

servlet-capture-2(1)

Microsoft PowerPoint - [JavaEE实战系列-servlet第二讲.ppt]

servlet开发流程

servlet的生命周期

Servlet 部署在容器里(我们使用的是tomcat,也可能是别的比如boss, weblogic.), 它的生命周期由容器来管理:
Servlet的生命周期分为以下几个阶段:

1. 装载servlet, 由相应的容器来完成。
2. 创建一个Servlet实例。
3. 调用servlet的init()方法, 该方法只在第一次访问该servlet时被调用一次。
4. 服务: 调用servlet的service()方法, 一般业务逻辑在这里处理。该方法在访问该servlet时, 都会被调用。
5. 销毁: 调用servlet的destroy()方法, 销毁该servlet实例。该方法在以下情况被调用:
a: tomcat重新启动 b: reload该Webapp c: 重新启动电脑

School of Continuing Education, Tsinghua University 清华大学继续教育学院

幻灯片 9 / 15

开始

2018/1/22

Microsoft PowerPoint - [JavaEE实战系列-servlet第二讲.ppt]

Servlet开发实例

通过继承GenericServlet 开发servlet

通过GenericServlet去开发servlet,只需要重写service方法
相对来说要简单一些。

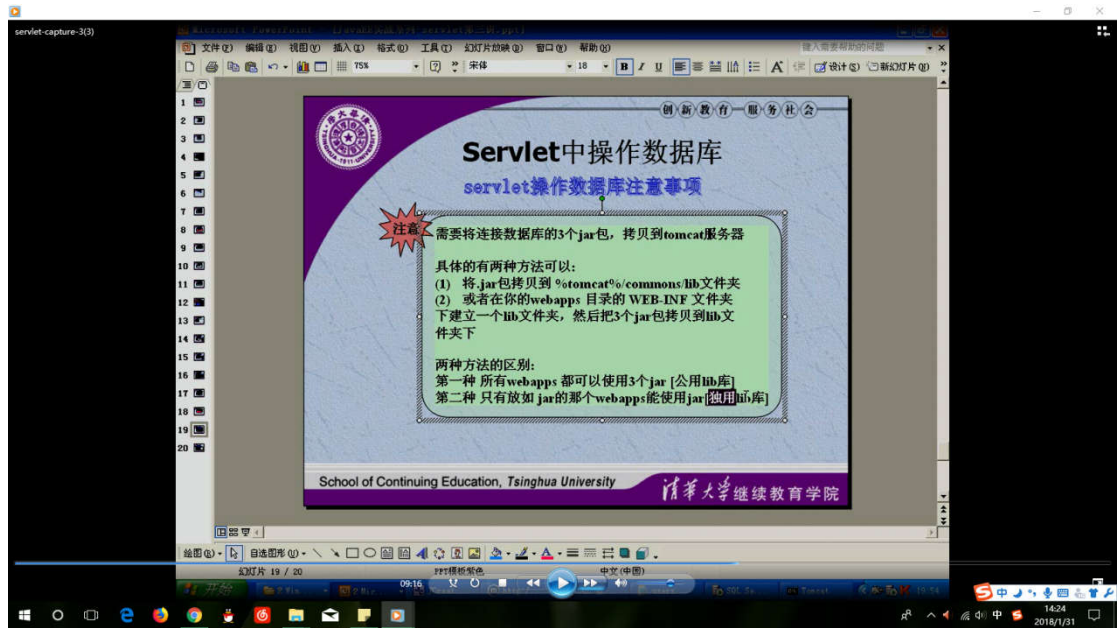
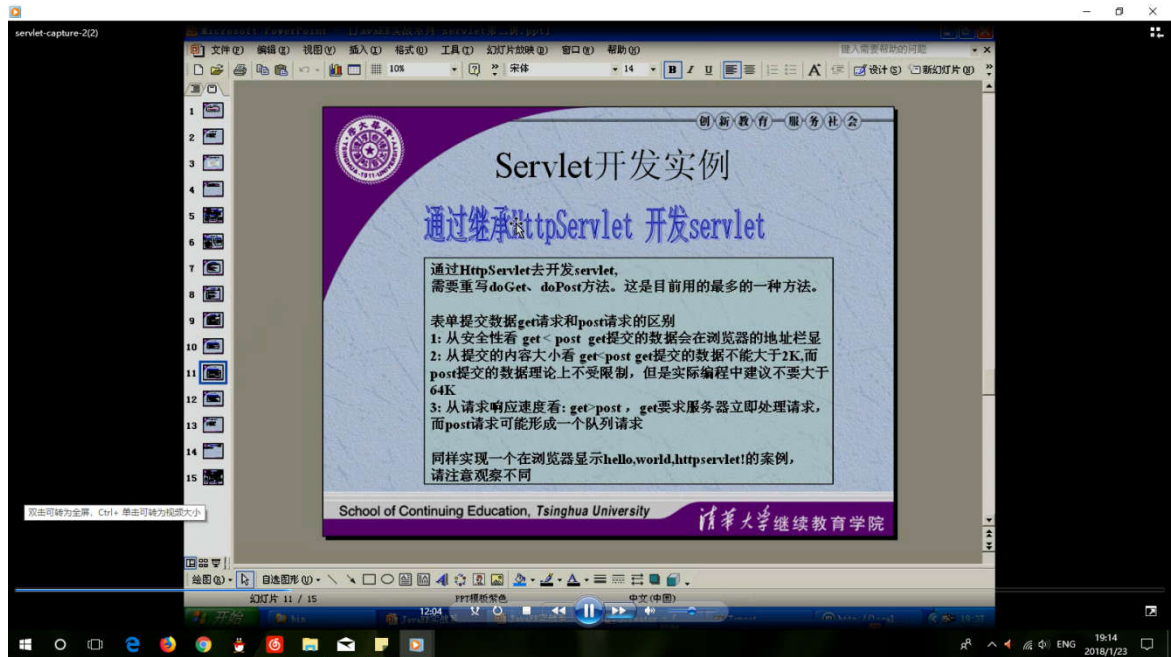
同样实现一个在浏览器显示hello.world,generic!的案例,
请注意观察两种方法的不同

