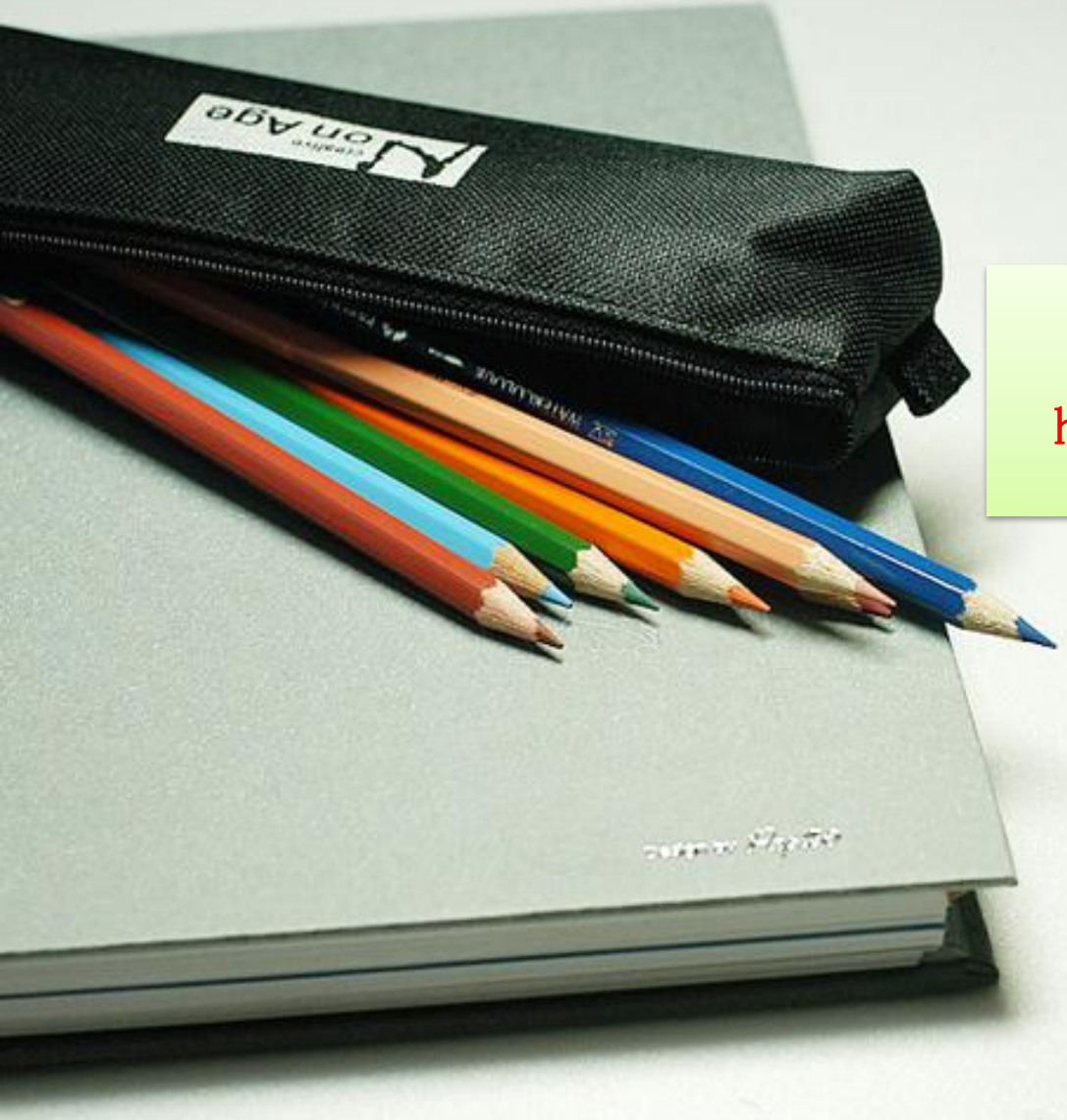


北京圣思园科技有限公司
http://www.shengsiyuan.com

主讲人：张龙

使用Eclipse结合Tomcat开发一个简单的Servlet和JSP

首先需要配置Tomcat，使得能够正常启动

Tomcat安装完毕需要将jre换成jdk

需要配置好如下三个环境变量：

JAVA_HOME

CATALINA_HOME

PATH

使用Eclipse结合Tomcat开发一个简单的Servlet和JSP

Servlet是JAVA服务器端编程，不同于我们之前写的一般的JAVA应用程序，Servlet程序是运行在服务器上的，服务器有很多种，本课程用到的服务器就是Tomcat

使用Servlet生成静态页面

首先新建一个 web 工程

使用Servlet生成静态页面

在src目录下新建一个包，名字叫com.test.servlet

在我们建的包下面新建一个类，名字叫MyServlet

该类需要继承HttpServlet类（一般来说，Servlet都需要继承此类）

使用Servlet生成静态页面

使得我们生成的类覆盖超类的doGet()和doPost()方法，Eclipse会为我们自动生成相关代码，并且为我们导入相关的包和类，这些类都在Servlet.jar包中

使用Servlet生成静态页面

现在我们需要编写项目的配置文件web.xml，每一个JAVA WEB项目都需要一个这样的配置文件，并且它的存放路径必须在WEB-INF目录下（因为这是J2EE规范规定的）

使用Servlet生成静态页面

在该文件中增加如下内容：

使用Servlet生成静态页面

- <servlet>
- <servlet-name>MyFirstServlet</servlet-name>
- <servlet-class>com.test.servlet.MyServlet</servlet-class>
- </servlet>

