## http状态码

常见的HTTP状态码：

200 ----请求成功

301 ----资源（网页等）被永久转移到其他url

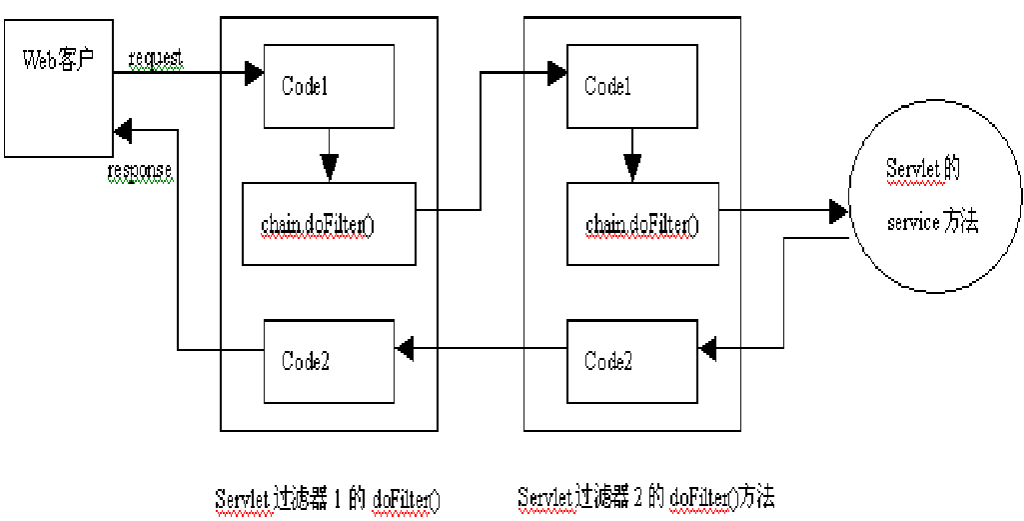
404 ----请求的页面不存在

500 ----内部服务器错误

具体请看：<http://www.runoob.com/http/http-status-codes.html>教程

## Filter学习

首先来看一下执行流程：



过滤器的生命周期，一般经历三个阶段：

初始化：

当容器第一次加载过滤器时，init()方法将被调用，该类在这个方法中包含了一个指向 Filter Config 对象的引用。

过滤：

过滤器的大多数时间都消耗在这里。doFilter方法被容器调用，同时传入分别指向这个请求/响应链中的 Servlet Request、Servlet Response 和 Filter Chain 对象的引用。然后过滤器就有机会处理请求，将处理任务传递给链中的下一个资源(通过调用 Filter Chain 对象引用上的 doFilter方法)，之后在处理控制权返回该过滤器时处理响应。

析构：

容器紧跟在垃圾收集之前调用 destroy()方法，以便能够执行任何必需的清理代码。

其中chain.doFilter(req, resp)方法是把请求转发给过滤器的下一个对象，可以是web.xml里面配置的另外的filter，也可以是servlet。

在这里我们用过滤器配置一个抛出异常到指定错误页面来实践一下。

首先是web.xml的配置：

<filter>

<filter-name>Page Request Timer</filter-name>

<filter-class>cn.filter.UserFilter</filter-class><!-- 指定Filter路径 -->

</filter>

<filter-mapping>

<filter-name>Page Request Timer</filter-name>

<servlet-name>Main Servlet</servlet-name><!-- 指定在哪个servlet下去使用Filter -->

</filter-mapping>

<servlet>

<servlet-name>Main Servlet</servlet-name>

<servlet-class>cn.servlet.FilterAction</servlet-class><!-- 指定Servlet路径 -->

</servlet>

<servlet-mapping>

<servlet-name>Main Servlet</servlet-name>

<url-pattern>/filter</url-pattern><!-- 通过什么请求来访问servlet -->

</servlet-mapping>

然后是Filter类：

package cn.filter;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.annotation.WebFilter;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import cn.exception.BsException;

@WebFilter("/UserFilter")

public class UserFilter implements Filter {

private String errorPage;//跳转的错误信息页

public void destroy() {

}

public void doFilter(ServletRequest req, ServletResponse resp,

FilterChain chain) throws IOException, ServletException {

HttpServletRequest request = (HttpServletRequest) req;

