

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

## 《服务器集群技术》JSP/Servlet 速成教学手册 (3)

1 周半 3 堂共 9 节课郭氏速成法

**集群**的学习的原则和技巧 10 分: 安装 4, 配置 3, 实验 2, 原理 1

I = Gitee+Eclipse+Clustering+DB+Middleware = Some Internet Coding Keys

我 = 代码仓库+IDE+集群+数据库+中间件 = 一些互联网开发的关键技术

作者: 郭鹏@北京理工大学珠海学院-计算机学院, 用于《服务器集群技术》授课

2019-09-26

**前言:** 对于缺少项目经验和编程不多的在校的大学生来说, 好的工具非常重要, 好的系统集成更是重要!

代码仓库使用和 IDE 集成的重要性在于:

团队之间可以合作, 代码历史可以跟踪, 有利于大学生创业, 提前进入工业界的状态。

### 0. 本手册主要的内容, 欲学 Tomcat, 先懂 JSP/Servlet !

I-1) 下载和安装 Tomcat 8, 以及 Eclipse 的一些常用功能。

II-2) 为什么要学 JSP 和 Servlet? 概括比较一下 CGI, PHP, ASP/C#, Node.js, Python。

3) 互联网 Web, B/S 的主要特点和要素。

4) 一个 Hello World 的 JSP 和 Servlet 的样子。

5) JSP/Servlet 要素之 1: 在哪运行? 主要在 Tomcat 容器。

6) JSP/Servlet 要素之 2: 这种特殊的 Java Class 的生命周期是什么? 生与死。

7) JSP/Servlet 要素之 3: 最重要的两大系统级 Object: 请求 (request) 和 响应 (response)。

8) JSP/Servlet 要素之 4: 页面跳转 (登录, 出错, 业务分流), 页面包含 (模块化), 上传。

9) JSP/Servlet 要素之 5: Cookie, Session 状态信息在哪? 存在的意义, Http 连接的无状态性。

III-10) JSP/Servlet 高级用法 1: Ajax, WebSocket, Tomcat 服务器的互动。

11) JSP/Servlet 高级用法 2: 不同级别的变量的传递 (外部环境, 静态), Thread 变量呢? 在哪?

IV-12) JSP/Servlet 高级用法 3: 框架 SSM, SSH, MVC 概念和框架的配置。

这一切都发生在 IDE 工具 Eclipse 上, 安转, 配置一到多个 Tomcat, 而且学会 JSP/Servlet, JavaScript 的断点调试

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

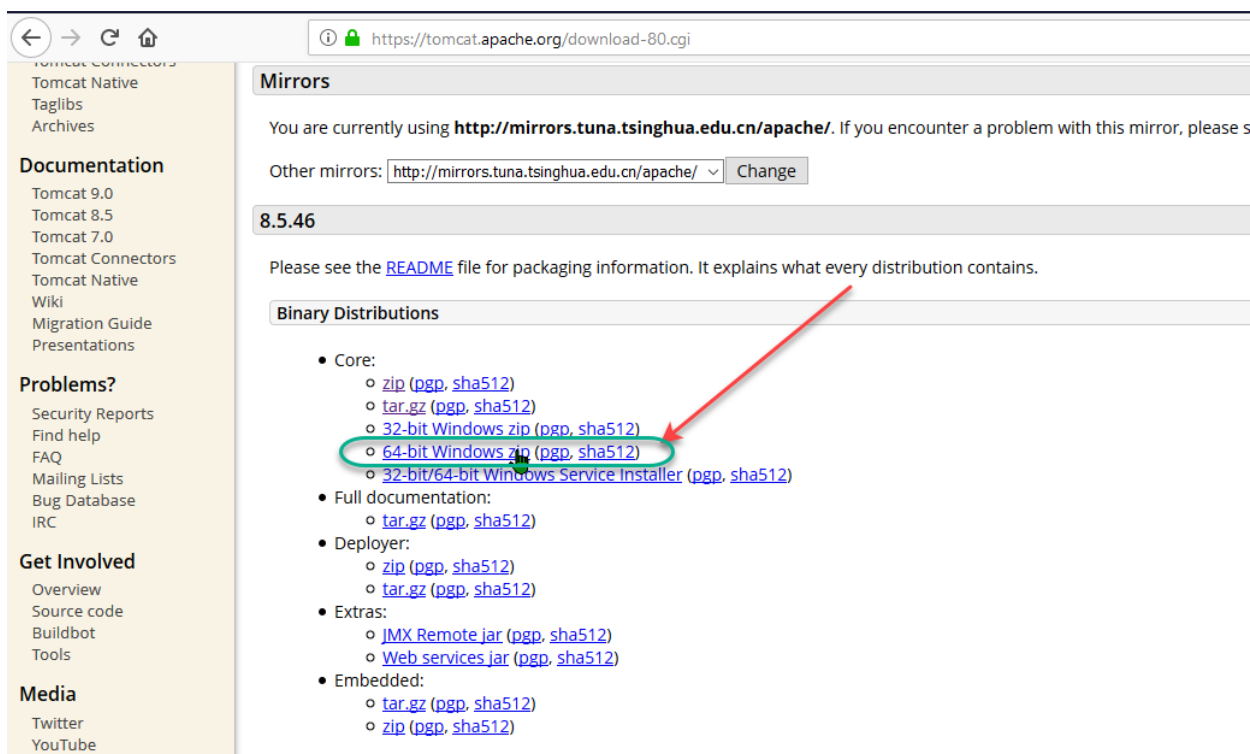
声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止,  
使用时不得擅自改动, 不得商用牟利。

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

## I-1) 下载和安装 Tomcat 8, 以及 Eclipse 的一些常用功能。

步骤 1) 链接: <https://tomcat.apache.org/download-80.cgi> 选择适合 windows10 的版本如下:



步骤 2) 下载后移动 apache-tomcat-8.5.46-windows-x64.zip 到 \SCT\Downloads 下. (文件名因版本变化可能有所不同)

步骤 3) 解压后, 拷贝到 \SCT\Servers\tomcat 中, 最后的目录结构如如下:

SCT > Servers > tomcat				
<input type="checkbox"/> Name	Date modified	Type	Size	
bin	4/10/2019 3:31 PM	File folder		
conf	4/10/2019 3:31 PM	File folder		
lib	4/10/2019 3:31 PM	File folder		
logs	4/10/2019 3:31 PM	File folder		
temp	4/10/2019 3:31 PM	File folder		
webapps	4/10/2019 3:31 PM	File folder		
work	4/10/2019 3:31 PM	File folder		
BUILDING	4/10/2019 3:31 PM	Text Document	20 KB	
CONTRIBUTING.md	4/10/2019 3:31 PM	MD File	7 KB	
LICENSE	4/10/2019 3:31 PM	File	57 KB	
NOTICE	4/10/2019 3:31 PM	File	2 KB	

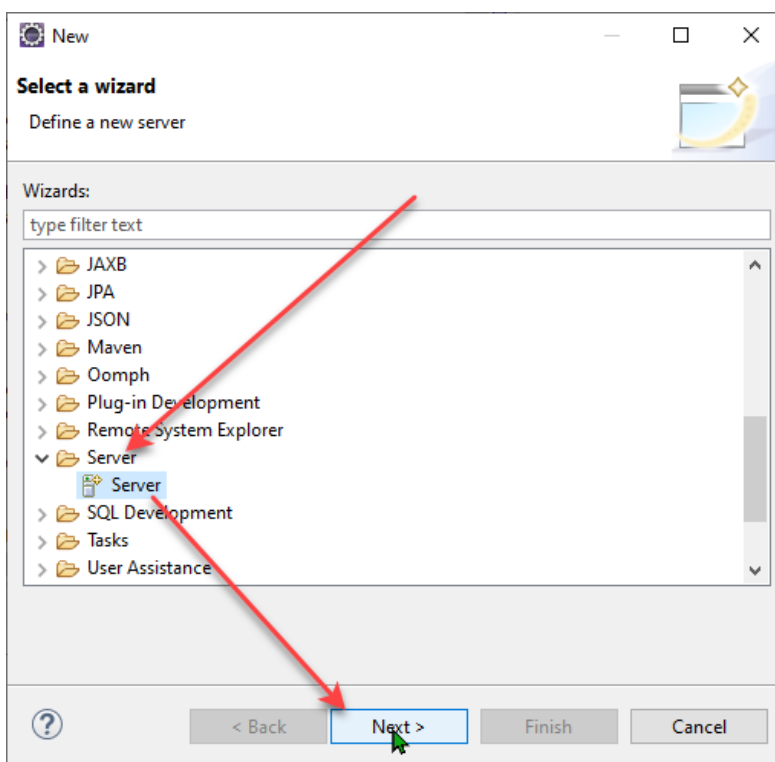
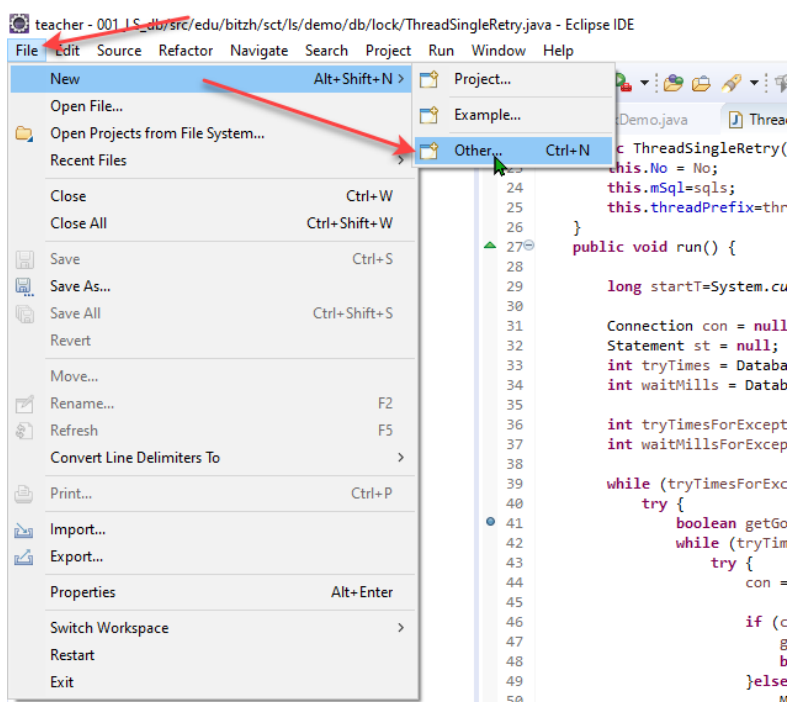
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止,  
使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

步骤 4) 在 Eclipse 生成 4 个 Tomcat 服务器，分别命名：Tomcat-1-8080，Tomcat-2-8081，Tomcat-3-8082，Tomcat-4-8083

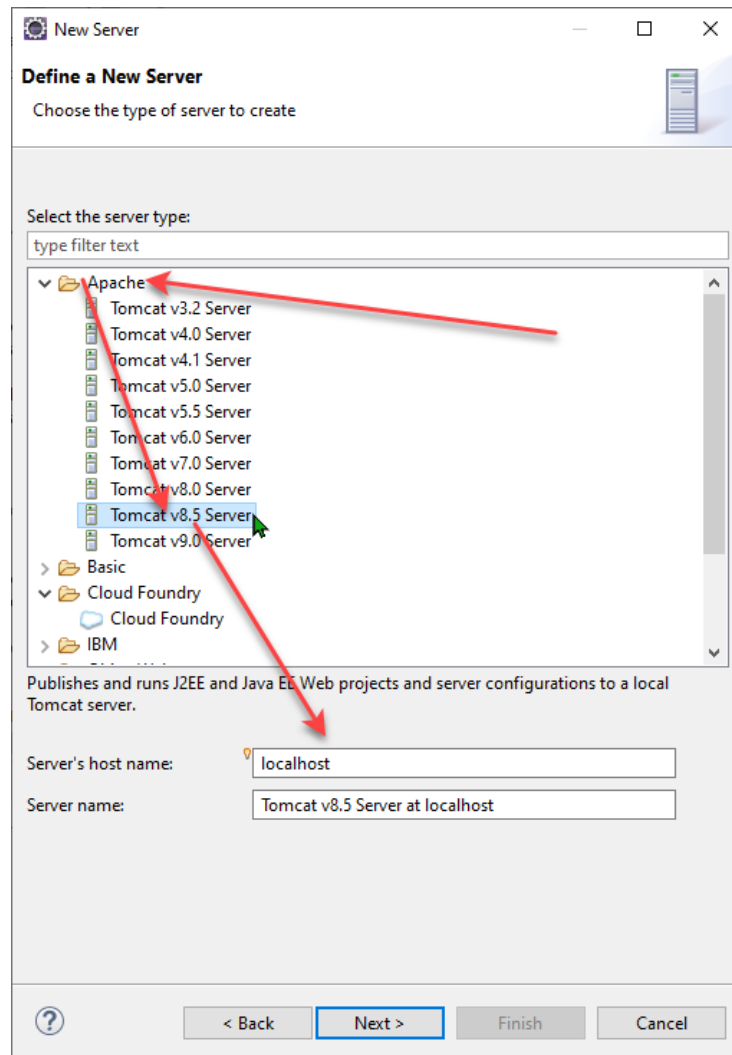


郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

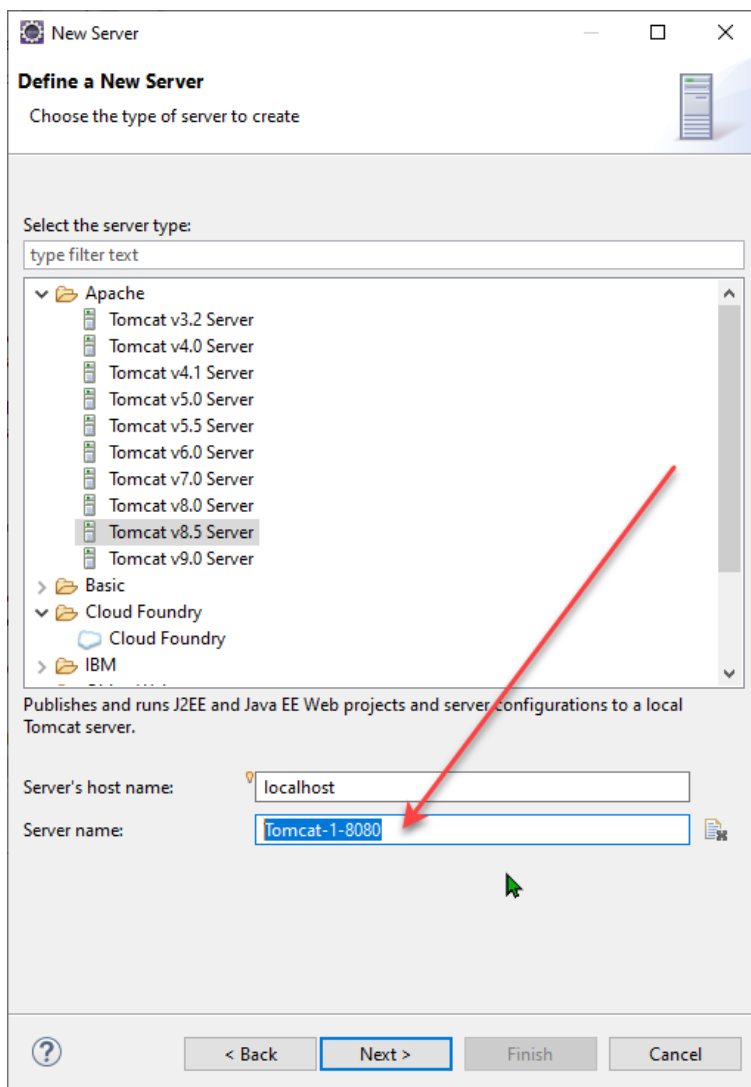
联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