使用Servlet生成静态页面

```
<servlet-mapping>
    <servlet-
  name>MyFirstServlet</servlet-
  name>
    <url-pattern>/MyServlet</url-
  pattern>
  </servlet-mapping>
</web-app>
```

使用Servlet生成静态页面

以上所输入的内容为本项目的配置信息，它来告诉服务器应该按照什么方法来找到我们写好的文件

使用Servlet生成静态页面

接下来，我们在生成的类中编写JAVA代码。

使用Servlet生成静态页面

```
protected void
  process(HttpServletRequest arg0,
  HttpServletResponse arg1)
            throws ServletException,
  IOException
    {

  arg1.setContentType("text/html");
      PrintWriter out =
  arg1.getWriter();
```

使用Servlet生成静态页面

out.println("<HTML><HEAD><TITLE>My First Servlet</TITLE></HEAD>");

- out.println("<BODY>");
- out.println("<H1>Hello World</H1>");
- out.println("</BODY></HTML>");

使用Servlet生成静态页面

接下来，在doGet()与doPost()方法中分别调用我们刚刚写好的Process()方法

形式如下： process(arg0, arg1);

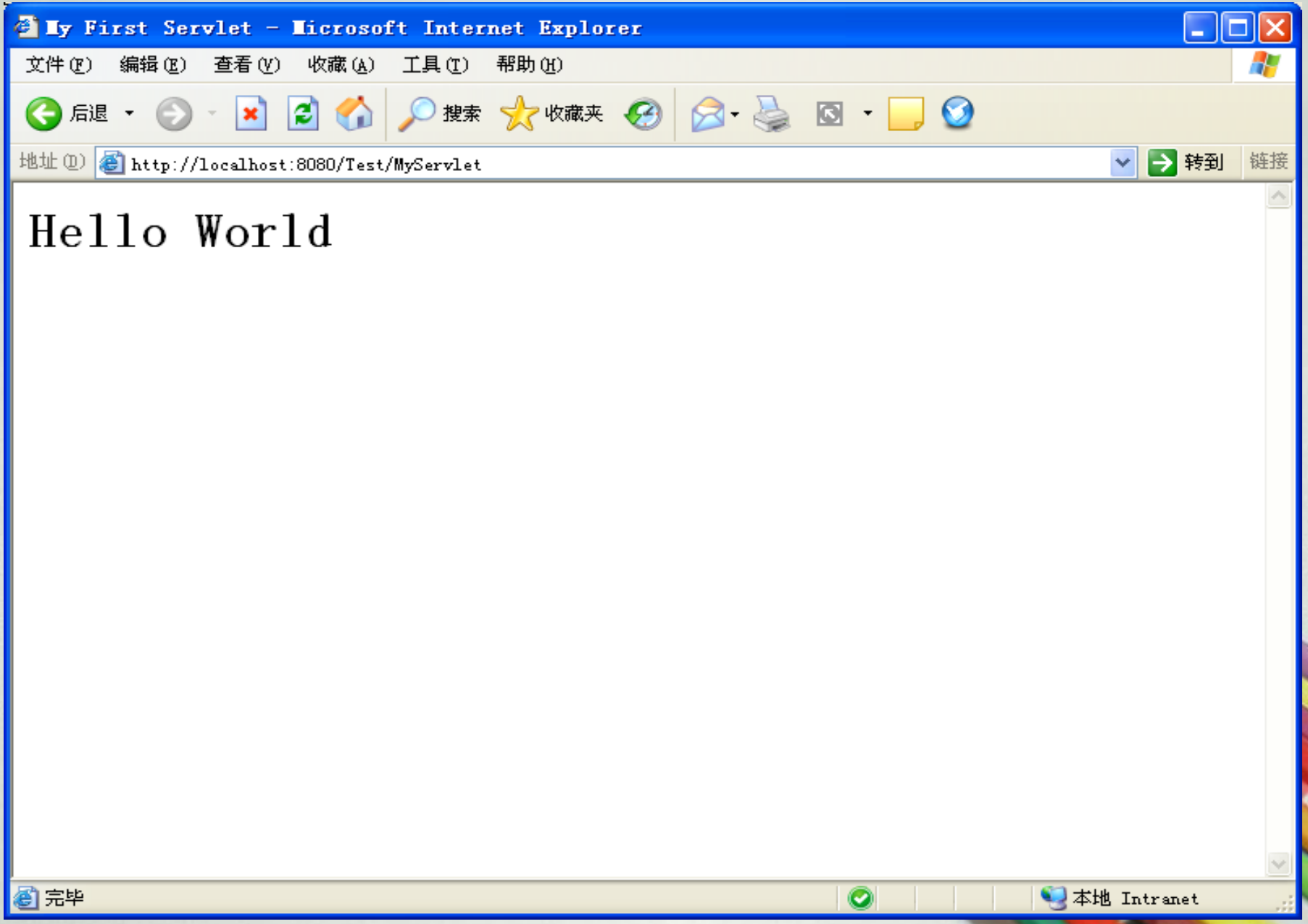
使用Servlet生成静态页面

配置好Tomcat的参数

现在启动Tomcat服务器

打开IE浏览器，输入如下地址：

http://localhost:8080/test/MyServlet


My First Servlet - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/MyServlet
转到
链接
Hello World
完毕
本地 Intranet

使用Servlet生成静态页面

如图所示，会生成一个页面，其中的内容就是我们输入的内容：Hello World

并且网页的标题也是我们输入的标题：My First Servlet

使用Servlet生成静态页面

现在我们对process()方法进行修改：

加上如下的语句：

```
out.println("<br>" + new
java.util.Date().toLocaleString());
```

保存，我们会发现Tomcat已经重新启动了

使用Servlet生成静态页面

Tomcat: Reloading this Context has started

这时我们再刷新刚才的页面

使用Servlet生成静态页面

尝试不断刷新浏览器，会发现时间在不断的变化

使用Servlet生成静态页面

继续对MyServlet类进行修改

增加如下语句：

```
System.out.println("当前系统时间为：" + new java.util.Date().toLocaleString());
```

