

## 知识点列表

编号	名称	描述	级别
1	http 协议是什么？	对超文本协议有基本了解	*
2	通讯的过程	了解通讯的过程	*
3	数据包的结构	了解并掌握基本的数据包的结构	*
4	MyEclipse 工具演示：TCP/IP Monitor	了解 MyEclipse 提供的抓包工具	*
5	如何获得请求参数值	掌握表单处理中如何获取请求参数	**
6	如何处理表单中的中文（难点）	重点掌握表单处理中的中文乱码解决方案，	***
7	MySql 安装步骤	了解 MySql 数据库的下载与安装是注意事项	*
8	mysql 的简单使用	掌握 Mysql 数据库的基本操作，学习并掌握与 Oracle 数据库之间基本用法的一些差异。	*
9	访问数据库步骤	重点掌握，如何利用 Servlet 访问数据库。	**

注：    \*\*"理解级别    \*\*\*"掌握级别    \*\*\*\*"应用级别

### 1. 知识点回顾

#### 什么是 servlet？

用于扩展 web 服务器功能的组件与规范。

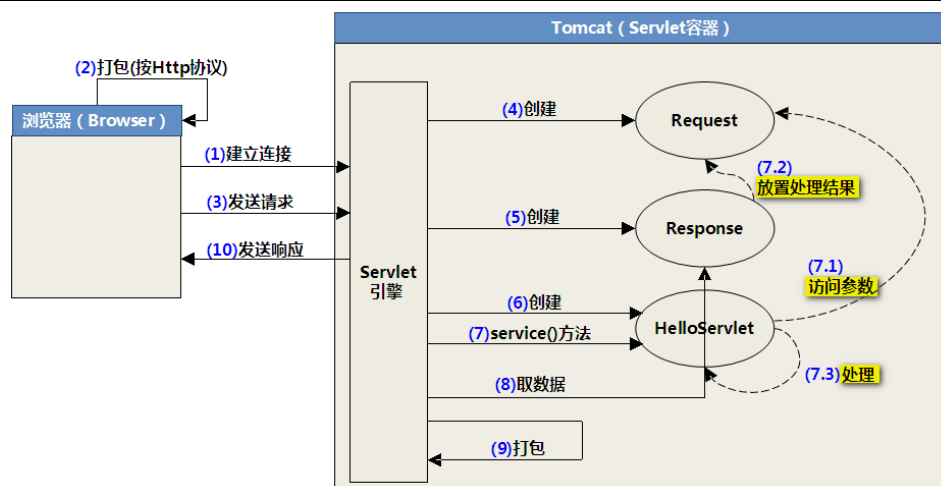
#### 扩展 web 服务器功能？

让服务器能够生成动态的页面

#### Servlet 是一种组件和规范？

Servlet 写好后可以在任何服务器上运行，但是有个前提，要求服务器必须遵守这些组件规范。

#### Servlet 运行图



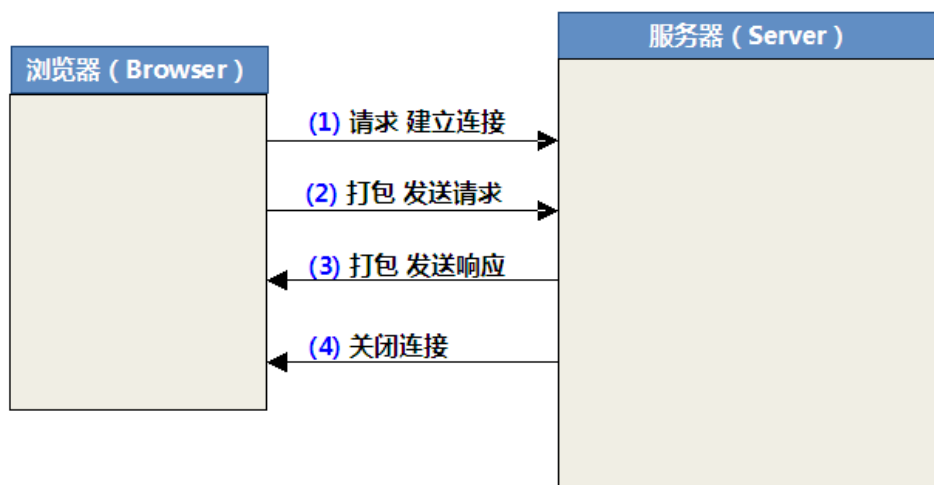
- ✓ 在程序开发过程中，我们只需要注意（7.1）从 Request 对象中访问参数、（7.2）将处理结果放置到 Response 对象中、（7.3）HelloServlet 实例处理这三部分即可。

## 2. http 协议(了解)

### 2.1. http 协议是什么? \*

超文本传输控制协议(hypertext transport protocol)。是一种应用层协议，定义了浏览器(也可以是其它程序)与 web 服务器之间通讯的过程与数据的格式。

### 2.2. 通讯的过程 \*



- 1) 浏览器向服务器发送建立连接的请求。
- 2) 浏览器先将请求数据打包，向服务器发送请求。
- 3) 服务器处理完请求，然后将数据打包，发送给浏览器。
- 4) 服务器发送完数据，并闭连接。

如果浏览器要向服务器再次发送请求，需要重新建立连接。也就是说，浏览器与服务器之间的连接，只能处理一次请求，然后立即关闭。这种通讯方式，可以让服务器以有限的资源为更多的客户端服务。

## 2.3. 数据包的结构 \*

### 1) 请求数据包的结构

第一部分：**请求行（数据包中的一行内容）**

请求行包括三部分内容：

- ✓ 请求方式(get/post)
- ✓ 请求资源路径（端口号之后的内容，比如/appname/servlet）
- ✓ 协议的类型与版本

第二部分：**若干消息头（消息头是由 w3c 定义的一些有特殊含义的键值对）**

消息头的样式，比如: **content-type= text/html;**

服务器和浏览器都会遵守这些消息头的约定。

消息头一般由服务器或者浏览器自动生成，但是也可以通过编程的方式生成

第三部分：**实体内容**

如果请求方式是 **post 方式**，请求参数及值会放在这儿。

如果请求方式是 **get 方式**，请求参数与值是包含在**请求资源路径**里面。

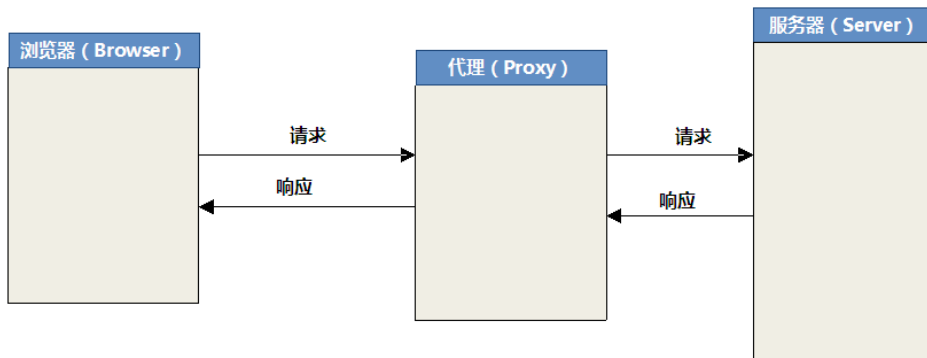
### 2) 响应数据包的结构

- 第一部分：**状态行**  
 协议的类型与版本  
 状态码(状态码是一个数字，不同的数字代表不同的含义，比如  
 ✓ 500: 系统错误(即程序代码有误)  
 ✓ 404: 找不到资源(访问路径错误)  
 ✓ 200: 正确  
 状态码的描述
- 第二部分：**若干消息头**
- 第三部分：**实体内容**  
 服务器返回给浏览器的处理结果

## 2.4. MyEclipse 工具演示：TCP/IP Monitor \*

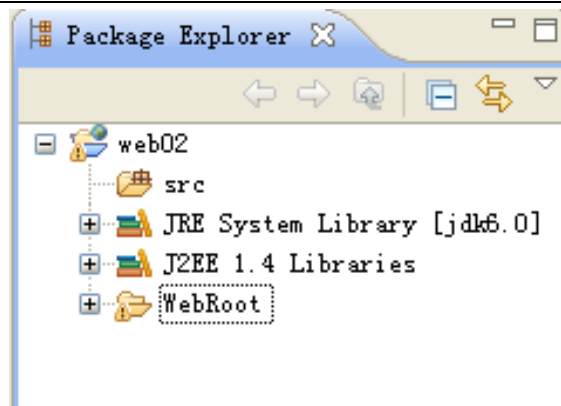
TCP/IP Monitor 相当于一个代理服务器。

**代理服务器原理**



### 步骤 1

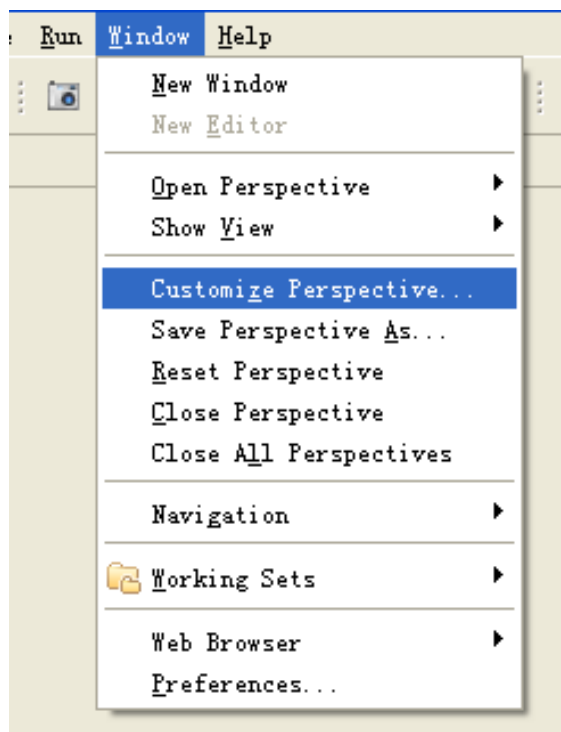
新建 web02 工程



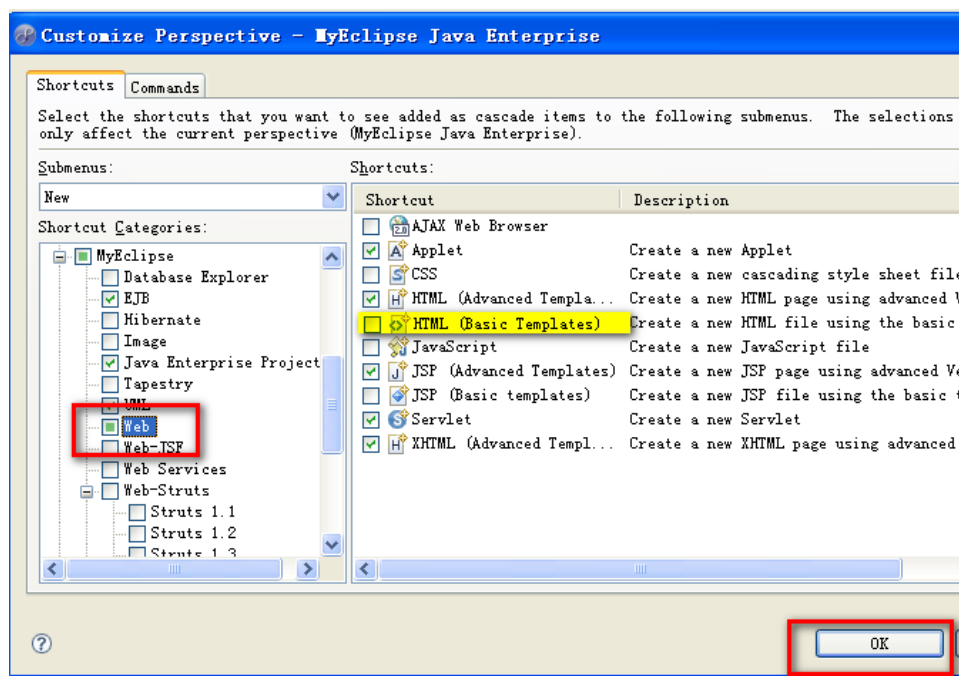
## 步骤 2 (可选)