HttpServletResponse response = (HttpServletResponse) resp;

try {

chain.doFilter(req, resp);

} catch (RuntimeException e) {

if(e instanceof BsException){//如果是你定义的业务异常

request.setAttribute("BsException", e);//存储业务异常信息类

errorPage = "error.jsp";

request.getRequestDispatcher(errorPage).forward(request, response);//跳转到信息提示页面！！

}

e.printStackTrace();

}

}

public void init(FilterConfig config) throws ServletException {

}

}

其他的只要写一个错误页面error.jsp，一个异常实体类BsException，一个FileDaoImpl方法即可。在浏览器中输入：<http://localhost:8080/Day0918/filter>。

## 一些注意事项

From表单中action中地址=/项目名/servlet名，用submit按钮提交，前提是servlet中有注解@WebServlet("/UserAction")

Servlet中转发重定向地址从WebContent目录下开始

# Day0919

## 总结：今天主要是通过编写一个注册模块和用户列表的增删改查，来熟悉关于jsp+servlet+jdbc的基本操作，通过画图了解有关servlet的生命周期，http的请求与响应，http状态码等相关流程

## Ajax复习

Ajax的定义主要看四个步骤，分别是：

1. New一个新的xmlHttpRequest对象
2. 设置回调函数
3. 设置路径，请求方式，请求方法
4. 调用send传送数据

代码如下：

var xmlHttp = new XMLHttpRequest();

xmlHttp.onreadystatechange = callback;//不要用callback();

xmlHttp.open("post","/Day0919/RegisterAction?method=2&name="+name,true);

xmlHttp.send(null);

function callback(){

if(xmlHttp.status==200&&xmlHttp.readyState==4){

var flag = xmlHttp.responseText;

if(flag=="true"){

message="当前用户名已被使用";

}else{

message="";

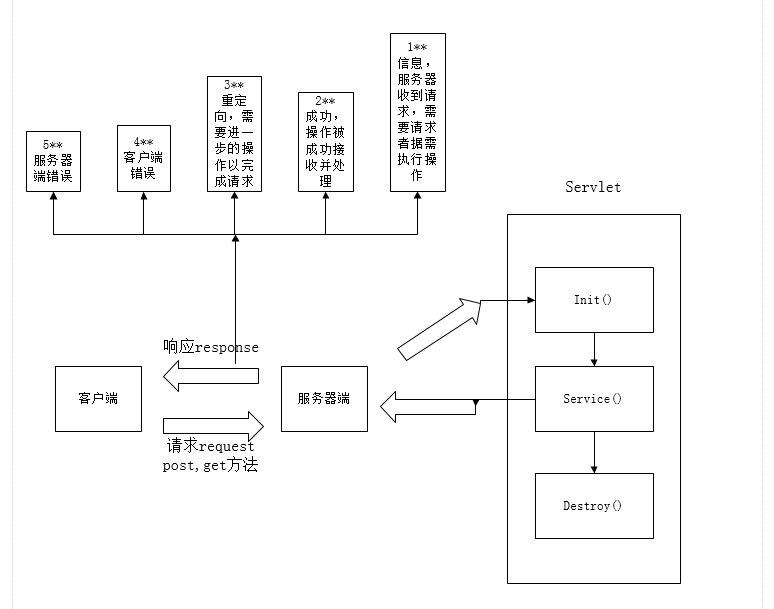
}

document.getElementById("showMsg").innerHTML=message;

}

}

## Servlet生命周期，http请求与响应，http状态码



# Day0920

## maven的功能和用法

1. **为什么要使用maven：**

它可以对项目依赖的jar包进行管理，可以让你的项目保持基本的依赖，排除冗余jar包，并且可以让你非常轻松的对依赖的jar包进行版本升级。而这些仅仅是Maven最基本的功能，它可以在这基础上对项目进行清理、编译、测试、打包、发布等等构建项目的工作。

