1. JavaWeb概述

Client/S架构

Browser(html css js)/**Server架构**

Servlet技术

JSP技术

1. Web服务器

学习Web开发，服务器是必不可少的

Web开发常用的服务器软件：

1. WebLogic服务器，是BEA公司的产品，也是目前Web开发使用最广泛的服务器之一。
2. WebSphere服务器，是IBM公司的产品。
3. Tomcat服务器，是中小型的应用系统中使用最广泛的服务器，是开源免费的，是Apache公司的产品。
4. Tomcat服务器的下载、配置
5. Tomcat服务器的下载

下载地址:http://tomcat.apahce.org

1. zip (pgp, sha512) 是Window操作系统下的Tomcat免安装版本的压缩文件
2. tar.gz (pgp, sha512) Linux操作系统下的版本



1. 32-bit Windows zip (pgp, sha512) Window系统下32位安装版本



1. 64-bit Windows zip (pgp, sha512) Window系统下64位安装版本

Tomcat服务器运行需要JDK的支持

Tomcat5 jdk4以上版本

Tomcat6 jdk5以上版本

Tomcat7 jdk6以上版本

1. 介绍一下Tomcat目录结构
2. bin目录 -- 存储Tomcat的命令
3. conf目录 -- Tomcat的配置文件，最重要的是server.xml文件
4. **webapps目录 -- 存储web项目**
5. 将Tomcat服务器部署到开发工具中

Tomcat服务器的默认端口号是8080

浏览器地址栏访问:

<http://localhost:8080>

1. Servlet技术
2. Servlet是什么？

Servlet是sun公司提供的一门用于开发动态web资源的技术

Servlet可以在服务器端接收浏览器端发送过来的请求，也可以向浏览器端响应内容，同时Servlet也用于和数据库做连接。

1. 使用开发工具开发第一个web程序—HelloServlet

Step1：新建一个web项目

**Step2：新建包，在包下创建类，HelloServlet类**

**Step3：让HelloServlet去继承HttpServlet，在该类中调出service方法，在service方法完成该servlet要做的事情**

**Step4：在web.xml文件中对HelloServlet进行配置**

Step5：将web项目部署到tomcat服务器中

Step6：在浏览器地址栏中输入地址访问：

<http://localhost:8080/web>项目名/url-pattern地址

<http://localhost:8080/servlet/hello>

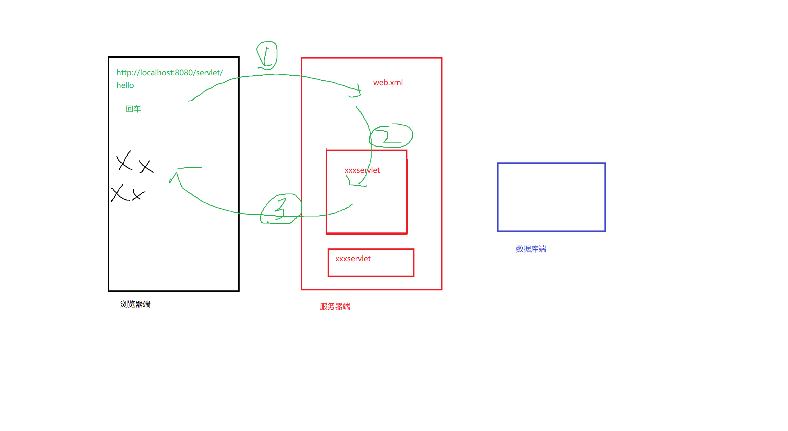
1. servlet的调用过程
2. 浏览器地址栏输入地址，会向服务器发送请求
3. 服务器端先检查web项目中的web.xml文件，根据配置找到要调用执行哪个servlet
4. 接下来会执行servlet中的service方法，将结果返回到浏览器端显示。

B---->S---->B

整个调用执行过程都是服务器自动进行的，开发人员只需要写

好servlet以及web.xml相关配置即可，无需关注底层的实现，

这个也是容器，框架的思想。



1. servlet的配置细节
2. 基本配置

< servlet >

<servlet-name></servlet-name>

<servlet-class></servlet-class>

</servlet>

<servlet-mapping>

<servlet-name></servlet-name>

<url-pattern></url-pattern>

</servlet-mapping>

其中一个<servlet>可以对应多个<servlet-mapping>

1. 带星号的通配符配置

语法**：1) \*.加内容的配置**

<url-pattern>\*.do</url-pattern>

**2)以/开头，以/\*结尾的配置**

<url-pattern>

/mytest/\*

</url-pattern>

localhost:8080/web项目名/mytest/xxx

（3）缺省配置

url-pattern的路径中，只有一个/

<url-pattern>/</url-pattern>

如果一个servlet的对外访问路径设置为/，则该servlet就是

一个缺省servlet，其它servlet不处理的请求都由它来处理。

1. 表单提交案例引入
2. HttpServletRequest request对象
3. **通过request对象获取浏览器端相关的信息**
4. **通过request对象获取请求参数**  **name=value**
5. getParameter(String name)

根据参数中的name来获取value的值

1. getParameterValues(String name)

该方法返回字符串数组String[]