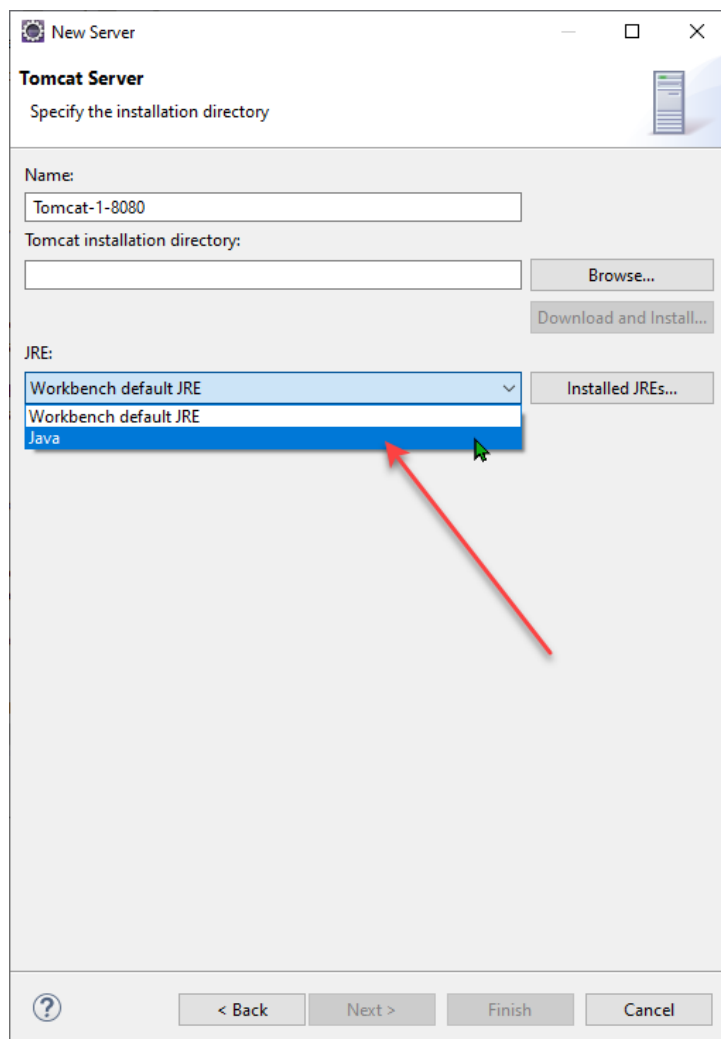


改名 Server Name 为 : Tomcat-1-8080

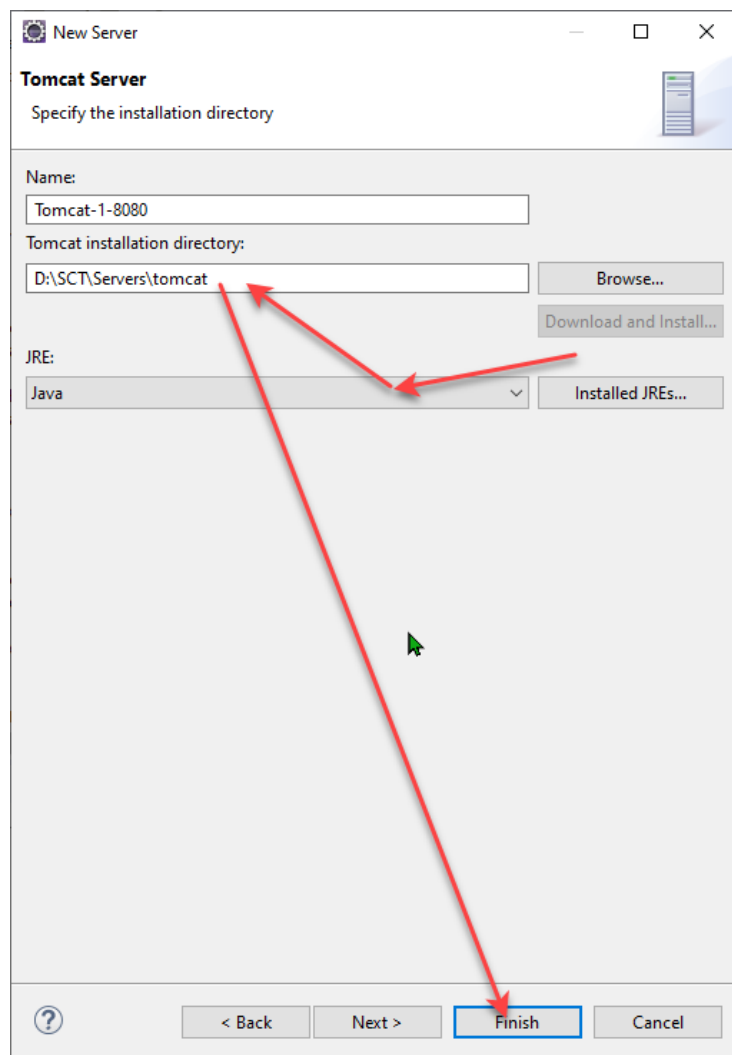
Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

点击“Finish”后, Eclipse 就会出现 Servers 的项目 Project, 点击展开后, 打开 server.xml, 这是 Tomcat 的最重要的配置文件, 额可以看到这 3 行:

63 行: 端口 8080

```
<Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>
```

这个影响到 Tomcat-1-8080 这个 Tomcat 容器 (一个网页服务器) 在后面网站的入口链接的端口: <http://localhost:8080>

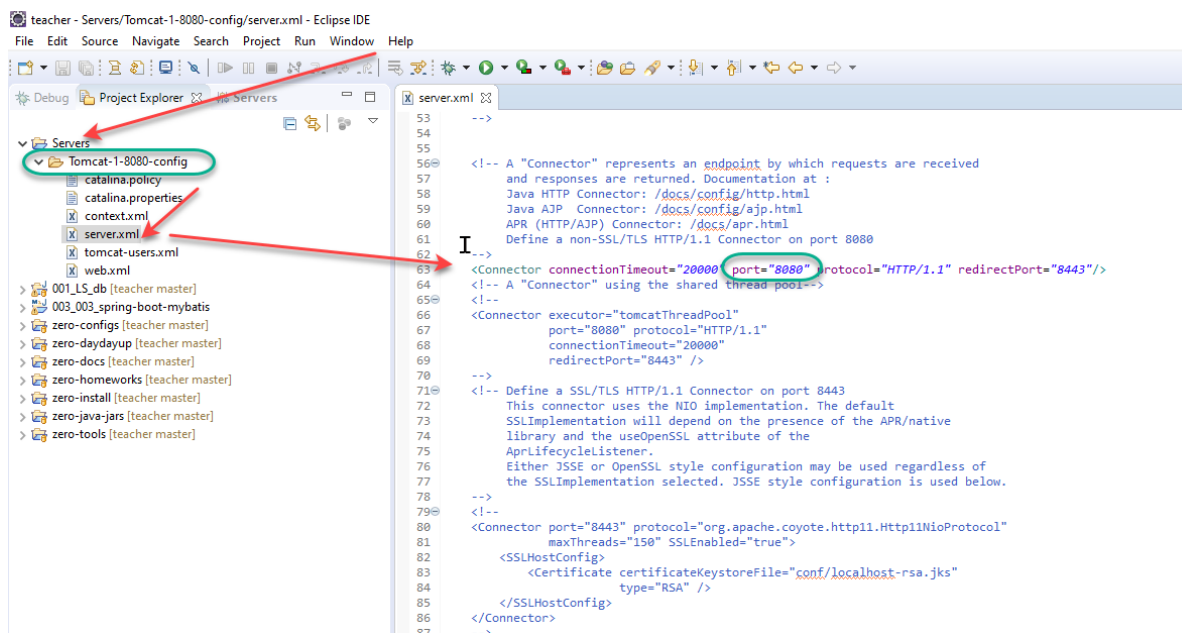
20 行: 端口 8005

```
<Server port="8005" shutdown="SHUTDOWN">
```