School of Continuing Education, Tsinghua University 清华大学继续教育学院

幻灯片 10 / 15

开始

2018/1/22



分页技术

分页方法(1)

我们先讲讲分页的算法:
 我们需要定义四个变量, 它们有各自的用处
 int pageSize: 每页显示多少条记录 int pageNow: 希望显示第几页
 int pageCount: 一共有多少页 int rowCount: 一共有多少条记录

说明:
 pageSize 是指定的 pageNow 是用户选择的
 rowCount 是从表中查询得到的
 pageCount 是计算出来的 该计算式为
 if(rowCount%pageSize==0){
 pageCount=rowCount/pageSize;
 }else{
 pageCount=rowCount/pageSize+1;
 }

举例说明:
 有用户表 共 9 条记录:
 rowCount=9;
 每页显示 3 条记录: pageSize=3;
 那么根据我们的算法:
 pageCount=3 刚好不多不少
 如果每页显示 4 条记录
 pageSize=4 那么
 pageCount=3, 那么最后一页
 只有 1 条记录
 所以, 不管怎样都是正确的

School of Continuing Education, Tsinghua University

同一用户的不同页面共享数据(cookie)

什么是Cookie(小甜饼)

服务器在客户端保存用户的信息, 比如登录名, 密码等..就是cookie,

这些信息就像是小甜饼一样, 数据量并不大, 服务器端在需要的时候可以从客户端读取, 一般保存在客户端的C:\Documents and Settings目录下, 可以通过右边的图来理解



同一用户的不同页面共享数据(cookie)

Cookie(小甜饼)可以用来做什么

- 1: 保存用户名、密码, 在一定时间不用重新登录
- 2: 记录用户访问网站的喜好(比如有无背景音乐、网页的背景色是什么)
- 3: **网站的个性化**, 比如定制网站的服务, 内容。

用户名 best

密码

验证码 FPP4F

重新计算验证码

☐ 2周内不用再登录

登录 没有注册? 点此处注册



cookie详解

Cookie(小甜饼)其它说明

- 1: 可以通过 IE--工具--internet选项--隐私--高级来启用或是禁用 cookie
- 2: 由于 cookie 的信息是保存在客户端的, 因此安全性不高.
- 3: cookie 信息的生命周期可以在创建时设置(比如 30s), 从创建那一时刻起, 就开始记时, 到时该 cookie 信息就无效了(如果 30s 后)
- 4: 关于 cookie 的其它注意事项, 我们在后面还要讲, 因为 cookie 对 b/s 编程是非常重要的