通过name来获得多个value值的情况

1. getParameterNames()

该方法返回Enumeration类型，理解为容器

获取所有的请求参数中的name值

1. **通过request对象设置参数中的中文乱码问题**
2. 乱码产生的原因

浏览器端一般设置的编码使用utf-8编码，提交表单时使用的也是utf-8编码；tomcat服务器默认情况下是iso8859-1的编码，所以对于英文数字不受影响，但是iso8859-1中没有中文，所以会用？代替。

1. 处理乱码的办法
2. get请求的处理方式

byte[] bs=xxStr.getBytes(“iso8859-1”);

String str=new String(bs,”utf-8”);

1. post请求的处理方式

setCharacterEncoding(“utf-8”)

post请求方式中参数的中文编码设置

（4）通过request对象进行请求**转发**

1）概念：将未完成的工作交给下一个组件(servlet、jsp)继续完成。

转发是用来实现资源的跳转，浏览器地址栏不会发生变化。

Servlet--->Servlet

2）方法：

RequestDispatcher dis=request.getRequestDispatcher(“xx”)

获得转发器

xx表示转发的下一个组件的地址

dis.forward(request,response);

实现请求转发，请求和响应对象要最为参数设置进来

1. 通过request对象请求转发—实现请求包含

1）方法

RequestDispatcher rd=request.getRequestDispatcher(“xx”);

获得转发器，指定转发的地址

rd.include(request,response);

实现请求包含的转发

2)说明

请求包含的转发，会将当前响应结果和转达到xx地址以

后的响应结果，共同处理，一起响应给浏览器端。

1. 使用request作为域对象来使用

域对象：一个对象具有可以被看见的范围，可以存储资源从而实现资源的共享。

request作为域对象，一般和转发器一起使用。

request.setAttribute(String name, Object obj);

往request域对象中存储数据

request.getAttribute(String name)

从request与对象中获取数据

1. HttpServletResponse对象
2. 概念

在doGet、doPost、service方法的参数中都会有HttpServletRequest和HttpServletResponse对象。

HttpServletRequest代表的是http请求，想要获取请求中数据的时候使用；

HttpServletResponse代表的是http响应，有数据需要发送给浏览器端的时候使用。

1. 常用方法

1）setHeader(String name, String value)

设置响应头：①设置文本格式和编码

②定时刷新

定时刷新是通过Refresh设置为响应头，可以实现

在多少时间后实现跳转 url

2）getWriter() --> PrintWriter

getOutputStream() -->ServletOuputStream

输出信息到浏览器端

3）setContentType()

设置响应到浏览器端的中文编码以及文本格式

4）重定向

sendRedirect(“xxxx”)

xxx代表重定向后的地址

1. 用于服务器端内部资源的跳转，也可以用于不同应用和不同服务器之间的跳转
2. 重定向浏览器地址栏会发生变量，浏览器地址栏会是重定向以后的地址
3. 重定向相当于是2次请求，2次响应
4. 重定向一般用于事情已经做完了，只是为了跳转到下一个组件。

8、EMS-员工信息管理系统 --servlet版本

员工信息的增删~~改~~查

B S DB

Step1:开发DB，数据库，表和表中的数据

打开命令行，进入到解压后的mysql数据库文件的bin目录中

(dir cd xxx d: c: )

进入bin目录以后

1. 启动mysql服务(不能关闭) mysqld --console
2. 启动mysql数据库

mysql -h localhost -u root -p

操作库

show databases;

create database ems;

use ems;

操作表

create table user(

id int primary key auto\_increment,

name varchar(50),

salary double,

age int

);

操作数据

insert into user values(1,’zs’,3000,20);

insert into user values(2,’ls’,1000,30);

insert into user values(3,’ww’,5000,25);

insert into user values(4,’xh’,4000,23);

9、ServletConfig和ServletContext对象

（1）ServletConfig

代表当前Servlet在web.xml文件配置信息的对象

1. 获取ServletConfig对象

ServletConfig scf=this.getServletConfig()

1. 功能
   1. **获取当前servlet的初始化参数**

在web.xml文件中的<servlet>标签的内部可以为当前servlet配置0个或者多个初始化参数，可以使用ServletConfig来获取配置的这些初始化参数

getInitParameter(name) -->value

getInitParameterNames() -->Enumeration

* 1. 获取ServletContext对象

ServletConfig scf=this.getServletConfig();

ServletContext sct=scf.getServletContext();

（2）ServletContext对象

代表的是当前web应用(web项目)对象

在web应用被加载以后，服务器会立刻创建出代表当前web

应用的ServletContext对象，创建后对象会一直驻留在内存

中，直到服务器关闭或者web应用移出服务器为止，随着

web应用的销毁，ServletContext对象也会跟着销毁。

ServletContext对象 ---> servlet web应用（项目）

ServletConfig对象 --> ServletConfigDemo.java

ResponseDemo1.java

以上servlet在web.xml配置的初始

化信息

1. 获取ServletContext对象

方法一：先获得ServletConfig，然后通过ServletConfig

在获得ServletContext。

ServletConfig scf=this.getServletConfig();

ServletContext sct=scf.getServletContext();