108 行: 端口 8009

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443"/>
```

后面安装的其他 Tomcat 服务器, 这以上 3 个端口必须修改, 保持唯一性, 因为我们有时需要这 4 个 Tomcat 同时启动运行!



OK, 大家依样画葫芦, 按照这些步骤, 安装 Tomcat-2-8081, Tomcat-3-8082, Tomcat-4-8083。而且各自的 server.xml 中的以上端口有规律地改为:

Tomcat-2-8081 : (8081,8006,8010)

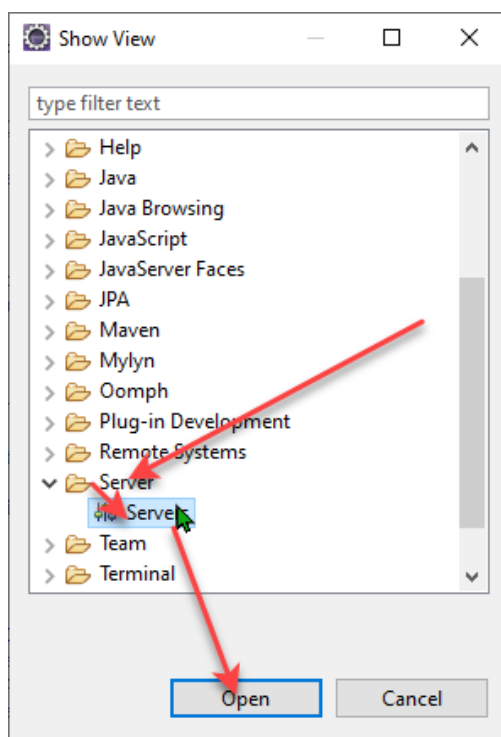
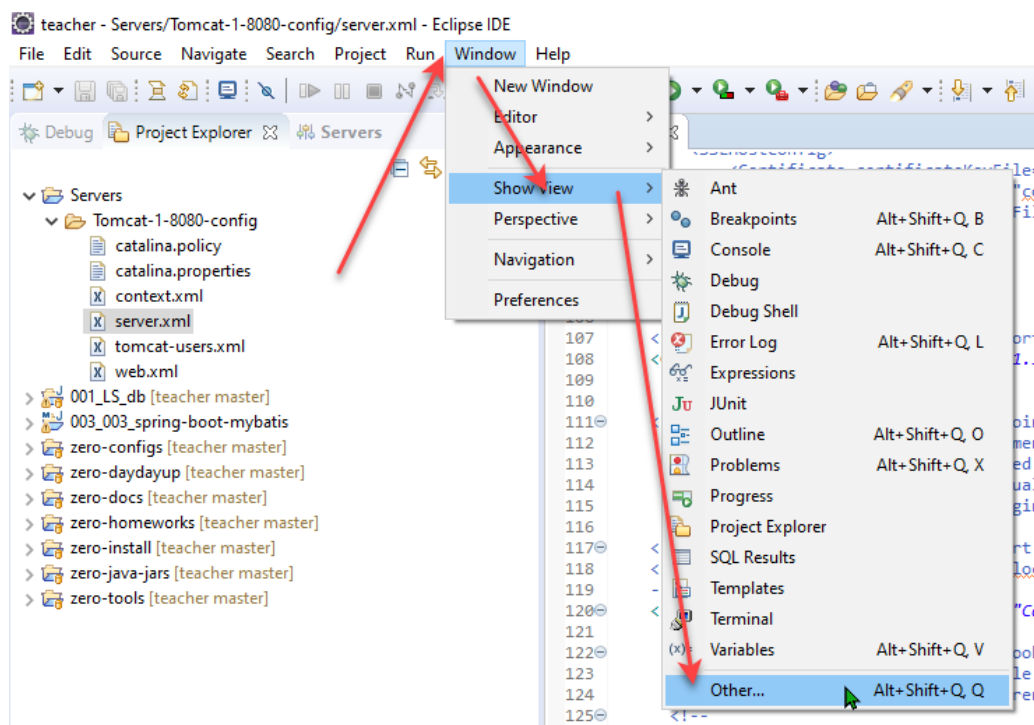
Tomcat-3-8082 : (8082,8007,8011)

Tomcat-4-8083: (8083,8008,8012)



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

那么我们打开 server 的界面看看服务器的样子，并运行一下，看是否正常吧。

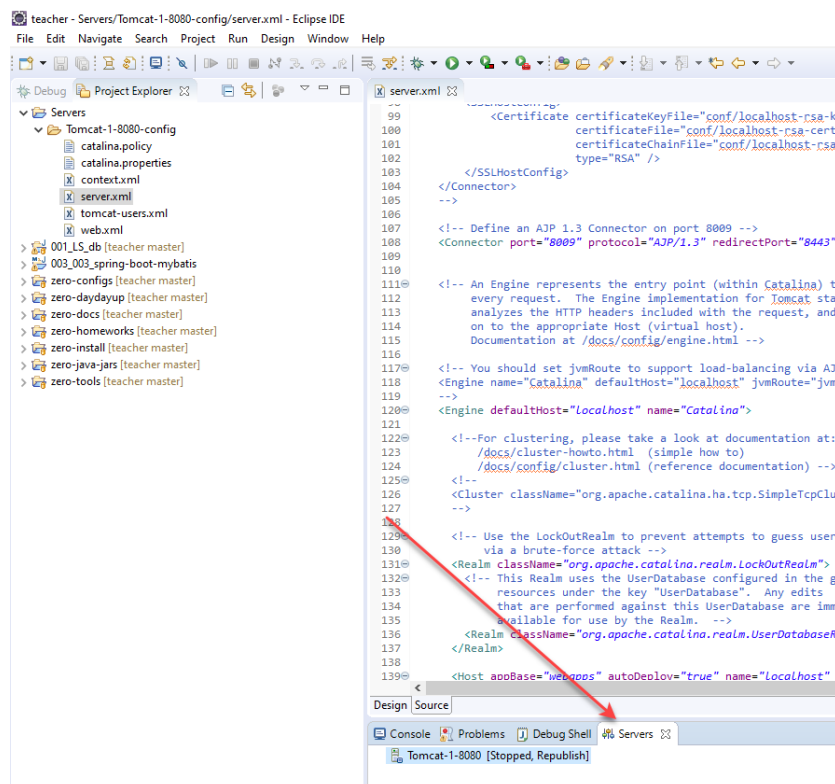
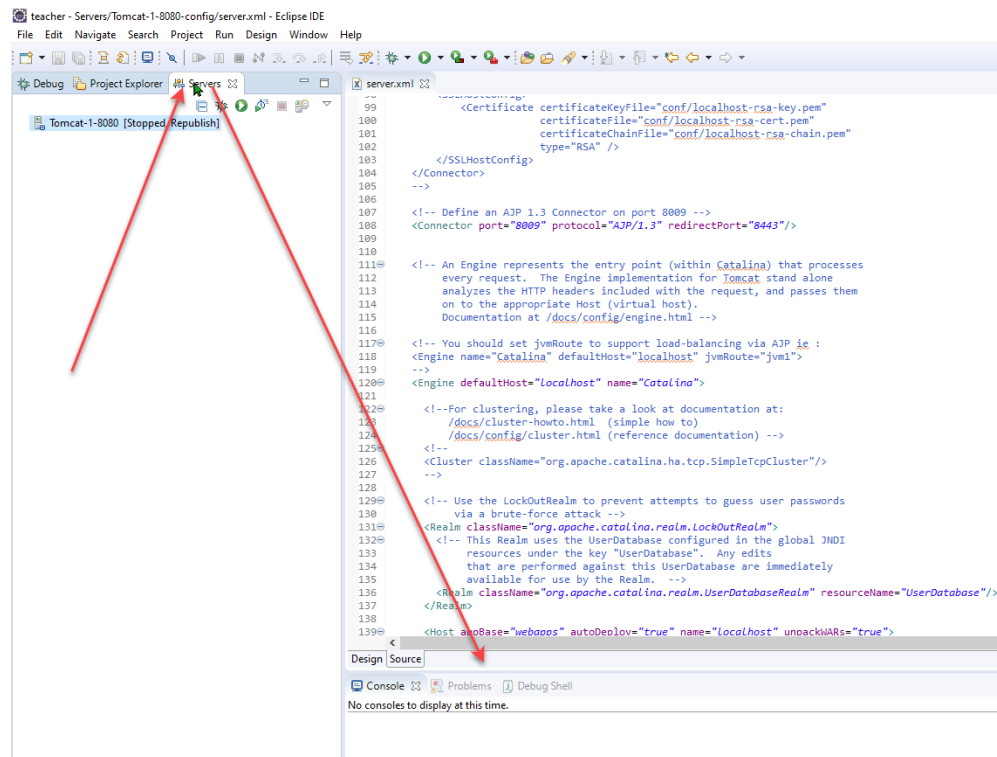


郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！



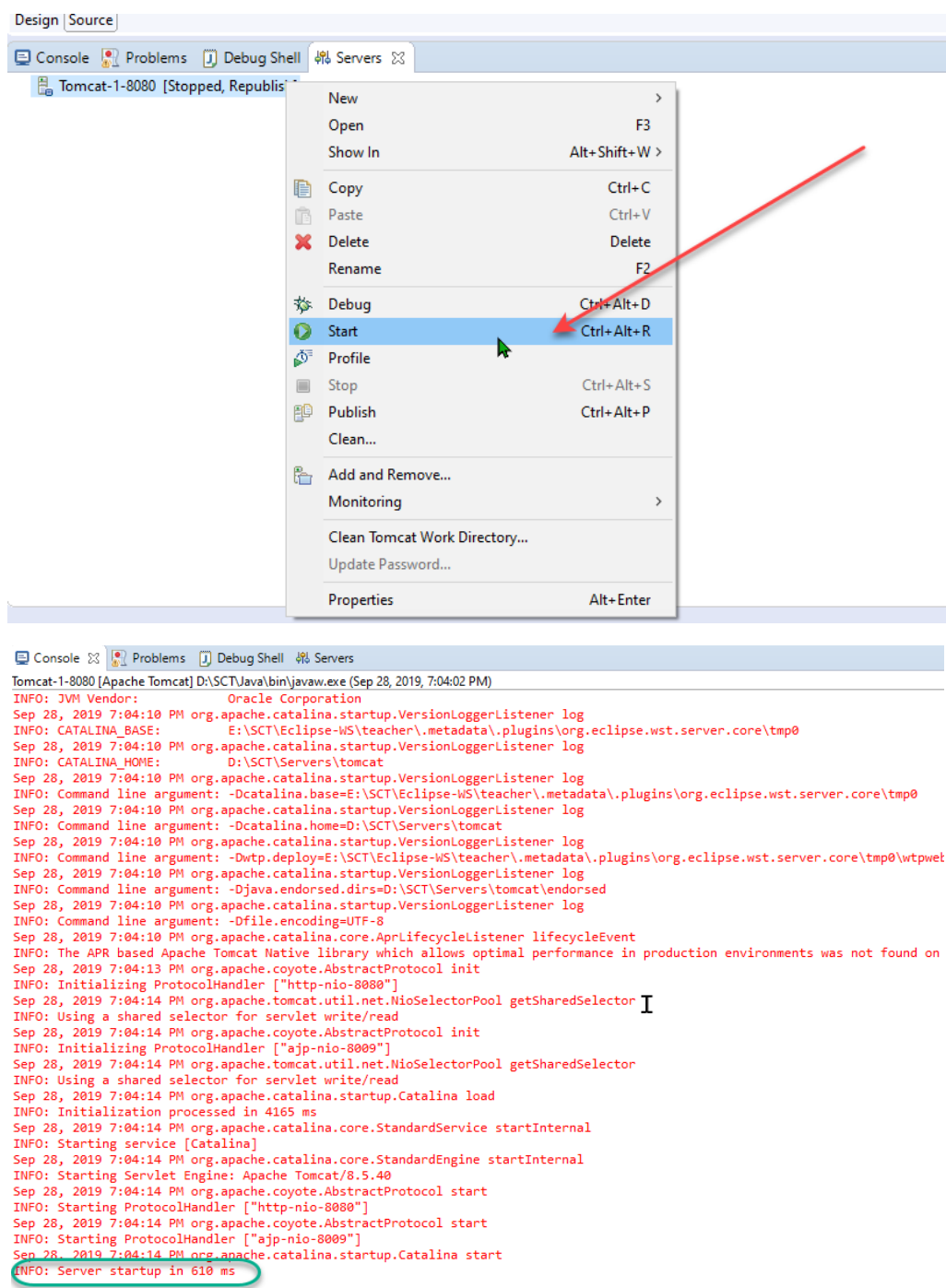
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止,  
使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

OK, 我们来启动 Tomcat, 大概 1 秒钟后, 可以从 console 控制台和 Servers 看结果和状态可以从 console 控制台和 Servers 看结果和状态:

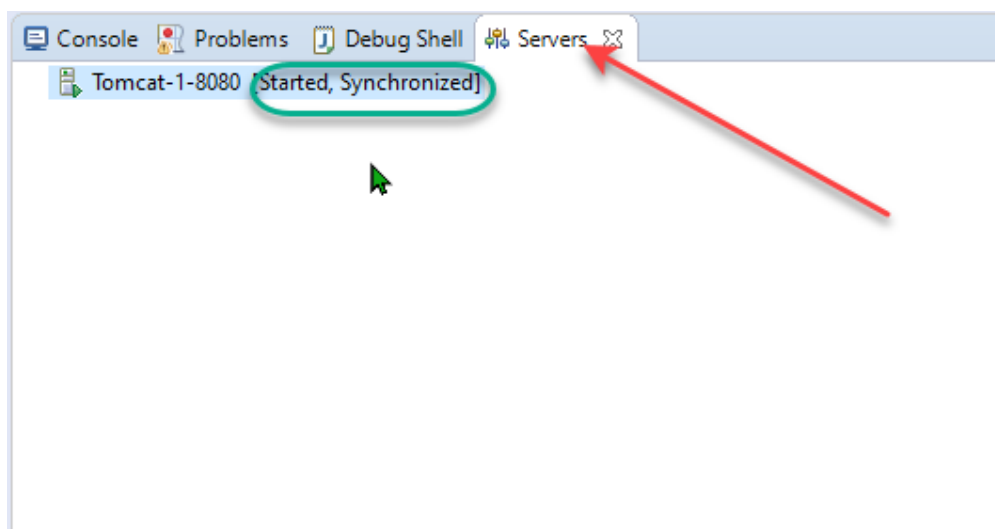


郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

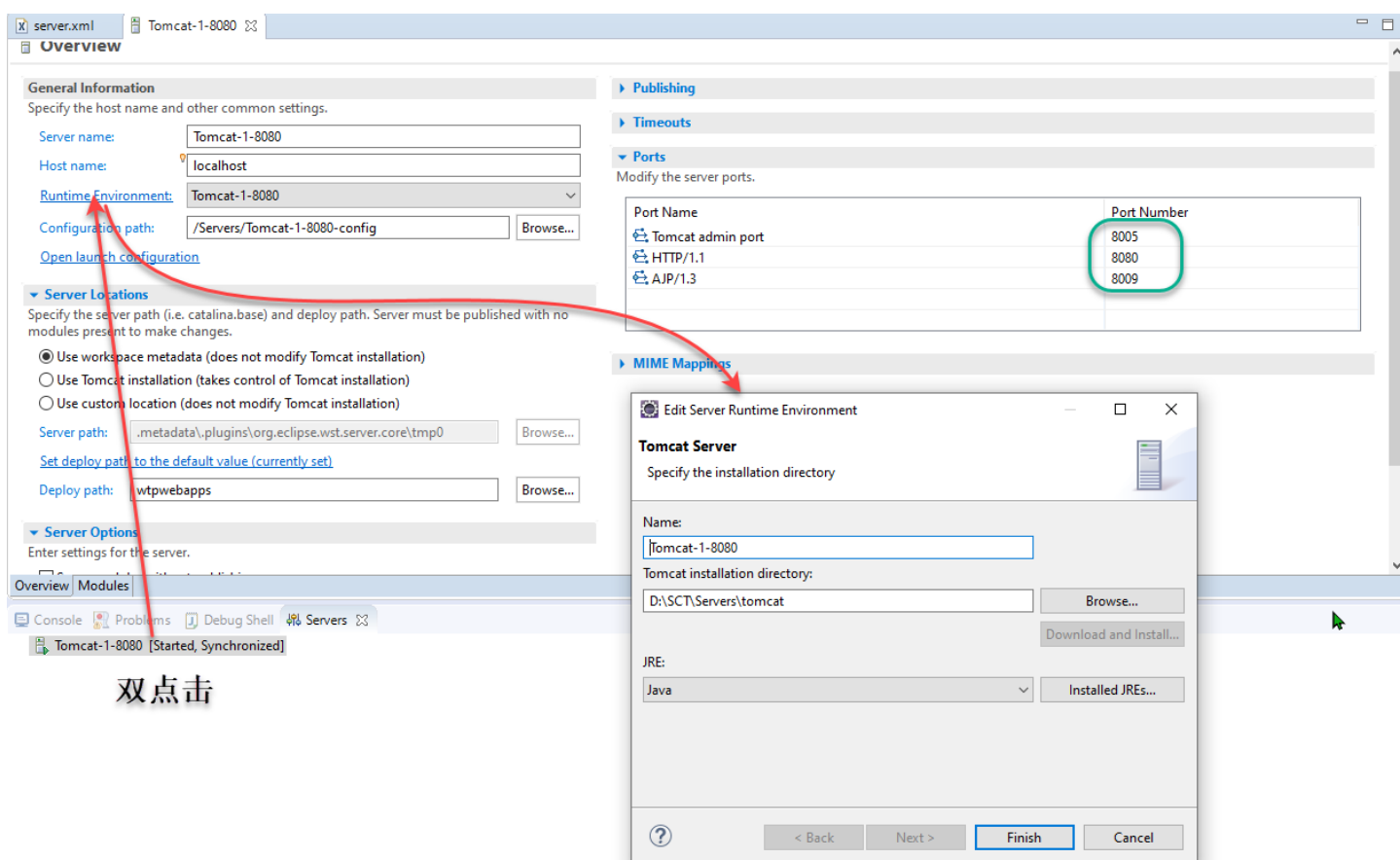
声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止, 使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！



另一角度查看 Tomcat 服务器，如下操作：



双点击

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

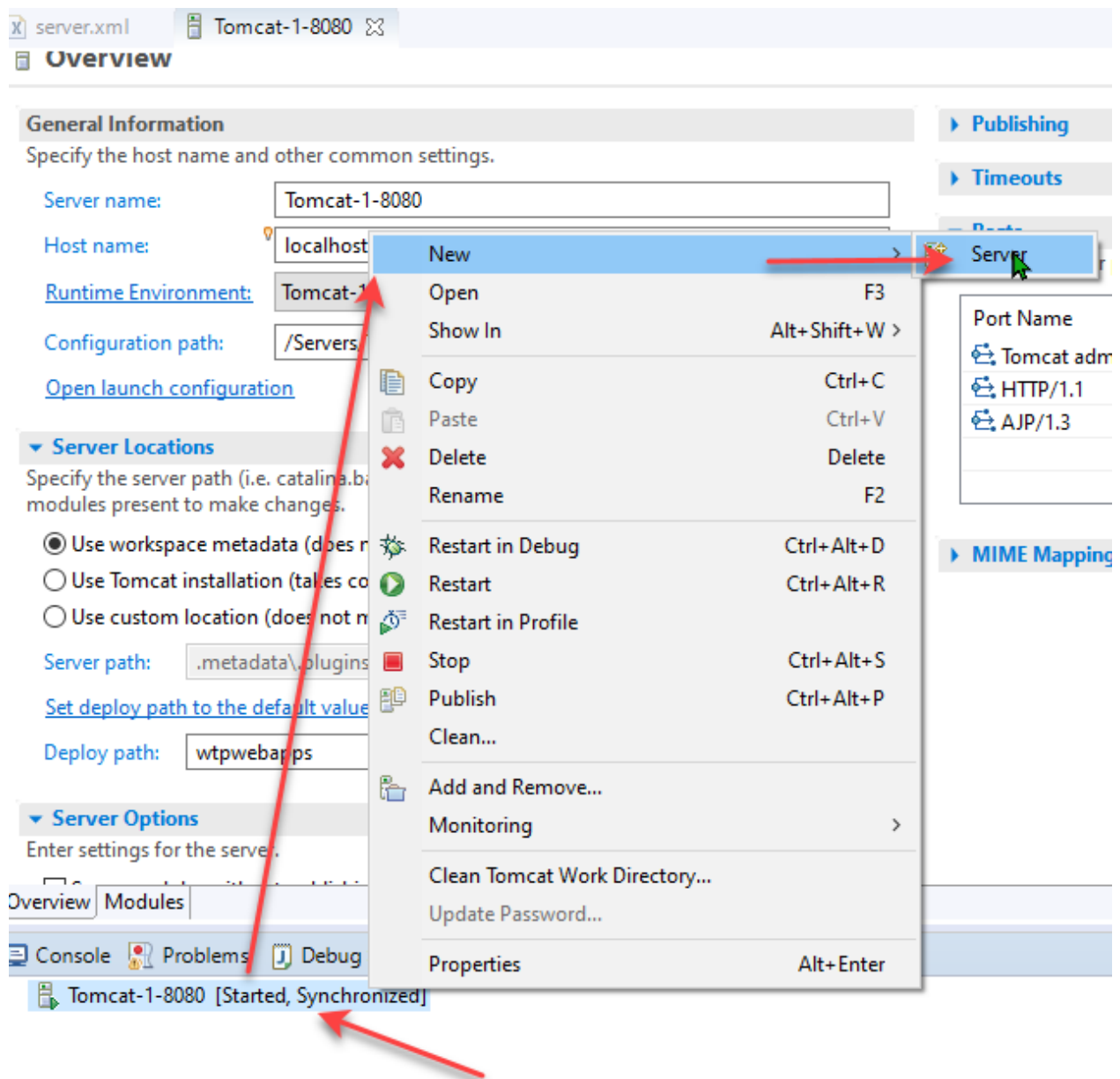
声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

下面是如何依样安装的要点：

### 步骤 1)



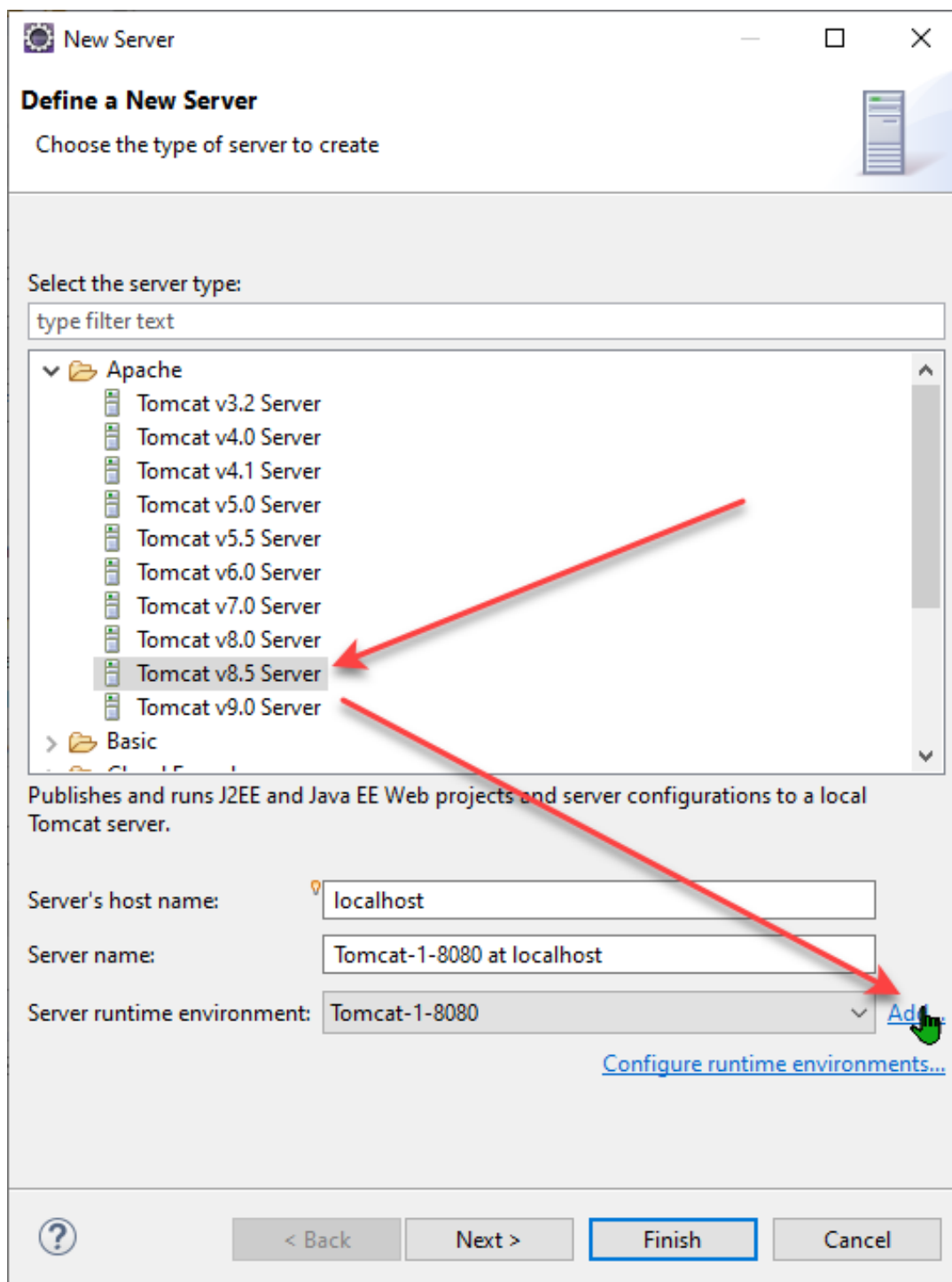
### 步骤 2)

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

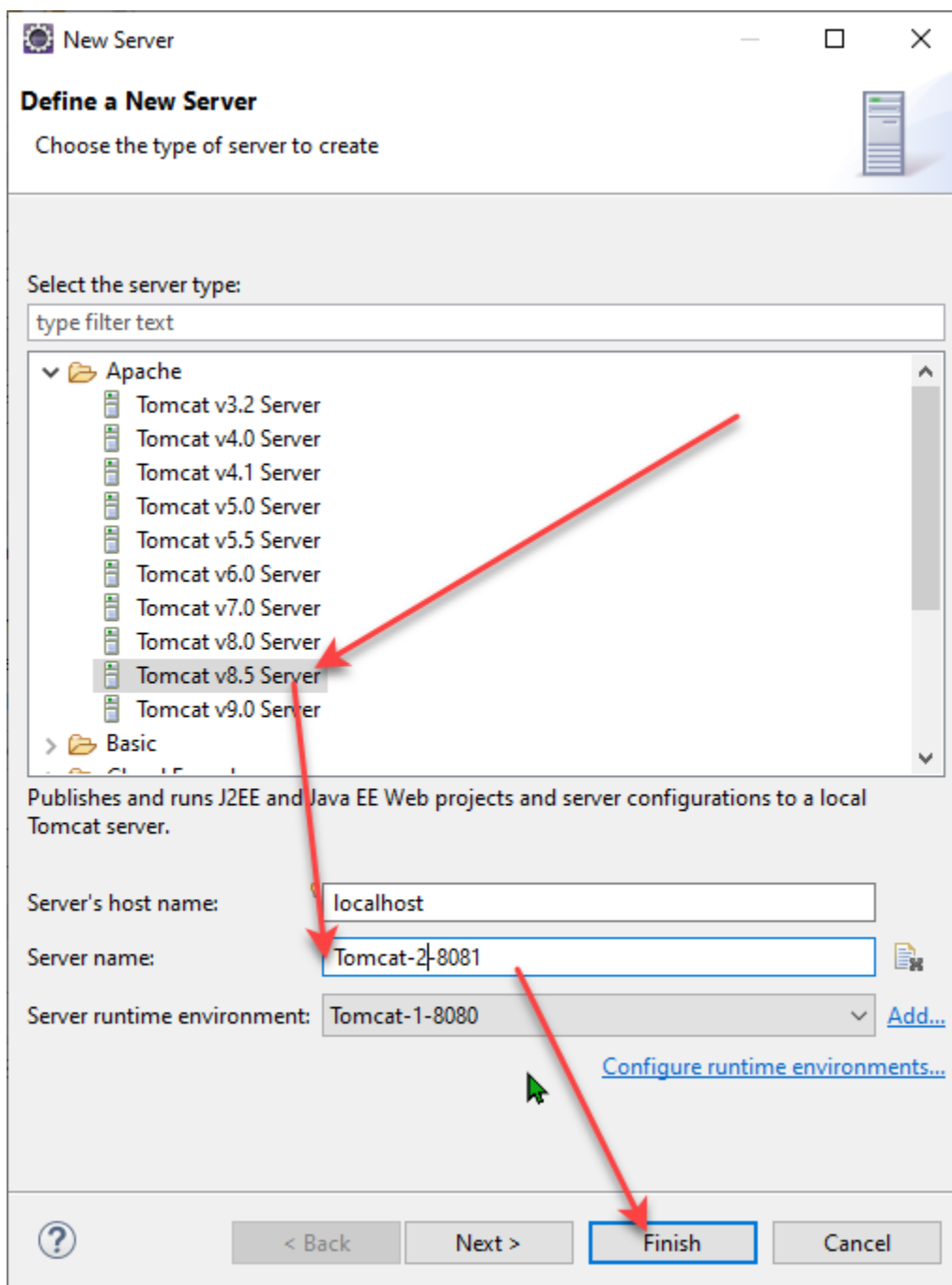
联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

### 步骤 3)



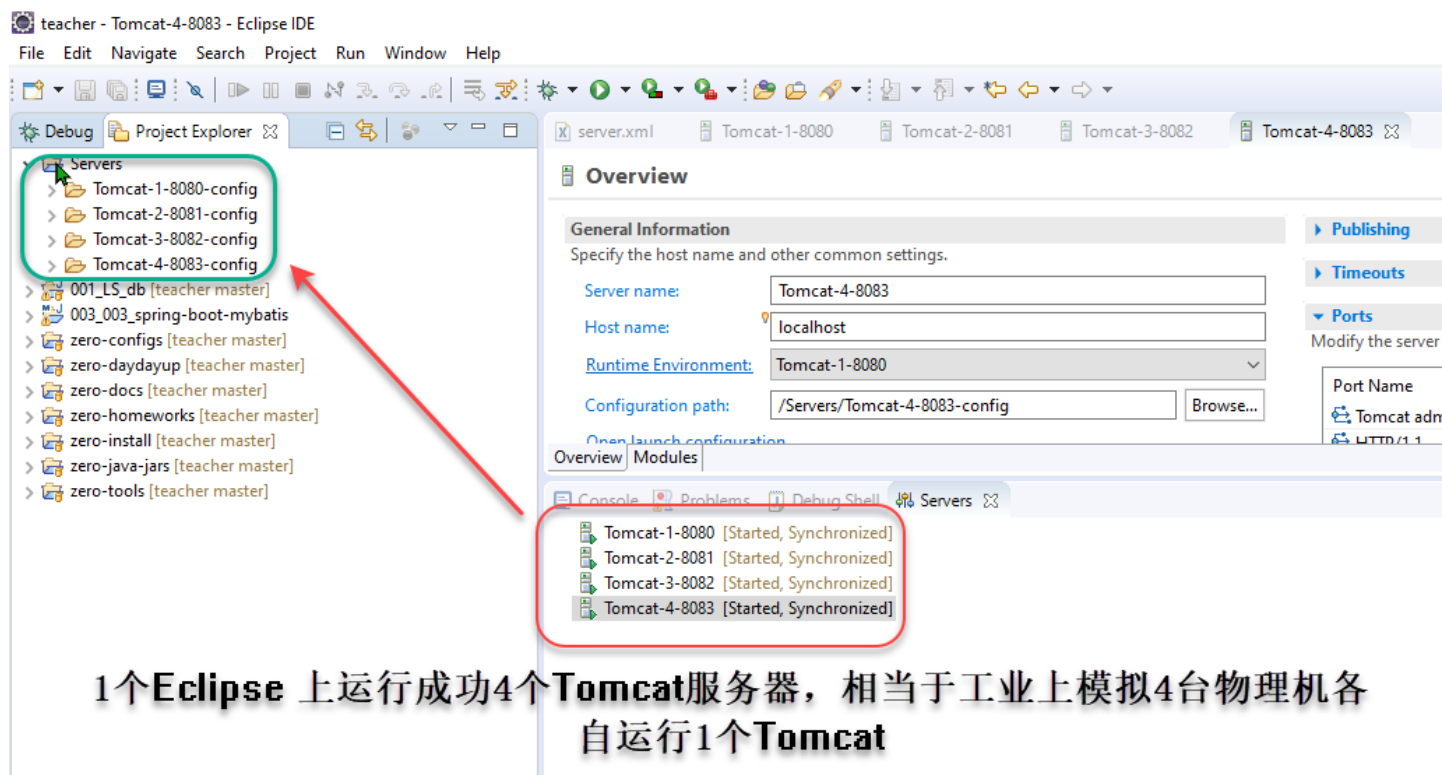
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止,  
使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

最后全部启动，正常的效果：



至此，你们应该掌握在 Eclipse 安装和配置多台 Tomcat 的技巧了，**关键点**在于**不要引起端口冲突！**