同一用户的不同页面共享数据

sendRedirect() 方法

通过该方法将信息传递给下一个页面:比如
`sendRedirect("welcome?uname=shunping");`
的形式

优点: 传送信息的速度比较快

缺点: 它只能传送字符串, 而不能传送一个对象

同一用户的不同页面共享数据

sendRedirect() 方法注意点

`sendRedirect("welcome?uname=shunping");`
的形式:

注意点: 1 `welcome`代表你要跳转的那个servlet的url

2 `servlet名`和变量之间有?号

3 如要传递两个以上的值, 它们间要用&号

分开: 比如

`sendRedirect("welcome?uname=shunping&pass=ok");`

4 如果传递的是中文, 那你将得到乱码, 需
处理一下

同一用户的不同页面共享数据

隐藏表单

这是最常见的一种方式,也是最简单的,但有时该技巧非常管用.形如:

```
<from action=login >  
<input type=hidden name=a value=b>  
</from>
```

同一用户的不同页面共享数据(cookie)

Session的注意事项

- 1: session中属性存在的默认时间是30min,你也可修改它存在时间: a 修改web.xml
b 在程序中去修改
- 2: 上面说的这个30min指的是用户的发呆时间,而不是累计时间,这点我会后面详细说明,别急

3: 当某个浏览器访问网站时,服务器会给浏览器分配一个唯一的session id,并以此来区分不同的浏览器(即客户端)

4: 因为session的各个属性要占用服务器的内存,因此软件公司都是在迫不得已的情况下,才使用.

session

属性

名字 String	值 Object



同一用户的不同页面共享数据(cookie)

Session应用实例

session是非常重要的，也是非常有用的技术，在网站开发中可以完成的功能很多，比如：

- 1: 网上商城中的购物车
- 2: 保存登录用户的信息
- 3: 将某些数据放入到Session中，供同一用户的各个页面使用
- 4: 防止用户非法登录到某个页面**

.....

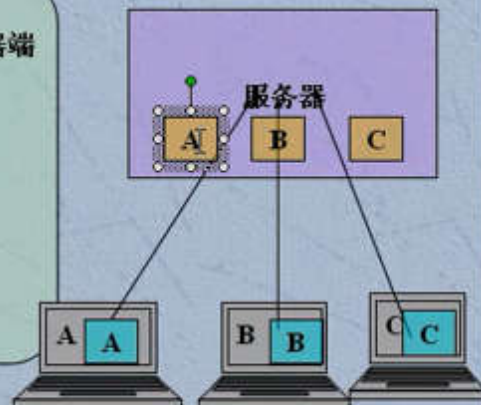
我们使用session完成2、4的功能吧！
又要动手实践了呵呵！




cookie vs session

- 1: 存在的位置
cookie保存在客户端，session保存在服务器端
- 2: 安全性
比较而言，cookie的安全性比session要弱
- 3: 网络传输量
cookie通过网络在客户端与服务器端传输。
而session保存在服务器端，不需要传输。

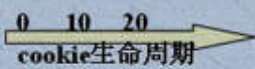
 cookie
 session





创新教育 服务社会

cookie vs session



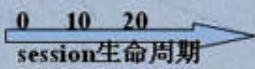
cookie生命周期

4: 生命周期(20分钟为例)

(1) cookie的生命周期是累计的, 从创建时, 就开始计时, 20分钟后cookie生命周期结束, cookie就无效

(2) session的生命周期是间隔的, 从创建时, 开始计时如在20分钟, 没有访问过 session, 那么session信息无效, 如果在20分钟内, 比如第19分钟时, 访问过session, 那么, 它的生命周期将重新开始计算.