MyEclipse 小技巧 (定制创建的文件模板)

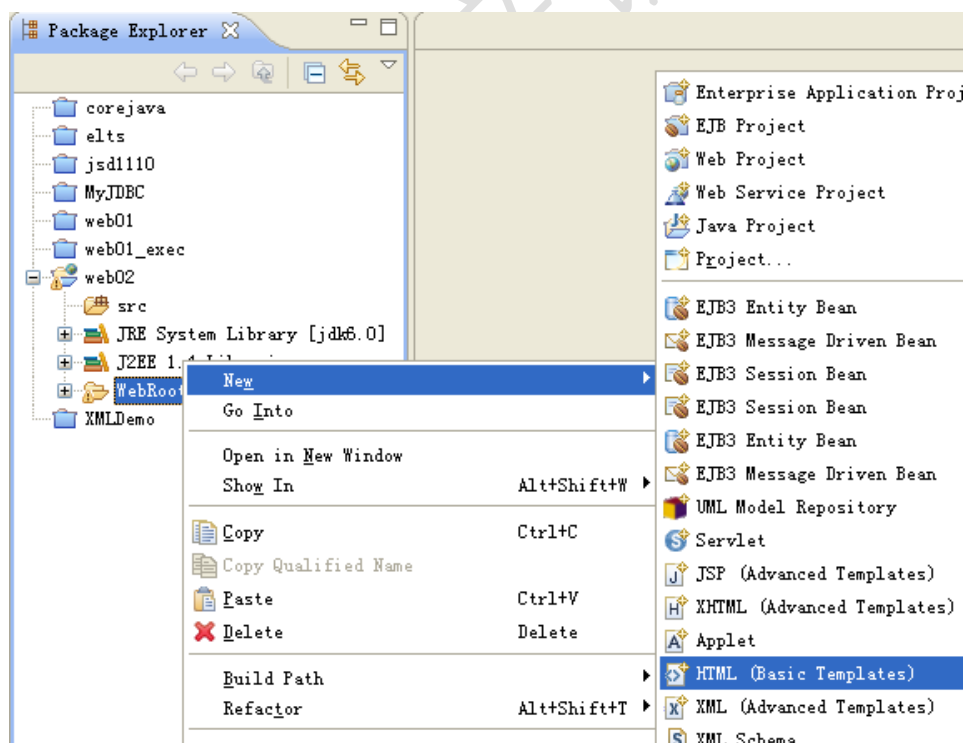
1) “Customize Perspective” 选项



2) 选择你需要的模板 (此处我需要 HTML Basic Templates)

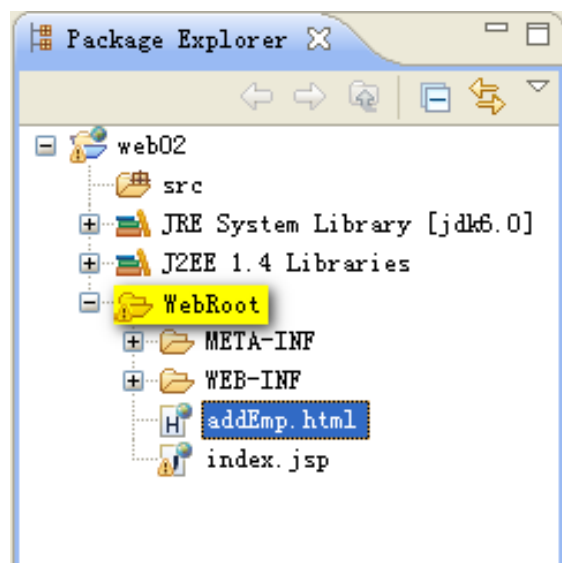


### 3) 新建文档时就可以找到你定制模板类型了



### 步骤 3

在 WebRoot 下新建 addEmp.html



#### 步骤 4

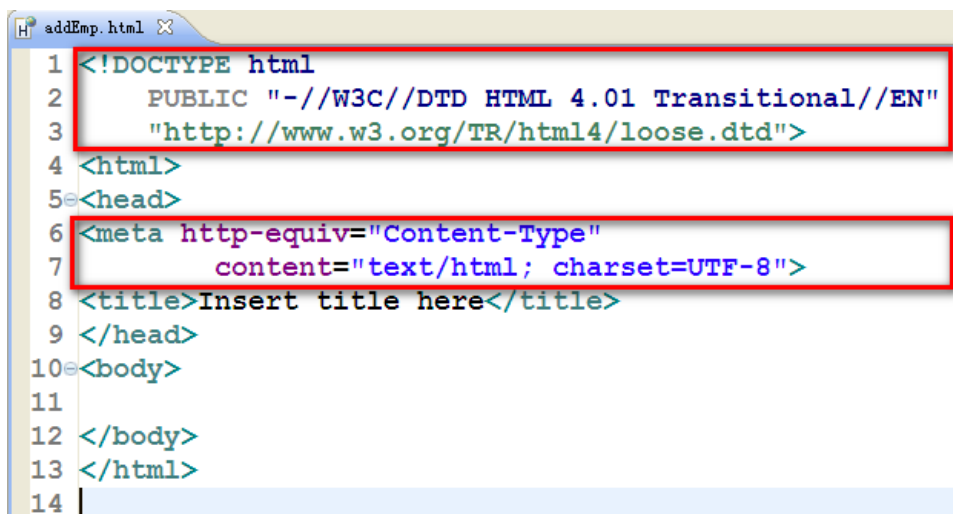
编辑 addEmp.html



#### 说明

使用 MyEclipse 的 HTML Basic Templates 生成的 HTML 模板样式如下

- ✓ `<!DOCTYPE >`在正式开发时是必须加上的；学习练习时则可加可不加
- ✓ `<meta >`标记必须加上，这是一种规范写法，后续会讲



```

1 <!DOCTYPE html
2 PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3 "http://www.w3.org/TR/html4/loose.dtd">
4 <html>
5 <head>
6 <meta http-equiv="Content-Type"
7 content="text/html; charset=UTF-8">
8 <title>Insert title here</title>
9 </head>
10 <body>
11
12 </body>
13 </html>
14

```

#### 步骤 5

部署应用并访问 addEmp.html



### 添加雇员

姓名:

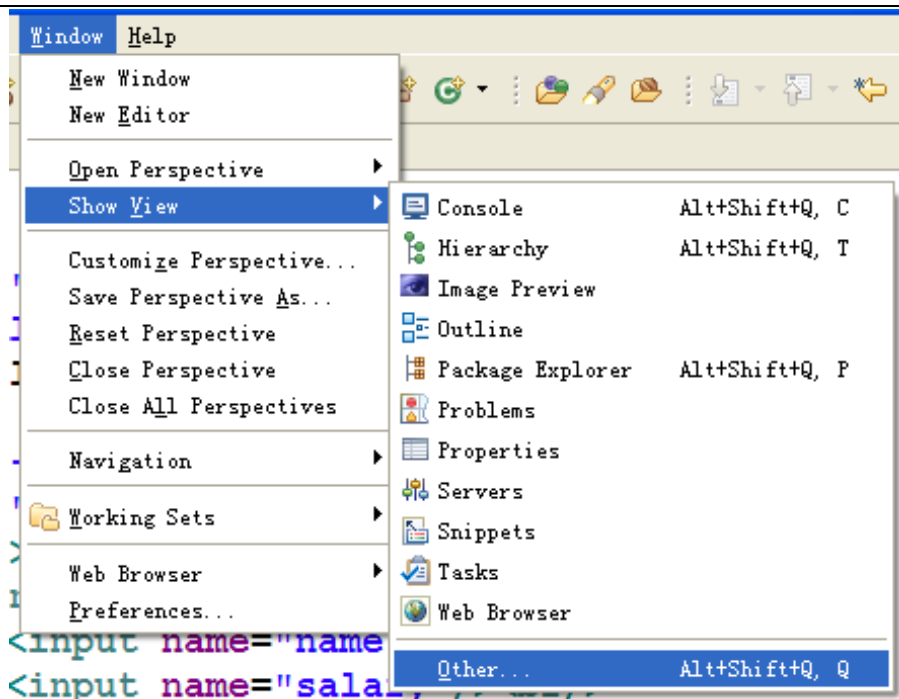
薪水:

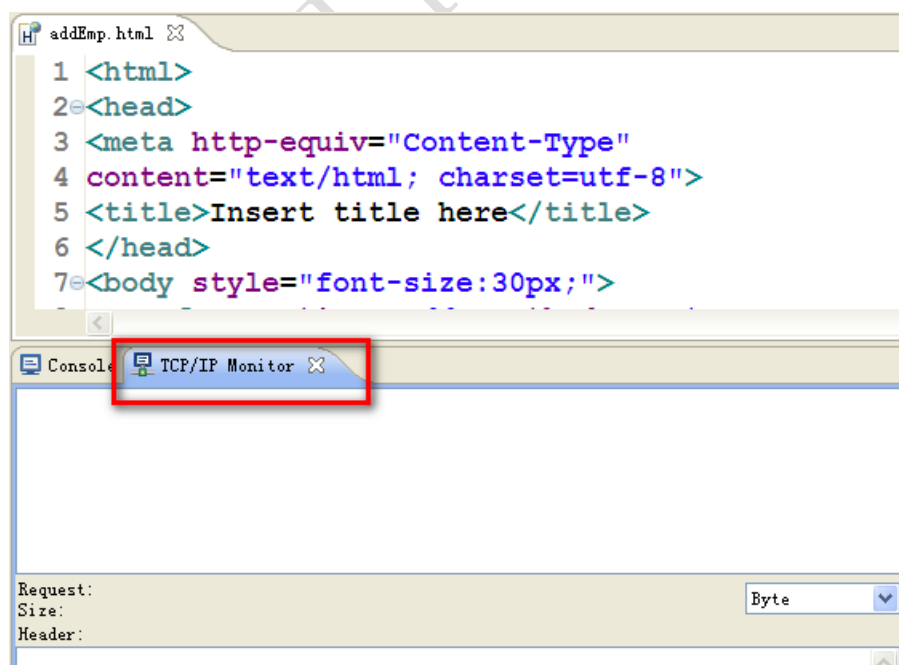
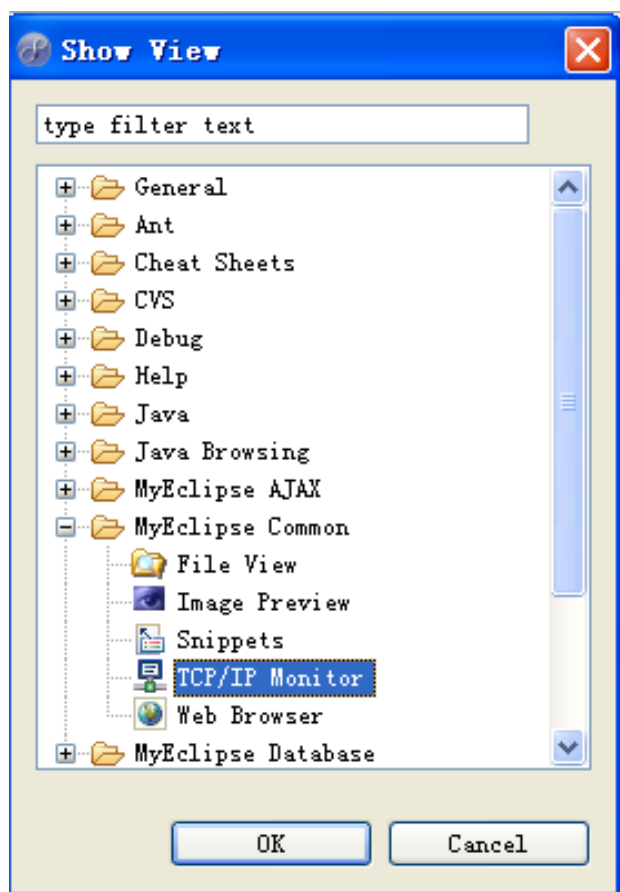
年龄:

#### 步骤 6

打开 TCP/IP Monitor



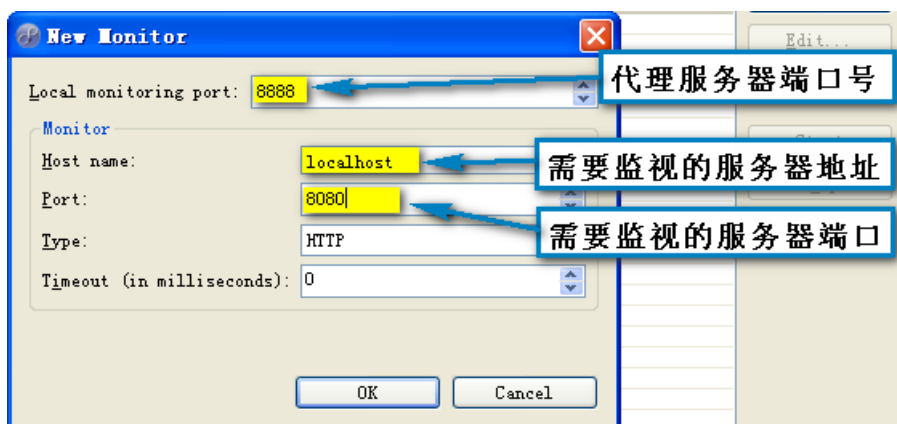




在“TCP/IP Monitor”视图空白处点击右键，出现“Properties”

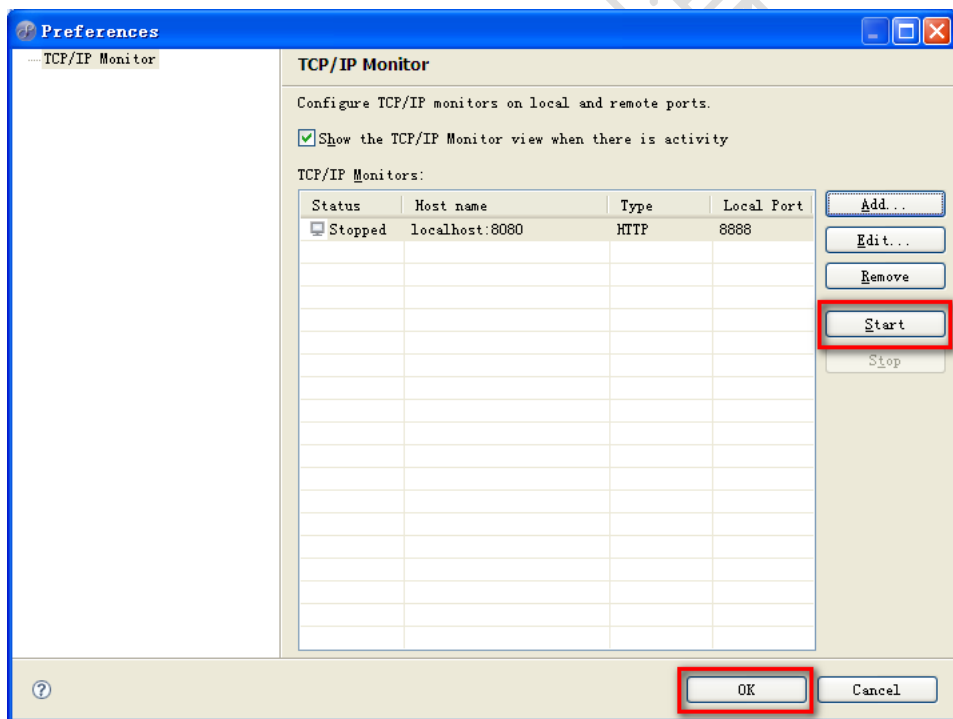
出现对话框点击 “Add” 按钮

## 增加新的监视器



### 步骤 10

点击 “Start” 启动代理服务器，“OK”



### 步骤 11

访问 地址栏输入 “<http://localhost:8888/web02/addEmp.html>”



## 添加雇员

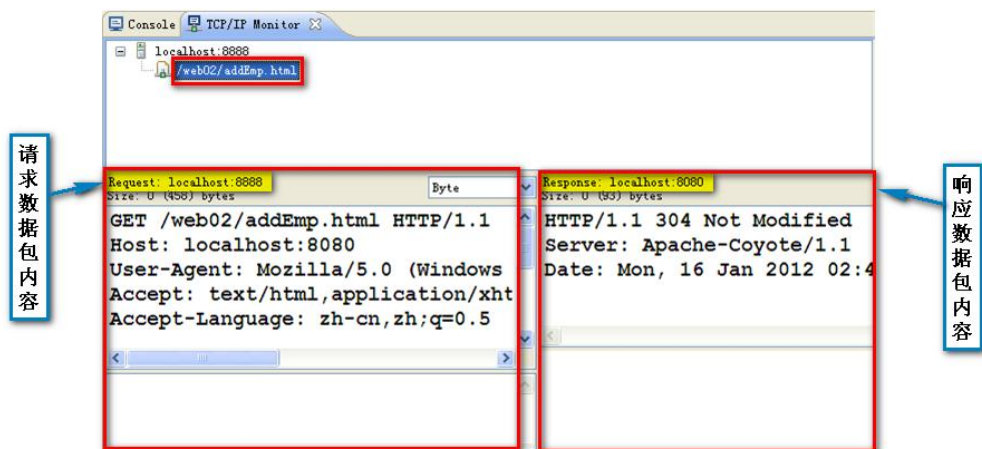
姓名:

薪水:

年龄:

### 步骤 12

查看数据包内容



显示内容说明

### 请求行

GET /web02/addEmp.html HTTP/1.1

请求方式

请求资源路径

协议类型及版本

### 若干消息头

Host: localhost:8888  
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv  
Accept: text/html,application/xhtml+xml,app  
Accept-Language: zh-cn,zh;q=0.5  
Accept-Encoding: gzip, deflate  
Accept-Charset: GB2312,utf-8;q=0.7,\*;q=0.7  
Connection: keep-alive

#### 1) AddEmpServlet.java

```
1 package web;
2
3 import java.io.IOException;
4
15 public class AddEmpServlet extends HttpServlet{
16     public void service(HttpServletRequest request,
17         HttpServletResponse response)
18         throws ServletException, IOException{
19         String name = request.getParameter("name");
20         double salary = Double.parseDouble(
21             request.getParameter("salary"));
22         int age = Integer.parseInt(
23             request.getParameter("age"));
24         System.out.println("name:" + name);
25         System.out.println("salary:" + salary);
26         System.out.println("age:" + age);
27     }
28 }
```

#### 2) addEmp.html

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type"
4 content="text/html; charset=utf-8">
5 <title>addEmp</title>
6 </head>
7 <body style="font-size:30px;">
8     <form action="add" method="get">
9         <fieldset>
10             <legend>添加雇员</legend>
11             姓名:<input name="name"/><br/>
12             薪水:<input name="salary"/><br/>
13             年龄:<input name="age"/><br/>
14             <input type="submit" value="确认"/>
15         </fieldset>
16     </form>
17 </body>
18 </html>
19 |

```

### 3. get/post 请求 \*\*\*

#### 1) 哪一些是 get 请求

- ✓ 在浏览器地址栏直接输入一个地址。
- ✓ 表单默认的提交方式。
- ✓ 点击链接

#### 2) 哪一些是 post 请求

- ✓ 给表单设置 method="post"。

#### 3) get/post 方式的区别

- ✓ get 方式会将请求参数及参数值放在请求资源路径里面,携带的数据大小有限制,不适合提交大量的数据;post 方式会将请求参数及参数值放在实体内容里面,理论上没有限制,适合大量数据的提交。
- ✓ 安全上来讲,post 方式相对安全(因为请求参数及值存放在实体内容里面,而 get 方式会将请求参数及值显示在浏览器地址栏)。但是要注意,post 方式并没有将数据加密。

## 【案例 1】提交方式演示 \*\*

### 1) addEmp.html

```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
<title>addEmp</title>
</head>
<body style="font-size:30px;">
    <form action="add" method="get">        <!--或者改为post-->
        <fieldset>
            <legend>添加雇员</legend>
            姓名:<input name="name"/><br/>
            薪水:<input name="salary"/><br/>
            年龄:<input name="age"/><br/>
            <input type="submit" value="确认"/>
        </fieldset>
    </form>
</body>
</html>
```

### 2) AddEmpServlet.java

```
package web;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```



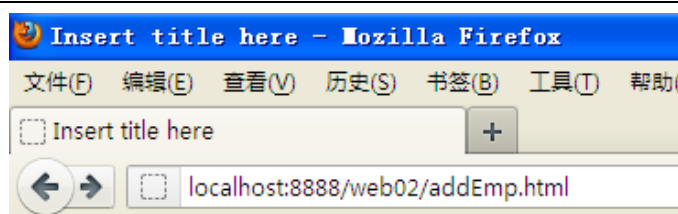
```
public class AddEmpServlet extends HttpServlet{
    public void service(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException,IOException{
        String name = request.getParameter("name");
        double salary = Double.parseDouble(
            request.getParameter("salary"));
        int age = Integer.parseInt(
            request.getParameter("age"));
        System.out.println("name:" + name);
        System.out.println("salary:" + salary);
        System.out.println("age:" + age);
    }
}
```

### 3) web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <servlet>
        <servlet-name>addEmpServlet</servlet-name>
        <servlet-class>web.AddEmpServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>addEmpServlet</servlet-name>
        <url-pattern>/add</url-pattern>
    </servlet-mapping>
</web-app>
```

### 结果演示

#### 1) 访问 addEmp.html



## 添加雇员

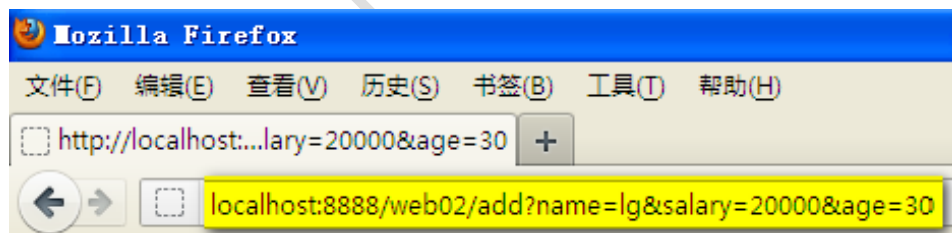
姓名:

薪水:

年龄:

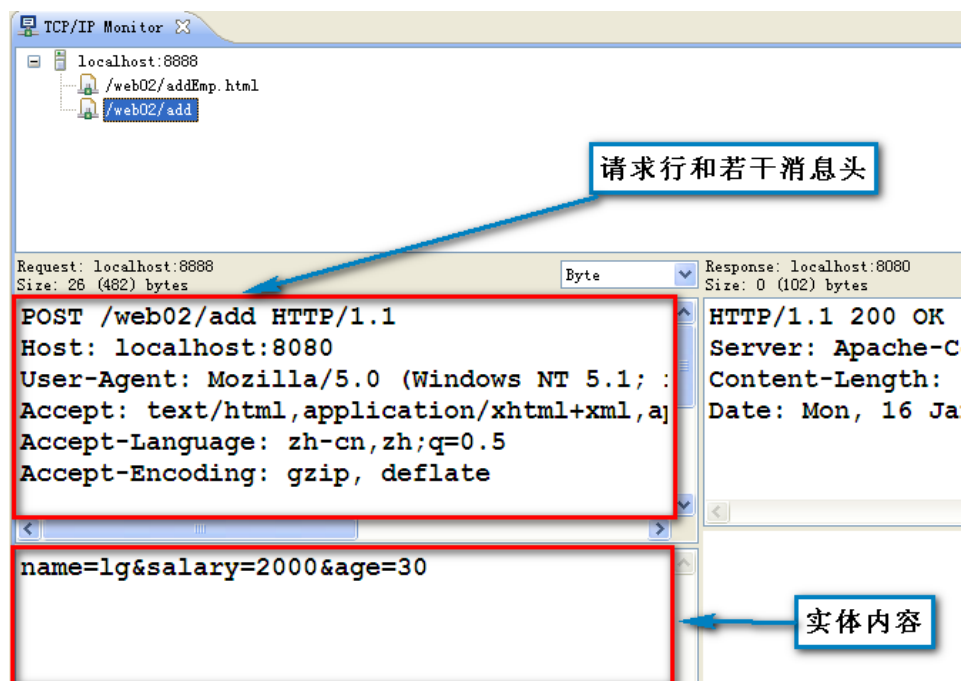
### 2) 使用 get 方式提交

get 方式会将请求参数及参数值放在请求资源路径里面  
携带的数据大小有限制，不适合提交大量的数据



### 3) 使用 post 方式提交

post 方式会将请求参数及参数值放在实体内容里面，理论上没有限制，**适合大量数据的提交**



## 4. 表单处理 \*\*\*

### 4.1. 如何获得请求参数值 \*\*

- 1) String request.getParameter(String paraName);  
要注意的问题: 如果 paraName 与实际的请求参数名不一致, 则返回 null。  
如果没有输入参数值, 则返回 ""。
- 2) String[] request.getParameterValues(String paraName);  
用在有多个请求参数名相同的情况下使用。  
比如 ?interest=fishing&interest=cooking

#### 演示 1

如果 paraName 与实际的请求参数名不一致, 则返回 null ;

#### 1) web.xml

