* **MyBatis**

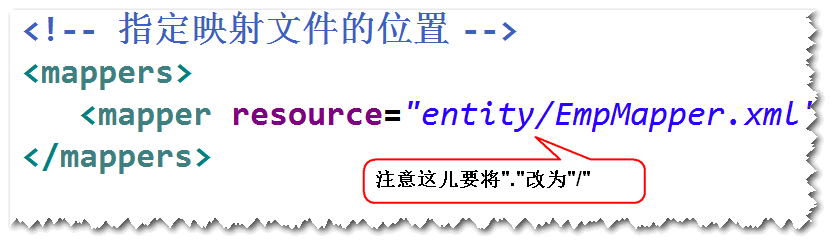
## (1)MyBatis是什么?

开源的持久层框架。 注：MyBatis底层仍然是jdbc。

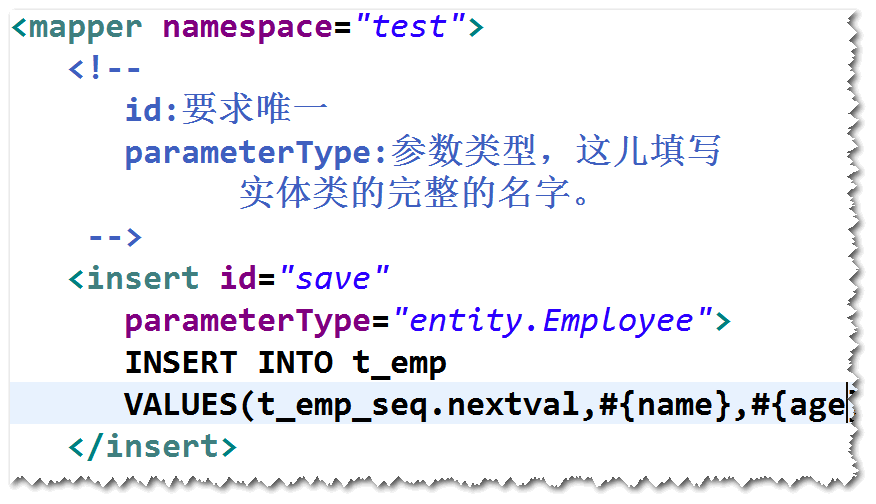
## (2)编程步骤

step1. 导包。 mybatis,ojdbc,junit

step2. 添加配置文件。 注：主要是连接池的配置和映射文件的位置。