后面还会讲解 Tomcat 使用**不同协议**和**线程池**配置引起的不同，这些是关于**性能**和**微调**的内容。



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

## II-2) 为什么要学 JSP 和 Servlet ?

### 概括比较一下 CGI, PHP, ASP/C#, Node.js, Python。

因为 JSP/Servlet 是目前最流行的互联网编程语言, 无论在国外的大厂 (facebook , google, snashop ) 还是国内的大厂 (阿里巴巴, 京东, 华为等), 而且本身就是 Java 语言, Java 语言是手机开发安卓的主要语言, 也是大数据开发的第一语言, 同时在人工智能和机器语言应用上也有一席之地, 所以学好 JSP/Servlet 有连带的好处和学习上的规模效应。

在互联网开发语言上:

**CGI** 是早期的编程语言, Perl 是一个广泛被用来编写 CGI 程序的语言, 但 CGI 的一个目的是要独立于任何语言的。Web 服务器无须在这个问题上对语言有任何了解。事实上, CGI 程序可以用任何脚本语言或者是完全独立编程语言实现, 只要这个语言可以在这个系统上运行。除 Perl 外, 像 Unix shellscript, Python, Ruby, PHP, Tcl, C/C++, 和 Visual Basic 都可以用来编写 CGI 程序。

**PHP** 使用方便, 开发快捷, 变量定义简单和灵活, 但是不容易实现**多线程**, 缺乏**异步**, 缺乏**非阻塞**编程模式, **内存不驻留**, 所以近十年来只要被局限在中小网站开发上, 组合开发包 WAMP, LAMP 中的 P 就是 PHP。

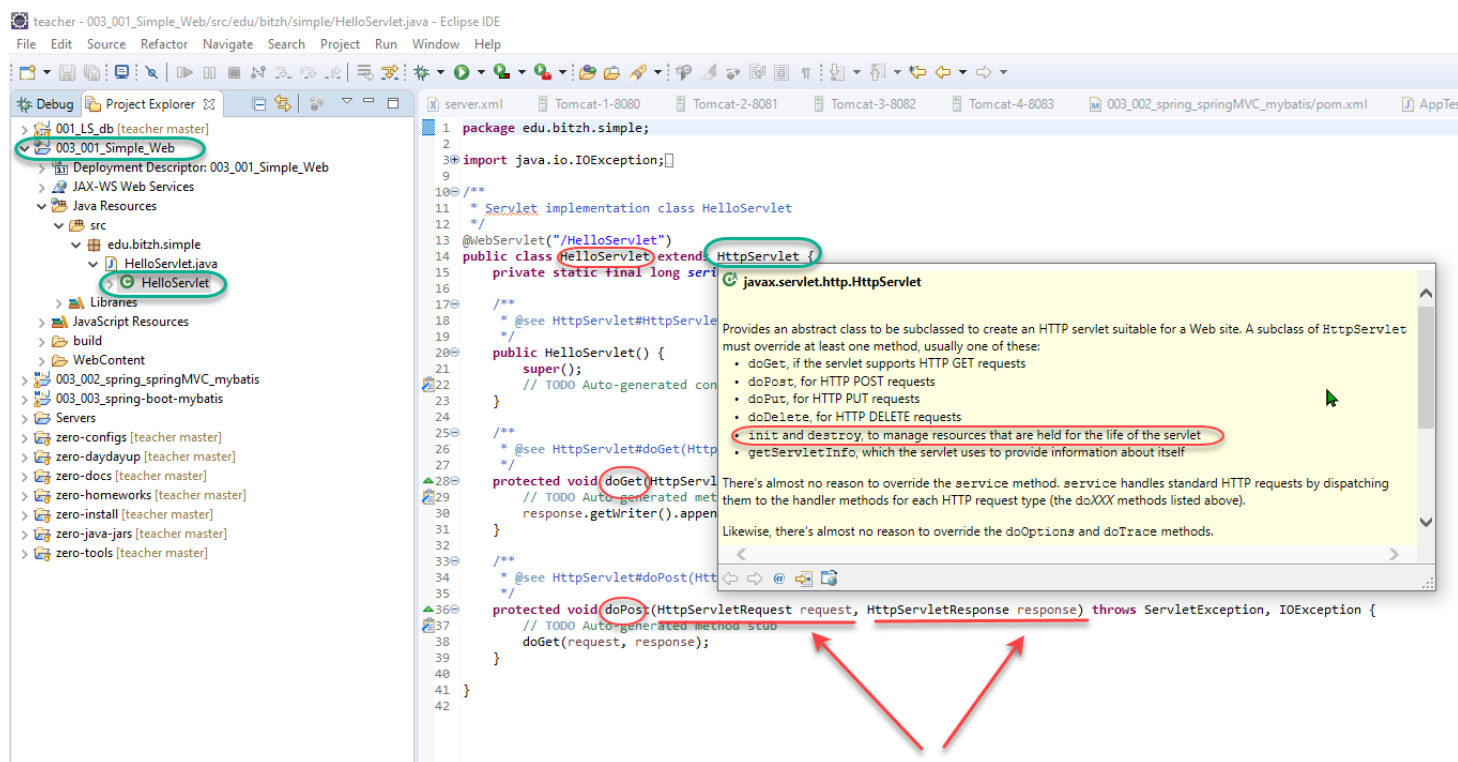
**ASP/C#** 是微软公司的出品, 虽然不错, 但遭到了其他工业界大佬们的忌惮, 而且被捆绑在微软的操作系统上也阻碍了推广。

**Node.js** 号称第一个全栈语言, 因为本身就是 Java Script 在服务器后端的应用, 加上前端使用 Java Script, 所以就成了前后端通吃的全栈开发语言, Node.js 的**非阻塞**比较好, 市场需求在上升。

**Python** 是人工智能的第一语言, 在 Linux 服务器的脚本编写等也用得比较多, 网页开发不是其强项, 本身以轻量 and 库多为特点。

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

一个最简单的 Servlet 就是下面这样，可以由 Eclipse 点击选择几步完成：



一个简单的 Servlet 的样子，一般存在一个 Java Dynamic Web Project，是系统的 `javax.servlet.http.HttpServlet` 类的子类

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

带来个方法 `doGet` 和 `doPost`，两个系统的参数，Object 类别的参数：`request` 和 `response`，这就是标配！

Servlet 的兄弟 JSP 呢？也一样，因为 JSP 编译后就是 Servlet，Servlet 从英文的含义就是：一个小的服务。

let 后缀是“小”，serv 当然是“服务”。

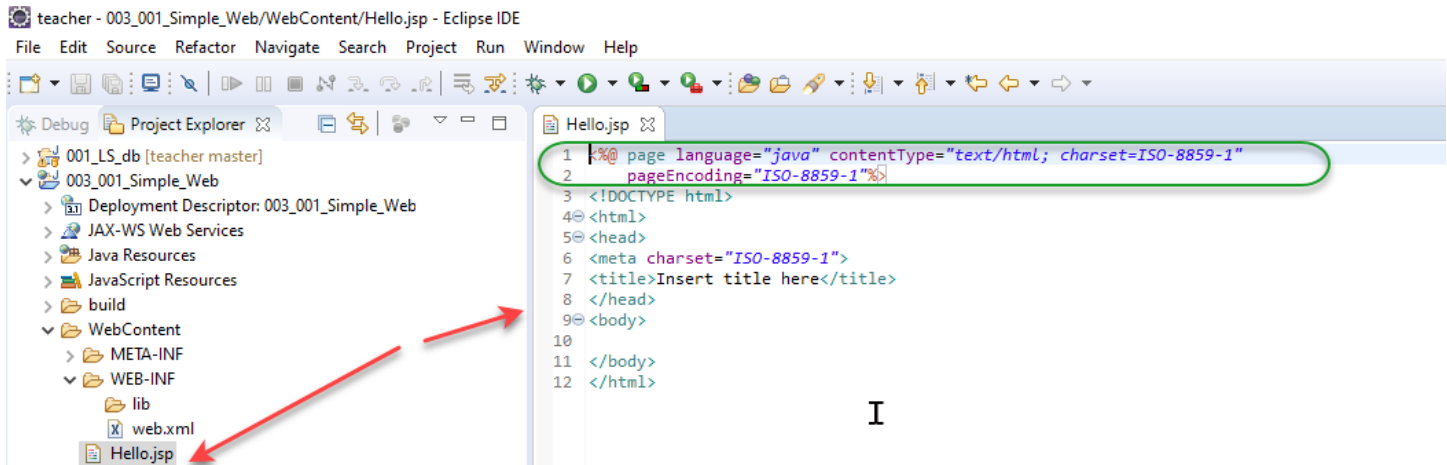
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册（3）

声明：版权属于郭鹏（Peng Guo），允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

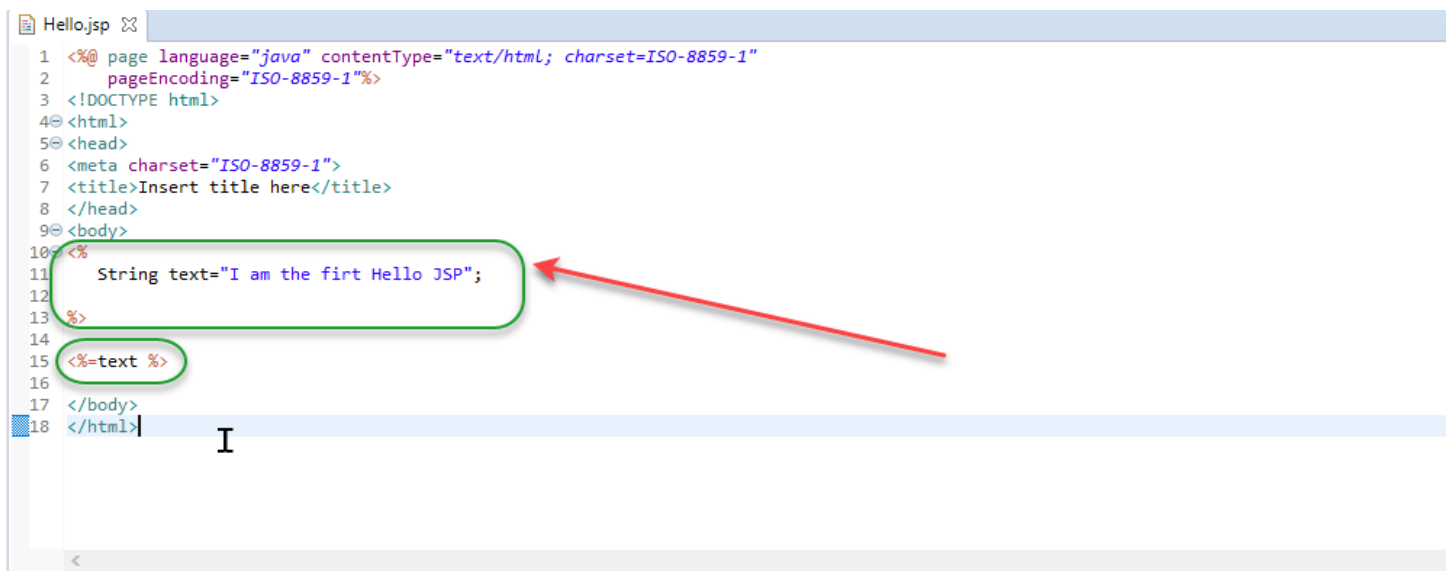
Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

一个最简单的 JSP 就是下面这样，可以由 Eclipse 点击选择几步完成：