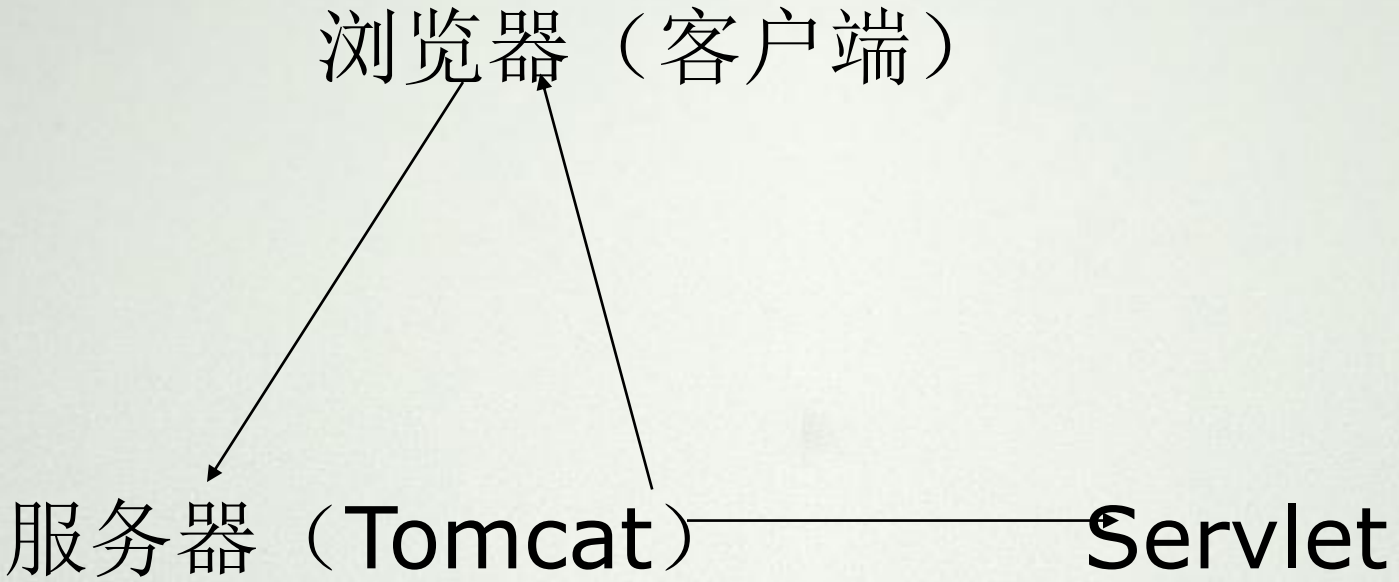
保存：会发现Tomcat又重新启动了

使用Servlet生成静态页面

刷新刚才的网页，并查看Tomcat的窗口

使用Servlet生成静态页面

生成页面的流程：



使用Servlet生成静态页面

服务器调用生成好的class文件，并把执行结果以html代码的形式发送给浏览器（即客户端）

使用JSP

在web目录下新建一个JSP页面：test.jsp

使用JSP

在页面中输入如下的代码：

```
<h1>Hello World</h1>
<%= new
  java.util.Date().toLocaleString()%>
```

保存文件

使用JSP

在浏览器中输入如下地址：

http://localhost:8080/test/test.jsp

使用JSP

我们发现其显示效果与方才Servlet显示效果一模一样

继续在jsp页面中加入如下语句：

```
<%System.out.println(new
java.util.Date().toLocaleString
() );%>
```