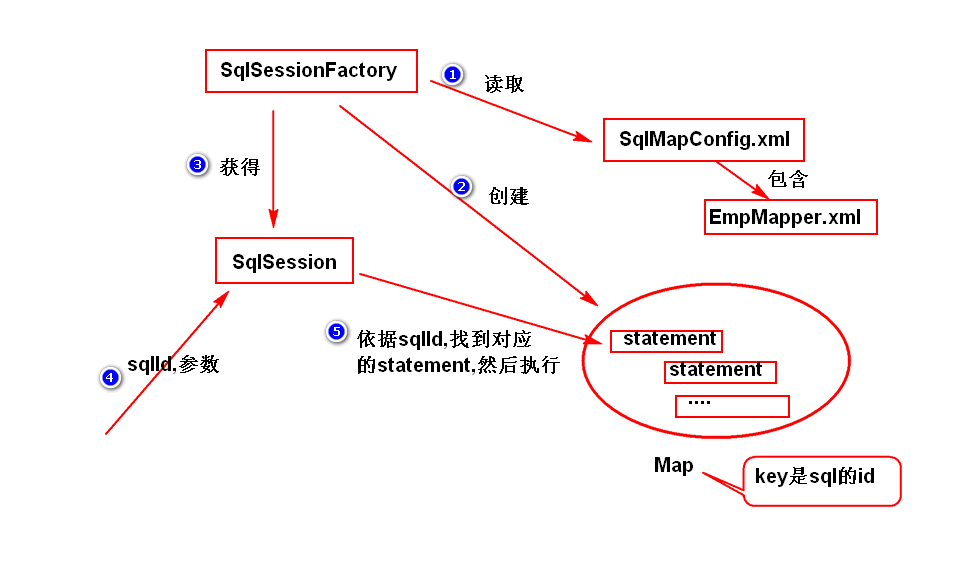


step3. 写实体类。 注：实体类的属性名与表的字段名要一样（大小写可以忽略）。 step4. 添加映射文件。 注：里面主要是sql语句。



step5. 调用SqlSession对象提供的方法来访问数据库。

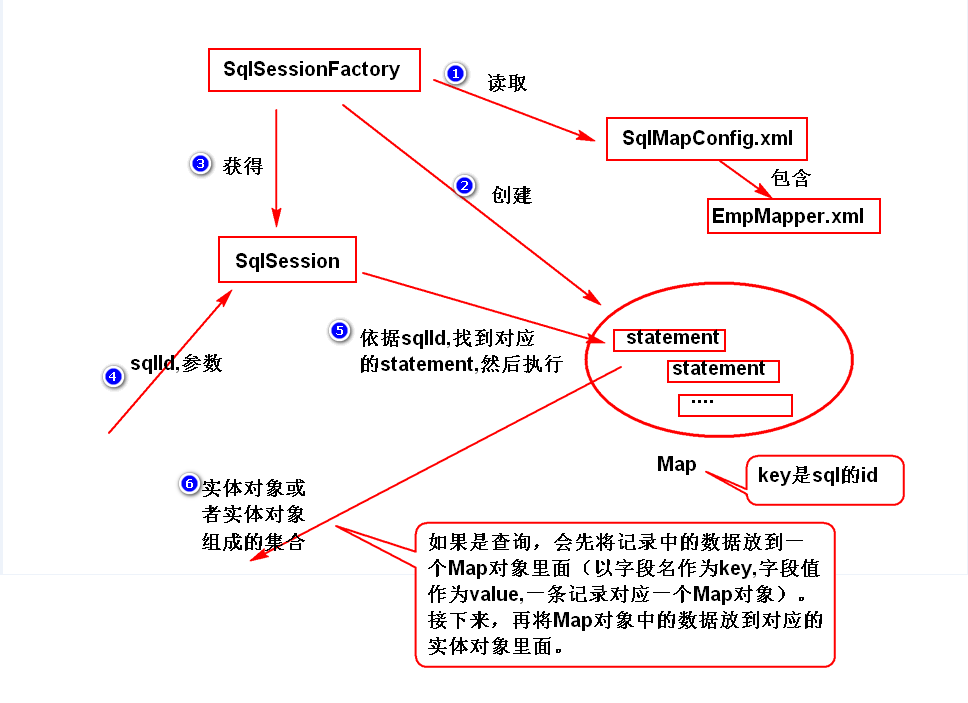
## (3)工作原理 (了解)