```

7<body style="font-size:30px;">
8    <form action="add" method="post">
9        <fieldset>
10            <legend>添加雇员</legend>
11            姓名:<input name="name"/><br/>
12            薪水:<input name="salary"/><br/>
13            年龄:<input name="age"/><br/>
14            <input type="submit" value="确认"/>
15        </fieldset>
16    </form>
17 </body>

```

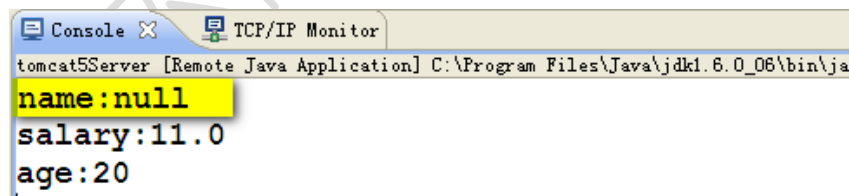
## 2) AddServlet.java

```

1 package web;
2
3 import java.io.IOException;
4
14 public class AddEmpServlet extends HttpServlet{
15     public void service(HttpServletRequest request,
16         HttpServletResponse response)
17         throws ServletException, IOException{
18         String name = request.getParameter("name");
19         double salary = Double.parseDouble(
20             request.getParameter("salary"));
21         int age = Integer.parseInt(
22             request.getParameter("age"));
23         System.out.println("name:" + name);
24         System.out.println("salary:" + salary);
25     }

```

## 3) 控制台打印



```

tomcat5Server [Remote Java Application] C:\Program Files\Java\jdk1.6.0_06\bin\ja
name:null
salary:11.0
age:20

```

## 演示 2

如果没有输入参数值则返回空字符串

### 1) 姓名为空



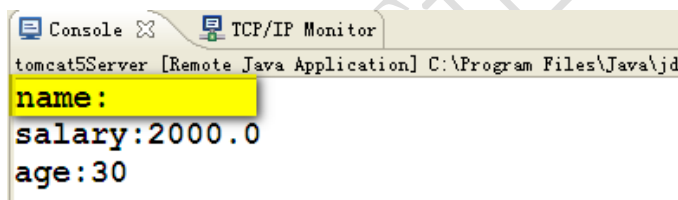
## 添加雇员

姓名:

薪水:

年龄:

### 2) 控制台打印



## 4.2. 如何处理表单中的中文(难点) \*\*\*

### 1) 浏览器会如何对表单中的数据进行编码?

当表单采用 post 方式提交时,浏览器会按照打开该表单所在的页面的编码来对表单中的数据进行编码。

### 2) 在 html 文件当中,添加以下代码的作用

```
<meta http-equiv="content-type" content="text/html;charset=utf-8">
```

作用 1:

模拟 http 消息头(content-type),让浏览器以 utf-8 的编码格式来打开该页面(要确保 html 文件本身确定是使用 utf-8 保存的)。

作用 2:

确保浏览器按指定的编码来对表单中的数据进行编码。

### 3) 中文乱码解决方案

#### step1:

在 html 文件中，添加

```
<meta http-equiv="content-type" content="text/html;charset=utf-8">
```

另外，表单的提交方式必须是 post。

#### step2:

在服务器端，使用 servlet 读取表单中的请求参数时：

```
request.setCharacterEncoding("utf-8");
```

这行代码的作用：设置解码时的编码格式。

#### step3:

如果 servlet 输出中文，要添加如下代码。

```
response.setContentType("text/html;charset=utf-8");
```

这行代码的作用：

作用 1：

指定 out.println 输出时所使用的编码。

作用 2：

生成一个消息头 content-type:text/html;charset=utf-8

告诉浏览器，返回的数据类型是 html，编码是 utf-8。这样，浏览器一定会以指定的编码来显示该页面。

### 演示 1

#### 中文乱码问题

##### 步骤 1 提交数据为中文



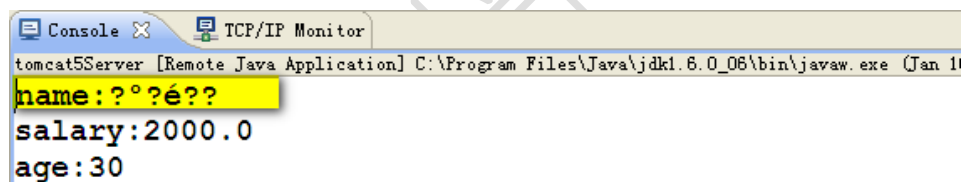
## 添加雇员

姓名:

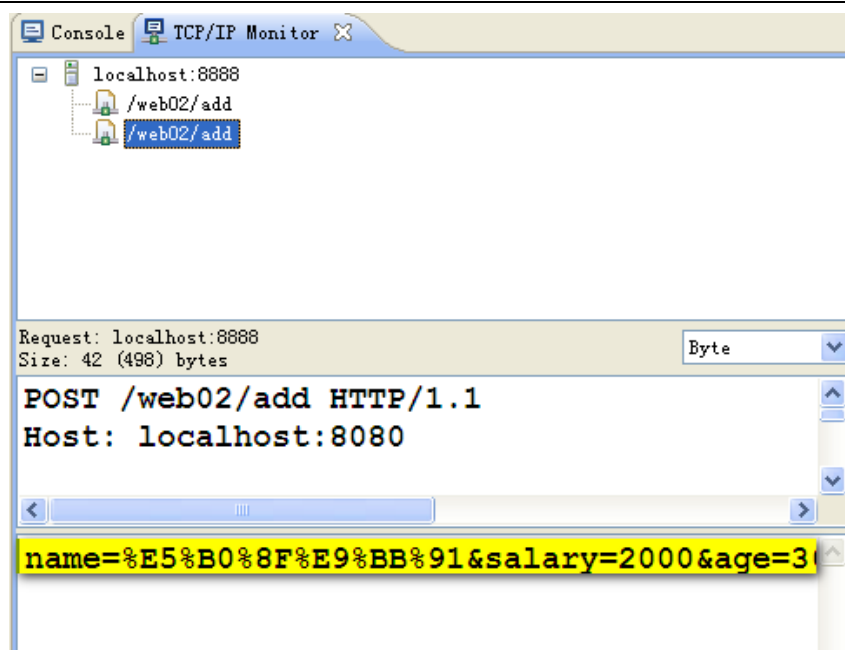
薪水:

年龄:

### 步骤 2 控制台显示为乱码



### 步骤 3 捕获的数据包中显示为 Unicode 编码



#### 步骤 4 测试 (新建 Test.java)

测试显示“小黑”的 Unicode 编码为“%E5%B0%8F%E9%BB%91”



The screenshot shows a Java IDE with a file named 'Test.java'. The code defines a package 'test', imports 'java.io.UnsupportedEncodingException', and creates a public class 'Test' with a 'main' method. The 'main' method throws 'UnsupportedEncodingException' and contains a 'TODO' comment. It then encodes the string '小黑' (Xiao Hei) into UTF-8 using 'URLEncoder.encode' and decodes it back using 'URLDecoder.decode', printing both results. The console output shows the hexadecimal representation of the UTF-8 bytes '%E5%B0%8F%E9%BB%91' and the original string '小黑'.

```

1 package test;
2
3 import java.io.UnsupportedEncodingException;
4
5
6
7 public class Test {
8     public static void main(String[] args)
9         throws UnsupportedEncodingException {
10        // TODO Auto-generated method stub
11        String str = "小黑";
12        String str2 =
13            URLEncoder.encode(str, "utf-8");
14        System.out.println(str2);
15        String str3 =
16            URLDecoder.decode(str2, "utf-8");
17        System.out.println(str3);
18    }
19 }

```

Console Output:

```

<terminated> Test (3) [Java Application] C:\Program Files\Java\jdk1.6.0_06\bin\javaw.exe (Jan 16
%E5%B0%8F%E9%BB%91
小黑

```

## 演示 2

### 中文乱码解决方案

#### step1

- 1) 在 html 文件中，添加  
`<meta http-equiv="content-type" content="text/html;charset=utf-8">`
- 2) 另外，表单的提交方式必须是 post

```

1 <html>
2 <head>
3   <meta http-equiv="Content-Type"
4     content="text/html; charset=utf-8">
5   <title>addEmp</title>
6 </head>
7 <body style="font-size:30px;">
8     <form action="add" method="post">
9         <fieldset>
10             <legend>添加雇员</legend>
11             姓名:<input name="name"/><br/>
12             薪水:<input name="salary"/><br/>
13             年龄:<input name="age"/><br/>
14             <input type="submit" value="确认"/>
15         </fieldset>
16     </form>
17 </body>
18 </html>

```

## step2

在服务器端，使用 servlet 读取表单中的请求参数时：request.setCharacterEncoding("utf-8");  
这行代码的作用：设置解码时的编码格式。

```

1 package web;
2
3 import java.io.IOException;
14
15 public class AddEmpServlet extends HttpServlet{
16     public void service(HttpServletRequest request,
17         HttpServletResponse response)
18         throws ServletException, IOException{
19
20         //这行代码要放在getParameter()执行之前。
21         request.setCharacterEncoding("utf-8");
22
23         String name = request.getParameter("name");
24         double salary = Double.parseDouble(
25             request.getParameter("salary"));
26         int age = Integer.parseInt(
27             request.getParameter("age"));
28         System.out.println("name:" + name);
29         System.out.println("salary:" + salary);
30         System.out.println("age:" + age);
31     }
32 }
33

```

### step3

如果 servlet 输出中文，要添加如下代码。

```
response.setContentType("text/html;charset=utf-8");
```

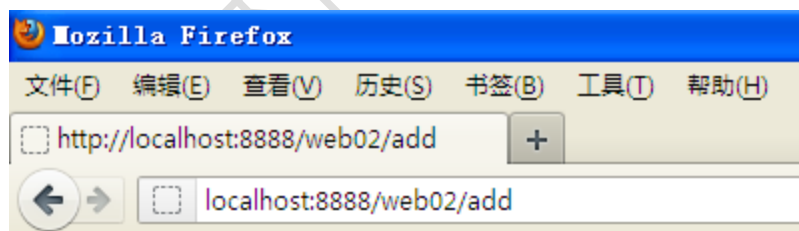
```

15 public class AddEmpServlet extends HttpServlet{
16     public void service(HttpServletRequest request,
17         HttpServletResponse response)
18         throws ServletException, IOException{
19
20         //这行代码要放在getParameter()执行之前。
21         request.setCharacterEncoding("utf-8");
22
23         String name = request.getParameter("name");
24         double salary = Double.parseDouble(
25             request.getParameter("salary"));
26         int age = Integer.parseInt(
27             request.getParameter("age"));
28         System.out.println("name:" + name);
29         System.out.println("salary:" + salary);
30         System.out.println("age:" + age);
31
32         response.
33             setContentType("text/html;charset=utf-8");
34         PrintWriter out = response.getWriter();
35         out.println("<h1>" + name + "</h1>");
36
37     }
38 }

```

## 结果演示

### 1) 页面



**小黑**

### 2) 控制台