使用JSP

刷新方才的页面，并查看Tomcat的窗口

练习表单的使用

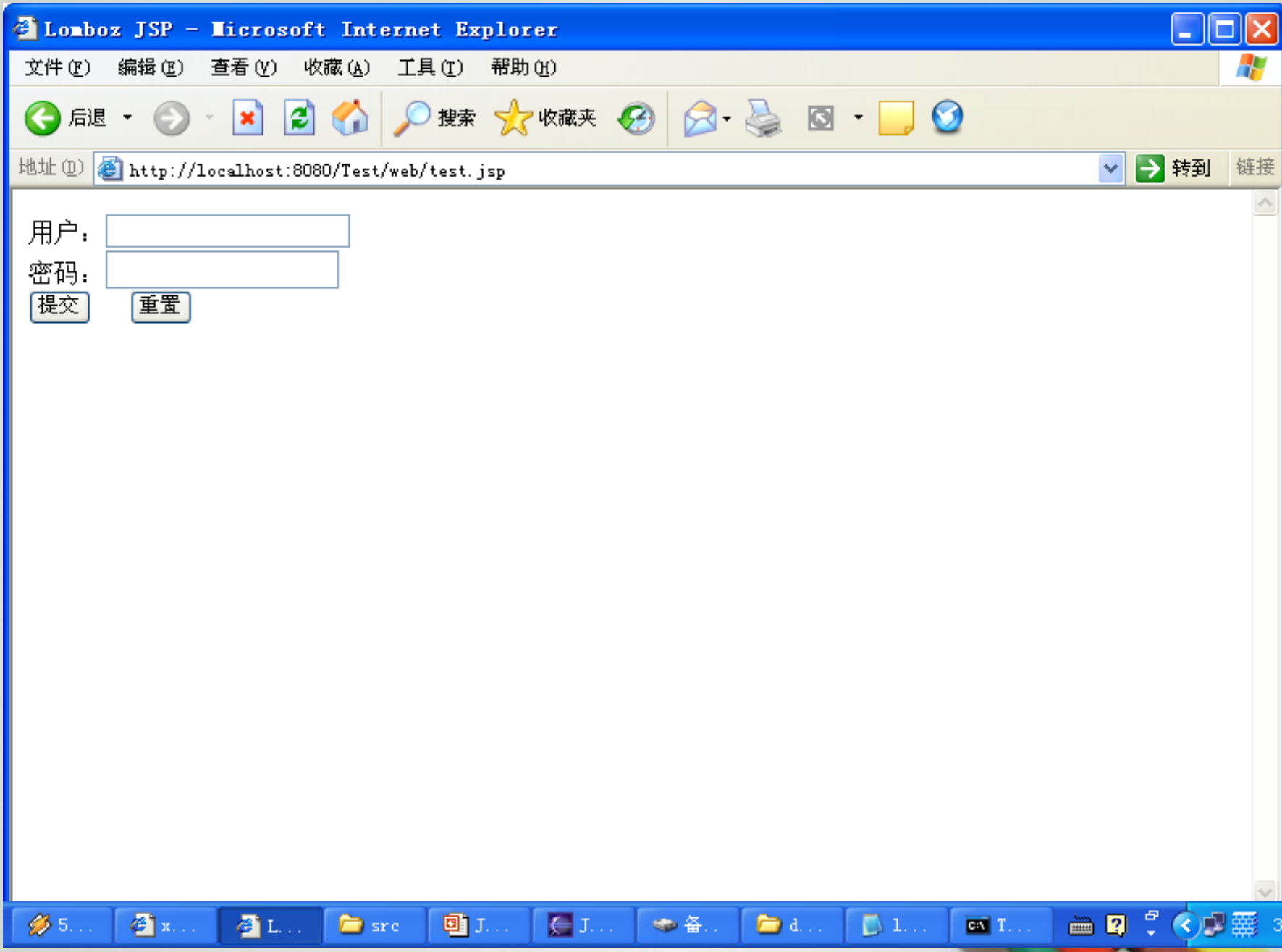
修改test.jsp页面，加入如下的表单代码：

```
<form action="/test/SecondServlet">
用户：<input type="text"
  name="username" size="20"><br>
密码：<input type="password"
  name="password" size="20"><br>
```

练习表单的使用

```
<input type="submit" value="提交
  ">   <input
  type="reset" value="重置">
</form>
```

保存并刷新页面


Lomboz JSP - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/web/test.jsp
转到
链接
用户：
密码：
提交
重置

练习表单的使用

处理我们输入的值

在包com.test.servlet下新建一个Servlet,名字为： SecondServlet

练习表单的使用

此时建立Servlet我们可以采用MyEclipse的向导来建立，这样可以节省我们的时间，向导会为我们生成好基础代码，在生成向导中我们选择生成doGet()和doPost()方法

练习表单的使用

在生成的类中添加如下代码：

```
protected void process(HttpServletRequest
   request,
            HttpServletResponse response)
   throws ServletException, IOException
    {
        String username =
   request.getParameter("username");
        String password =
   request.getParameter("password");
```

练习表单的使用

- response.setContentType("text/html");
- PrintWriter out = response.getWriter();

练习表单的使用

- out.println("<HTML><HEAD><TITLE>Result</TITLE></HEAD>");
- out.println("<BODY>");
- out.println(username + "
");
- out.println(password);
- out.println("</BODY></HTML>");

练习表单的使用

修改web.xml文件，添加如下代码：

```
<servlet>
<servlet-name>
  SecondServlet
</servlet-name>
<servlet-class>
  com.test.servlet.SecondServlet
</servlet-class>
</servlet>
```

练习表单的使用

- <servlet-mapping>
- <servlet-name>
  SecondServlet
- </servlet-name>
- <url-pattern>
  /SecondServlet
- </url-pattern>
- </servlet-mapping>