**1. 返回Map类型的结果**

**(1) MyBatis会将记录中的数据先放到一个Map对象里面**

（以字段名作为key,以字段值作为value,一条记录对应一个Map对象）， 然后 再将Map中的数据放到对应的实体对象里面。



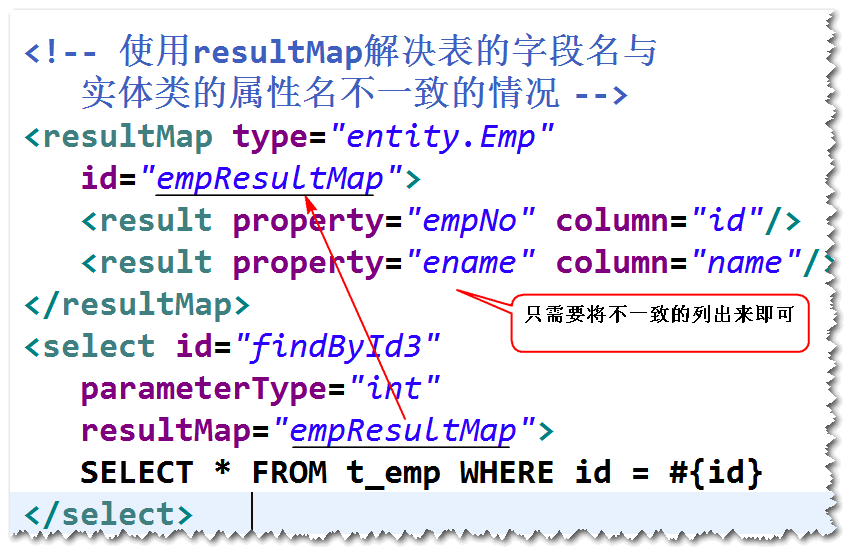
**(2)返回Map类型的结果，好处是不用实体类了，但是不方便**

（因为要获得字段值，还需要调用Map对象提供的get方法, 注意，oracle数据库中，字段名统一都是大写）。

**2.解决字段名与实体类的属性名不一致**

**(1)方式一 使用别名。(就是让别名与属性名一样)。**

**(2)方式二 使用resultMap解决。**



**3.Mapper映射器**

**(1)什么是Mapper映射器?**

符合映射文件要求的接口。 注：MyBatis会生成符合该接口要求的对象。

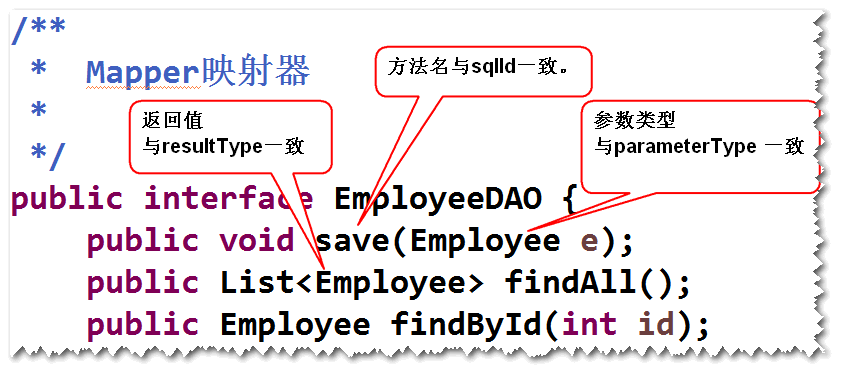
**(2)具体要求**

a.方法名要与sql的id一致。

b.方法的参数类型要与parameterType一致。

c.方法的返回类型要与resultType一致。

d.映射文件的namespace要等于接口的完整的名字。





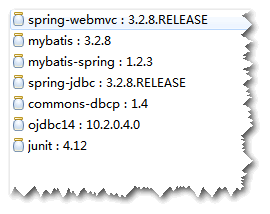
**(3)如何使用Mapper映射器**



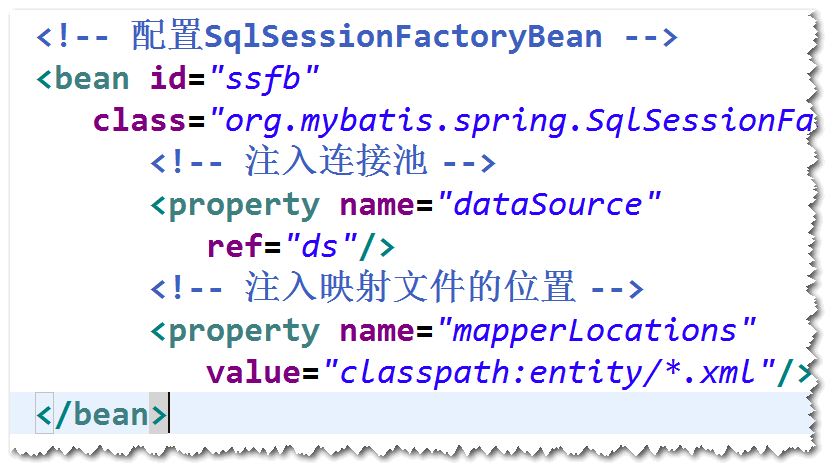
* **Spring集成MyBatis （方式一 使用Mapper映射器）**

**(1)集成步骤**

step1.导包。 spring-webmvc,mybatis,mybatis-spring, dbcp,ojdbc,spring-jdbc,junit。



step2.添加spring配置文件。 注：不再需要MyBatis的配置文件,可以在spring的配置文件里面 添加SqlSessionFactoryBean来代替。