(3) 另外, 关机造成session生命周期结果, 但是对cookie没有任何影响



session生命周期

School of Continuing Education, Tsinghua University

清华大学继续教育学院

session机制:

客户端第一次请求服务端时, 服务端会产生一个session对象 (用于保存该客户的信息);
 并且每个session对象 都会有一个唯一的 sessionId (用于区分其他session);
 服务端由会 产生一个cookie, 并且 该cookie的name=JSESSIONID, value=服务端sessionId的值;
 然后 服务端会在 响应客户端的同时 将该cookie发送给客户端, 至此 客户端就有了 一个cookie(JSESSIONID);
 因此, 客户端的cookie就可以和服务端的session一一对应 (JSESSION - sessionId)

第一次请求是就会进行 jsessionid-sessionid 匹配, 但是肯定失败, 英文 jsessionid 不存在
 第二次/n 次访问服务器时, 就每次都会进行匹配查询



分页技术

分页方法(1)

我们先讲讲分页的算法:

我们需要定义四个变量, 它们有各自的用处

int pageSize: 每页显示多少条记录 **int pageNow**: 希望显示第几页
int pageCount: 一共有多少页 **int rowCount**: 一共有多少条记录

说明:

pageSize 是指定的 **pageNow** 是用户选择的
rowCount 是从表中查询得到的

pageCount 是计算出来的 该计算式为

```
if(rowCount%pageSize==0){  
    pageCount=rowCount/pageSize;  
}  
else{  
    pageCount=rowCount/pageSize+1;  
}
```

举例说明:

有用户表 共 9 条记录:

rowCount=9;

每页显示 3 条记录: **pageSize=3;**

那么根据我们的算法:

pageCount=3 刚好不多不少

如果每页显示 4 条记录

pageSize=4 那么

pageCount=3, 那么最后一页

只有 1 条记录

所以, 不管怎样都是正确的



分页技术

分页方法(2)

针对前面提出的问题, 我们可能很自然的想到, 用

select 字段名列表 from 表名 where id between ? and ?

以我们前面的 **users** 表为例, 显示第三页, 该查询语句就是:

select * from users wher userId between 7 and 9

这个 **sql** 语句确实是比较快, 但是它有一个问题, 就是如果表的 **id** 被删除了, 那么, 某页可能就会少一条记录.

我们在 **sql 2000** 的查询分析器中实验一下

因此我们的最终解决方法是如下语句:

**select top pageSize 字段名列表 from 表名 where id not in
(select top pageSize*(pageNow-1) id from 表名)**

以我们前面的 **users** 表为例, 显示第三页, 该查询语句就是:

select top 3 * from users where userId not in (select top 6 userId from users)

创新教育 服务社会

网站框架的改进

网站框架改进方案

为什么在UserBeanCI中我们返回ArrayList集合,而不是直接返回ResultSet

1. 如果返回ResultSet,那么我们在使用ResultSet时,是不能关闭与该ResultSet相互关联的数据库连接等资源,从而造成资源浪费.
2. 如果返回ResultSet,我们只能使用 rs.getInt(?) ,rs.getString(?)...这样的方法来得到结果,代码的可读性不好,维护不方便.

rs

userid	username	password	email	grade
1	admin	admin	admin@tsinghua.com	1
2	shangqing	shangqing	shangqing@tsinghua.com	1
3	tester1	tester1	tester1@tsinghua.com	0
4	tester2	tester2	tester2@tsinghua.com	0
5	tester3	tester3	tester3@tsinghua.com	0
6	tester4	tester4	tester4@tsinghua.com	0
7	tester5	tester5	tester5@tsinghua.com	0
8	tester6	tester6	tester6@tsinghua.com	0
9	tester7	tester7	tester7@tsinghua.com	0
10	tester8	tester8	tester8@tsinghua.com	0

db

School of Continuing Education, Tsinghua University

创新教育 服务社会

网站框架改进方案

rs

userid	username	password	email	grade
1	admin	admin	admin@tsinghua.com	1
2	shangqing	shangqing	shangqing@tsinghua.com	1
3	tester1	tester1	tester1@tsinghua.com	0
4	tester2	tester2	tester2@tsinghua.com	0
5	tester3	tester3	tester3@tsinghua.com	0
6	tester4	tester4	tester4@tsinghua.com	0
7	tester5	tester5	tester5@tsinghua.com	0
8	tester6	tester6	tester6@tsinghua.com	0
9	tester7	tester7	tester7@tsinghua.com	0
10	tester8	tester8	tester8@tsinghua.com	0

db

ArrayList al

1 UserBean ub 2 3 UserBean

根据前面的说明,我们可以利用集合 比如:ArrayList作为中转,这样,就可以尽快的关闭rs,数据库连接...,同时更能体现面向对象编程,代码的可读性更好.