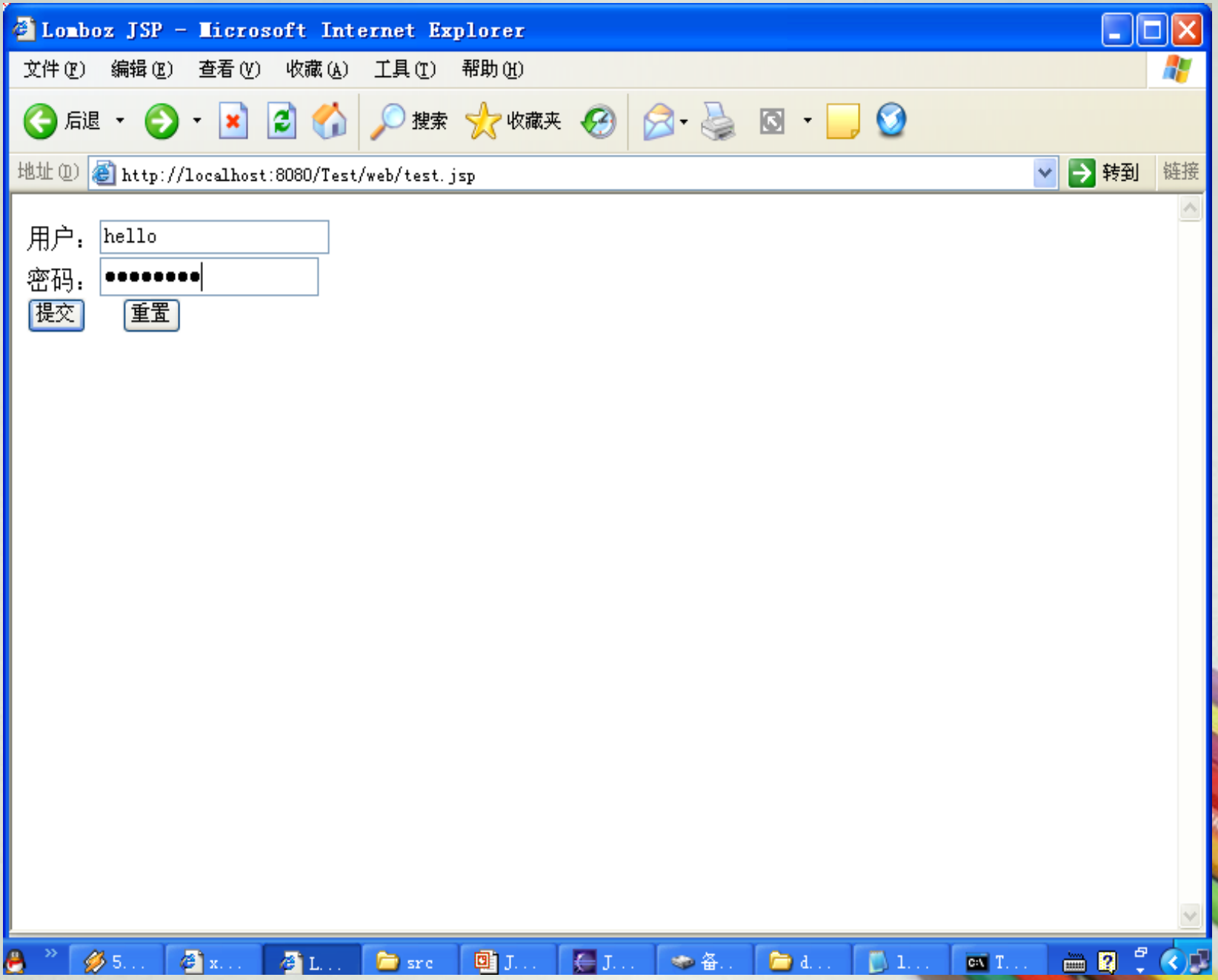
练习表单的使用

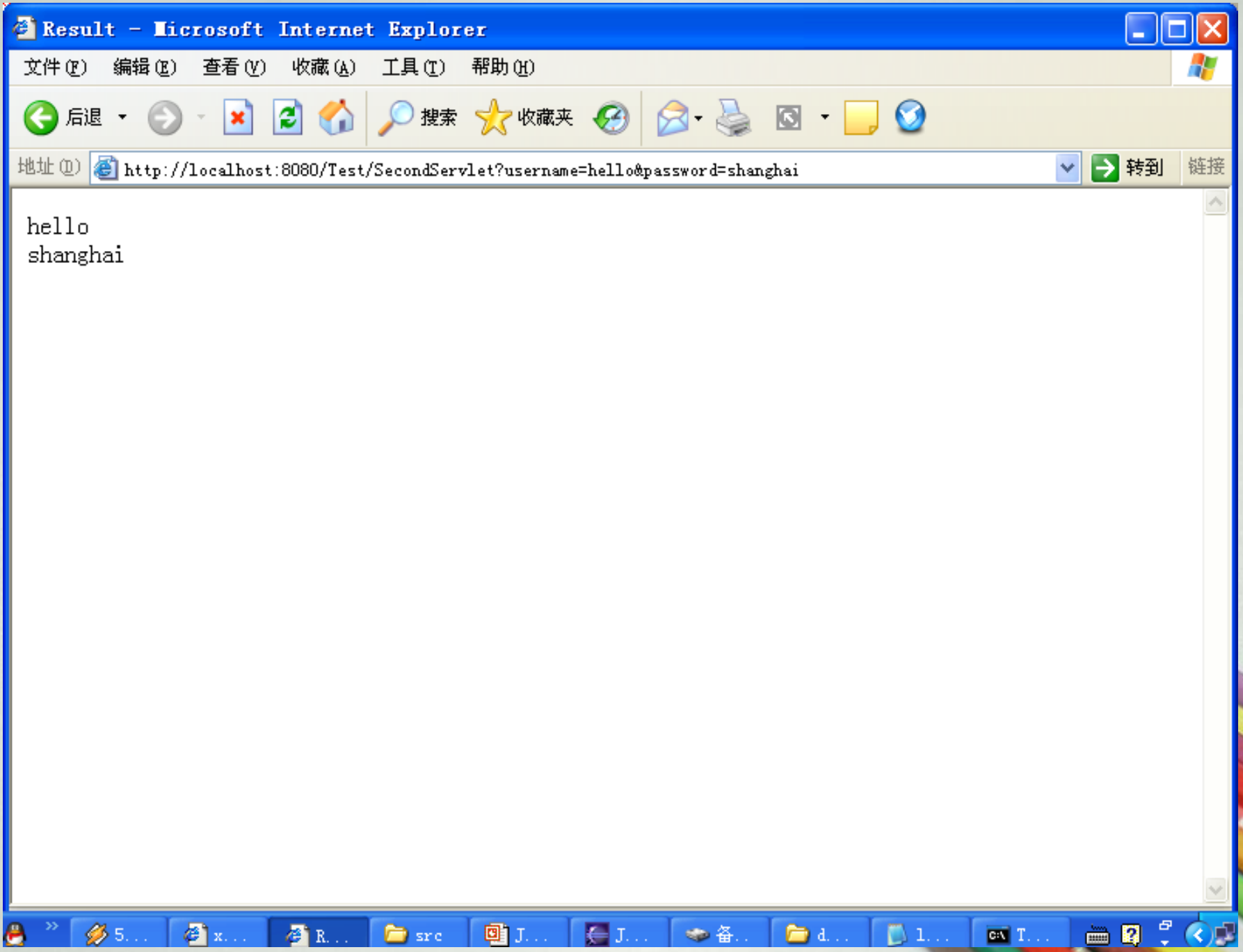
重新启动Tomcat

(Ctrl + C 或者直接关闭 或者用shutdown.bat命令)