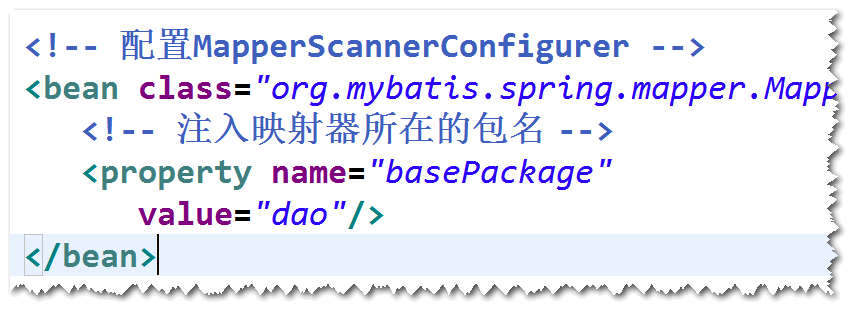
step3.实体类

step4.映射文件

step5.Mapper映射器

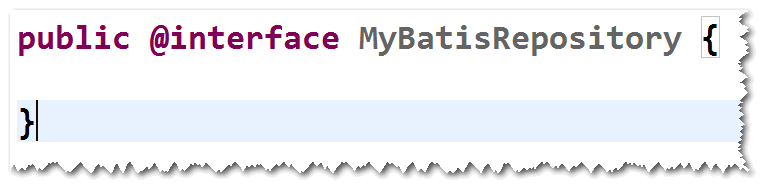
step6.配置MapperScannerConfigurer

注：该bean会扫描指定包及其子包下面的所有的Mapper映射器 (即接口),然后调用getMapper方法获得映射器的实现（比如, 调用 EmployeeDAO dao = SqlSession.getMapper(EmployeeDAO.class))。并且，将这些对象添加到Spring容器里面 (默认的id是首字母小写之后的接口名，可以使用@Repository重命名)。