方法二：通过this直接获取

ServletContext sct=this.getServletContext();

1. ServletContext中的方法：

方法一：

getInitParameter(String name) ---->value

根据name来获得value的值

getInitParameterNames() ---->Enumeration

获得所有的name值

方法二：

ServletContext对象也是一个域对象，利用这个对象可以在整个web应用内实现资源的共享

setAttribute(String name, Object obj)

getAttribute(String name) --->Object obj

方法三、获取web资源文件的路径：

String path=sct.getRealPath(“xx文件”)

获得的是一个绝对路径，传入一个相对于web应用跟目录的资源文件路径，这个方法会在传入的路径前面动态的拼接上当前web应用跟目录的磁盘路径，这种路径称为动态获取的路径，换个环境，也能够正确的获取路径。

10、会话技术

（1）什么是会话？

用户打开一个浏览器，访问服务器端的多个web资源，然后关

闭浏览器，从打开到关闭的整个过程称之为一个会话。

会话技术多用于数据的存取。

（2）如何来保存会话过程中产生的数据

每个用户在使用浏览器和服务器进行会话过程中，不可避免的

会产生一些数据，这些数据有时候需要保存下来的，比如用户登录过后，应该保存一个用户的状态，会话结束钱（关掉浏览器前）表明用户一直是登录状态，并且不用的用户之间的登录状态应该互不影响，也有是需要在浏览器端永久保存一些数据。

已经学习的可以存储数据的两个域对象:request ServletContext

request域太小，request表示一次请求，可以在一次请求中实现资源的共享，但是在多次请求中，每次请求都是新的，request存储的数据就实现不了资源的共享。

ServletContext域太大，所在的所用用户都操作这个域，太混乱了。

会话技术包含Cookie技术和Session技术

（3）Cookie技术

1）Cookie的概述

Cookie的原理是通过set-Cookie响应头和Cookie请求头将

会话中产生的数据保存在浏览器端。

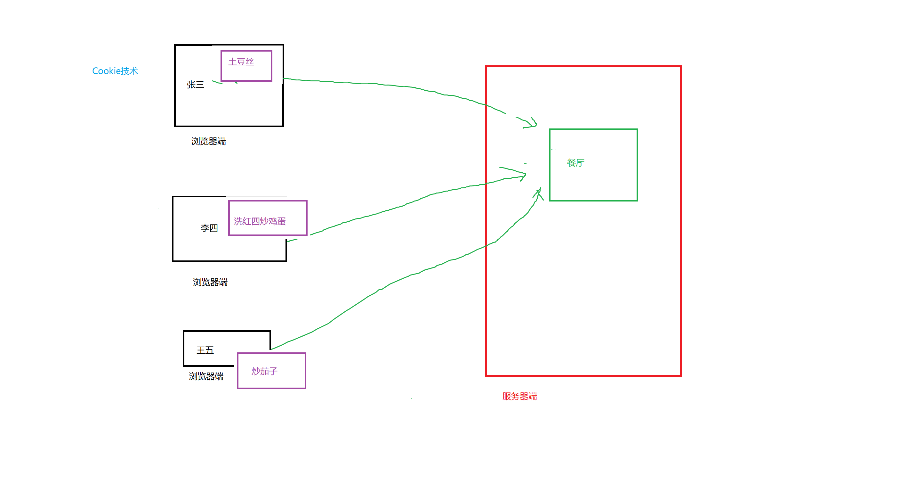
浏览器端请求服务器，服务器会将需要保存的数据通过set-cookie响应头发送给浏览器端，浏览器端接收到数据会保存在浏览器的内部；

当浏览器再次请求服务器时，通过Cookie请求头将上次保存的数据在带给服务器端，服务器端通过Cookie请求头来获取数据。

通过这种方式可以保存在会话过程中产生的数据。

Cookie技术是将需要保存的数据保存在了浏览器端，是一种浏览器端的会话技术，每个浏览器端各自保存各自的数据，再次访问服务器时会带着自己的数据来访问服务器端，每个浏览器端都持有自己的数据，因此数据在存取的时候也不会混乱。

理解图：



2）Cookie的常用方法

①创建Cookie

Cookie c=new Cookie(String name,String value);

name和value是需要往Cookie存储的数据

②添加Cookie

response.addCookie(c);

向响应中添加一个Cookie，可以在一次响应中添加多个Cookie

③设置Cookie存活时间

setMaxAge(时间) 单位是秒

如果不设置，默认情况下Cookie会将数据保存在浏览器内存中，一旦关闭浏览器，Cookie也会随着浏览器内存的释放而销毁；通过setMaxAge()方法设置的Cookie存活时间以后，Cookie会以文件的形式保存在浏览器端，在指定的时间来到之间，即使多次关闭浏览器，Cookie信息也会一直存在。

④设置Cookie的路径

setPath(String path)

设置浏览器在访问哪个地址及其子地址时，带着Cookie信息过来，

如果没有设置setPath，那么默认的路径是当前web应用

的路径 -- /web应用

⑤获取Cookie

Cookie[] cs=request.getCookies();