练习表单的使用

页面会显示出你所输入的用户名和密码


Lomboz JSP - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/web/test.jsp
转到
链接
用户：
hello
密码：
提交
重置


Result - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/SecondServlet?username=hello&password=shanghai
转到
链接
hello
shanghai

练习表单的使用

注意地址栏：会出现输入的用户名与密码

在form表单里面加上method=“post”语句

刷新页面，地址栏不会再出现输入的用户名和密码了

Get 与 Post的区别（重要）

1 处理方式不一样

2 处理的方法不一样

3 地址栏呈现的结果不一样

JSP 与Servlet的生成方式

Servlet 首先被编译为class文件，然后由服务器调用

JSP首先被转化为Servlet（Java文件），然后再被编译为class文件，最后由服务器调用

编写稍复杂的Servlet

新建一个Servlet，在包com.test.servlet下

名字叫：ThirdServlet

覆盖doPost()与doGet()方法，同时修改web.xml文件

编写稍复杂的Servlet

在类中添加如下代码：

```
private static String[] responses = {
  "Yes", "No", "Maybe", "Later",
            "It is your call", "Not a
  chance" };
```

编写稍复杂的Servlet

```
private String generateReply()
    {
        //randomly pick a response
        int respIndex = new Random().nextInt(responses.length);
        // return the response back to the caller
        return responses[respIndex];
    }
```

编写稍复杂的Servlet

- private void process(HttpServletRequest request,
- HttpServletResponse response) throws IOException
- {
- // first get the dynamic response
- String reply = generateReply();
- response.setContentType("text/html");
- PrintWriter out = response.getWriter();

编写稍复杂的Servlet

- out.println("<HTML><HEAD><TITLE>Decision Maker</TITLE></HEAD>");
- out.println("<BODY>");
-

编写稍复杂的Servlet

- out.println("<H1>Your answer is: '" + reply + "'</H1>");
- out.println("</BODY></HTML>");
- }

编写稍复杂的Servlet

- 修改web.xml文件
- <servlet>
- <servlet-name>
- ThirdServlet
- </servlet-name>
- <servlet-class>
- com.test.servlet.ThirdServlet
- </servlet-class>
- </servlet>

编写稍复杂的Servlet

```
<servlet-mapping>
  <servlet-name>
    ThirdServlet
</servlet-name>
      <url-pattern>
  /ThirdServlet
</url-pattern>
</servlet-mapping>
```