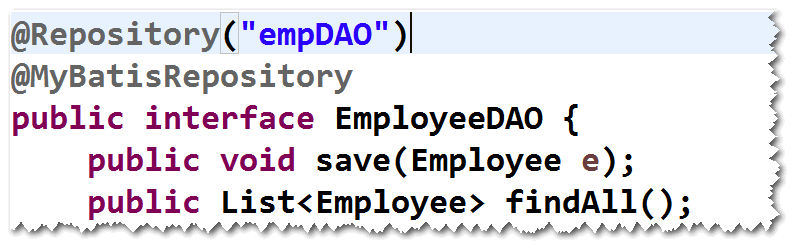


**(2)只扫描特定的接口。**

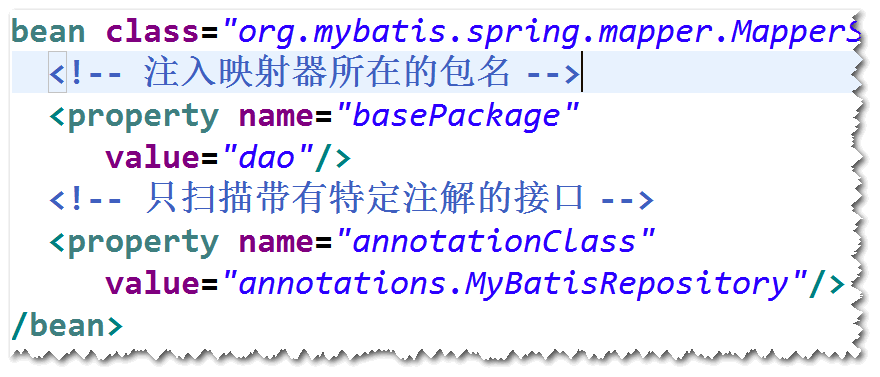
step1. 开发一个注解。



step2. 将该注解添加到需要扫描的接口之上。



step3. 修改MapperScannerConfigurer的配置。



* **Spring集成MyBatis (方式二 了解)**

**(1)集成步骤**

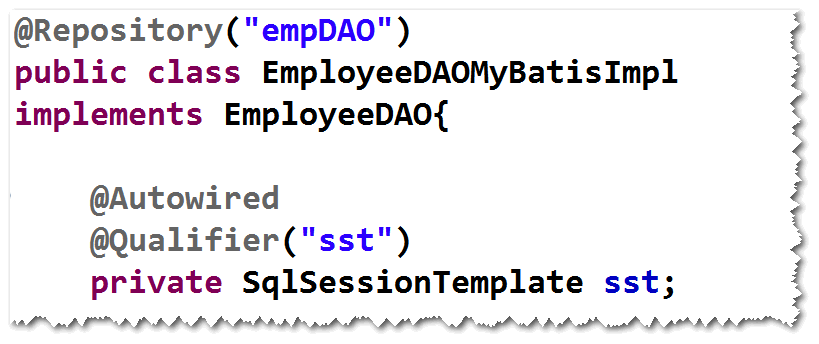
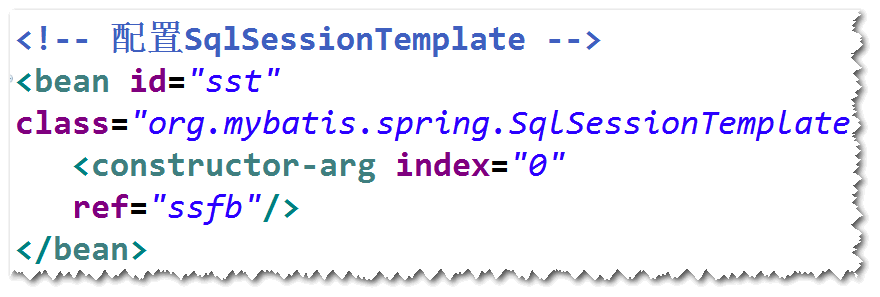
step1.导包。 spring-webmvc,mybatis,mybatis-spring, dbcp,ojdbc,spring-jdbc,junit。

step2.添加spring配置文件。 注：不再需要MyBatis的配置文件,可以在spring的配置文件里面 添加SqlSessionFactoryBean来代替。

step3.实体类

step4.映射文件 注：namespace不再要求等于接口名。

step5.DAO接口 注：接口方法没有特定要求 step6.写一个DAO接口的实现类 注：可以注入SqlSessionTemplate。



**2. 使用Spring集成MyBatis的方式重写AdminDAO**

step1. 导包 需要添加 mybatis,mybatis-spring,spring-jdbc

step2. 在配置文件当中，添加 SqlSessionFactoryBean

step3. 实体类Admin 要注意属性与表的字段名不一样，建议用别名解决

step4. 映射文件 AdminMapper.xml namespace="cn.tedu.netctoss.dao.AdminDAO" <select id="findByAdminCode" parameterType="java.lang.String" resultType="cn.tedu.netctoss.entity.Admin"> SELECT ... </select>