这里面有两大特点，一是好像是 html 代码和一些 Java 代码混合，二是看不到 servlet 里那样明显的 request 和 response 的 Java Object（对象和实例），因为 JSP 的 request 和 response 是隐式的，即存在但一般看不到，你不知不觉中就用到了，这两个 Object 有什么作用呢？后面我们会谈到，而且大谈特谈，因为他们非常重要，现在先按下不表。

我们稍加改动，变成这样，可以让 Hello.jsp 从 Tomcat 送回一串文本：**I am the first Hello JSP**



**那么，我们该如何运行以上的 Hello Servlet 和 JSP 呢？**

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册（3）

声明：版权属于郭鹏（Peng Guo），允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

首先, Servlet 和 JSP 是运行在 Tomcat 里的, 我们已经安装了几个 Tomcat, 那么还需要什么呢? 大家要知道 Servlet 和 JSP 是特殊的 Java class, 需要在一般 JDK 上提供的 JVM 上, 再和系统能解释和编译 Servlet/JSP 的系统类 (class):

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

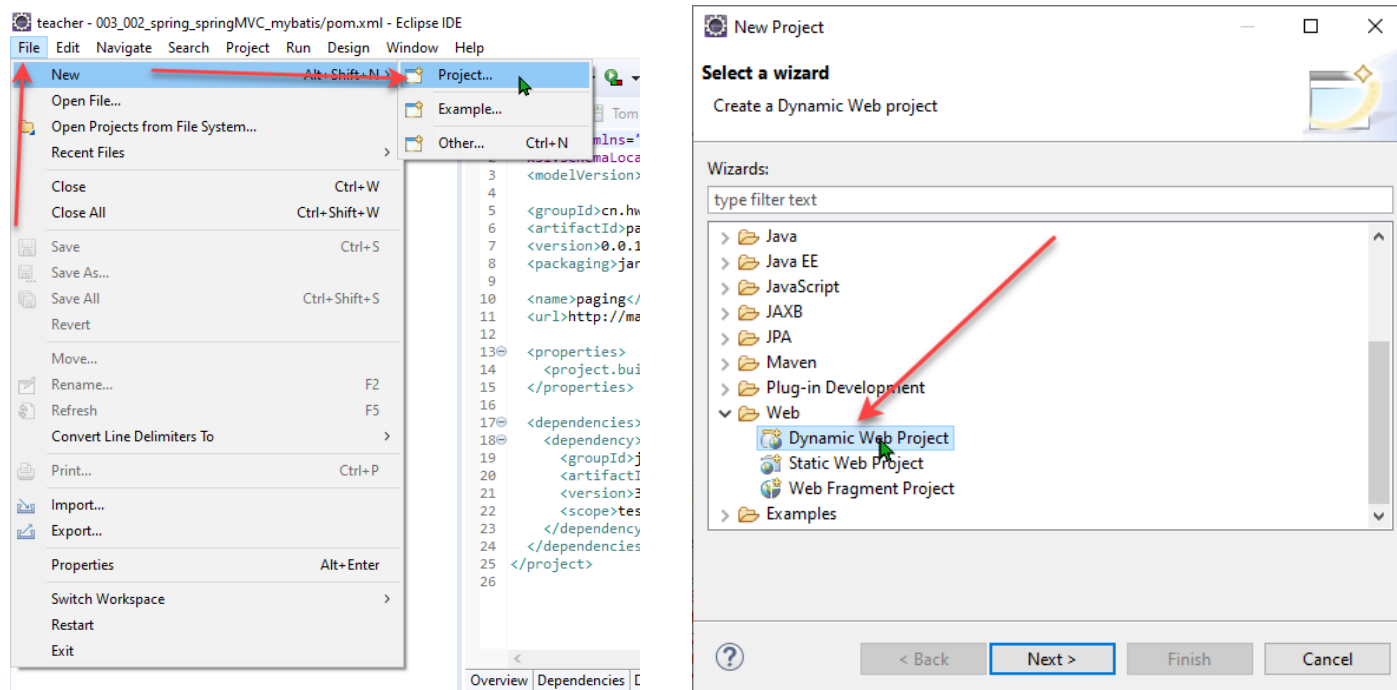
一起打包成一个特殊的项目 project, 为什么要用项目呢? 因为将各种配置和代码, 系统代码 (或叫做库) 打包在一起, 犹如一只军队一样方便指挥, 管理, 调动和打仗, 因为军队里也有不同的角色, 例如有军队里有卫生兵负责医护, 炊事员负责解决吃饭问题等等, 大家分工不同, 协调合作, 就构成了一个整体。

**JSP/Servlet (士兵) → ? (军队) → JDK (社会) → Tomcat (战场) → 打仗 (胜王败寇)**

这个 ? 就是一个动态网页项目 (Dynamic Web Project)。

Eclipse 可以方便地一步一步建立一个动态网页项目:

步骤 1)



郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止, 使用时不得擅自改动, 不得商用牟利。

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

步骤 2)

**New Dynamic Web Project**

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location

☒ Use default location

Location:

Target runtime

Dynamic web module version

Configuration

A good starting point for working with Apache Tomcat v8.5 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name:

Working sets

☐ Add project to working sets

Working sets:

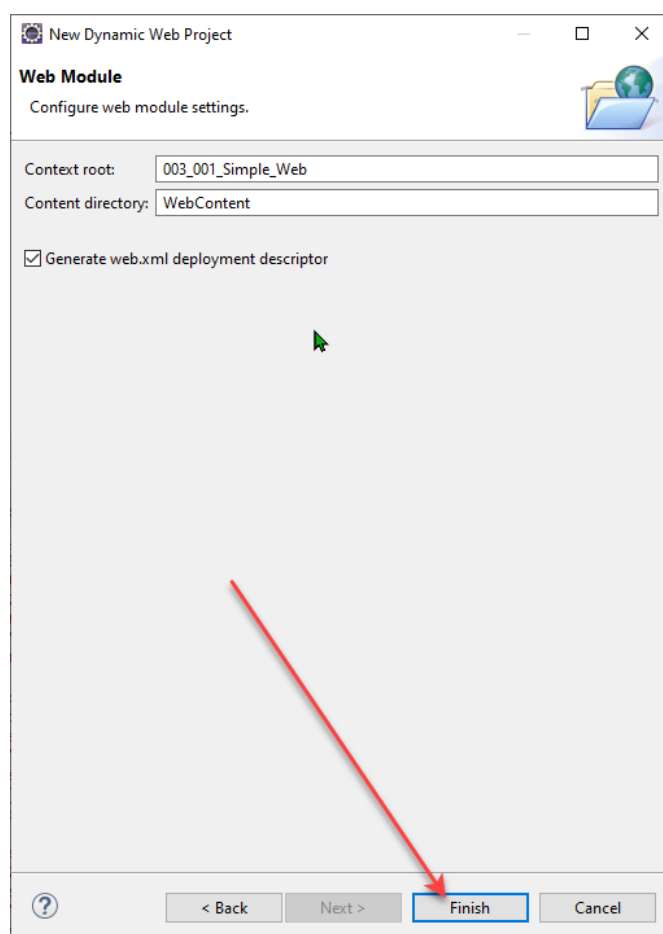
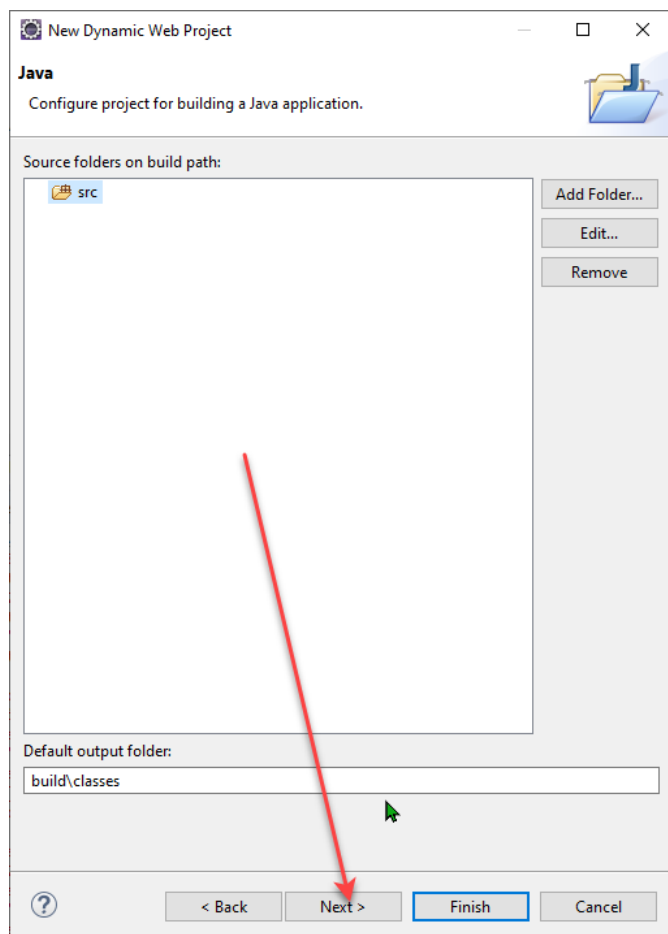
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止, 使用时不得擅自改动, 不得商用牟利.

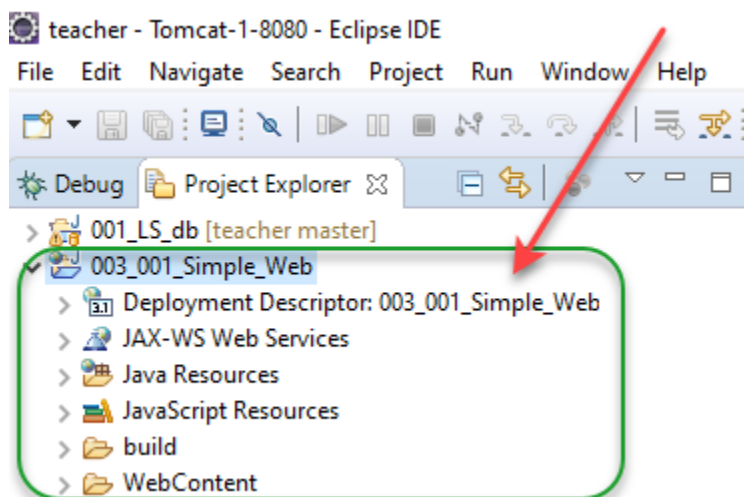
联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

### 步骤 3)



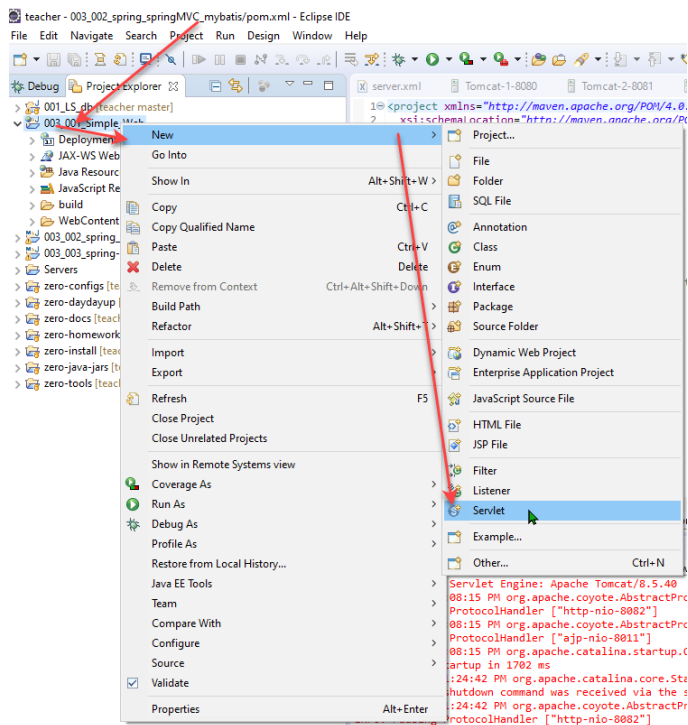