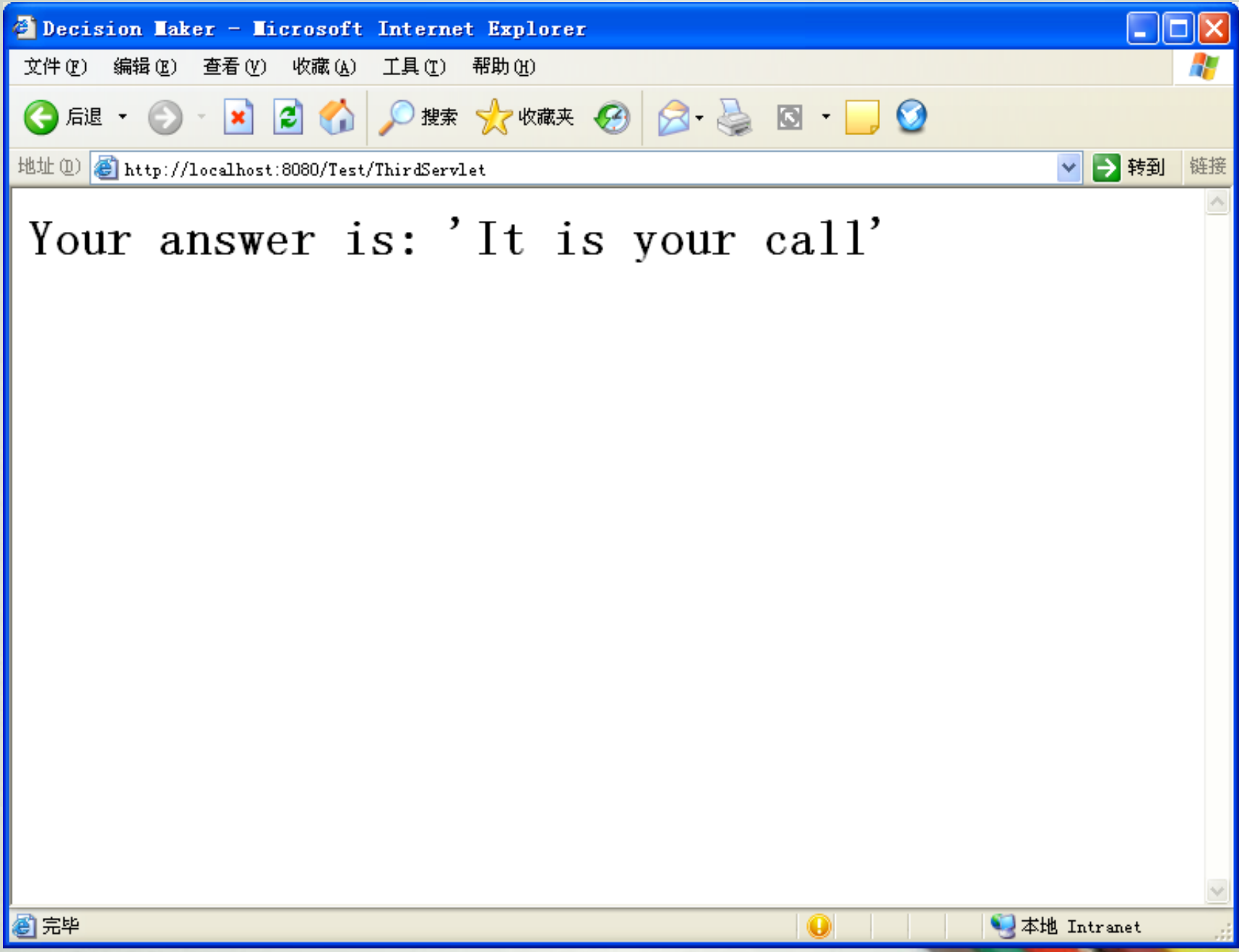
编写稍复杂的Servlet

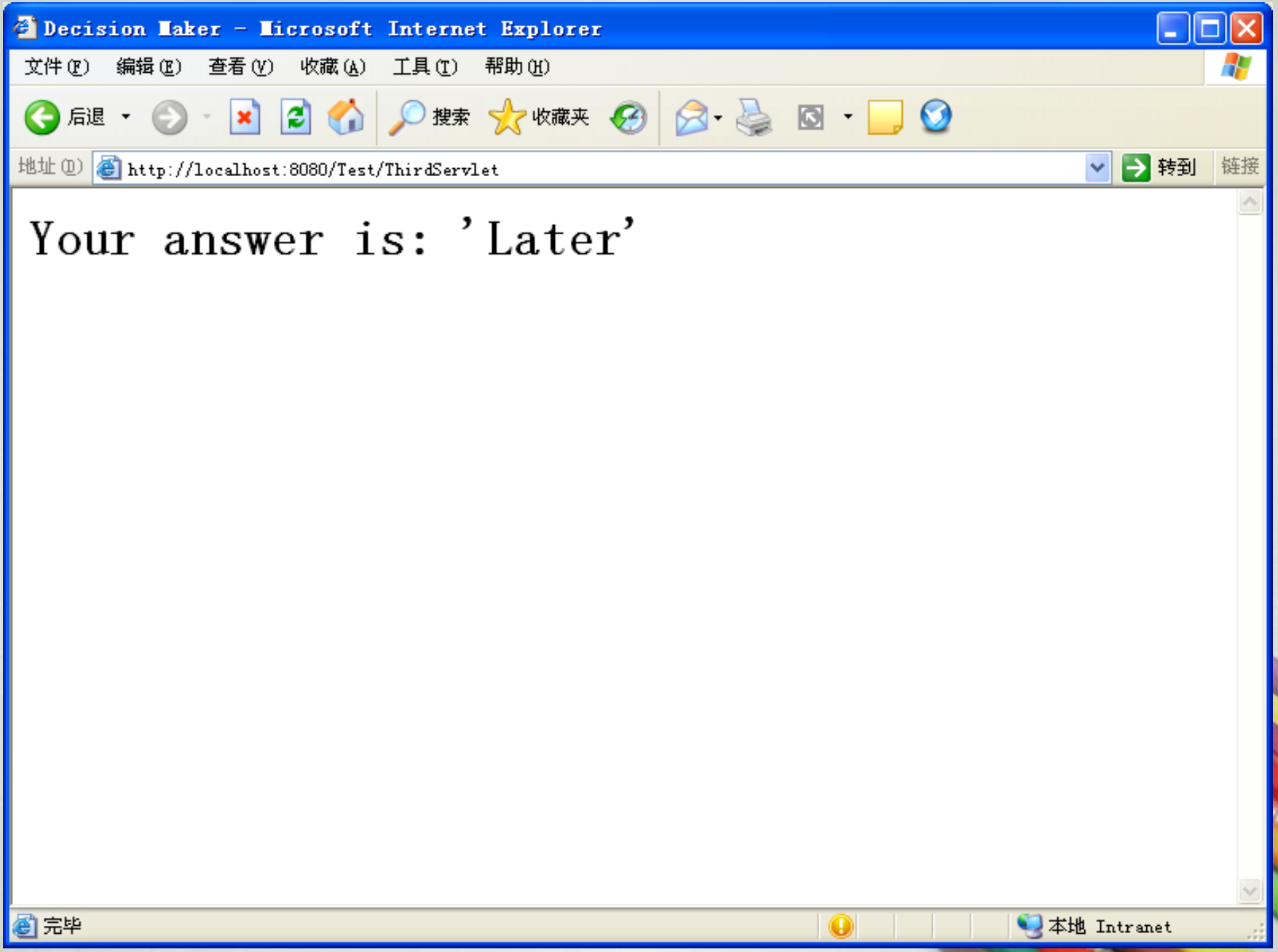
保存，重新启动服务器

在浏览器地址栏输入：

http://localhost:8080/test/ThirdServlet

并不断刷新页面


Decision Maker - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/ThirdServlet
转到
链接
Your answer is: 'It is your call'
完毕
本地 Intranet


Decision Maker - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/Test/ThirdServlet
转到
链接
Your answer is: 'Later'
完毕
本地 Intranet

编写稍复杂的JSP

在web目录下新建一个JSP页面，名字叫result.jsp

输入如下的代码

编写稍复杂的JSP

- <%String username = **request**.getParameter("name");
- String password = **request**.getParameter("passoword");
- **out**.println(username);
- **out**.println(password);
- %>