1. **Maven的用法**

创建一个maven项目：

mvn archetype:generate

-DgroupId　　项目包名com.springmvc

-DartifactId　　项目名称springMVCStudy

-DarchetypeArtifactId　　使用的项目模板

（1）Java工程模版-maven-archetype-quickstart

（2）Web项目模版 maven-archetype-webapp

-DinteractiveMode=false 是否使用交互模式

例如创建一个Javaweb项目：

mvn archetype:generate -DgroupId=com.springmvc -DartifactId=demo-springmvc -DarchetypeArtifactId=maven-archetype-webapp

-DinteractiveMode=false

注意：

1. 控制台窗口要在想要创建项目的目录下去执行以上的命令
2. 进入到项目工程后要手动添加src/main/java、src/test/java、src/test/resources目录 ，少什么补什么
3. **Maven命令**

1．编译源代码： mvn compile

2. 编译测试代码：mvn test-compile

3. 运行测试：mvn test

4．产生site：mvn site

5. 打包：mvn package

6. 在本地Repository中安装jar：mvn install

7. 清除产生的项目：mvn clean

8. 生成eclipse项目：mvn eclipse:eclipse

9. 生成idea项目：mvn idea:idea

10. 组合使用goal命令，如只打包不测试：mvn -Dtest package

11. 编译测试的内容：mvn test-compile

12. 只打jar包: mvn jar:jar

13. 只测试而不编译，也不测试编译：mvn test -skipping compile -skipping test-compile

( -skipping 的灵活运用，当然也可以用于其他组合命令)

14. 清除eclipse的一些系统设置:mvn eclipse:clean

1. **pom.xml文件节点元素说明**：

　<project> pom文件的顶级节点  
<modelVersion>　object model版本，对Maven2和Maven3来说，只能是4.0.0　  
<groupId>　　　　项目创建组织的标识符，一般是域名的倒写  
<artifactId>　　　定义了项目在所属组织的标识符下的唯一标识，一个组织下可以有多个项目  
<version>　　　　当前项目的版本，SNAPSHOT，表示是快照版本，在开发中

　<packaging>　　 打包的方式，有jar、war、ear等  
　<name>　　　　　项目的名称  
　<url>　　　　　　项目的地址

<properties>　　属性配置，比如：

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
<dependencies>　　   构建项目依赖的jar

其中**由groupId、artifactId和version唯一的确定了一个项目坐标**

**注：maven项目几种部署：**

<http://www.micmiu.com/software/build/maven-web-eclipse-deploy/>

## ****SVN的功能与用法****

1. **svn是什么**

svn是版本管理工具，譬如团队进行项目开发，项目代码都储存在服务器上，成员可用svn在本地获得并更新代码

1. **为什么要使用它**
2. 备份工作是十分重要的，您永远不知道计算机上的硬盘何时会坏掉。
3. 版本控制非常重要。您无法保证手头上最新版本永远都是对的。很多时候，在经过数天努力工作之后，您才发现走错方向，需要将所有的修改恢复到数天前的版本。没有几个人能够完全记住自己修改过什么东西。如果没有做好版本控制，那么，最差的状况就是要全部重来。
4. 伙伴间的数据同步也是非常重要的，很多时候，除了您个人外，还有其他的同伴也需要您的档案，怎么把最新的档案提供给伙伴呢，就是通过svn。
5. 如果没有一个更好的办法，备份不同版本是很耗费硬盘空间的。
6. **用法**

详情请看如下教程：

<http://blog.csdn.net/u014486880/article/details/45822405>

## MYSQL语法大全

1. 命令行进入mysql数据库：mysql –u root –p
2. 退出MySQL数据库：exit;
3. 修改密码：首先在DOS 下进入mysql安装路径的bin目录下，然后键入以下命令:

mysqladmin -uroot –p原密码 password 新密码;

1. 增加用户：

#格式:grant 权限 on 数据库.\* to 用户名@登录主机 identified by '密码'

/\*

如，增加一个用户user1密码为password1，让其可以在本机上登录， 并对所有数据库有查询、插入、修改、删除的权限。首先用以root用户连入mysql，然后键入以下命令：

grant select,insert,update,delete on \*.\* to user1@localhost Identified by "password1";

如果希望该用户能够在任何机器上登陆mysql，则将localhost改为"%"。

如果你不想user1有密码，可以再打一个命令将密码去掉。

grant select,insert,update,delete on mydb.\* to user1@localhost identified by "";