1号线 将rs中的每条记录,封装成一个UserBean 对象 ub
2号线 将封装成一个UserBean 对象 ub放入到ArrayList集合中,利于管理.
3号线 从ArrayList集合中取出ub,然后使用..

School of Continuing Education, Tsinghua University

清华大学继续教育学院

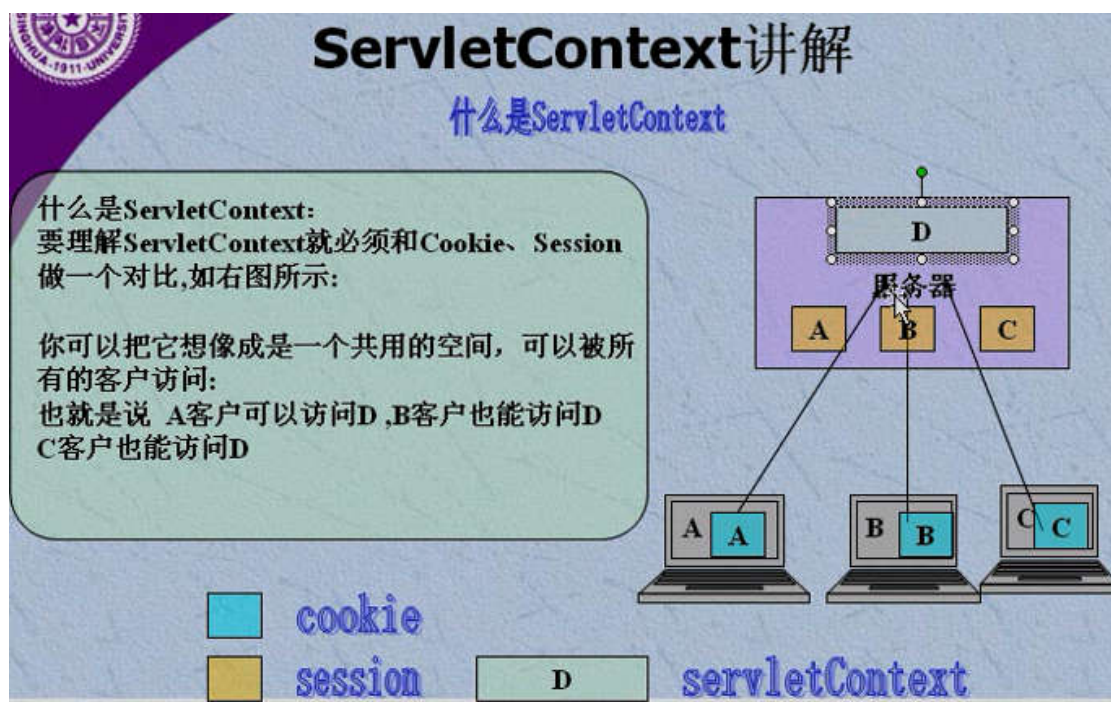
今天用 JCreator 写小程序连接 mysql 5.5l 数据库时出现 ClassNotFoundException :

com.mysql.jdbc.Driver

原因是要将驱动包放入 tomcat,不然 tomcat 找不到该包

在 java 项目中，只需要引入 mysql-connector-java-5.1.7-bin.jar 就可以运行 java 项目。

在 web 项目中，当 `Class.forName("com.mysql.jdbc.Driver");` 时 jcreator 是不会去查找字符串，不会去查找驱动的。所以只需要把 mysql-connector-java-5.1.7-bin.jar 拷贝到 tomcat 下 lib 目录就可以了。



3: 生命周期

ServletContext中的属性的生命周期从创建开始,到服务器关闭而结束

使用ServletContext: 的注意事项

因为存在ServletContext中的数据会长时间的保存在服务器,会占用内存,因此我们建议不要向ServletContext中添加过大的数据...切记

在网站开发中，有很多功能需要使用ServletContext, 比如:

- 1: 网站计数器
- 2: 网站在线用户的显示
- 3: 简单的聊天系统.