获取所有的Cookie中存储的数据，这些数据放到数组中

⑥Cookie的销毁

没有提供直接删除Cookie的方法

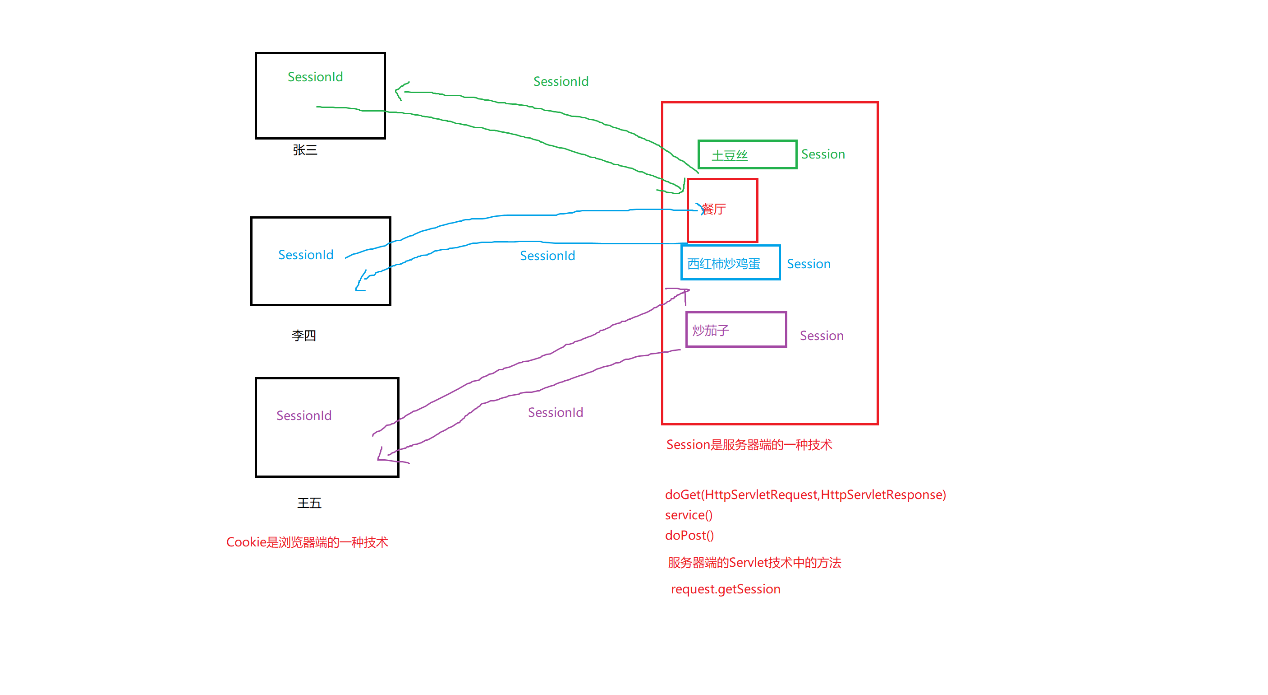
如果想删除一个Cookie，可以向浏览器发送一个同名、同Path的Cookie，只需要将Cookie的maxAge设置为0即可。

（4）Session技术

1）概念

Session也是一种会话技术，Session是将会话中产生的数据保存在了服务器端，**是服务端的一种会话技术**。

浏览器访问服务器端，服务器会为每一个浏览器单独创建一个Session对象，该对象有一个ID属性，是唯一的，一般称之为SessionId，并且服务器会将这个SessionId使用Cookie的方式保存在浏览器端，当浏览器再次访问服务器时，会将SessionId发送给服务器，服务器可以根据SessionId查找响应的Session对象。



2）Session技术中常用的方法：

①获取Session对象

HttpSession session=request.getSession();

②Session对象可以存取数据

session.setAttribute(String name, String value)

session.getAttribute(String name) --> value

③设置Session的有效时间

如果不设置，Session默认是30分钟的有效时间

在web.xml文件中配置Session的有效时间

④销毁Session对象

session.invalidate() 立刻马上销毁session

Session中存储的数据几点说明：

1）重启，重新部署等，服务器端使用Session存储的数据依然有效

2）关闭浏览器，相当于结束会话。

重新打开浏览器去访问服务器端Session存储的数据，是访问不

到的。

3）Session的有效时间如果过期了，Session中存储的数据也就获取

不到了

（5）Cookie和Session的比较

①Cookie是将会话中产生的数据保存在浏览器端(客户端)，

是客户端的技术。

②Session是将会话中产生的数据保存在服务器端，是服务器端

的技术。

③Cookie保存的信息的时间比较长，但是安全性不高，可能随

着用户的操作，Cookie会被清空，所以Cookie存储数据的 稳定性比较差。

因此Cookie适合存放要保存时间较长，但安全性不高的信息

④Session通常保存信息的时间比较有限，但是安全性较高。

因为数据是保存在服务器端，不会随着用户的操作而导致

Session中的数据意外丢失。

因此Session适合存放安全性比较高，但是不需要长时间保存

的数据。

五、JSP技术

1、JSP介绍

JSP是由SUN公司提供动态web资源的开发技术，看起来非常的像HTML，但是JSP可以在里面写java代码，而HTML是不可以的。所以JSP是一种动态网页开发技术。