step5. Mapper映射器 (AdminDAO) 不用写了

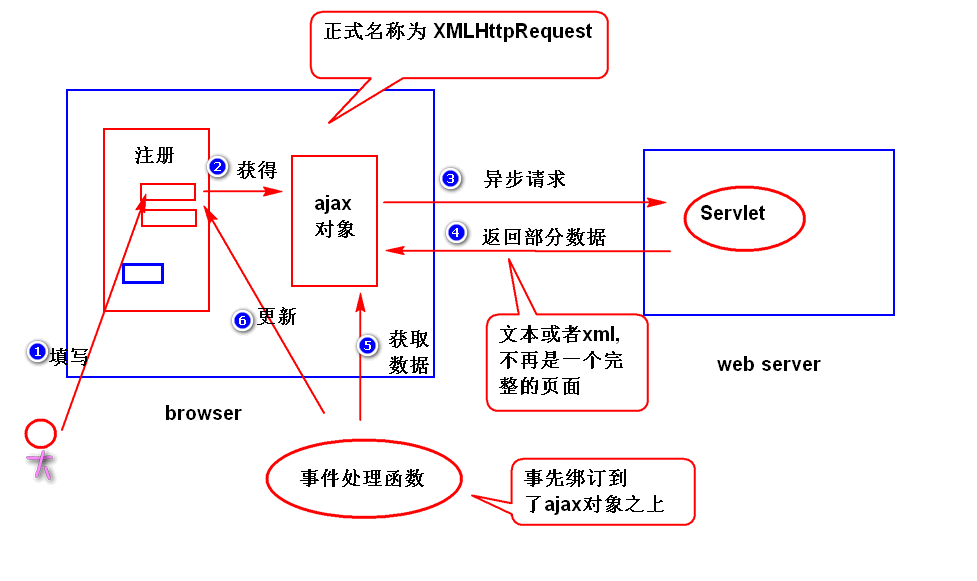
step6. 配置MapperScannerConfigurer

step7. 测试 AdminDAO

* **ajax (asynchronous javascript and xml 异步的javascript和xml)**

**(1)ajax是什么?**

是一种用来改善用户体验的技术，本质上是利用浏览器提供的一个 特殊对象(XMLHttpRequest对象，一般也可以称之为ajax对象)向 服务器发送异步请求；服务器返回部分数据，浏览器利用这些数据 对当前页面做部分更新；整个过程，页面无刷新，不打断用户的操作。 注： 异步请求，指的是，当ajax对象发送请求时，浏览器不会销毁 当前页面，用户仍然可以对当前页面做其它操作。



**(2)如何获得ajax对象?**

function getXhr(){

var xhr = null;

if(window.XMLHttpRequest){

//非ie浏览器

xhr = new XMLHttpRequest();

}else{

xhr = new ActiveXObject('MicroSoft.XMLHttp');

} return xhr; }

**(3)ajax对象的几个重要属性**

onreadystatechange: 绑订事件处理函数，用来处理readystatechange 事件。

注：当ajax对象的readyState属性值发生了任何的改变，比如 从0变成了1，就会产生readystatechange事件。

readyState：有5个值(分别是0,1,2,3,4),用来获取ajax对象与服务 器通信的进展。其中,4表示ajax对象已经获得了服务器返回的所 有的数据。

responseText:获得服务器返回的文本数据。

responseXML:获得服务器返回的xml数据。

status:获得状态码。

**(4)编程步骤**

step1. 获得ajax对象 比如 var xhr = getXhr();

step2. 调用ajax对象的方法，发送请求 方式一 get请求

xhr.open('get','check.do?adminCode=king',true);

xhr.onreadystatechange = f1;

xhr.send(null);

注: true: 异步 (浏览器不会销毁当前页面，用户仍然可以 对当前页面做其它操作) false:同步 (浏览器不会销毁当前页面,但是会锁定当前 页面，用户不能够对当前页面做任何操作)。 方式二 post请求

step3. 编写服务器端的程序 注：不需要返回完整的页面，只需要返回部分数据。

step4. 编写事件处理函数

if(xhr.readyState == 4 && xhr.status == 200)

{ //ajax对象已经获得了服务器返回的所有数据， //而且服务器处理正确。

var txt = xhr.responseText; 页面更新... }

**编码问题**

**(1)get请求**

1)为什么会有乱码? ie浏览器提供的ajax对象，在发送get请求时，会对中文参数 值使用"gbk"来编码，其它浏览器会使用"utf-8"来编码。服务器 端默认会使用"iso-8859-1"来解码。所以会有乱码。

2)如何解决?

step1. 服务器端，统一使用"utf-8"来解码。 可以修改 <Connector URIEncoding="utf-8" />

step2.客户端，使用encodeURI函数对中文参数值进行编码。 注： encodeURI函数是javascript内置的函数，会使用"utf-8" 来编码。

**(2)post请求**

1)为什么会有乱码? 浏览器提供的ajax对象在发送post请求时，对中文参数值会使用 "utf-8"来编码，而服务器端会使用"iso-8859-1"来解码。

2)如何解决? request.setCharacterEncoding("utf-8")

**缓存问题**

**(1)什么是缓存问题?**

如果使用ie浏览器提供的ajax对象发送get请求，会比较请求地址 是否访问过，如果访问过，则不再发送新的请求，而是将第一次访问 的结果显示出来（也就是说，第一次访问时，会将服务器返回的结果缓存下来了）。