\*/

grant all privileges on wpj1105.\* to sunxiao@localhost identified by '123'; #all privileges 所有权限

1. Mysql数据库操作基础
   1. 显示数据库

Show databases;

* 1. 判断是否存在数据库wpj，有的话先删除

Drop database if exists wpj;

* 1. 创建数据库

Create database wpj;

* 1. 删除数据库：

Drop database wpj;

* 1. 使用数据库：

Use wpj;

* 1. 显示数据库的表：

Show tables

* 1. 先判断表是否存在,存在先删除:

drop table if exists student;

* 1. 创建表：

Create table student(

Id int auto\_increment primary key,

Name varchar(50),

Sex varchar(50),

Date varchar(50)

)default charset=utf8;

* 1. 删除表：

drop table student;

* 1. 查看表结构：

Desc student;

* 1. 插入数据：

insert into student values(null,'aa','男','1988-10-2');  
insert into student values(null,'bb','女','1889-03-6');  
insert into student values(null,'cc','男','1889-08-8');  
insert into student values(null,'dd','女','1889-12-8');  
insert into student values(null,'ee','女','1889-09-6');  
insert into student values(null,'ff','null','1889-09-6');

* 1. 查询表中数据

select \* from student;  
select id,name from student;

* 1. 修改某一条数据

update student set sex='男' where id=4;

* 1. 删除数据

delete from student where id=5;

* 1. And 且

select \* from student where date>'1988-1-2' and date<'1988-12-1';

* 1. Or 或

select \* from student where date<'1988-11-2' or date>'1988-12-1';

* 1. Between

select \* from student where date between '1988-1-2' and '1988-12-1';

* 1. 查询指定集合内的数据

select \* from student where id in (1,3,5);

* 1. 排序 asc升序，desc降序

select \* from student order by id asc;

* 1. 分组查询聚合函数

select max(id),name,sex from student group by sex;

select min(date) from student;

select avg(id) as '求平均' from student;

select count(\*) from student;   #统计表中总数

select count(sex) from student;   #统计表中性别总数  若有一条数据中sex为空的话,就不予以统计~

select sum(id) from student;

* 1. 查询从坐标x开始，往后y条数据

select \* from student limit x,y(下标从0开始，包括x);

* 1. 修改表名：

alter table c rename to a;

* 1. 向表中添加一个字段：  
     alter table test add  columnname varchar(20);
  2. 修改表中某个字段的名字

alter table test change name uname varchar(50);

* 1. 导入.sql文件命令（例如D:/mysql.sql）

source d:/mysql.sql;

## JDBC事务和原理

1. 事务的特性：
2. 原子性

事务是数据库的逻辑工作单位，而且是必须是原子工作单位，对于其数据修改，要么全部执行，要么全部不执行。

1. 一致性

事务在完成时，必须是所有的数据都保持一致状态。在相关数据库中，所有规则都必须应用于事务的修改，以保持所有数据的完整性。

1. 隔离性

一个事务的执行不能被其他事务所影响。

1. 持久性

一个事务一旦提交，事物的操作便永久性的保存在DB中。即使此时再执行回滚操作也不能撤消所做的更改。

事务(Transaction):是并发控制的单元，是用户定义的一个操作序列。这些操作要么都做，要么都不做，是一个不可分割的工作单位。通过事务，sql server 能将逻辑相关的一组操作绑定在一起，以便服务器 保持数据的完整性。事务通常是以begin transaction开始，以commit或rollback结束。Commint表示提交，即提交事务的所有操作。具体地说就是将事务中所有对数据的更新写回到磁盘上的物理数据库中去，事务正常结束。Rollback表示回滚，即在事务运行的过程中发生了某种故障，事务不能继续进行，系统将事务中对数据库的所有已完成的操作全部撤消，滚回到事务开始的状态。