JSP的出现是为了解决Servlet在响应到页面端时不适合往外输出

的问题。

HTML适合用来开发页面，展示数据，但是HTML开发出来的页面

本质上就是一个文档，无法展示动态的数据。

JSP非常适合编写HTML代码，合适作为响应页面向外输出，同时JSP里面可以写Java代码，可以展示动态数据。

Java Server Page Java服务器端的一种页面技术

2、编写JSP以及JSP的文档结构

1）写一个以.jsp为后缀的文件

2）在该文件中，可以包含如下内容：

--HTML CSS JavaScript

--Java代码

--注释

--指令

--隐含对象

--el表达式

--jstl标准标签库

3、JSP页面中的Java代码

Java代码可以写在jsp文件中的任意位置

两种方式：

（1）jsp表达式写法

语法: <%=……… %>

内容：变量、变量加运算符组合的表达式、常量值、有返回值

的方法

功能：将内容输出到页面上显示， =是输出的功能

（2）jsp脚本片段写法（代码块写法）

语法: <%

……..

…….

%>

内容：java代码的片段

功能：通过java代码的片完成一段业务逻辑的功能

4、指令：jsp声明写法

语法: <%@指令名 属性1=值1 属性2=值2 ... %>

(1) page指令（一般写在jsp页面的开头）

pageEncoding=”utf-8” 设置jsp文件保存时的中文编码

import=”” 在jsp页面章导包使用

（2）include指令

功能：能够将其它文件包含进来，可以包含**jsp文件**，也可以包含html文件

语法: <%@include file=”引入的文件.jsp” %>

（3）taglib指令

语法：<%@taglib uri=”” prefix=”” %>

功能：用来引入标签库 （ el表达式、jstl标准标签库）

5、注释

（1）<!-- 注释内容 -->

Html注释，注释中的内容如果包含了java代码，java代码注释不了，这些java代码会被执行的。

（2）<%-- 注释内容 --%>

Jsp注释，如果注释的内容包含了java代码，会被注释掉

6、九大隐式对象

可以不需要在JSP中预先定义就可以直接使用的。

一共九个，统称为JSP的九大隐式对象。

分别是：

①request对象 请求信息

②response对象 响应信息

③out对象 输出的输出流

④session对象 会话

⑤application对象 全局的上下文对象，（ServletContext对象）

⑥pageContext对象 当前JSP页面上下文

⑦page对象 JSP页面本身，了解即可，不常用

⑧config对象 Servlet配置对象（ServletConfig对象）

⑨exception对象 捕获网页（jsp页面）异常

（1）request隐式对象

①获取请求参数

②作为域对象存储值（注意域的范围）

③获取Cookie对象

（2）response隐式对象

①重定向

②处理HTTP文件头

--网页自刷新

--定时跳转

（3）application对象

application对象用于保存当前web应用的共有数据。

在服务器启动以后application对象会自动创建，在服务器停止时销毁。和ServletContext的功能是一样的。

application在JSP页面中使用；

ServletContext在Servlet中使用

功能1：获取web.xml文件中的全局初始化参数

功能2：作为域对象存取数据（域对象的范围—整个web应用）

（4）pageContext对象

获取页面上下文对象，可以通过pageContext在jsp页面中获取request、response等对象，在实际开发中，该功能使用较少，因为request、response等对象本身也是隐式对象，可以在jsp页面中直接使用。

一般使用pageContext是用来作为域对象，来存储数据的。

pageContext域对象的范围只能在当前页面中获取，范围是当前页面。

getAttribute()

setAttribute()

（5）page对象

page对象代表JSP本身，只有在JSP页面内才是合法的，可以看作是this关键字的别名，使用较少，了解即可。

（6）config对象

config对象用于在web.xml文件中配置信息，和ServletConfig的功能是一致的。

config对象代表当前JSP的配置信息，但是JSP页面通常无需配置，因此也就不存在配置的问题了。该对象在JSP中使用较少。

但是ServletConfig在Servlet中用处相对较大，因为Servlet是一定要在web.xml中配置的，可以配置相关的初始化参数，可以用ServletConfig来获取。

（7）exception对象

Exception对象用来处理JSP文件执行时发生的错误和异常，只有在page指令中设置isErrorPage属性值为true的页面才可以使用exception对象。

<%@page errorPage=”” isErrorPage=”” %>

errorPage是指定异常信息页面的

isErrorPage=”true” 指定是否可以使用exception对象

7、JSP中的四大域对象

request、session、application、pageContext是JSP页面中的四大域对象，也是JSP页面中的隐式对象。

setAttribute()

getAttribute()

--application作用域：有效范围是整个web应用，整个应用是指从应用启动到应用结束。

--session作用域：有效范围是当前会话，所谓当前会话，是指从用户打开浏览器开始，到用户关闭浏览器结束。

--request作用域：一次请求跨越的前后两个组件(servlet jsp)，一次请求指的是从一个组件开始，到另一个组件结束。

--pageContext作用域：作用范围仅限于用户请求的当前页面

什么时候用什么作用域？

如果数据只在当前jsp页面中使用，用pageContext域；

如果数据需要在转发时资源共享，用request域；

如果数据我现在需要用，过一会我还需要用，用session域；

如果数据我现在需要用，过一会别人也需要用，用application域

EMS员工信息管理系统

--servlet版本

--servlet+jsp版本

中文

MVC的设计模式

M层：Model模型层，负责业务，封装数据库的相关操作

Jdbc+javabean+方法—dao工厂

--com.lddx.dao

--com.lddx.util

V层：View视图层，负责浏览器端页面的显示 在WebRoot下

Jsp(html css js java代码)