编写稍复杂的JSP

修改test.jsp, action=“result.jsp”

保存，并在浏览器中输入如下地址：

http://localhost:8080/test/test.jsp

输入用户名和密码

编写稍复杂的JSP

理解jsp中默认对象request的含义

理解jsp script的含义与写法

理解<%= %>与out.print();的作用以及他们与System.out.println();的区别

out.print()与out.println()区别

编写稍复杂的JSP

```
<%@ page language="java"
  pageEncoding="GB2312" %>
```

理解<%@ page %>标签的作用

<%@ page import=“”%>的作用

理解表单中hidden元素的作用

修改test.jsp与result.jsp文件，得到实际的效果

理解JSP 标记

<%@ page language=“java”
contentType=
“text/html;charset=“gb2312””%>

继续编写稍复杂的JSP

新建一个JSP文件，名字叫third.jsp

```
<%@ page import="java.util.*"%>
<%Calendar calendar =
  Calendar.getInstance();
int currentHour =
  calendar.get(Calendar.HOUR_OF_D
  AY);
%>
```

继续编写稍复杂的JSP

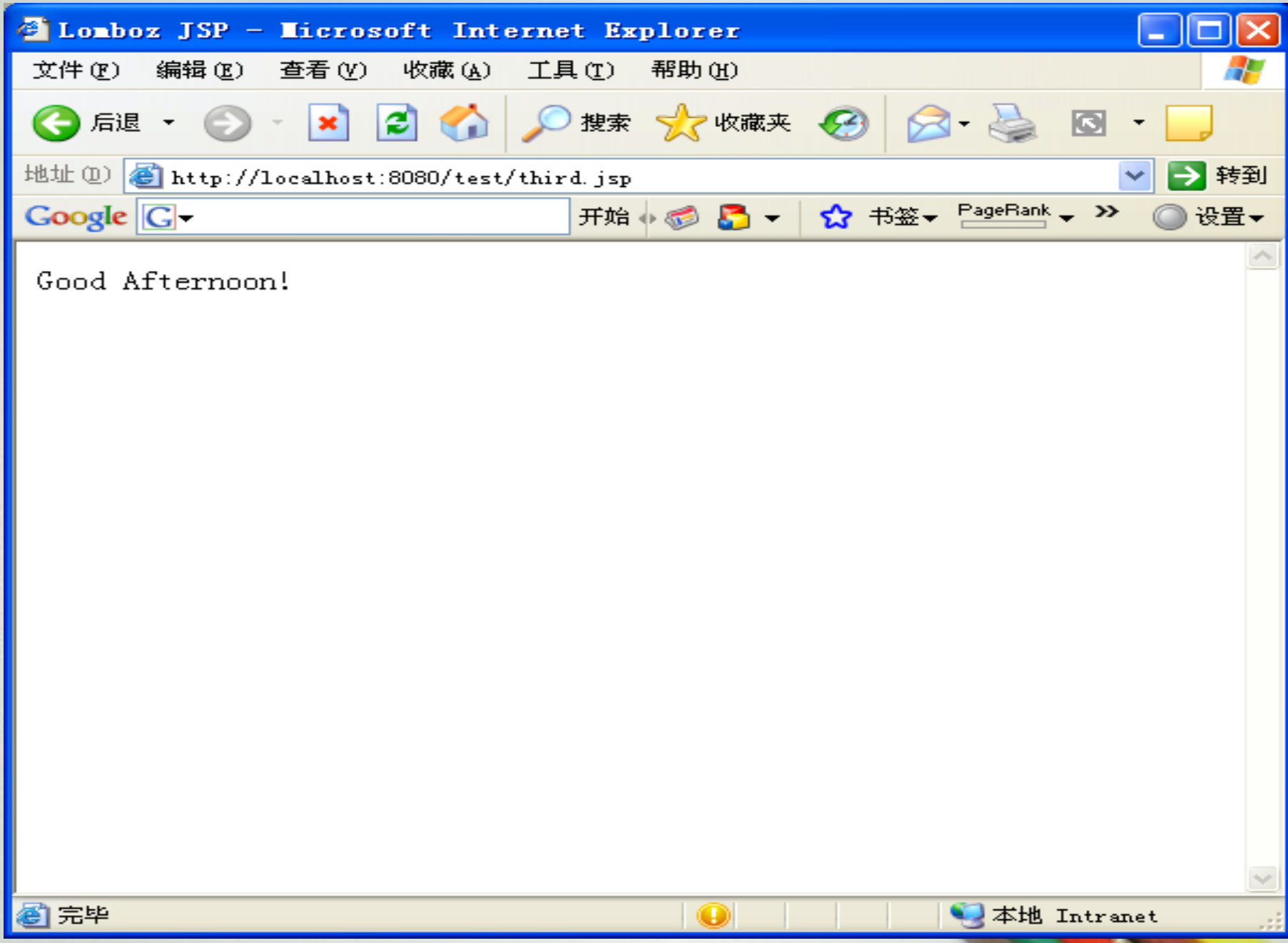
- <%**if**(currentHour < 12){%>
- Good Morning!
- <%} **else if**(currentHour > 12 && currentHour < 18){%>
- Good Afternoon!
- <%}**else**{%>
- Good Evening!
- <%}%>

继续编写稍复杂的JSP

保存

在浏览器地址栏输入：

http://localhost:8080/test/third.jsp


Lomboz JSP - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
收藏夹
地址(D) http://localhost:8080/test/third.jsp
转到
Google
开始
书签
PageRank
设置
Good Afternoon!
完毕
本地 Intranet

继续编写稍复杂的JSP

深入理解JSP Script的写法与意义