2,3 思路?.jsp 会讲

如何修改tomcat的端口

修改tomcat端口的方法:


修改~tomcat~/conf/server.xml 文件中的 port="8080" 数据将8080修改成你需要的端口号即可.

如何设置虚拟目录

为什么要设置虚拟目录

目前，我们的网站站点都是放在默认的目录下~tomcat~/webapps下。但是，在某些情况下，可能需要把站点放在别的目录下，比如:

- ◆ tomcat所在的磁盘空间不够用了。
- ◆ 为了统一管理，希望放在某个特定的目录下，而不是放在默认的 ~tomcat~/webapps下。





给tomcat的管理员设密码



利用密码为空搞破坏过程



- (1) 利用jdk自带的jar工具将有搞破坏的站点打包成*.war文件
这个步骤,首先需要设置路径,命令如下:

set path=%path%;你的jdk目录\bin,这样即可在任何目录下
使用jar命令,然后切换路径到需要打包的那个文件夹目录下;
打包的命令是:jar -cvf war文件名 *.*

- (1) 通过tomcat管理页面将站点发布到tomcat
- (2) 访问有破坏代码的servlet,这样你就被黑了.

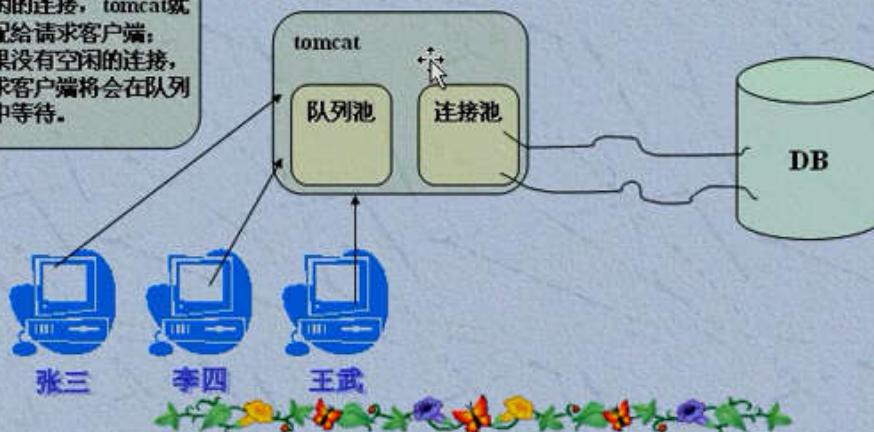


配置数据源和连接池



原理示意图

说明:如果连接池中有
空闲的连接, tomcat就
分配给请求客户端;
如果没有空闲的连接,
请求客户端将会在队列
池中等待。



如何使用

◆ 如果使用连接池的方式来连接数据库，那么就要这样：

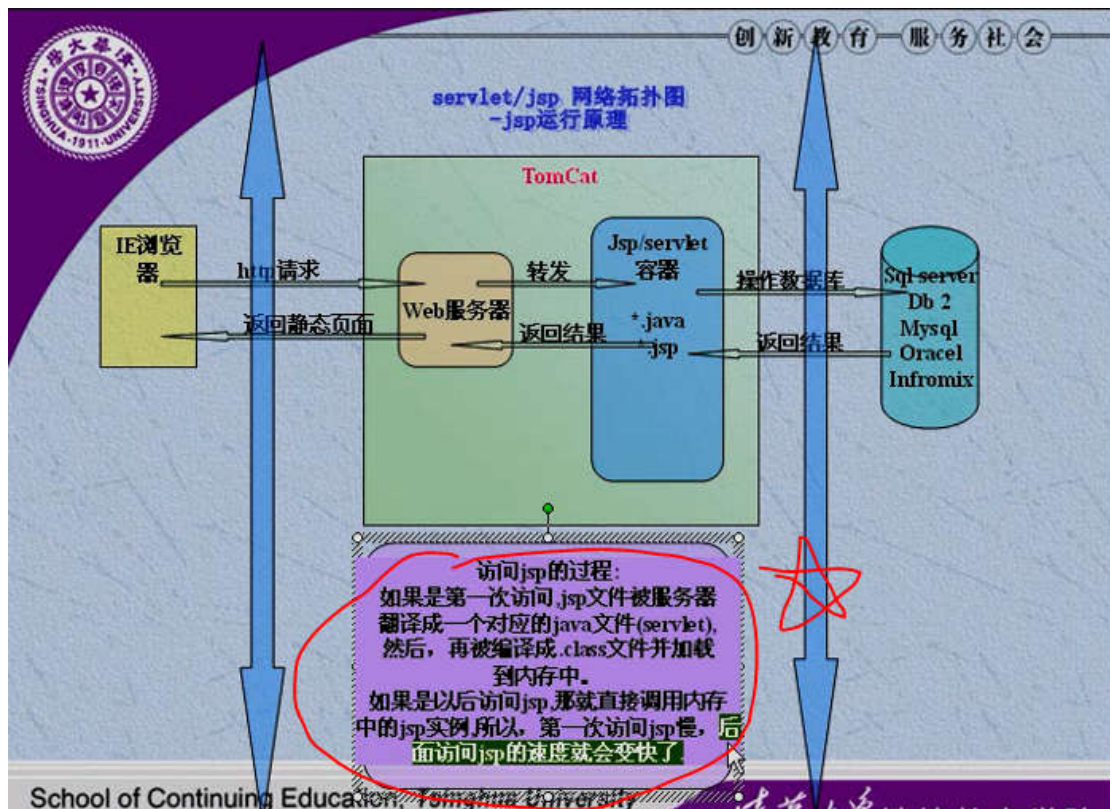
```
Context ctt=new javax.naming.InitialContext();
```

```
DataSource ds=(DataSource)ctt.lookup("java:comp/env/数据源的名");
```

```
ct=ds.getConnection();
```

得到配置
环境，记
住这个是
固定写法

JSP 部分





jsp的概述(5)

--jsp的九大内置对象

- (1) out //向客户端输出数据,字节流
out.println("");
- (2) request //接受客户端的http请求←→servlet中的 HttpServletRequest
getParameter(String name); //name表示表单的参数名