```

Console x TCP/IP Monitor
tomcat5Server [Remote Java Application] C:\Program Files\Java\jdk1.6.0_06\bin\ja
name: 小黑
salary: 20000.0
age: 30
    
```

### 注意：

出现如下错误可以不用理会，是因为 Tomcat 热部署造成的，重新手动部署一下即可

```

Problems Tasks Web Browser Console x Servers TCP/IP Monitor
tomcat5Server [Remote Java Application] C:\Program Files\Java\jdk1.6.0_01\bin\javaw.exe (Nov 22, 2011 9:27:33 AM)
salary: 3000.0
age: 22
2011-11-22 14:16:46 org.apache.catalina.loader.WebappClassLoader
信息: Illegal access: this web application instance has been st
java.lang.IllegalStateException
    at org.apache.catalina.loader.WebappClassLoader.loadCl
    at org.apache.catalina.loader.WebappClassLoader.loadCl
    at java.lang.ClassLoader.loadClassInternal(ClassLoader.j
2011-11-22 14:16:46 org.apache.catalina.loader.WebappClassLoader
信息: Illegal access: this web application instance has been st
java.lang.IllegalStateException
    
```

## 【案例 2】中文乱码处理方案 \*\*

### 1) addEmp.html

```

<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
<title>addEmp</title>
</head>
<body style="font-size:30px;">
    <form action="add" method="post">
        <fieldset>
            <legend>添加雇员</legend>
        
```

```

        姓名:<input name="name"/> <br/>
        薪水:<input name="salary"/> <br/>
        年龄:<input name="age"/> <br/>
        <input type="submit" value="确认"/>
    </fieldset>
</form>
</body>
</html>

```

## 2) AddEmpServlet.java

```

package web;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class AddEmpServlet extends HttpServlet{
    public void service(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException,IOException{

        //这行代码要放在 getParameter()执行之前。
        request.setCharacterEncoding("utf-8");

        String name = request.getParameter("name");
        double salary = Double.parseDouble(
            request.getParameter("salary"));
        int age = Integer.parseInt(
            request.getParameter("age"));
    }
}

```

```

        System.out.println("name:" + name);
        System.out.println("salary:" + salary);
        System.out.println("age:" + age);

        response.
            setContentType("text/html;charset=utf-8");
        PrintWriter out = response.getWriter();
        out.println("<h1>" + name + "</h1>");

    }
}

```

### 3) web.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <servlet>
        <servlet-name>addEmpServlet</servlet-name>
        <servlet-class>web.AddEmpServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>addEmpServlet</servlet-name>
        <url-pattern>/add</url-pattern>
    </servlet-mapping>
</web-app>

```

### 结果演示

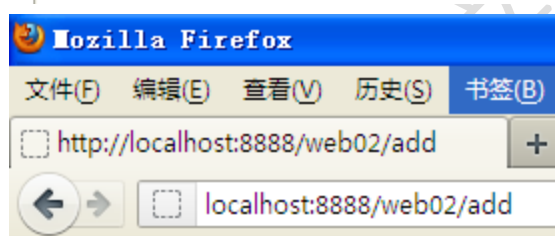


## 添加雇员

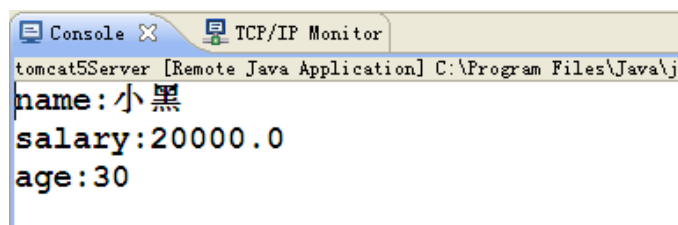
姓名:

薪水:

年龄:



**小黑**



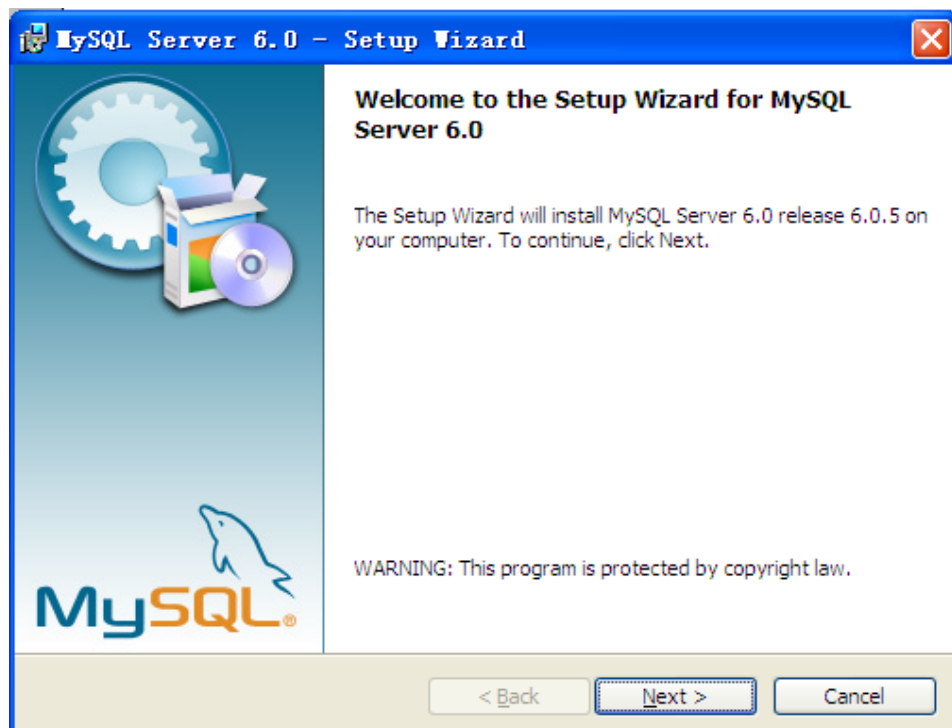


## 5. 访问数据库 \*\*

### 5.1. MySQL 安装步骤 \*

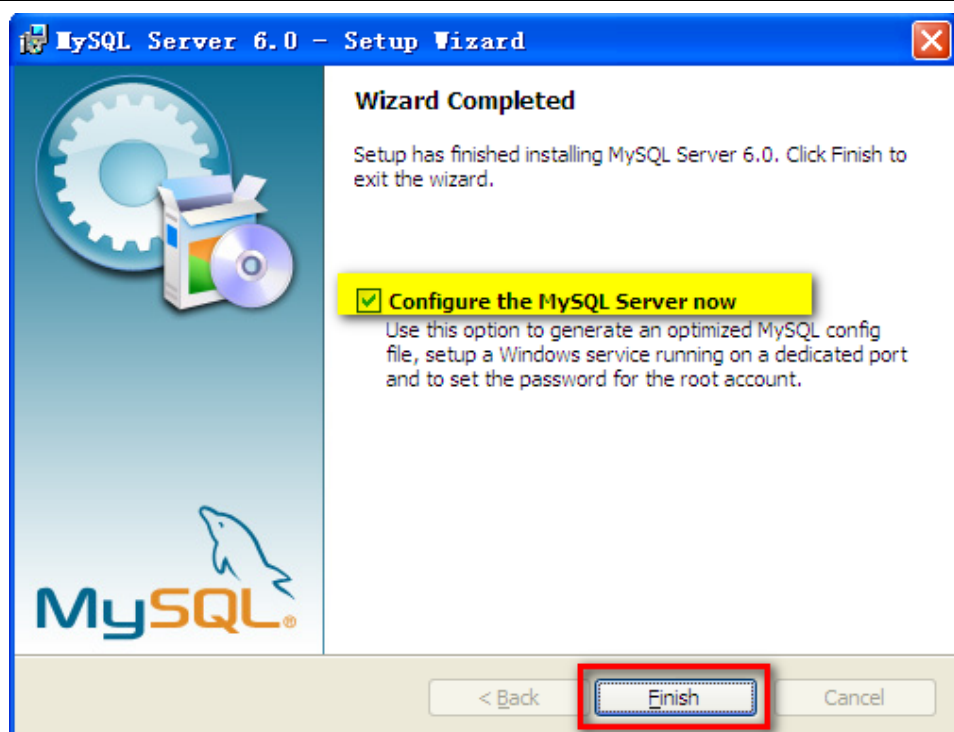
以 mysql-6.0.5-alpha-win32 为例

#### 1) 双击安装程序源文件

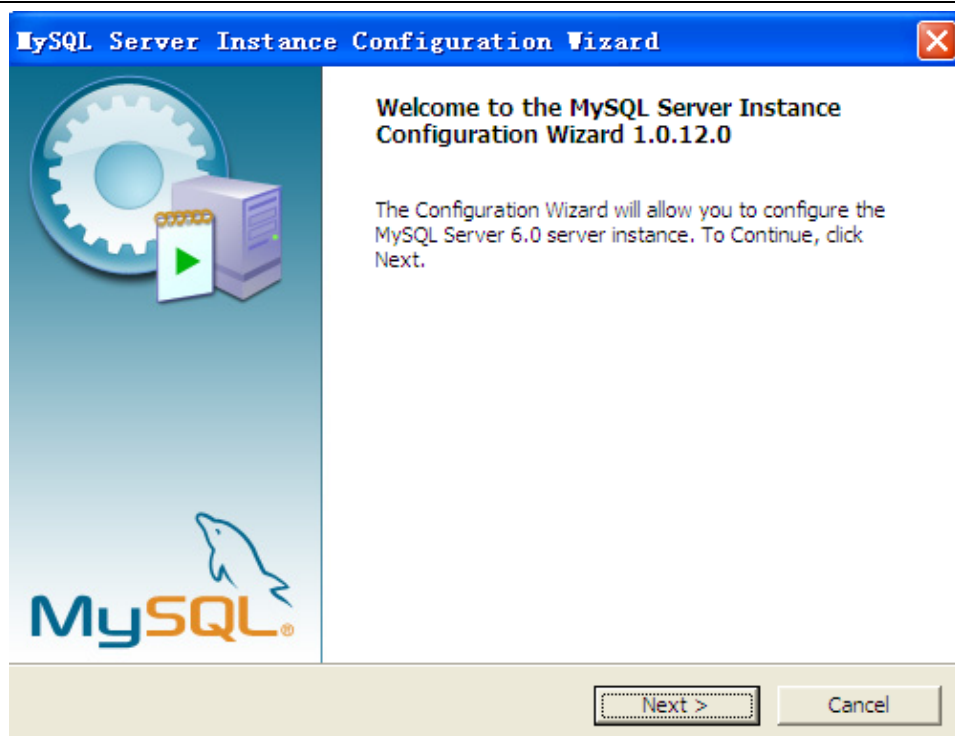


#### 2) 默认安装

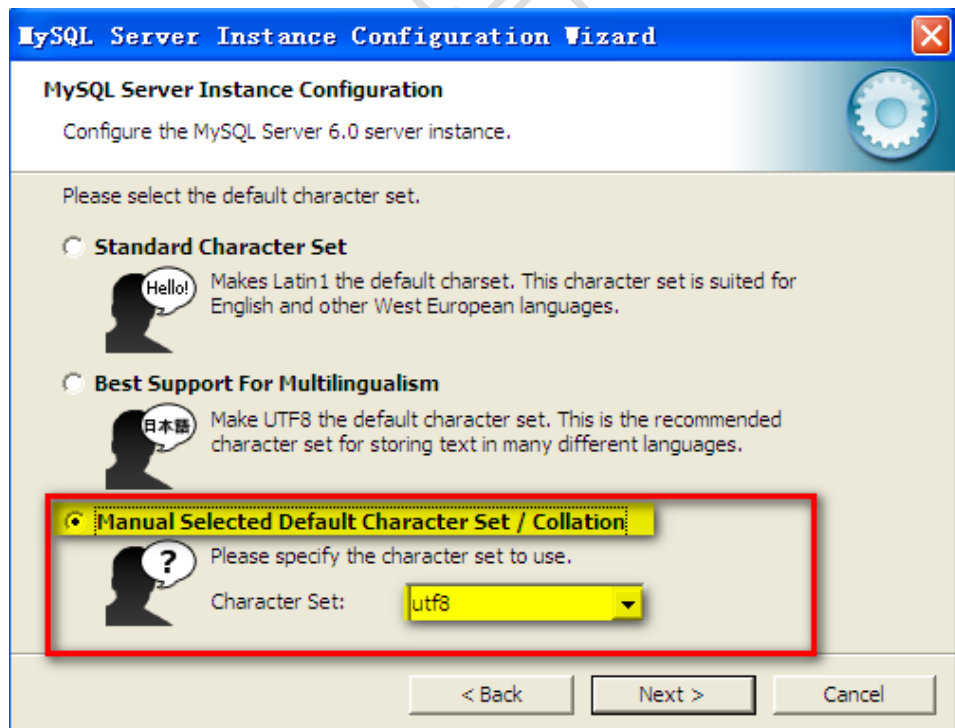
注意：勾选“Configure the MySQL Server now”（默认）