C层：Controller控制层：属于服务器端，用来接收请求和响应

Servlet技术

--com.lddx.web

--servlet+jsp+el+jstl版本

8、EL表达式和JSTL标准标签库

（1）为什么要用EL表达式

在JSP中根据开发的需要嵌入了大量的java代码，增加了页面的复杂度，是页面不够简洁，不方便代码的开发和维护。

因此Sun公司制定了JSP标签（类似于html标签）来代替java代码。

Apache组织开发了一套标签，被sun公司整合后，成为标准标签库（jstl），jstl标签标签库可以配合EL表达式一起使用，减轻JSP页面的复杂度，代替Java代码，使JSP页面更加简洁，方便开发和维护。

（2）什么是EL表达式

EL表达式是一套简单的计算规则，用于给标签（jstl标准标签库）的属性赋值，也可以单独使用来计算和输出。

EL表达式单独用

EL+JSTL一起用

语法：${………}

使用${}来替代<% %>

（3）EL表达式的功能

1）输出简单的运算结果

--算术运算 + - \* / %

--逻辑运算 && || !

--关系运算 > >= < <= == !=

--empty运算：判断对象是否为空

2）获取请求参数

请求的地址：

localhost:8080/项目名/url-pattern地址

?key1=value1&key2=value2…

根据key来获取value的值

${param.key}

根据key来获取多个对应的value的值

${paramValues.key[下标]}

3）从域对象中直接获取数据

域对象—jsp中的四大域对象—jsp中的隐式对象

pageContext

request

session

appliaction

共同的方法：setAttribute(name, value)

getAttribute(name) ----> value

el表达式获取域对象中的数据的语法：

${name} ---> value

或者${xxScope.name} --->value

xx指定哪个域

4）获取JavaBean属性的值

什么是JavaBean？

是一种规范，符合该规范的类就是一个javabean类。

规范：①该类的属性/成员变量是private修饰的

②这些属性有对应的get和set方法

③该类中设有无参和有参构造方法

④该类中设有toString方法

如果一个类是javabean类，在jsp中可以很方便的使用

el表达式来获取类中的属性/成员变量。

语法：${对象名.属性名}

（4）jstl标准标签库

使用步骤：1）导入standar.jar和jstl.jar这两个jar文件

2）使用tablig指令引入标签库的信息

1）<c:out>标签：用于输出内容到页面上

<%= %>

<% out.println() %>

输出常量

输出作用域中存储的数据

转义输出

输出默认值等

2）<c:set>标签：设置用的标签

向四大域对象中增加域属性

向Map集合中增加，修改键值对

修改JavaBean的属性

3）<c:remove>标签

该标签用于删除四个域对象中的属性

4）<c:if>标签：可以进行条件判断

该标签执行达到的效果和java中的if语句效果一致

5）<c:choose>标签

实现多种分支情况的判断

语法：

<c:choose>

<c:when></c:when>

<c:when></c:when>

<c:when></c:when>

…….

<c:otherwise></c:otherwise>

</c:choose>

和java中if…else if…else语句结构相似

6）<c:forEach>标签：循环

①完成有循环次数的循环

②可以对集合的数据进行迭代操作

③可以对集合中存储的JavaBean对象进行操作

7）<c:forTokens>标签

用来浏览字符串中所有的成员，每个成员由定义的符号进行

分隔。

[www.baidu.com](http://www.baidu.com)