getParameterValues(String name); //使用得到是String []
setAttribute(String name, Object obj); //设置名字为name的obj, 值为obj
getAttribute(String name); //返回由name指定的属性值, 如果不存在就返回null;
getCookie();
- (3) response //封装jsp的产生的回应
addCookie(Cookie cookie);
sendRedirect("/welcome.jsp");
- (4) session //用于保存用户的信息, 跟踪用户的行为
setAttribute(String name, Object obj);
getAttribute(String name);
- (5) application //多个用户共享该对象, 可以做计数器.
- (6) pageContext //代表jsp页面的上下文
- (7) exception //代表运行时的一个异常.
getMessage();
- (8) page //代表jsp这个实例本身 (使用比较少)
- (9) config //代表jsp对应的servlet的配置, 可以得到web.xml中的参数

购物车和用户信息保存用 session

sendRedirect() 是请求从定向, 和超连接是一个意思, 比如你在 A 页面中写一个 request.setAttribute, sendRedirect 到 B 页面, 就是说服务器从 A 页面中给你一个 response, 然后你的浏览器再去 request 到 B 页面, 由于有两次 request 和 response, 是不能在 B 页面取到 request.setAttribute 里的值, 能从地址栏看到 url 的改变。

request.getRequestDispatcher().forward(request, response) 是请求分发器, 比如你在 A 页面中写一个 request.setAttribute, request.getRequestDispatcher().forward(request, response) 到 B 页面, 那就是说服务器给你的 response 是 B 页面的, 并且只有一次 request 和 response, 所以是能在 B 页面取到 request.setAttribute 里的值, 地址栏的 url 仍然是 A 页面的。