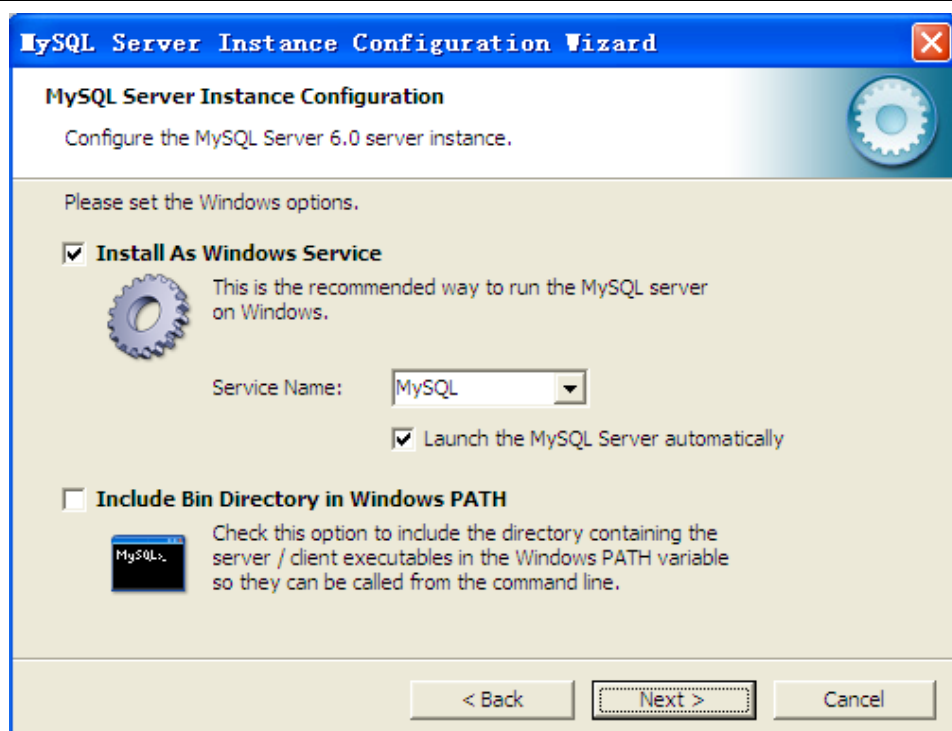
- 3) 配置 MySQL 服务器
- a. 默认一直点击 “Next”



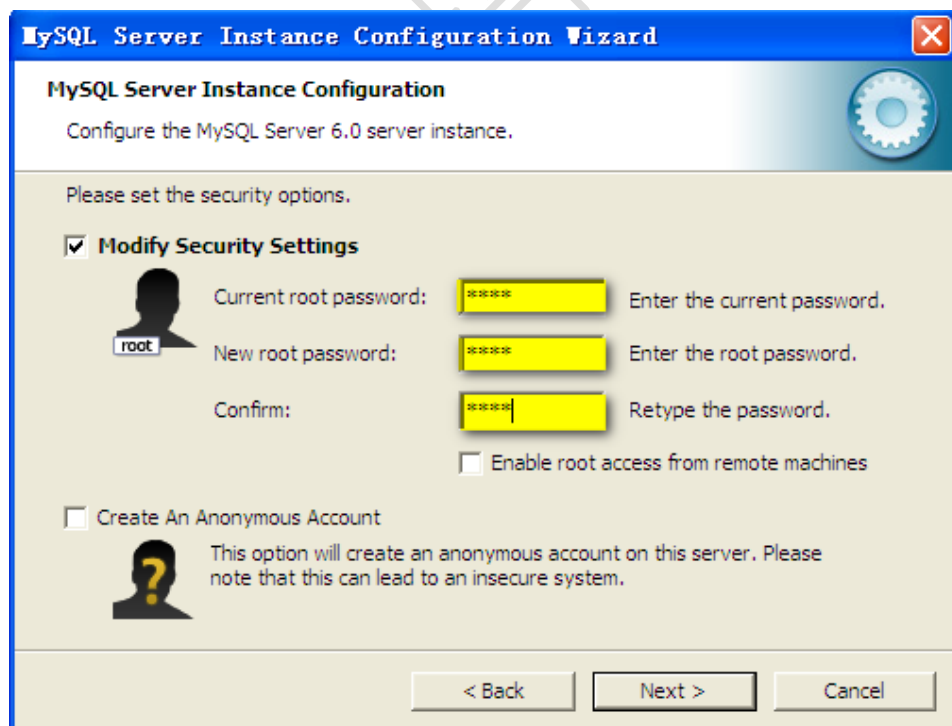
b. 注意在选择“默认编码集”的时候，点选默认的编码集为“UTF8”



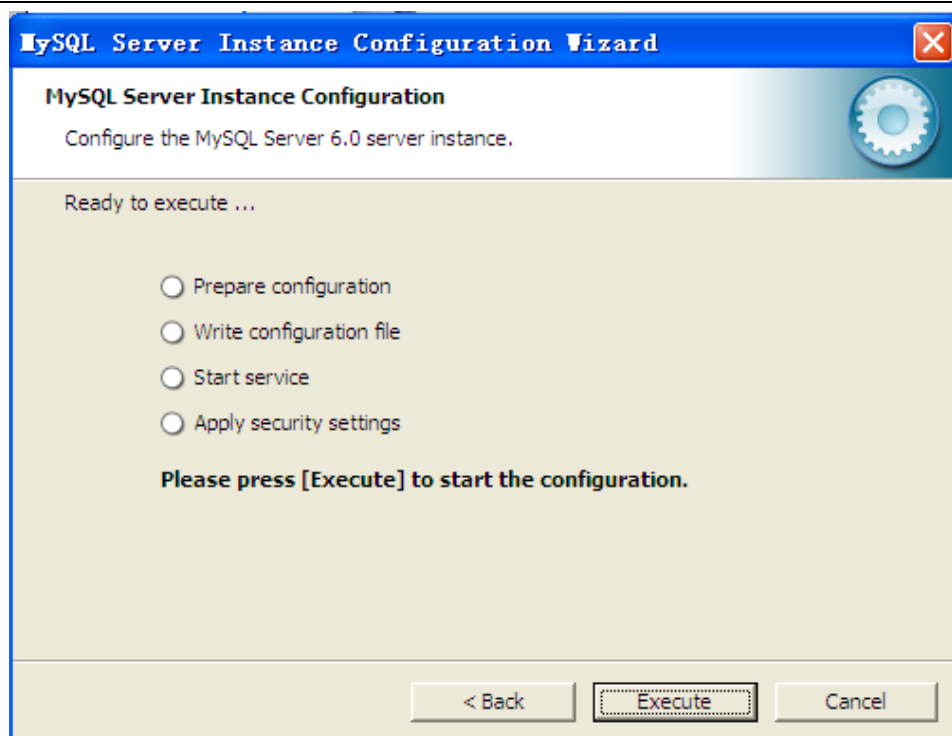
c. 默认



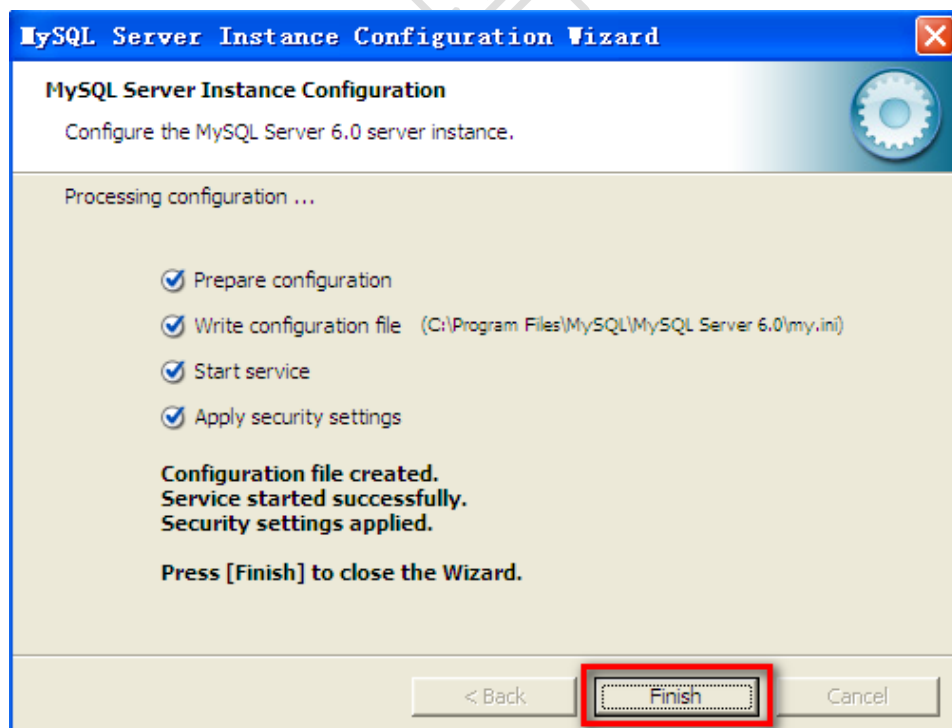
d. 注意输入密码，演示密码为“root”



e. 点击“Execute”执行



f. 点击“Finish”完成



## 5.2. mysql 的简单使用 \*

### 1) 进入 mysql

中关村校区 linux 系统

mysql -uroot; //以 root 用户登录

### 2) 几个简单指令

#### ✓ 查看数据库

show databases;

#### ✓ 使用某个数据库

use test;

#### ✓ 查看该数据库所拥有的表

show tables;

#### ✓ 建一个新数据库

create database jd1109db2;

或者

create database jd1109db2 default character set utf8;

创建名叫 jd1109db2 的数据库，并且设置默认的编码是 utf-8。

#### ✓ 建表

```
SQL> use jd1109db2;
SQL> create table t_emp(
    id bigint primary key auto_increment,    --主键自增长
    name varchar(50) unique,                --mysql 没有 varchar2
    salary double,
    age int
);
SQL> insert into t_emp(name,salary,age) values('lg',2000,22);
```

primary key: 主键

auto\_increment: 自增长列，即每插入一条记录，数据库会自动生成一个主键值。

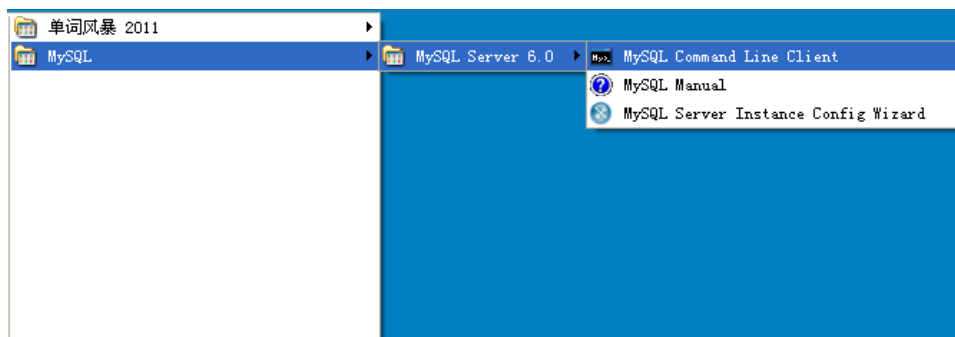
unique: 唯一性约束

```
SQL> create table t_user(
    id bigint primary key auto_increment,
    username varchar(50) unique,
    name varchar(50),
    age int,
    gendar char(1),
```