EMS—jsp—el -- 最终版

M—模型层：接口、javabean

V—视图层JSP—el+jstl

C --控制层

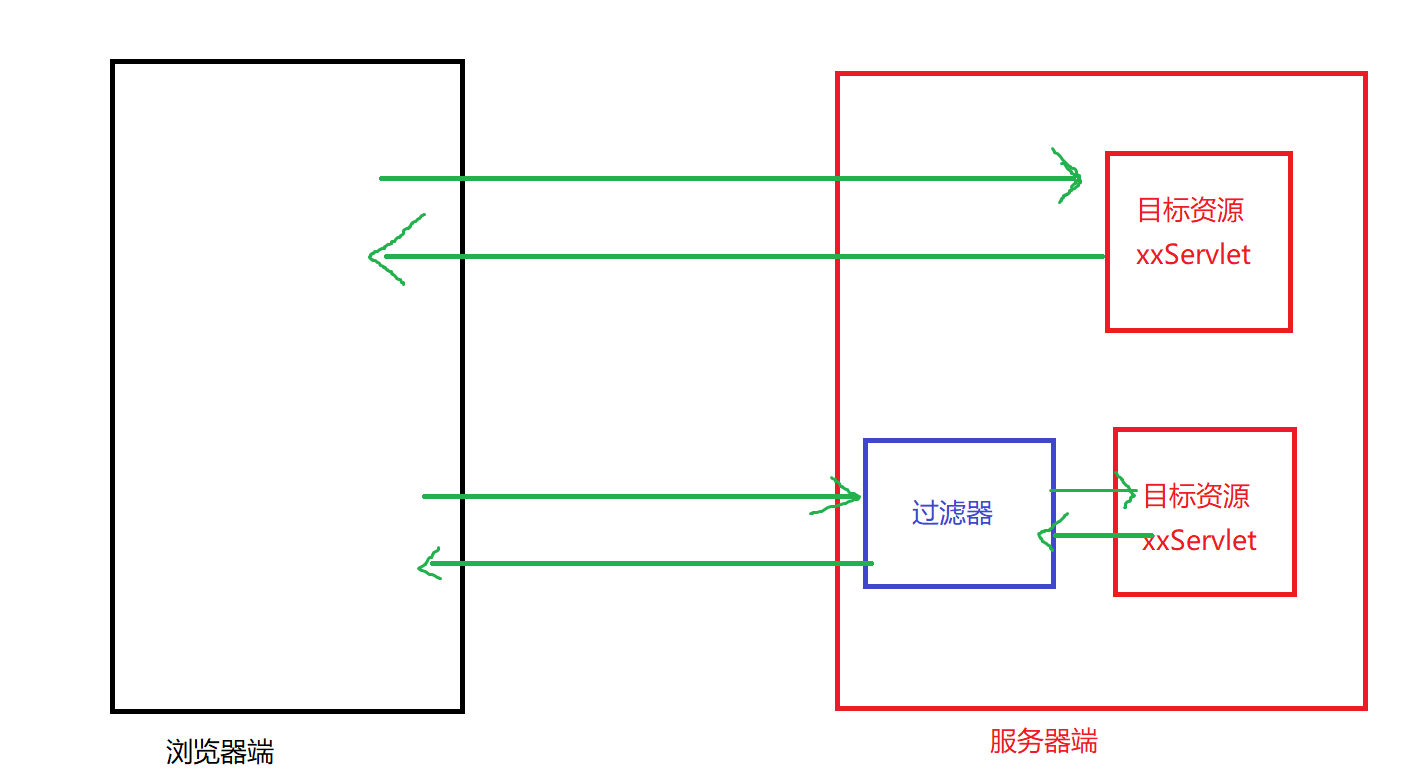
六、Servlet过滤器

1、什么是过滤器

Servlet过滤器具有拦截浏览器端请求的功能，Servlet过滤器可以

改变请求中的内容来满足实际开发中的需求。

实质上Servlet过滤器就是在web应用服务器中的一个web应用组件，用于拦截浏览器与目标资源的请求，并对这些请求进行一定的过滤处理在发送给目标资源。

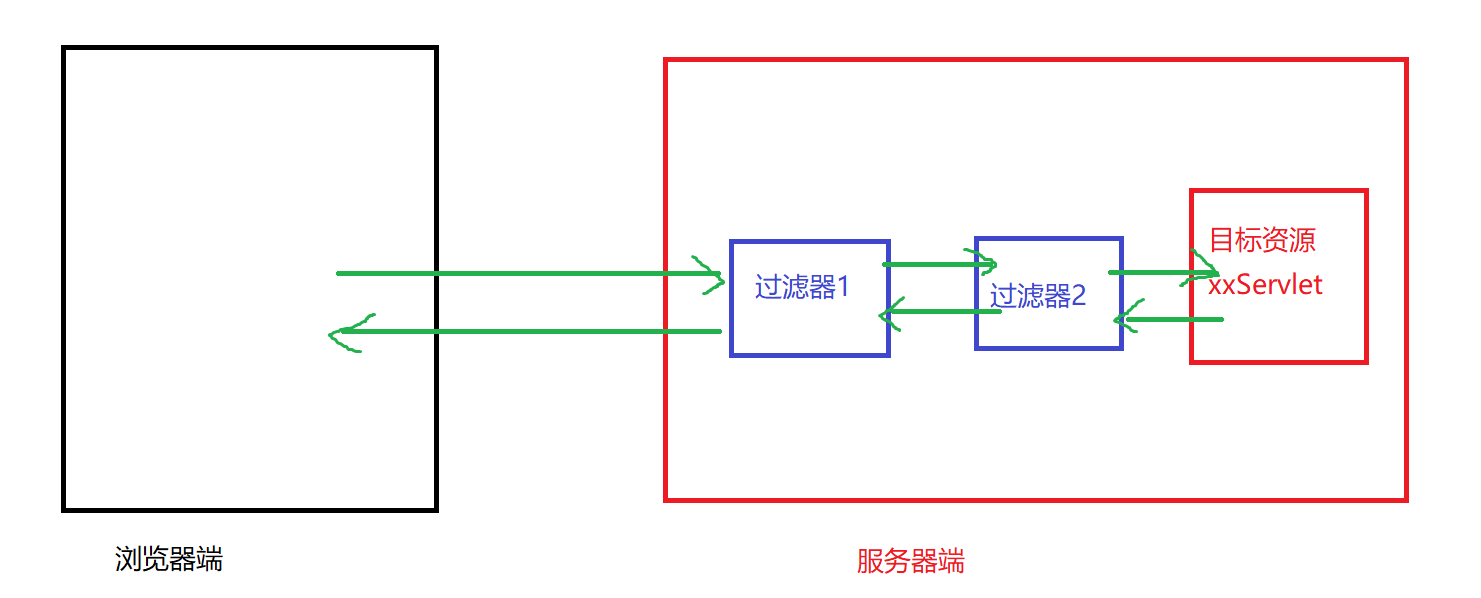


说明：web服务器部署了过滤器以后，不仅浏览器端发送请求会经过过滤器的处理，当发送到目标资源以后，响应时也同样会经过过滤器。

过滤器链：

当一个过滤器不满足实际业务的需求时，可以配置部署多个

过滤器进行处理，这样就组成了一个过滤器链。



2、过滤器的创建与配置

（过滤器和Servlet同属于web组件，所以过滤器的创建和配置与Servlet的创建和配置极其类似。）