**(2)如何解决?**

在请求地址后面添加一个随机数。

* **json (javascript object notation)**

**(1)json是什么?**

是一种轻量级的数据交换格式。

注：json借鉴了javascript的部分语法

注： 数据交换:指的是将要交换的数据转换成一种与平台无关的 数据格式（比如xml）,然后发送给接收方来处理。

轻量级：json相对于xml而言，文档更小，解析速度更快。

**(2)语法**

1)表示一个对象 {属性名：属性值,属性值:属性值...}

注意：

a.属性名必须使用双引号括起来。

b.属性值可以string,number,true/false,null,object。

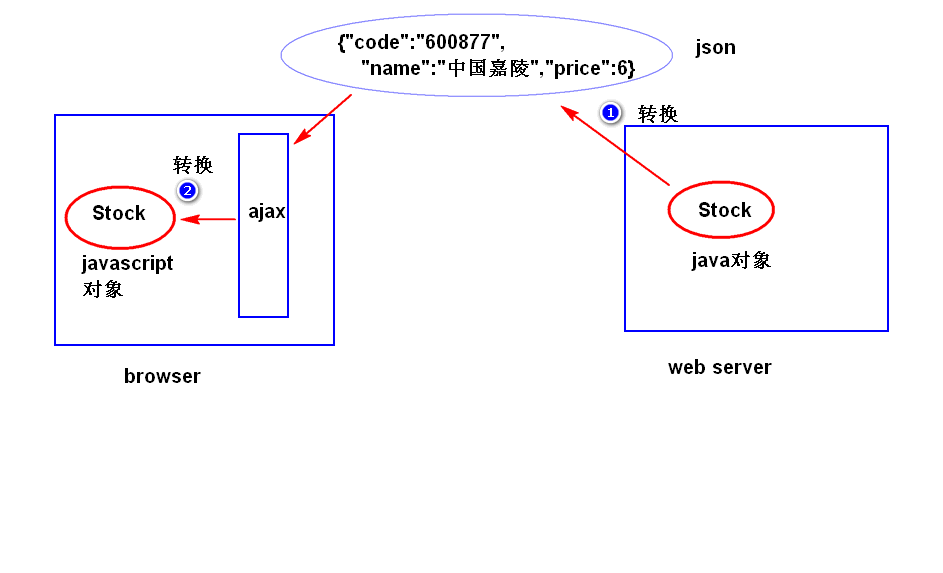
c.属性值如果是string,必须使用双引号括起来。

2)表示对象组成的数组 [{},{},{}...]

**(3)使用json**

1)java对象如何转换成json字符串? 使用jackson提供的api(ObjectMapper)。

2)将json字符串转换成javascript对象? 使用javascript内置对象JSON提供的parse()函数。



* **jQuery对ajax编程的支持**

**(1)$.ajax方法**

1)用法: $.ajax({});

注：{}是一个对象，其属性用来控制ajax对象如何向服务器发送请求，常见的属性如下。

url: 请求地址

type: 请求类型

data: 请求参数(可以是请求字符串形式，比如 "adminCode=king&age=22", 还可以是对象形式，比如{"adminCode":"king","age":22}) dataType: 服务器返回的数据类型,有如下类型： "json" json字符串 "text" 文本 "html" html文档 "xml" xml文档 "script" javascript脚本

success: 绑订事件处理函数(服务器已经返回了所有数据， 并且没有出错)。

error: 绑订事件处理函数（服务器出错了）

比如: $.ajax({"url":"check.do","type":"get","data":"adminCode=king","dataType":"text", "success":function(obj){ //处理服务器返回的数据了 obj就是服务器端返回的数据。 如果返回的是json字符串，会 自动转换成javascript对象。 }， "error": function(){ } });

**(2) load方法**

1)该方法会向服务器发送异步请求，并且将服务器返回的数据 直接添加到符合要求的节点之上。

2)用法 $obj.load(url,[data]);

url : 是请求地址。

data: 是请求参数。有两种写法（请求字符串和对象的形式）

注: $obj,是jQuery对象