */
    public void delProduct(String pid) throws SQLException {
        //编写SQL语句 ?占位符
        String sql = "DELETE  FROM product Where pid=?";
        //执行SQL语句
        qr.update(sql, pid);
    }
```

《09(JavaEE综合练习)查询商品的页面修改_》

改admin/product/list.jsp

```jsp
<%--修改部分1--%>
        <tr>
            <td class="ta_01" align="center" bgColor="#afd1f3">
                商品名称<input type="text" name="pname">
                价格区间<input type="text" name="min_price">--<input type="text"
name="max_price">
                是否热门<select name="is_hot">
                <option value="">不限制</option>
                <option value="1">是</option>
                <option value="0">不是</option>
            </select>
                <input type="submit" value="搜索">
            </td>
        </tr>
        <%--修改结束1--%>
```

《10(JavaEE综合练习)查询商品的分层架构_》



生成JavaBean

```
/**
 * 封装了所有查询条件的对象
 */
public class Search {
    private String pname;
    //最好不用double因为用户写数字有可能不写不写就没有赋值
    //成员变量就会是默认值0.0
    //可以用String String默认值是null只要不是null就填了
    private String min_price;
    private String max_price;
    private String is_hot;
```

《11(JavaEE综合练习)查询商品dao层_》

实现步骤

- 客户端填写数据,提交服务器
- Servlet获取客户端提交的数据(中文处理)
- 提交的数据存储到JavaBean中(Search)
- JavaBean传递业务层
- 业务层将JavaBean传递到dao层
- 取出JavaBean中存储的查询条件
    - 根据条件内容，动态的拼接处需要的SQL
    - 查询结果返回到业务层
- Servlet获取结果集存储到域对象,转发页面

dao层

```
    /**
     * 方法 findProduct()
     * 传递封装了查询条件的JavaBean
     * 查询后,返回List
     */
    public List<Product> findProduct(Search search) throws SQLException {
        //编写SQL语句
        String sql = "select * from product where 1=1 and shop_price >=1000";
```

```
        //创建集合，存储?占位符的实际参数
        ArrayList<Object> list = new ArrayList<>();
        //取出Search对象的数据,判断结果拼接SQL
        //判断pname商品名="a" 商品名模糊查
        if (search.getPname() != null && !"".equals(search.getPname())) {
            sql = sql + " and pname like ?";
            //集合添加商品名
            list.add("%" + search.getPname() + "%");//模糊查询用%%
        }

        //判断min_price
        if (search.getMin_price() != null && !"".equals(search.getMin_price()))
{
            sql = sql + " and shop_price >= ?";//大于最小价格大于最大价格
            //集合添加最小价格
            list.add(search.getMin_price());
        }

        //判断max_price
        if (search.getMax_price() != null && !"".equals(search.getMax_price()))
{
            sql = sql + " and shop_price <= ?";
            //集合添加最大价格
            list.add(search.getMax_price());
        }

        //判断is_hot
        if (search.getIs_hot() != null && !"".equals(search.getIs_hot())) {
            sql = sql + " and is_hot = ?";
            //集合添加是否热门
            list.add(search.getIs_hot());
        }
        System.out.println("打印" + list.toArray());
        //执行SQL语句
        return qr.query(sql, new BeanListHandler<Product>(Product.class),
list.toArray());

    }
```

《12(JavaEE综合练习)查询商品web层和service_》

service层

```
    /**
     * 方法 findProduct()
     * WEB层调用,传递封装好的Search对象
     * 调用dao层方法,传递Search
     * 返回结果List
     */
    public List<Product> findProduct(Search search) {
        List<Product> list = null;//提升作用域
        try {
            list = dao.findProduct(search);
        } catch (SQLException e) {
```