### 4) 结果如下：



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

好，我们就在这个被比喻为军队的新建立的动态网页项目（Dynamic Web Project）中生成两个士兵：Servlet 和 JSP

### 步骤 1) 生成 Servlet，Class 名为 **HelloServlet.java**



**Create Servlet**

Specify class file destination.

Project: 003\_001\_Simple\_Web

Source folder: \003\_001\_Simple\_Web\src

Java package: edu.bitzh.simple

Class name: HelloServlet

Superclass: javax.servlet.http.HttpServlet

☐ Use an existing Servlet class or JSP

Class name: HelloServlet

< Back Next > Finish Cancel

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册（3）

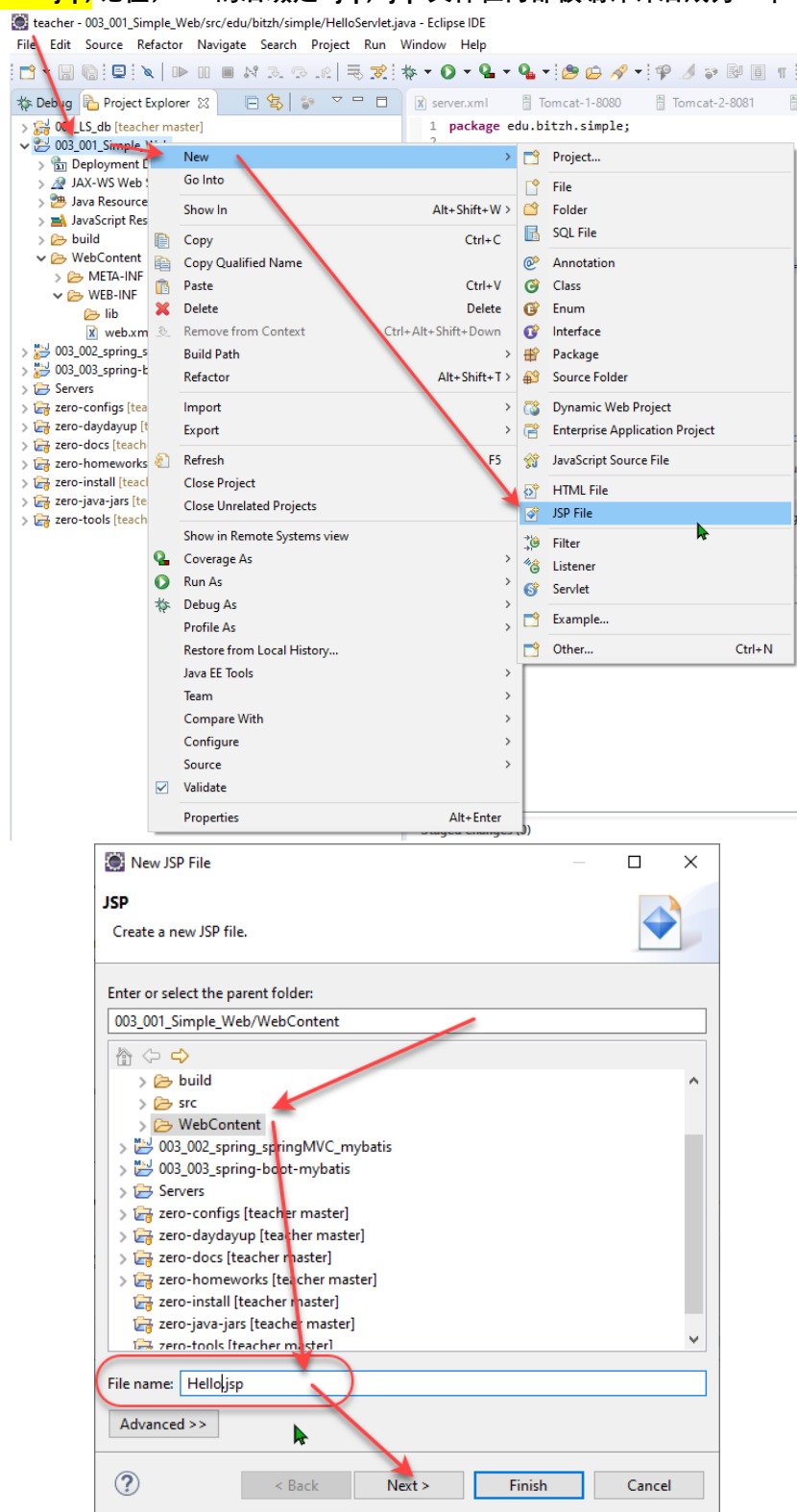
声明：版权属于郭鹏（Peng Guo），允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：3164801047@qq.com 或 guopeng6869@yeah.net



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

步骤 2) 生成 JSP, 名为 **Hello.jsp**, 记住, JSP 的后缀是 .jsp, .jsp 文件在内部被编译后成为一个 Servlet, 但我们一般看不到



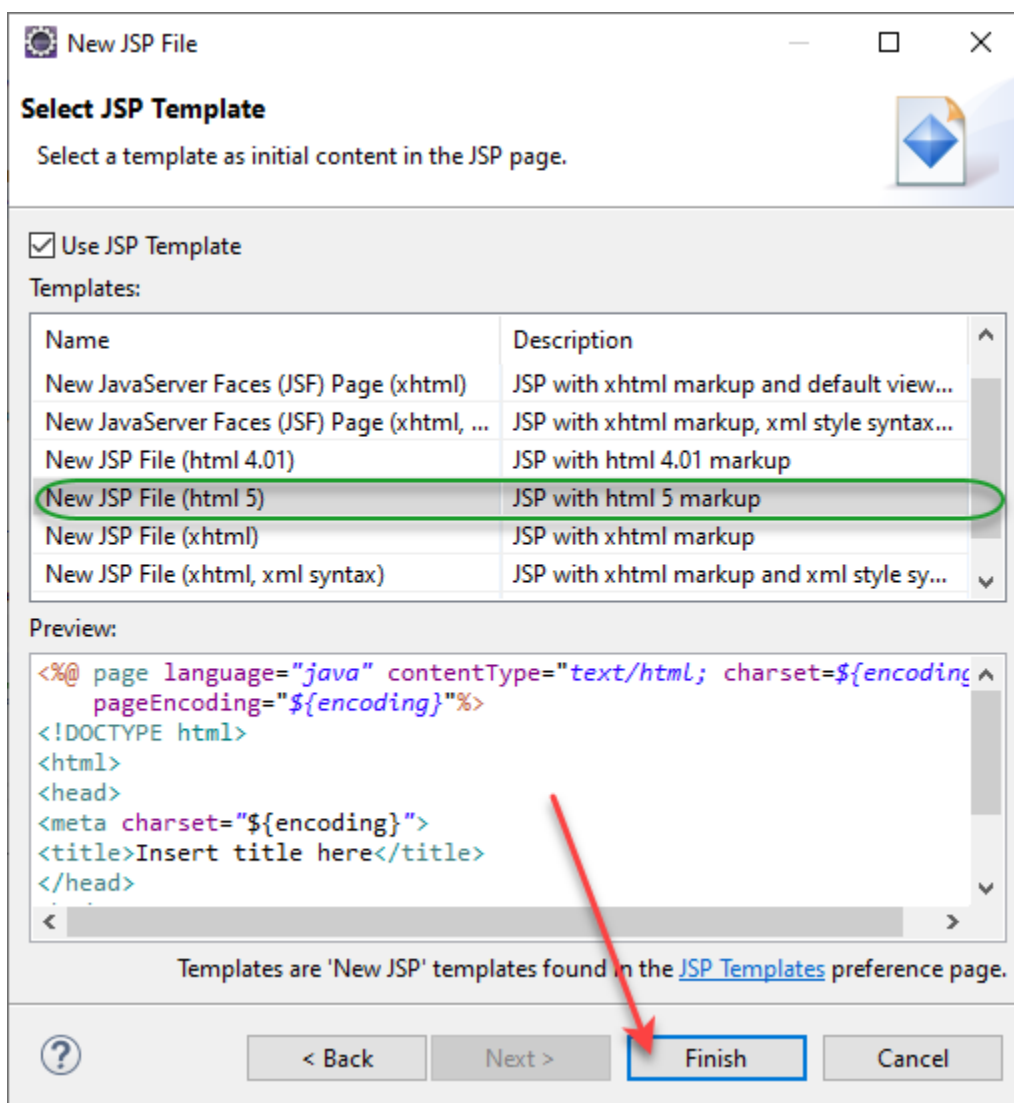
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止,  
使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

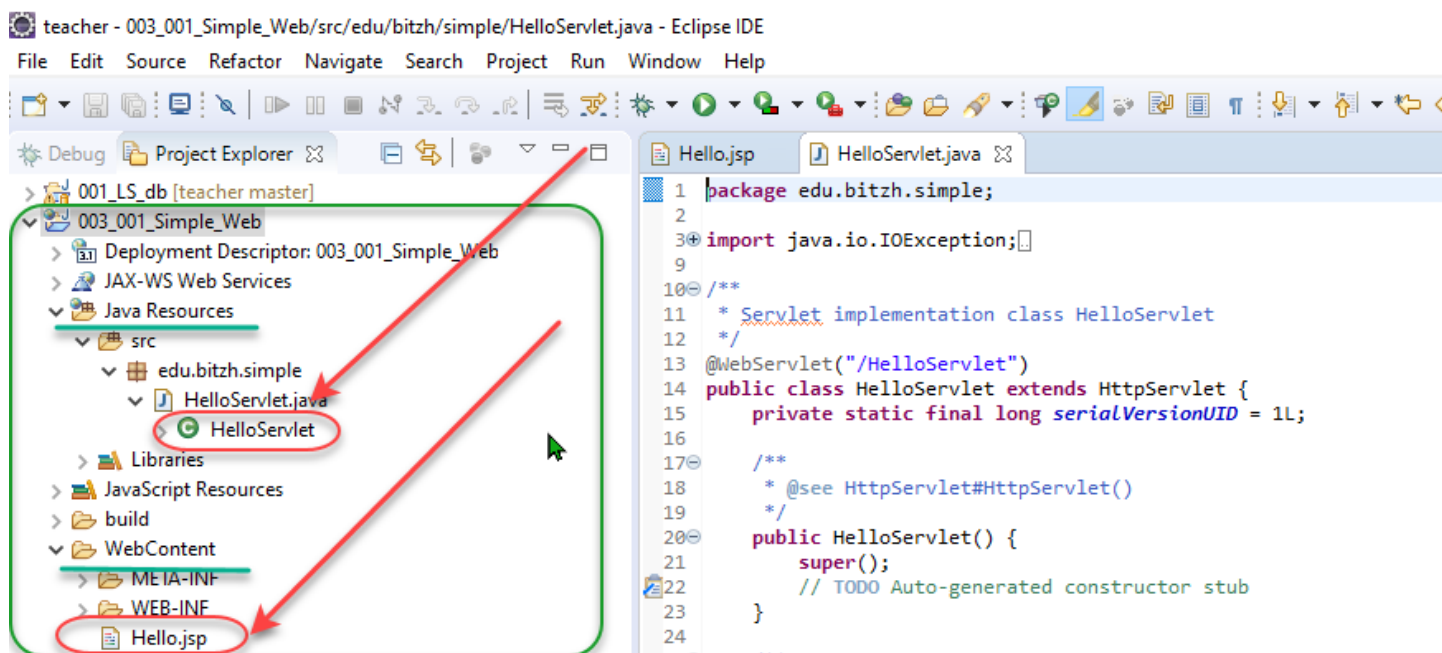


Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



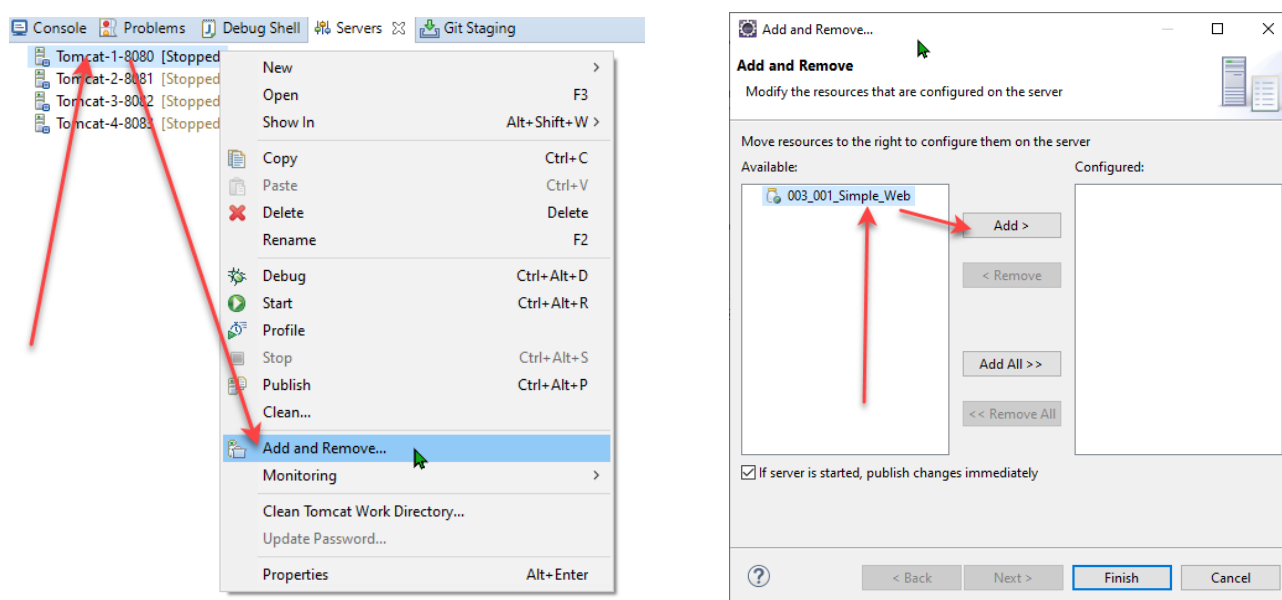
Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

最后，两个文件如下，第一在项目 1003\_001\_Simple\_Web 里，第二在项目的不同目录结构下 (src 和 WebContent) 中，大家理解这种现象为军队中不同的兵种就容易理解了。



打仗去！ 运行一下这两个简单的文件，看效果如何？

步骤 1) 将项目加到 Tomcat 上，犹如派军队上战场！我们试一下 Tomcat-1-8080 吧。

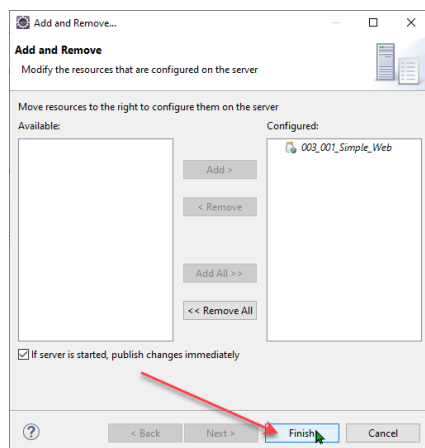


郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

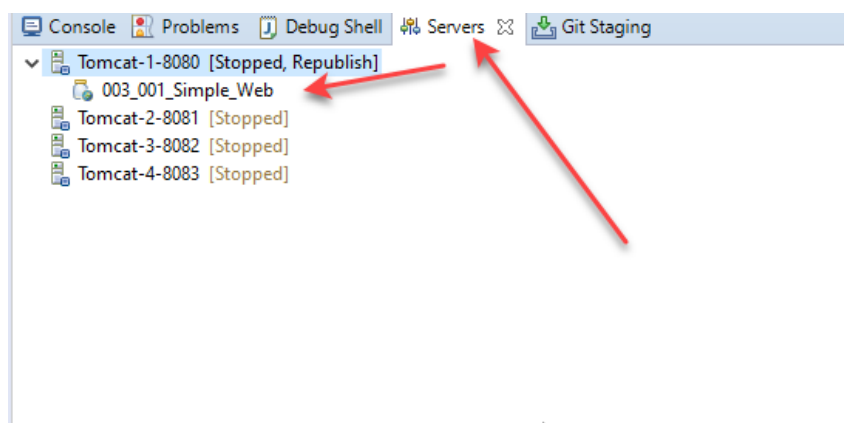
声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！



结果如下：



大家可以看到在 Servers -> Tomcat-1-8080 下有一个 003\_001\_Simple\_Web 的项目，也就是军队（003\_001\_Simple\_Web）已经被派到战场（Tomcat-1-8080）上，可以开始打仗了。开始运行之前，我们将战场设置简单点

将 Tomcat-1-8080 的配置文件 server.xml 中的 path 从

