/\*\*

\* 认识 xml文档

\* 作用：通过xml格式 传递数据到 服务器 或者客户端，结构简单 格式清除，更适合 开发人员对数据的 查看 信息的整理；

格式：树形结构，必须定义为标准的 结构；

根目录：

子元素

内部子元素

元素的属性

元素值

\* 例如：xml格式的文件：

\* <?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd" id="WebApp\_ID" version="3.1">

<display-name>JavaWebDemo</display-name>

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>com.baidu.servlet.LoginServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/servlet/LoginServlet</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

</web-app>

\*

\* 标准的xml格式文档：

\* 根目录下的子元素 是有规律的；

\* <?xml version="1.0" encoding="UTF-8"?>

\* <root>

\* <studInfo id="10" name="stu\_10">

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

<booksList>

<bookInfo>

<bookId>102</bookId>

<bookName>三国演义</bookName>

<bookPrice>88.88</bookPrice>

<bookDate>2011-10-8</bookDate>

</bookInfo>

<bookInfo>

<bookId>102</bookId>

<bookName>三国演义</bookName>

<bookPrice>88.88</bookPrice>

<bookDate>2011-10-8</bookDate>

</bookInfo>

<bookInfo>

<bookId>102</bookId>

<bookName>三国演义</bookName>

<bookPrice>88.88</bookPrice>

<bookDate>2011-10-8</bookDate>

</bookInfo>

</booksList>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

<booksList>

</booksList>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

\* </studInfo>

\* <studInfo>

\* <stuId>1001</stuId>

\* <stuName>刘德华</stuName>

<stuSex>男</stuSex>

<stuAge>30</stuAge>

\* </studInfo>

\* </root>

\*

\* 二章： WEB开发基础：

\* 1. web 万维网 WWW ：World wide Web;

\* 2. web 开发的三大核心

\* html 网页， url 定位， http 传输协议；

\*

\* 页面：动态 静态

\* html +css jsp asp(VB) ado.net(C#) php 电子商务

\*

\*

\* 定位：

\* url:统一资源定位器，用来定位web服务中的某个资源；

\* uri:通用资源标识符，不仅可以定位某个资源，还可以确定 资源的具体位置(url是uri的组成部分之一);

\*

\* 请求提交方式：get post put delete head options 六种提交方式

\* get：传输速度 快，不安全 数据 会在 url中显示，传输数据量小 1024字节

\* post：速度慢 相对比较安全，将数据 已打包的形式传输 传输数据量大 一般为4M;

\*

\* 3. B/S 和 C/S

\* B/S: 浏览器 服务器： 易于维护 成本低 分布式

\* C/S： 客户端 服务器：维护困难 处理数据量大 交换性强

\*

\* 4.http协议 /https fpt/

\* 俗称：超文本传输 协议

\* 特点：无状态 -- 不能记住 客户端请求的数据

\*

\* 5.状态码：

\* 200 ：操作成功

\* 301：资源绑定到其他URL

\* 404：请求url错误

\* 400：请求的服务器参数异常

\* 415：请求服务内部错误

\* 500：服务器逻辑错误

\* ...............

\* 自定义状态码：

\* NO8858885569\_599:这是一个什么错误........

\*

\* 6.Web服务器：tomcate jBoss webLogic Oracle.......

\*

\* 第三章 serlert:

\* 一 什么是 servlet？

\* 是有Java代码编写的 Java源文件，运行在web服务器中，用于接收用户请求，并且调用javaBean业务逻辑 获取响应得结果，响应给请求；

\*

\* 二 生命周期

\* 五个阶段 3方法：

\* 加载实例化：执行构造函数

\* 初始化：执行 inint() 方法

\* 服务：执行 service(),根据请求类型 调用 get post updata delete .....

\* 销毁：执行 destory()方法。服务停止 执行该方法

\* 不可用状态：系统自动回收，造成内存不可用状态；

\*

\* 服务的加载顺序：

\* 第一次：执行顺序：请求 ===》加载实例化===》初始化===》服务===》响应===》===》

\* 第二次：执行顺序：请求 ===》服务===》响应===》===》

\*

\* 最终====》销毁

\*

\* 三如何 使用servlet?

\*

\* Jsp动态页面：

\* 什么是 jsp?