```
            e.printStackTrace();
        }
        return list;
    }
```

```
@WebServlet(urlPatterns = "/findProduct")
public class FindProductServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        /**
         *封装到JavaBean
         * 调用业务层方法,传递JavaBean
         * 返回结果List集合
         * 存储域对象request
         * 转发页面list.jsp
         */
        request.setCharacterEncoding("utf-8");
        Map<String, String[]> map = request.getParameterMap();
        Search search = new Search();
        try {
            BeanUtils.populate(search, map);
        } catch (Exception e) {
            e.printStackTrace();
        }
        ProductService service = new ProductService();
        List<Product> list = service.findProduct(search);
        request.setAttribute("list", list);
        RequestDispatcher dispatcher =
request.getRequestDispatcher("/admin/product/list.jsp");
        dispatcher.forward(request, response);
    }
```

《13(JavaEE综合练习)分页中需要的数据_》

找出分页需要的5个数据

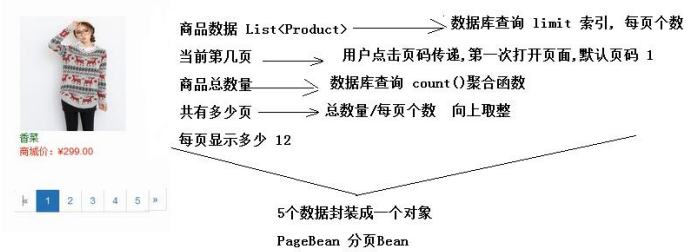
商品数据List---------->数据库查询 limit 索引,每页个数

当前第几页---------->用户点击页码传递,第一次打开页面,默认页码 1

商品总数量---------->数据库查询count()聚合函数

共有多少页---------->总数量/每页个数 向上取整

每页显示多少 12

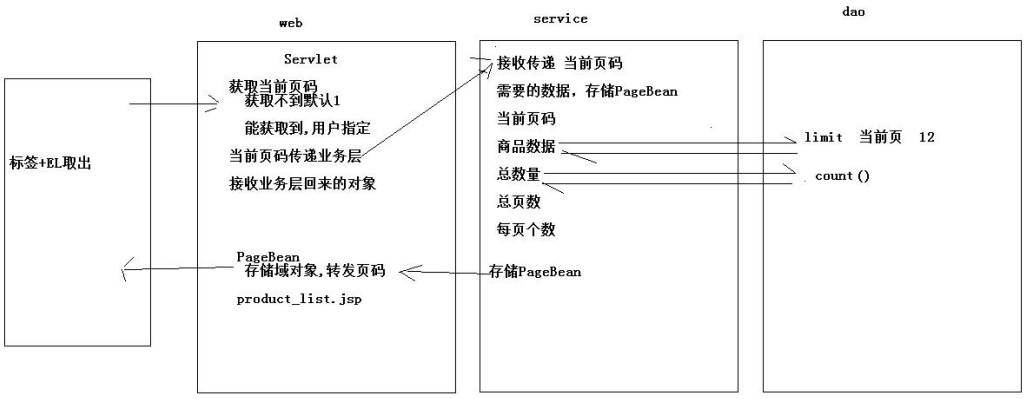


《14(JavaEE综合练习)分页定义PageBean_》

创建PageBean对象

```
/**
 * 用于封装实现分页查询的数据
 *
 * List<Product> 这种分页只能针对显示商品数据
 * 改成适合所有数据集合写泛型 List<T>  灵活性大
 * PageBean<User>  pb = new PageBean()
 */
public class PageBean <T>{//泛型类
    //当前页数
    private int currentPage;
    //总数量
    private long totalCount;
    //总页数
    private int totalPag;
    //每页显示条数
    private int pageSize;
    //分页中的数据
    private List<T> list;
```

《15(JavaEE综合练习)分页的三层架构_》



《16(JavaEE综合练习)业务层封装PageBean_》

PageBean

```
/**
 * 用于封装实现分页查询的数据
 * <p>
 * List<Product> 这种分页只能针对显示商品数据
 * 改成适合所有数据集合写泛型 List<T>  灵活性大
 * PageBean<User>  pb = new PageBean()
 */
public class PageBean<T> {//泛型类
    //当前页数
    private int currentPage;
    //总数量
    private long totalCount;
    //总页数
    private int totalPag;
    //每页显示条数
    private int pageSize;
    //分页中的数据
    private List<T> list;
```

dao层