```
path="/003_001_Simple_Web"
```

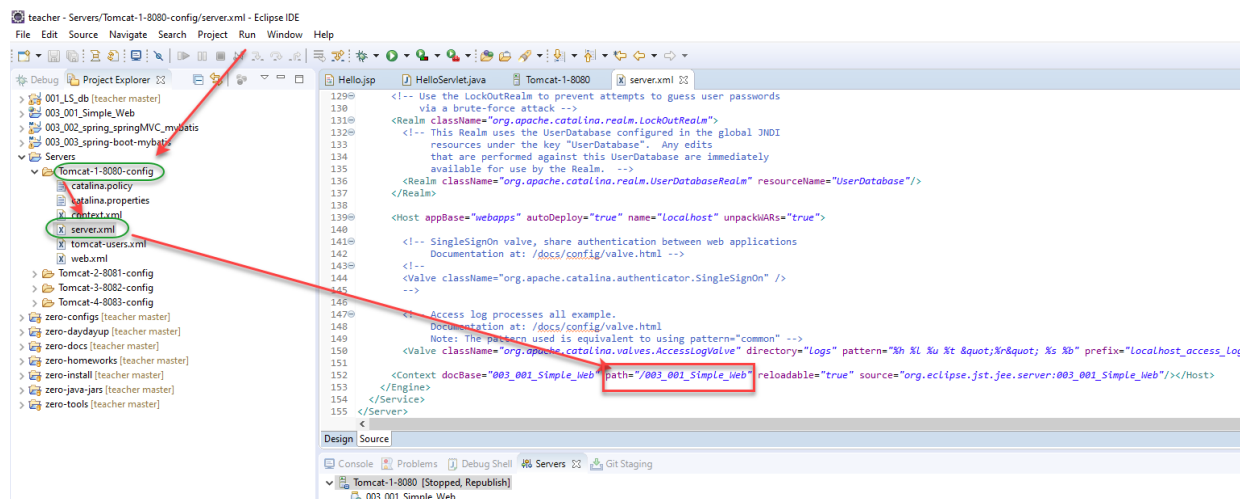
改为

```
path="/"
```

因为这样在浏览器上访问时久不用加路径 003\_001\_Simple\_Web 了，例如访问 Hello.jsp，

从 http://localhost:8080/003\_001\_Simple\_Web/Hello.jsp 变成简单的 http://localhost:8080/Hello.jsp

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!



改后变成, 然后记得存盘。

```

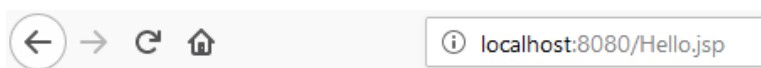
129<!-- Use the LockOutRealm to prevent attempts to guess user passwords
130via a brute-force attack -->
131<Realm className="org.apache.catalina.realm.LockOutRealm">
132<!-- This Realm uses the UserDatabase configured in the global JNDI
133resources under the key "UserDatabase". Any edits
134that are performed against this UserDatabase are immediately
135available for use by the Realm. -->
136<Realm className="org.apache.catalina.realm.UserDatabaseRealm" resourceName="UserDatabase"/>
137</Realm>
138
139<Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true">
140
141<!-- SingleSignOn valve, share authentication between web applications
142Documentation at: /docs/config/valve.html -->
143<!--
144<Valve className="org.apache.catalina.authenticator.SingleSignOn" />
145-->
146
147<!-- Access log processes all example.
148Documentation at: /docs/config/valve.html
149Note: The pattern used is equivalent to using pattern="common" -->
150<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs" pattern="%h %l %u %t &quot;%r&quot;%s %b" prefix="localhost_access_log" suffix=".txt" />
151
152<Context docBase="003_001_Simple_Web" path="/003_001_Simple_Web" reloadable="true" source="org.eclipse.jst.jee.server:003_001_Simple_Web"/></Host>
153</Engine>
154</Service>
155</Server>

```

I

那么开始启动 Tomcat-1-8080 吧

启动后, 开一个浏览器访问 Hello.jsp, 结果如下:



I am the first Hello JSP

正如前面所提到, 因为老师已经将 Hello.jsp 加了点内容:

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止, 使用时不得擅自改动, 不得商用牟利。

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

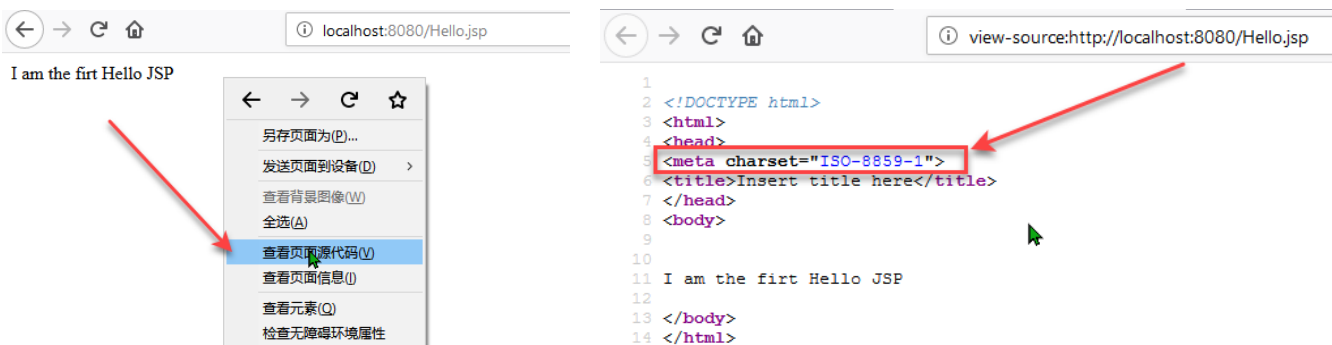
Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

```

1  <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2      pageEncoding="ISO-8859-1"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="ISO-8859-1">
7  <title>Insert title here</title>
8  </head>
9  <body>
10 <%
11     String text="I am the first Hello JSP";
12
13 %>
14
15 <%=text %>
16
17 </body>
18 </html>

```

如果想看浏览器中额的源代码，那么操作和结果如下：



如果想将送中文字符串会浏览器，那么修改 Hello.jsp 中的这两行：