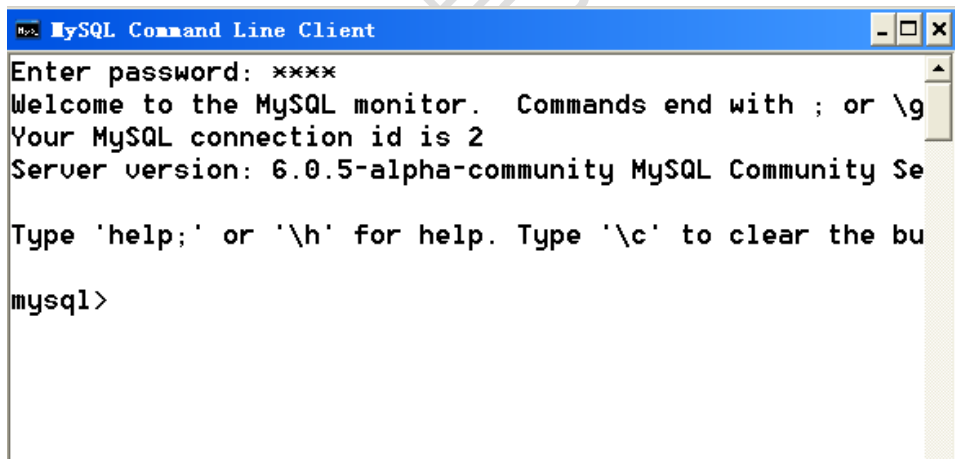
```
birthday date,      --日期
salary double,
info text           --大数据量的文本
);
```

**演示**（Windows 操作系统为例）

**1) 开始菜单启动 MySQL 终端**



**2) 输入密码进入 mysql 数据库**



**3) 查看数据库**

```
MySQL Command Line Client
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql      |
| test       |
+-----+
3 rows in set (0.00 sec)

mysql>
```

#### 4) 使用某个数据库

```
MySQL Command Line Client
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql      |
| test       |
+-----+
3 rows in set (0.00 sec)

mysql> use test;
Database changed
mysql>
```

#### 5) 查看该数据库所拥有的表



```
MySQL Command Line Client

mysql> use test;
Database changed
mysql> show tables;
+-----+
| Tables_in_test |
+-----+
| xxxbyx          |
+-----+
1 row in set (0.05 sec)

mysql>
```

6) 建立一个新数据库 jd1109db2，设置默认编码为 UTF8

```
MySQL Command Line Client

mysql> create database jd1109db2
-> default character set utf8;
Query OK, 1 row affected (0.02 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| jd1109db2 |
| mysql |
| test |
+-----+
4 rows in set (0.00 sec)

mysql>
```

删除自定义数据库的命令

```
MySQL Command Line Client
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| jd1109db2 |
| mysql |
| test |
+-----+
4 rows in set (0.00 sec)

mysql> drop database test;
Query OK, 1 row affected (0.03 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| jd1109db2 |
| mysql |
+-----+
3 rows in set (0.00 sec)

mysql>
```

## 7) 建表 t\_emp

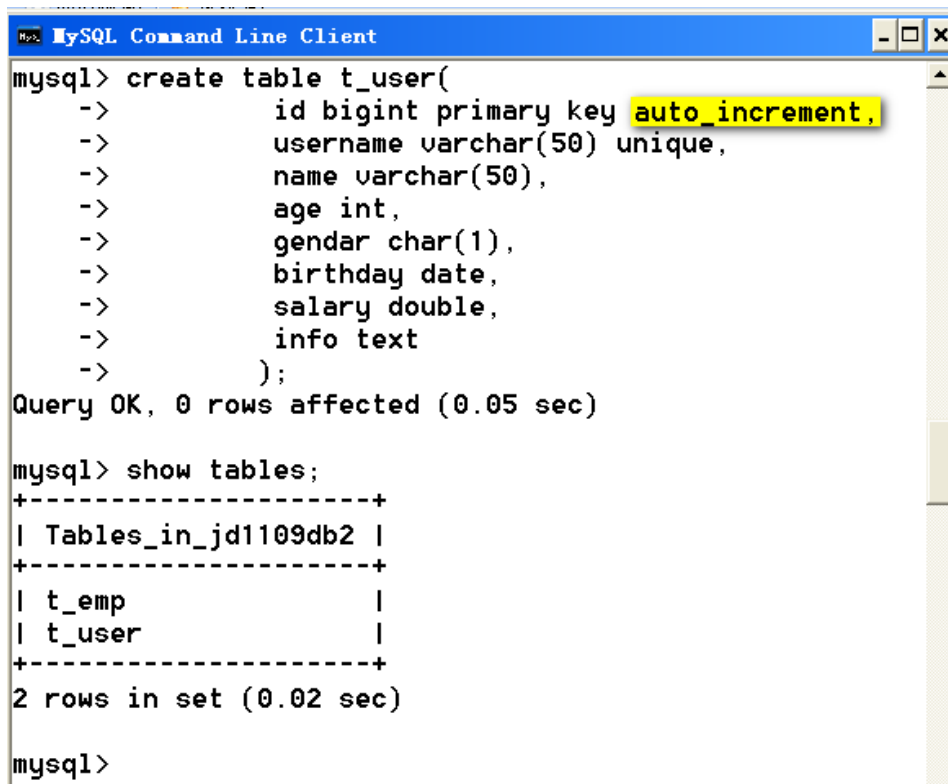
```
MySQL Command Line Client
mysql> use jd1109db2;
Database changed
mysql> create table t_emp(
->         id bigint primary key auto_increment,
->         name varchar(50) unique,
->         salary double,
->         age int
-> );
Query OK, 0 rows affected (0.06 sec)

mysql> insert into t_emp(name,salary,age)
-> values('lg',2000,22);
Query OK, 1 row affected (0.03 sec)

mysql>
```

## 8) 建表 t\_dept

主键自增长 : **auto\_increment**



```

MySQL Command Line Client
mysql> create table t_user(
->      id bigint primary key auto_increment,
->      username varchar(50) unique,
->      name varchar(50),
->      age int,
->      gendar char(1),
->      birthday date,
->      salary double,
->      info text
-> );
Query OK, 0 rows affected (0.05 sec)

mysql> show tables;
+-----+
| Tables_in_jd1109db2 |
+-----+
| t_emp                |
| t_user                |
+-----+
2 rows in set (0.02 sec)

mysql>
  
```

## 5.3. 访问数据库步骤 \*\*

- 1) 将 jdbc 驱动程序相关的 jar 包 copy 到 WEB-INF/lib 下
- 2) 在 servlet 代码当中, 使用 jdbc 访问数据库, 要注意如何处理异常。
- 3) 如何配置错误处理页面  
因为在访问数据库时, 可能会产生系统异常, 可以为服务器配置一个错误处理页面, 这样, 当发生系统异常时, 服务器会将对应的错误处理页面显示给用户。

**step1** 写一个错误处理页面 error.html

**step2** 在 web.xml 文件中, 配置

```

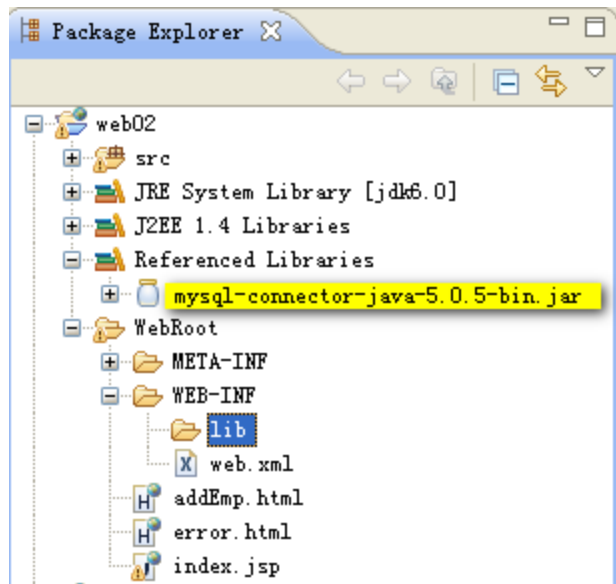
<error-page>
    <error-code>500</error-code>
    <location>/error.html</location>
</error-page>
  
```

## 演示：MyEclipse 连接 MySql 数据库

### 步骤 1

将 jdbc 驱动程序相关 jar 包 ( **mysql-connector-java-5.0.5-bin.jar** ) 拷贝到 WEB-INF/lib 下

- 1) 手工将 jar 包拷贝到 lib 目录
- 2) 在 web02 项目上点击右键“刷新”即可



### 步骤 2

在 servlet 代码当中，使用 jdbc 访问数据库，要注意如何处理异常

```

AddEmpServlet.java X web.xml
1 package web;
2
3 import java.io.IOException;
4
14
15 public class AddEmpServlet extends HttpServlet{
16
17     public void service(HttpServletRequest request,
18                         HttpServletResponse response)
19         throws ServletException, IOException{
20
21         //这行代码要放在getParameter()执行之前。
22         request.setCharacterEncoding("utf-8");
23
24         String name = request.getParameter("name");
25         double salary = Double.parseDouble(
26             request.getParameter("salary"));
27         int age = Integer.parseInt(
28             request.getParameter("age"));
29         System.out.println("name:" + name);
30         System.out.println("salary:" + salary);
31         System.out.println("age:" + age);
32
33         //访问数据库
34         Connection conn = null;
35         try {
36             Class.forName("com.mysql.jdbc.Driver");
37             conn = DriverManager
38                 .getConnection(
39                     "jdbc:mysql://localhost:3306/jd1109db2",
40                     "root", "1234");
41             PreparedStatement prep =
42                 conn.prepareStatement(
43                     "insert into t_emp(name,salary,age) " +
44                     "values(?,?,?)");
45             prep.setString(1, name);
46             prep.setDouble(2, salary);
47             prep.setInt(3, age);
48             prep.executeUpdate();
49
50             response.setContentType(
51                 "text/html;charset=utf-8");
52             PrintWriter out = response.getWriter();
53             out.println("添加雇员成功");

```

```

54
55         out.close();
56
57     } catch (Exception e) {
58         /* 异常catch之后,
59         * 要区分对待不同的异常类型。*/
60
61         /* 对于系统异常
62         * (不是由于程序本身的问题产生的异常,
63         * 而是由于环境,比如网络问题、数据库问
64         * 题等产生的异常,如数据库没有启动时,
65         * 使用DriverManager.getConnection()是
66         * 无论如何也得不到连接的,会报异常,这
67         * 种异常就是系统异常,程序无能为力,
68         * 但是,必须告诉用户)。
69         * 所以,对于系统异常,一般会提示用户 */
70
71         //step1 先记录日志
72         e.printStackTrace();
73         //step2 抛出
74         throw new ServletException(e);
75
76     } finally {
77         if (conn != null) {
78             try {
79                 conn.close();
80             } catch (SQLException e) {
81                 e.printStackTrace();
82             }
83         }
84     }
85 }
86 }

```

### 步骤 3

#### 配置错误处理页面

因为在访问数据库时,可能会产生系统异常,可以为服务器配置一个错误处理页面,这样,当发生系统异常时,服务器会将对应的错误处理页面显示给用户。

#### 步骤 3.1

写一个错误处理页面 error.html

```

error.html
1 <html>
2 <head>
3 <meta http-equiv="Content-Type"
4   content="text/html; charset=UTF-8">
5 <title>Insert title here</title>
6 </head>
7 <body style="font-size:30px;color:red;">
8   发生了系统错误, 请稍后
9   <a href="addEmp.html">重试</a>
10 </body>
11 </html>

```

### 步骤 3.2

在 web.xml 文件中，配置

```

<error-page>
  <error-code>500</error-code>
  <location>/error.html</location>
</error-page>