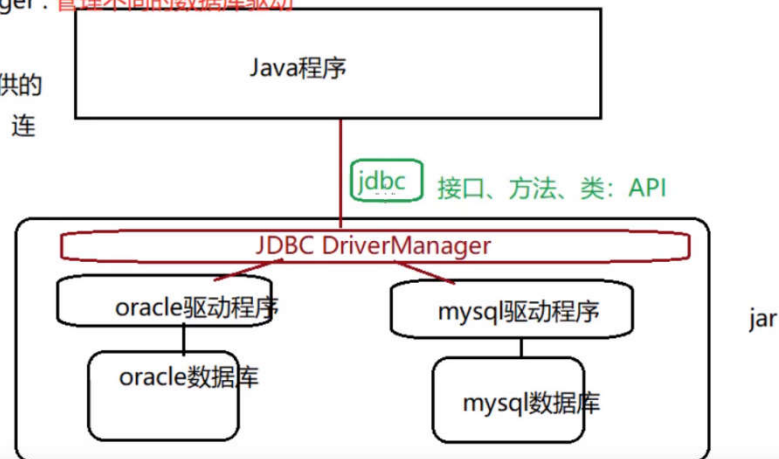
- pageContext 当前页面有效（页面跳转后无效）
 - request 同一次请求有效；其他请求无效（请求转发后有效；重定向后无效）
 - session 同一次会话有效（无论怎么跳转，都有效；关闭/切换浏览器后无效；从 登陆->退出 之间 全部有效）
 - application 全局变量；整个项目运行期间 都有效（切换浏览器 仍然有效）；关闭服务、其他项目 无效
->多个项目共享、重启后仍然有效：JNDI
- 以上的4个范围对象，通过 setAttribute()复制，通过getAttribute()取值；
 - 以上范围对象，尽量使用最小的范围_τ 因为 对象的范围越大，造成的性能损耗越多

JDBC 相关概念

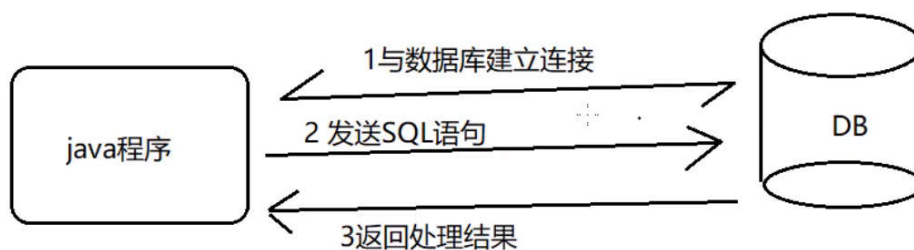
1.JDBC API: 提供各种操作访问接口， Connection Statement PreparedStatement ResultSet

2.JDBC DriverManager: 管理不同的数据库驱动

3. 各种数据库驱动:
相应的数据库厂商提供的
(第三方公司提供)，连接
直接操作数据库



jdbc API



1. JDBC:Java DataBase Connectivity 可以为多种关系型数据库DBMS 提供统一的访问方式，用Java来操作数据
 2. JDBC API 主要功能：
 三件事，具体是通过以下类/接口实现：
 DriverManager：管理jdbc驱动
 Connection：连接
 Statement (PreparedStatement)：增删改查
 CallableStatement：调用数据库中的 存储过程/存储函数
 Result：返回的结果集
 3. jdbc访问数据库的具体步骤：
 a. 导入驱动，加载具体的驱动类
 b. 与数据库建立连接
 c. 发送sql，执行
 d. 处理结果集（查询）

三层组成：

表示层 (USL, User Show Layer；视图层)

-前台：对应于MVC中的View，用于和用户交互、界面的显示
 jsp js html css jquery等web前端技术
 代码位置：WebContent

-后台：对用于MVC中Controller，用于 控制跳转、调用业务逻辑层
 Servlet (SpringMVC Struts2)，位于xxx.servlet包中

业务逻辑层 (BLL, Business Logic Layer；Service层)

-接收表示层的请求 调用
 -组装数据访问层，逻辑性的操作（增删改查，删：查+删），
 一般位于 xxx.service包（也可以成为：xxx.manager，xx.bll）

数据访问层 (DAL, Data Access Layer；Dao层)

-直接访问数据库的操作，原子性的操作（增删改查）
 一般位于 xxx.dao包

mvc 中的 M 对应 Service 和 dao，dao 层有 javabean（封装功能，增删改查）

mvc 中的 m 中 javabean 除了封装功能还要提供数据实体

AOP 相关术语：