```
    /**
     * 方法获取所有商品总数量
     * 返回long
     * 查询结果一个值
     * ScalarHandler
     */
    public long getTotalCount() throws SQLException {
        String sql = "select count(*) from product";
        return qr.query(sql, new ScalarHandler<Long>());//构造器不传 泛型传
    }

    /**
     * 方法获取分页查询的商品
```

```
     * 参数传递当前页,每页条数
     * 返回List
     */
    public List<Product> findByPage(int currentPage, int pageSize) throws
SQLException {
        //拼写分页查询的SQL
        String sql = "select * from product limit ?,?";//参数写成?防止注入
        return qr.query(sql, new BeanListHandler<Product>(Product.class),
(currentPage - 1) * pageSize, pageSize);


    }
```

service层

```
    /**
     * 定义方法,封装PageBean对象
     * 接收WEB传递的当前页数
     */
    public PageBean<Product> getPageBean(int currentPage) {//String可以转成int传递
过来
        PageBean<Product> pb = new PageBean<Product>();
        try {
            //封装PageBean对象的数据

            //封装当前页
            pb.setCurrentPage(currentPage);
            //封装每页个数
            pb.setPageSize(12);
            //封装需要的商品数据
            //dao层查询,需要当前页和每页的条数
            List<Product> list = dao.findByPage(currentPage, 12);
            pb.setList(list);
            //封装总数量，dao查询
            long totalCount = dao.getTotalCount();
            pb.setTotalCount(totalCount);
            //封装总页数 向上取整 double转成整数int
            int totalPage = (int) Math.ceil(totalCount * 1.0 / 12);//long/int永
远都是一个整数可是Math返回是double类型所以强转
            pb.setTotalPag(totalPage);

        } catch (Exception e) {
            e.printStackTrace();
        }

        return pb;
    }
```

web层

```
@WebServlet(urlPatterns = "/page")
public class PageServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        /**
```

```
         * 获取客户端提交的当前页
         * 当前页传递业务层
         * 接收业务层返回PageBean
         * 存储requwst域对象,转发页面
         */
        String currentPage = request.getParameter("currentPage");
        if (currentPage == null) {
            currentPage = "1";
        }
        //调用业务层传递当前页
        ProductService service = new ProductService();
        PageBean<Product> pageBean =
service.getPageBean(Integer.parseInt(currentPage));
        request.setAttribute("pb", pageBean);
        RequestDispatcher dispatcher =
request.getRequestDispatcher("/product_list.jsp");
        dispatcher.forward(request, response);
    }
```

《17(JavaEE综合练习)分页显示商品数据_》