```

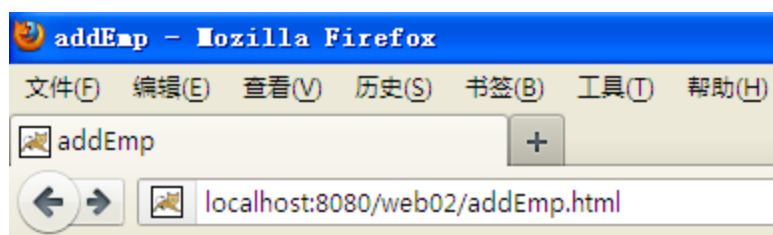
```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app version="2.4"
3   xmlns="http://java.sun.com/xml/ns/j2ee"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
6   http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
7   <servlet>
8     <servlet-name>addEmpServlet</servlet-name>
9     <servlet-class>web.AddEmpServlet</servlet-class>
10  </servlet>
11  <servlet-mapping>
12    <servlet-name>addEmpServlet</servlet-name>
13    <url-pattern>/add</url-pattern>
14  </servlet-mapping>
15
16  <error-page>
17    <!--方式1 -->
18    <error-code>500</error-code>
19    <!--方式2
20    <exception-type>javax.servlet.ServletException
21    </exception-type>
22    -->
23    <location>/error.html</location>
24  </error-page>

```

## 结果演示

### 1) 访问页面



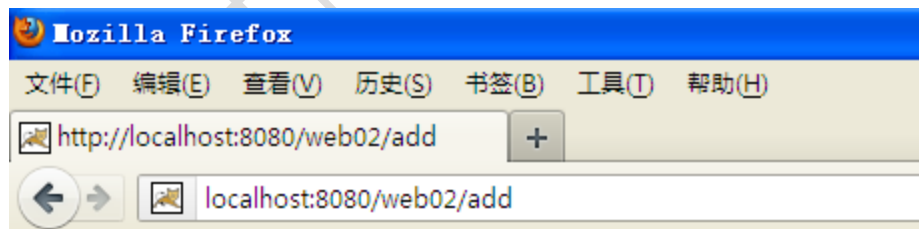
**添加雇员**

姓名:

薪水:

年龄:

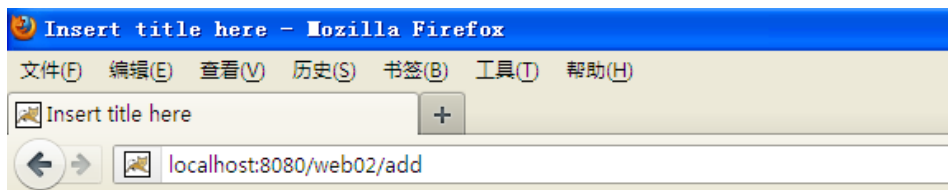
### 2) 当数据添加成功会显示“添加雇员成功”



添加雇员成功



### 3) 当数据添加失败会跳转到 err.html



发生了系统错误，请稍后 重试

## 【案例 3】Servlet 连接 MySql 数据库 \*\*

### 1) addEmp.html

```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
<title>addEmp</title>
</head>
<body style="font-size:30px;">
  <form action="add" method="post">
    <fieldset>
      <legend>添加雇员</legend>
      姓名:<input name="name"/><br/>
      薪水:<input name="salary"/><br/>
      年龄:<input name="age"/><br/>
      <input type="submit" value="确认"/>
    </fieldset>
  </form>
</body>
```

&lt;/html&gt;

## 2) AddEmpServlet.java

```
package web;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class AddEmpServlet extends HttpServlet{

    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws ServletException,IOException{

        //这行代码要放在 getParameter()执行之前。
        request.setCharacterEncoding("utf-8");

        String name = request.getParameter("name");
        double salary = Double.parseDouble(
            request.getParameter("salary"));
        int age = Integer.parseInt(
            request.getParameter("age"));
        System.out.println("name:" + name);
        System.out.println("salary:" + salary);
        System.out.println("age:" + age);

        //访问数据库
        Connection conn = null;
        try {
```

```

Class.forName("com.mysql.jdbc.Driver");
conn = DriverManager
    .getConnection(
        "jdbc:mysql://localhost:3306/jd1109db2",
        "root", "root");
PreparedStatement prep =
    conn.prepareStatement(
        "insert into t_emp(name,salary,age) " +
        "values(?,?,?)");
prep.setString(1, name);
prep.setDouble(2, salary);
prep.setInt(3, age);
prep.executeUpdate();

response.setContentType(
    "text/html;charset=utf-8");
PrintWriter out = response.getWriter();
out.println("添加雇员成功");

out.close();

} catch (Exception e) {
    /* 异常 catch 之后 ,
     * 要区分对待不同的异常类型。*/

    /* 对于系统异常
     * (不是由于程序本身的问题产生的异常,
     * 而是由于环境, 比如网络问题、数据库问
     * 题等产生的异常, 如数据库没有启动时,
     * 使用 DriverManager.getConnection()是
     * 无论如何也得不到连接的, 会报异常, 这
     * 种异常就是系统异常,程序无能为力,
     * 但是, 必须告诉用户)。
     * 所以, 对于系统异常, 一般会提示用户 */

    //step1 先记录日志
    e.printStackTrace();
    //step2 抛出

```

```

        throw new ServletException(e);

    }finally{
        if(conn!=null){
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
}

```

### 3) err.html

```

<html>
<head>
<meta http-equiv="Content-Type"
      content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body style="font-size:30px;color:red;">
    发生了系统错误，请稍后
    <a href="addEmp.html">重试</a>
</body>
</html>

```

### 4) web.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4"
  xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
  <servlet>
    <servlet-name>addEmpServlet</servlet-name>
    <servlet-class>web.AddEmpServlet</servlet-class>
  </servlet>
  <servlet-mapping>

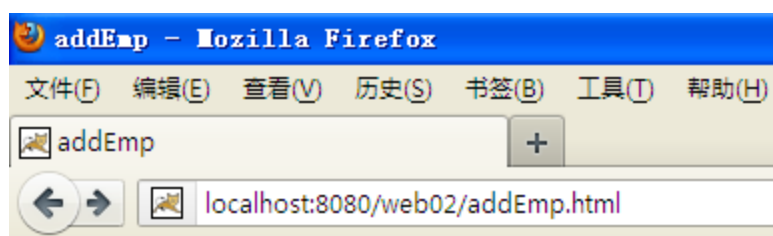
```

```
<servlet-name>addEmpServlet</servlet-name>
<url-pattern>/add</url-pattern>
</servlet-mapping>

<error-page>
  <!--方式1 -->
  <error-code>500</error-code>
  <!--方式2
  <exception-type>javax.servlet.ServletException
  </exception-type>
  -->
  <location>/error.html</location>
</error-page>
</web-app>
```

## 结果演示

### 1) 访问页面



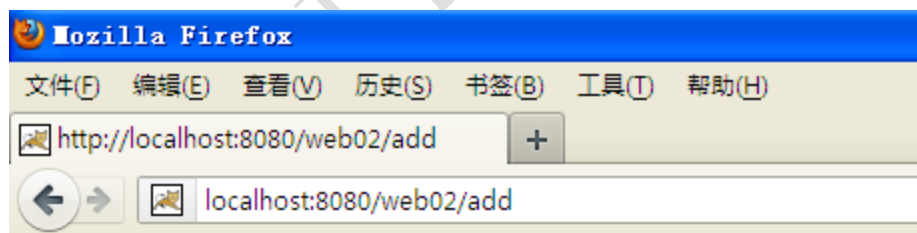
## 添加雇员

姓名:

薪水:

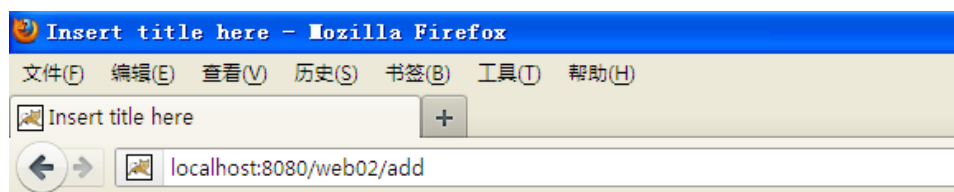
年龄:

2) 当数据添加成功会显示“添加雇员成功”



添加雇员成功

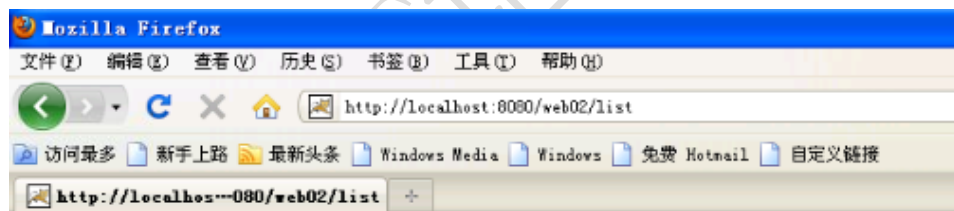
3) 当数据添加失败会跳转到 err.html



发生了系统错误，请稍后 **重试**

## 6. 【课堂练习】 \*\*

写一个 ListEmpServlet, 查询数据库，以表格的方式显示所有雇员信息。



id	姓名	薪水	年龄
1	lg	2000.0	22
2	lyg	2000.0	21
3	csb	2000.0	22

### 参考答案

#### 1) ListEmpServlet.java

```
package web;

import java.io.IOException;
```

```
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ListEmpServlet extends HttpServlet{
    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws ServletException,IOException{
        //访问数据库
        Connection conn = null;
        try {
            Class.forName(
                "com.mysql.jdbc.Driver");
            conn =
                DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306/jd1109db2",
                    "root","root");
            Statement stat =
                conn.createStatement();
            ResultSet rst = stat.executeQuery(
                "select * from t_emp");

            //使用查询得到的结果，生成一个表格
            response.setContentType(
                "text/html;charset=utf-8");
            PrintWriter out =
                response.getWriter();
            out.println(
                "<table border='1' " +
                "width='60%' " +
```



```

        "cellpadding='0' " +
        "cellspacing='0'>");
    out.println("<tr> " +
        "<td>id</td> " +
        "<td>姓名</td> " +
        "<td>薪水</td> " +
        "<td>年龄</td></tr>");
    while(rst.next()){
        long id =
            rst.getLong("id");
        String name =
            rst.getString("name");
        double salary =
            rst.getDouble("salary");
        int age =
            rst.getInt("age");
        out.println("<tr><td> "
            + id + "</td><td> "
            + name + "</td><td> "
            + salary + "</td><td> "
            + age + "</td></tr>");
    }
    out.println("</table>");
    out.close();
} catch (Exception e) {
    e.printStackTrace();
    throw new ServletException(e);
}finally{
    if(conn!=null){
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}
}
}

```

## 2) web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4"
  xmlns=
    "http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi=
    "http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation=
    "http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
  <servlet>
    <servlet-name>
      listEmpServlet</servlet-name>
    <servlet-class>
      web.ListEmpServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>
      listEmpServlet</servlet-name>
    <url-pattern>
      /list</url-pattern>
  </servlet-mapping>
  <error-page>
    <!-- <error-code>500</error-code> -->

    <exception-type>
      javax.servlet.ServletException
    </exception-type>

    <location>/error.html</location>
  </error-page>
</web-app>
```

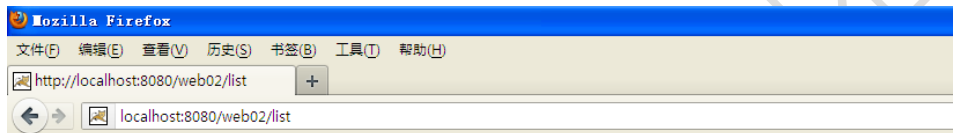
## 3) err.html

```
<html>
<head>
<meta http-equiv="Content-Type"
```

```
content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body style="font-size:30px;color:red;">
    发生了系统错误，请稍后
    <a href="addEmp.html">重试</a>
</body>
</html>
```

### 结果演示

访问 <http://localhost:8080/web02/list>



The screenshot shows the Mozilla Firefox browser window. The address bar displays the URL <http://localhost:8080/web02/list>. The page content displays a table with the following data:

id	姓名	薪水	年龄
1	lg	2000.0	22