（1）写一个类实现Filter接口

（2）在该类中重写父接口中的方法，在这些方法中实现过滤

器的功能。

（3）在web.xml中配置过滤器

3、过滤器的生命周期

过滤器在web项目中配置完成以后：

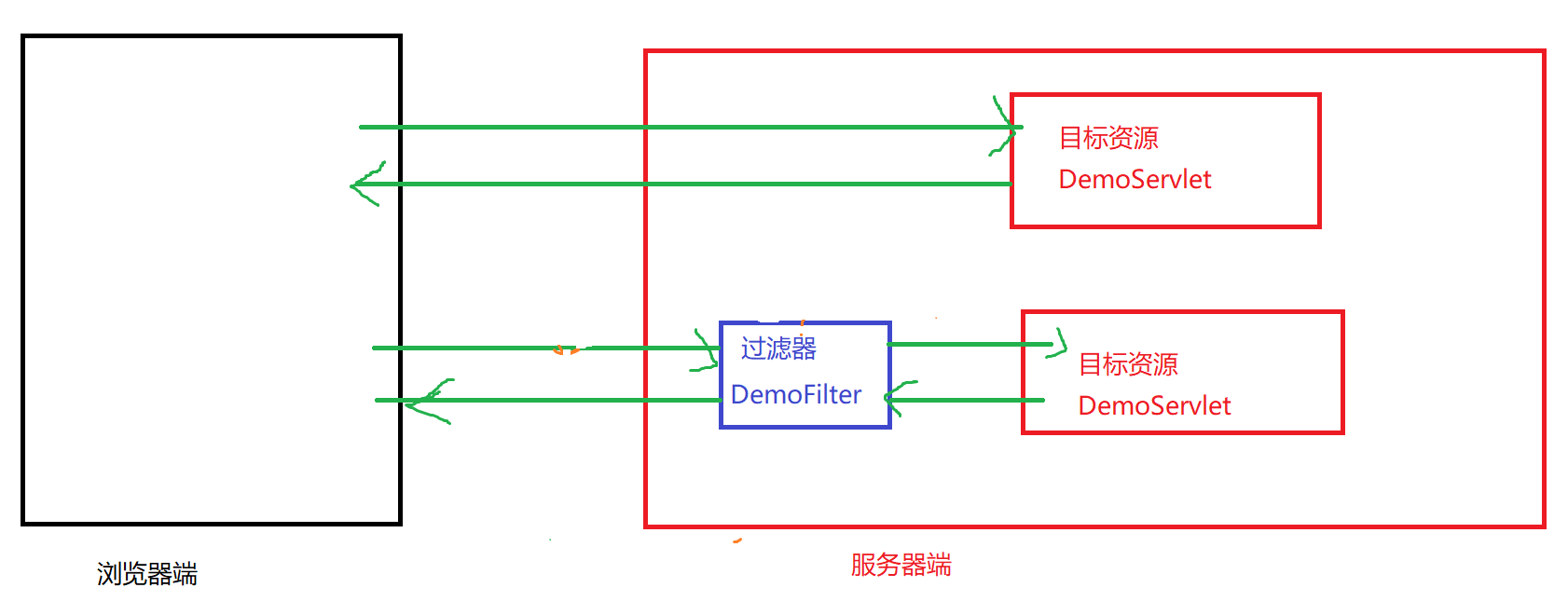
过滤器的创建：web项目第一次部署到tomcat服务器中，tomcat服务器会创建过滤器，自动执行过滤器中的init方法。

过滤器的销毁：web项目从tomcat服务器中移除时，tomcat服务器会销毁过滤器，自动执行过滤器中的destory方法。

Tomcat服务器重新部署（redeploy）web项目的时候，tomcat

服务器会对过滤器销毁和重新过创建，会自动先执行destory方

法，在执行init()方法



当浏览器端发送请求，过滤器拦截到该请求后，执行过滤器

的doFilter核心方法，如果有chain.doFilter(request, response)代

码，会向下一个过滤器或者目标资源进行传递，如果没有该代码，

程序会卡在doFilter核心方法不会往下执行了。