```
    <!--
        遍历标签,取出域对象中的商品
        pb取出的是域对象中的值
        值是一个PageBean对象,pb对象的属性List
        procuct装的集合中的每个元素（存在了pagecontext域中）
    -->
    <c:forEach items="${pb.list}" var="product">
        <div class="col-md-2" style="height: 240px">
            <a href="product_info.htm"> <img src="${product.pimage}"
                                             width="170" height="170"
style="display: inline-block;">
            </a>
            <p>
                <a href="product_info.html" style='color:
green'>${product.pname}</a>
            </p>
            <p>
                <font color="#FF0000">商城价：&yen;${product.shop_price}</font>
            </p>
        </div>
    </c:forEach>
    <!-- 遍历标签,取出域对象中的商品-->
```

《18(JavaEE综合练习)分页显示页码完成_》

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>会员登录</title>
```

```
<link rel="stylesheet" href="css/bootstrap.min.css" type="text/css" />
<script src="js/jquery-1.11.3.min.js" type="text/javascript"></script>
<script src="js/bootstrap.min.js" type="text/javascript"></script>
<!-- 引入自定义css文件 style.css -->
<link rel="stylesheet" href="css/style.css" type="text/css" />

<style>
body {
    margin-top: 20px;
    margin: 0 auto;
    width: 100%;
}

.carousel-inner .item img {
    width: 100%;
    height: 300px;
}
</style>
</head>

<body>


    <!-- 引入header.jsp -->
    <jsp:include page="/header.jsp"></jsp:include>


    <div class="row" style="width: 1210px; margin: 0 auto;">
        <div class="col-md-12">
            <ol class="breadcrumb">
                <li><a href="#">首页</a></li>
            </ol>
        </div>
     <!--
         遍历标签,取出域对象中的商品
         pb取出的是域对象中的值
         值是一个PageBean对象,pb对象的属性List
     -->
        <c:forEach items="${pb.list}" var="product">
        <div class="col-md-2" style="height: 240px">
            <a href="product_info.htm"> <img src="${product.pimage}"
                width="170" height="170" style="display: inline-block;">
            </a>
            <p>
                <a href="product_info.html" style='color:
green'>${product.pname}</a>
            </p>
            <p>
                <font color="#FF0000">商城价：&yen;${product.shop_price}</font>
            </p>
        </div>
        </c:forEach>
        <!-- 遍历标签,取出域对象中的商品结束-->

    </div>

    <!--分页 -->
    <div style="width: 380px; margin: 0 auto; margin-top: 50px;">
```

```
        <ul class="pagination" style="text-align: center; margin-top: 10px;">

            <!--
                 上一页=当前页-1
                 如果已经是第一页,不能点击
            -->
            <c:if test="${pb.currentPage==1}">
                <li class="disabled"><a aria-label="Previous"><span aria-
hidden="true">&laquo;</span></a></li>
            </c:if>

            <c:if test="${pb.currentPage!=1}">
                <li><a href="${pageContext.request.contextPath}/page?
currentPage=${pb.currentPage-1}" aria-label="Previous"><span
                        aria-hidden="true">&laquo;</span></a></li>
            </c:if>



            <!--
               页码: 不是死的
               而是循环出来的
               开始循环变量1
               循环结束到总页数
             -->
            <c:forEach begin="1" end="${pb.totalPage}" var="i">
                <!-- 超连接: 连接到分页Servlet,传递参数 页码-->

                <!-- 当前页数,页码不能点击,背景样式-->
                <c:if test="${i==pb.currentPage}">
                    <li class="active"><a>${i}</a></li>
                </c:if>

                <c:if test="${i!=pb.currentPage}">
                    <li><a href="${pageContext.request.contextPath}/page?
currentPage=${i}">${i}</a></li>
                </c:if>
            </c:forEach>

            <!--
                下一页=当前页+1
                如果是最后一页,不能点击
            -->
            <c:if test="${pb.currentPage==pb.totalPage}">
                <li class="disabled"><a  aria-label="Next"> <span aria-
hidden="true">&raquo;</span>
            </c:if>

            <c:if test="${pb.currentPage!=pb.totalPage}">
                <li><a href="${pageContext.request.contextPath}/page?
currentPage=${pb.currentPage+1}" aria-label="Next"> <span aria-
hidden="true">&raquo;</span>
            </c:if>

            </a></li>
        </ul>
    </div>
    <!-- 分页结束 -->
```

```
    <!--商品浏览记录-->
    <div
        style="width: 1210px; margin: 0 auto; padding: 0 9px; border: 1px solid
#ddd; border-top: 2px solid #999; height: 246px;">

        <h4 style="width: 50%; float: left; font: 14px/30px 微软雅黑">浏览记录
</h4>
        <div style="width: 50%; float: right; text-align: right;">
            <a href="">more</a>
        </div>
        <div style="clear: both;"></div>

        <div style="overflow: hidden;">

            <ul style="list-style: none;">
                <li
                    style="width: 150px; height: 216; float: left; margin: 0 8px
0 0; padding: 0 18px 15px; text-align: center;"><img
                    src="products/1/cs10001.jpg" width="130px" height="130px" />
</li>
            </ul>

        </div>
    </div>


    <!-- 引入footer.jsp -->
    <jsp:include page="/footer.jsp"></jsp:include>

</body>

</html>
```