1. 事务并发处理可能引起的问题

脏读(dirty read) 一个事务读取了另一个事务尚未提交的数据，

在事务未commit前进行查询

不可重复读(non-repeatable read) 一个事务的操作导致另一个事务前后两次读取到不同的数据

事务A有两次查询，事务B有一次修改操作，顺序是A第一次查询-->B修改，comiit🡪B第二次查询发现数据不一样

幻读(phantom read) 一个事务的操作导致另一个事务前后两次查询的结果数据量不同。

事务A有两次查询数据量（聚合函数），事务B有一次添加或删除操作，顺序是A第一次查询-->B修改，comiit🡪B第二次查询发现数据不一样

举例：

事务A、B并发执行时，

当A事务update后，B事务select读取到A尚未提交的数据，此时A事务rollback，则B读到的数据是无效的"脏"数据。

当B事务select读取数据后，A事务update操作更改B事务select到的数据，此时B事务再次读去该数据，发现前后两次的数据不一样。

当B事务select读取数据后，A事务insert或delete了一条满足A事务的select条件的记录，此时B事务再次select，发现查询到前次不存在的记录("幻影")，或者前次的某个记录不见了。

1. JDBC的事务支持

主要体现在三个方面：

1.自动提交模式(Auto-commit mode)

Connection提供了一个auto-commit的属性来指定事务何时结束。

a.当auto-commit为true时，当每个独立SQL操作的执行完毕，事务立即自动提交，也就是说每个SQL操作都是一个事务。

一个独立SQL操作什么时候算执行完毕，JDBC规范是这样规定的：

对数据操作语言(DML，如insert,update,delete)和数据定义语言(如create,drop)，语句一执行完就视为执行完毕。

对select语句，当与它关联的ResultSet对象关闭时，视为执行完毕。

对存储过程或其他返回多个结果的语句，当与它关联的所有ResultSet对象全部关闭，所有update count(update,delete等语句操作影响的行数)和output parameter(存储过程的输出参数)都已经获取之后，视为执行完毕。

b. 当auto-commit为false时，每个事务都必须显示调用commit方法进行提交，或者显示调用rollback方法进行回滚。auto-commit默认为true。

JDBC提供了5种不同的事务隔离级别，在Connection中进行了定义。

2.事务隔离级别(Transaction Isolation Levels)

JDBC定义了五种事务隔离级别：

TRANSACTION\_NONE JDBC驱动不支持事务

TRANSACTION\_READ\_UNCOMMITTED 允许脏读、不可重复读和幻读。

TRANSACTION\_READ\_COMMITTED 禁止脏读，但允许不可重复读和幻读。

TRANSACTION\_REPEATABLE\_READ 禁止脏读和不可重复读，单运行幻读。Mysql默认事务隔离级别

TRANSACTION\_SERIALIZABLE 禁止脏读、不可重复读和幻读。

3.保存点(SavePoint)

JDBC定义了SavePoint接口，提供在一个更细粒度的事务控制机制。当设置了一个保存点后，可以rollback到该保存点处的状态，而不是rollback整个事务。

Connection接口的setSavepoint和releaseSavepoint方法可以设置和释放保存点。

JDBC规范虽然定义了事务的以上支持行为，但是各个JDBC驱动，数据库厂商对事务的支持程度可能各不相同。如果在程序中任意设置，可能得不到想要的效果。为此，JDBC提供了DatabaseMetaData接口，提供了一系列JDBC特性支持情况的获取方法。比如，通过DatabaseMetaData.supportsTransactionIsolationLevel方法可以判断对事务隔离级别的支持情况，通过DatabaseMetaData.supportsSavepoints方法可以判断对保存点的支持情况。

四．PreparedStatement和statement区别：

PreparedStatement支持动态参数化查询

preparedStatement使用预处理语句查询比statement要快

PreparedStatement可以阻止常见的sql注入攻击

五．InnoDB锁描述

有两种锁分别是：共享锁和排它锁

共享锁：不同事务之间可以查询语句没有影响，增删改会等待锁释放，添加方法：在一个事务的查询语句后面加上lock in share mode

排它锁：增删改查都要等待锁释放，添加方法：在一个事务的查询语句后面加上 for update

注意要在不同事务里面，可以这么操作：命令行mysql下set autocommit = 0;取消自动commit，start transaction开启事务，在eclipse下connection对象调用setAutoCommit(false);禁止自动提交，相当于开启事务，后面要有commit操作和rollback操作。