\* jsp:(java Server page) 运行在服务器上

\* jsp中的业务逻辑代码：

\* <% Date nowTime = new Date();%>

\* 输出内容：<%= 输出结果%>

\*

\* 如何不使用servlet服务，可以在jsp页面 写 逻辑代码 + html 渲染数据；

\*

\* jsp 运行特点：

\* 第一次访问 ：

\* 将 jsp文件由jsp引擎，转换为servelt 的源文件(.java) ===>将源文件 编译为字节码文件（.class）

\* ===> jmv 执行字节码 ===》然后再向客户端输出===》浏览器渲染输出数据生成静态html页面；

\*

\* 第二次访问：

\* 在页面内容没有改变的情况下，直接执行 字节码文件；否则按照第一次编译过程执行；

\*

\* jsp页面的组成：

\* html 静态页面

\* <% script %> 脚本的引入

\* <% JAVA 业务逻辑代码%> Java代码

\* <%= %> 数据的输出， 等同于：servlet中的out.print("。。。。");

\* <%! 变量 方法....%>声明全局变量 / 方法

\* <%@ %> 指令：

\* page 指令： 导入包

\* pageEncoding 设置编码

\* errorPage :指定错误页面

\* include：引入页面，

\* 静态引入 和动态引入的区别？

\*

\* 案例 在jsp实现 后台业务逻辑的处理

\*

\* <% IUserDao userDao = new UserDaoImpl();

//调用 列表方法

List<UserInfo> list = userDao.querList();

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

%>

<!-- 循环的数据 -->

<%=list[i].userName %>,<%=list[i].userAge %>,<%=list[i].userSex %><br/>

<%

}

%>

\*

\* jsp 中的内置对象：

\* jsp页面中不需要创建的对象，可以直接调用

\* 有哪些 内置对象，分类

\* 输入输出：out request response

\* 作用域：page request session application

\* 通讯域：pageContext request/requestContext session/sessionContext

\* application/applicationContext

\* 其他：config exception

\*

\* 作用域的 范围大小：page request session application

\*

\* 页面取值：

\* <% out.print() %> <%= 变量名称%> ${变量名称}

\* pageContext.setAttribute("name",'小白');

\* request.setAttribute("name",'小花');

\* session.setAttribute("name",'小红');

\* application.setAttribute("name",'小灰');

\*

\* ${name} == 小白

\* ${sessionScore.name} == 小红

\*

\* 案例练习：

\* 使用 四大作用域存值:通过 不同页面+浏览器+ 客户端 进行访问 验证四大作用域的 范围；

\*

\* EL表达式的使用：

\* ${ }

\* 放变量， 作用域.变量， 对象.属性;

\* 还可以：

\* 运算符 + - \* / % || && ! eq ..... 三目运行符；

\*/

<body>

<%@ include file=*"index.jsp"* %>

欢迎${userNakme }登录 某某系统:你的个人信息：名称${userNakme },登录密码：${userPwd }<br>

<img alt=*""* src=*""*>

<hr/>

<%

SimpleDateFormat smp = **new** SimpleDateFormat("yyyy-MM-dd");

String nowDate = smp.format(**new** Date());

out.print("out时间为："+nowDate);

//页面中的内置对象

session.setAttribute("", "");

pageContext.getAttribute("");

%>

时间为：<%= nowDate %>

案例：

<%!

// 定义全局变量

**int** pageCon = 1;

**int** reqCon = 1;

**int** sessionCon = 1;

**int** applicationCon = 1;

%>

<%

//每次访问页面 后 变量加一

Object pc = pageContext.getAttribute("pageCon");

**if**(pc!=**null**){

pageCon+=1;

}

Object rc = request.getAttribute("reqCon");

**if**(rc!=**null**){

reqCon+=1;

}

Object sc = session.getAttribute("sessionCon");

**if**(sc!=**null**){

sessionCon+=1;

}

Object ac = application.getAttribute("applicationCon");

**if**(ac!=**null**){

applicationCon+=1;

}

pageContext.setAttribute("pageCon", pageCon);

request.setAttribute("reqCon", reqCon);

session.setAttribute("sessionCon", sessionCon);

application.setAttribute("applicationCon", applicationCon);

%>

<hr/>

上下文：<%= pageContext.getAttribute("pageCon") %><br/>

请求： <%= request.getAttribute("reqCon") %><br/>

会话： <%= session.getAttribute("sessionCon") %><br/>

服务： <%= application.getAttribute("applicationCon") %><br/>

<hr/>

EL:表达式<br/>

699+885= ${699+885 }<br/>

699%885= ${699%885 }<br/>

判断：

(3<58) && (549>995) 结果为：${(3<58) && (549>995) }<br/>

</body>