```
<meta charset="UTF-8">
```

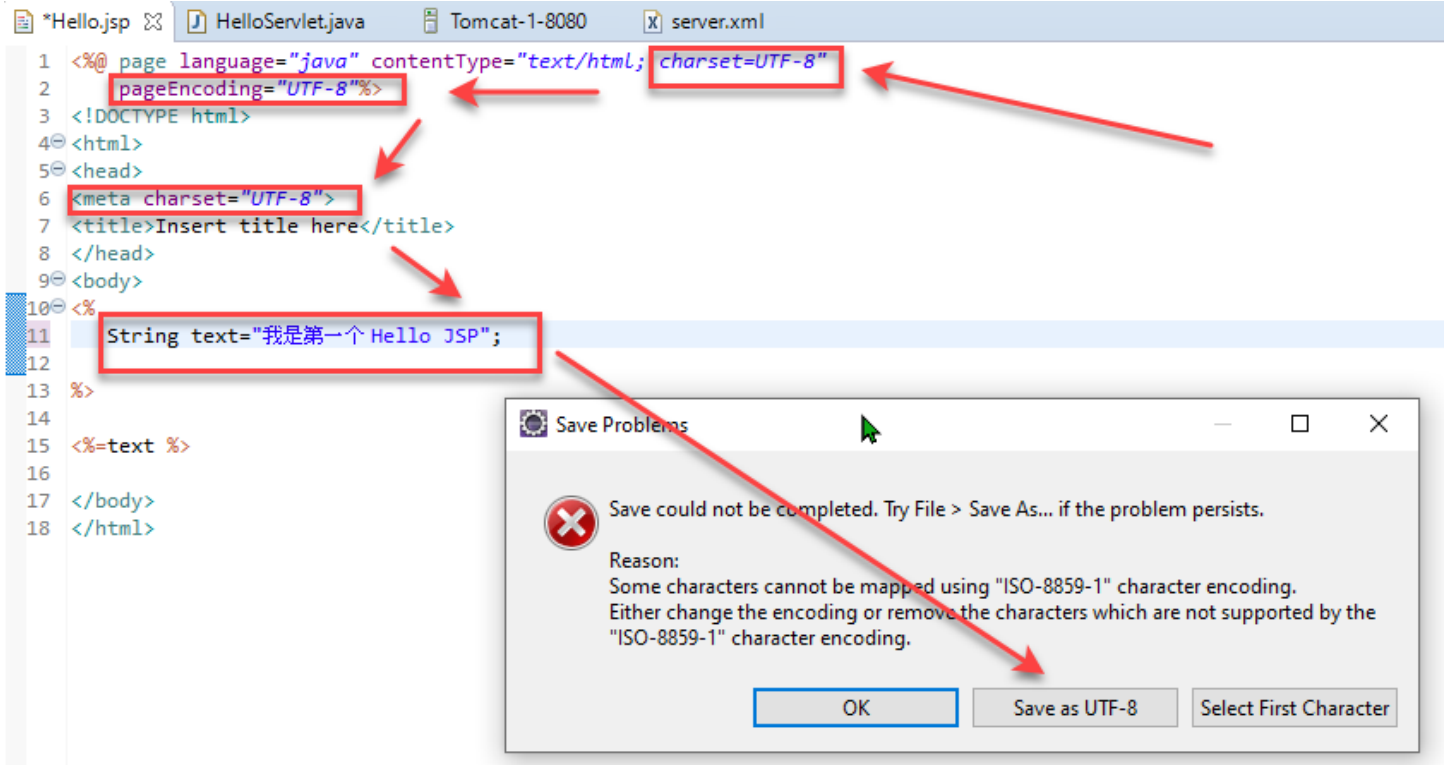
```

<%
String text="我是第一个 Hello JS";
%>

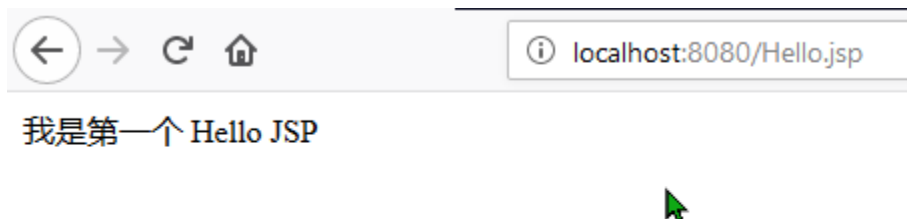
```

以及将 Hello.jsp 存盘时编码改为 UTF-8。

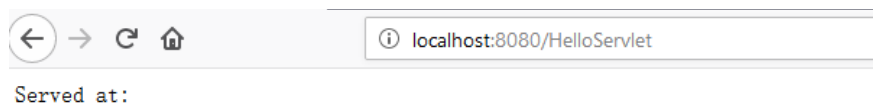
Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！



再到浏览器访问一下：结果是是中文了。



访问一下 HelloServlet 吧， <http://localhost:8080/HelloServlet>



郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

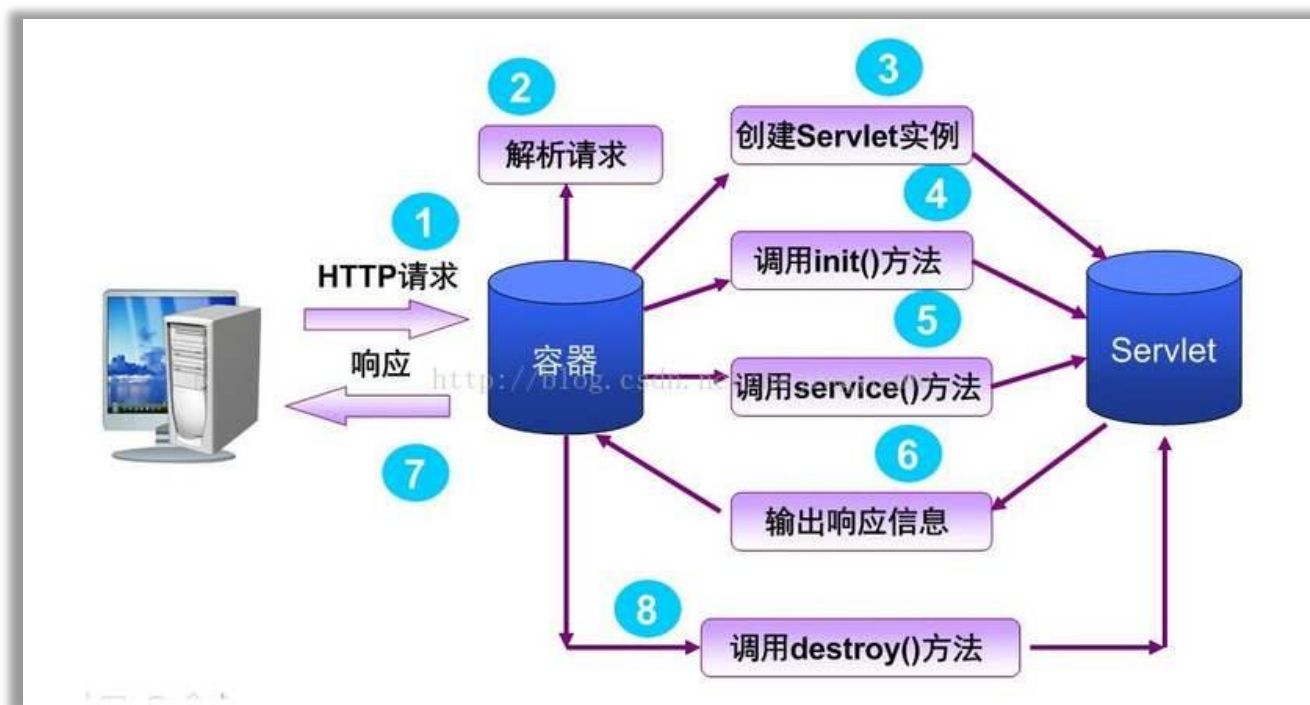
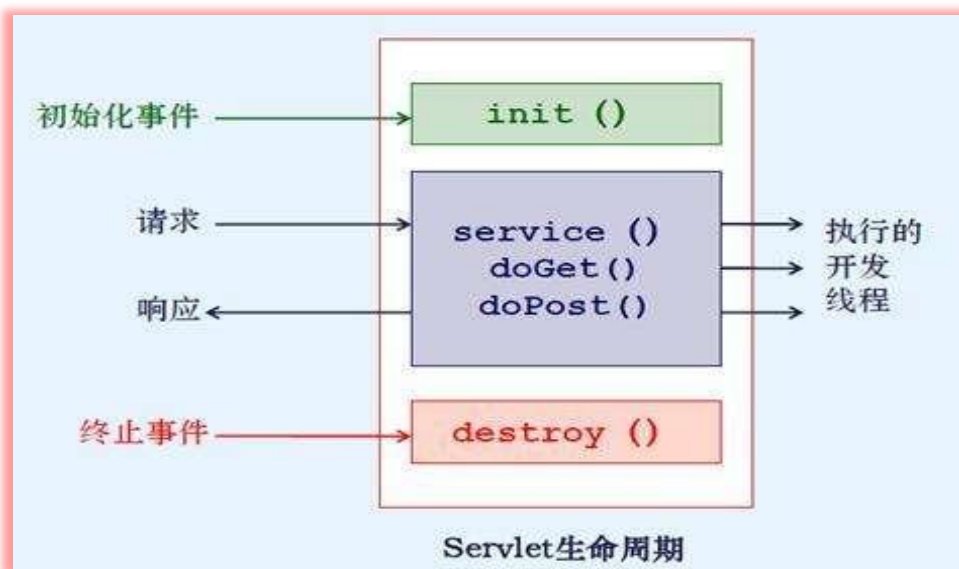
声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

## 2. JSP/Servlet 要素之 2 : 出生 `init ()` ,工作 `Service ()` 和死亡 `destroy ()` .

### Servlet 生命周期



一图胜千言万语, 因为直观!

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明: 版权属于郭鹏 (Peng Guo), 允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止, 使用时不得擅自改动, 不得商用牟利.

联系: [3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)



Servlet 生命周期可被定义为从创建直到毁灭的整个过程。以下是 Servlet 遵循的过程:

- Servlet 通过调用 **init ()** 方法进行初始化。
- Servlet 调用 **service()** 方法来处理客户端的请求。
- Servlet 通过调用 **destroy()** 方法终止 (结束) 。
- 最后, Servlet 是由 JVM 的垃圾回收器进行垃圾回收的。

现在让我们详细讨论生命周期的方法。

### init() 方法

init 方法被设计成只调用一次。它在第一次创建 Servlet 时被调用, 在后续每次用户请求时不再调用。因此, 它是用于一次性初始化, 就像 Applet 的 init 方法一样。

Servlet 创建于用户第一次调用对应于该 Servlet 的 URL 时, 但是您也可以指定 Servlet 在服务器第一次启动时被加载。

当用户调用一个 Servlet 时, 就会创建一个 Servlet 实例, 每一个用户请求都会产生一个新的线程, 适当的时候移交给 doGet 或 doPost 方法。init() 方法简单地创建或加载一些数据, 这些数据将被用于 Servlet 的整个生命周期。

init 方法的定义如下:

```
public void init() throws ServletException {  
    // 初始化代码...  
}
```

### service() 方法

service() 方法是执行实际任务的主要方法。Servlet 容器 (即 Web 服务器) 调用 service() 方法来处理来自客户端 (浏览器) 的请求, 并把格式化的响应写回给客户端。

每次服务器接收到一个 Servlet 请求时, 服务器会产生一个新的线程并调用服务。service() 方法检查 HTTP 请求类型 (GET、POST、PUT、DELETE 等), 并在适当的时候调用 doGet、doPost、doPut、doDelete 等方法。

下面是该方法的特征:

```
public void service(ServletRequest request,
```



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

```
        ServletResponse response)
    throws ServletException, IOException{
}
```

**service()** 方法由容器调用, service 方法在适当的时候调用 doGet、doPost、doPut、doDelete 等方法。所以, 您不用对 service() 方法做任何动作, 您只需要根据来自客户端的请求类型来重写 doGet() 或 doPost() 即可。

doGet() 和 doPost() 方法是每次服务请求中最常用的方法。下面是这两种方法的特征。

### doGet() 方法

GET 请求来自于一个 URL 的正常请求, 或者来自于一个未指定 METHOD 的 HTML 表单, 它由 doGet() 方法处理。

```
public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
    throws ServletException, IOException {
    // Servlet 代码
}
```

### doPost() 方法

POST 请求来自于一个特别指定了 METHOD 为 POST 的 HTML 表单, 它由 doPost() 方法处理。

```
public void doPost(HttpServletRequest request,
                   HttpServletResponse response)
    throws ServletException, IOException {
    // Servlet 代码
}
```

### destroy() 方法

destroy() 方法只会被调用一次, 在 Servlet 生命周期结束时被调用。destroy() 方法可以让您的 Servlet 关闭数据库连接、停止后台线程、把 Cookie 列表或点击计数器写入到磁盘, 并执行其他类似的清理活动。

在调用 destroy() 方法之后, servlet 对象被标记为垃圾回收。destroy 方法定义如下所示:

```
public void destroy() {
    // 终止化代码...
}
```

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

### 3. JSP/Servlet 要素之 3：最重要的两大系统级 Object：请求（request）和 响应（response）

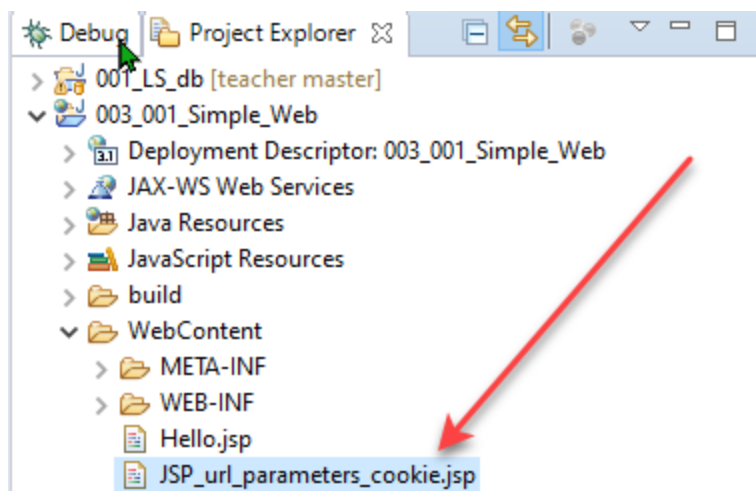
简单点解释，**request**（中文叫**请求**）是系统（简单地指 Tomcat）处理来自用户的请求后打包了几乎所有的有用的信息（**cookie**，**传递的参数**等等），然后用保留字 `request` 传给了服务器端的 JSP 和 Servlet，一旦用户的请求处理完毕返回给用户（手机或浏览器），这个 `request` 就结束寿命。

**response**（中文叫**响应**）是系统准备给 JSP 和 Servlet 用来处理如何返回信息（**html**，**Cookie** 或其他结果）给用户（手机或浏览器），简单地说，就是 JSP 和 Servlet 用 `Response` 来返回处理结果给用户，寿命和 `request` 一样。

注意，**request** 和 **response** 是线程级别的变量，也就是访问同个 JSP/Servlet 的不同请求（线程）有不同的各自一对不同的 `request` 和 `response`。

下面，我们用 `JSP_url_parameters_cookie.jsp` 的例子来说明这个两个非常重要的 Object 是如何被使用的，1) **处理参数和 cookie**。

大家按照前面 `Hello.jsp` 在 `003_001_Simple_Web` 这个动态网页项目中创建如下：



JSP 的内容如下：

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

```

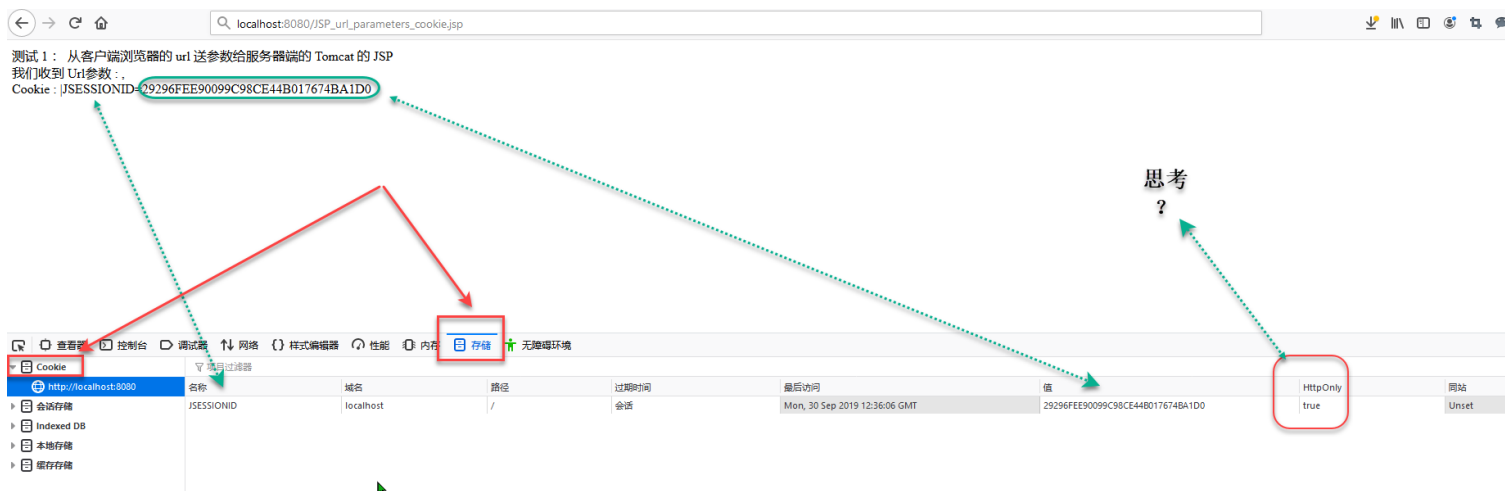
Hello.jsp  HelloServlet.java  Tomcat-1-8080  server.xml  JSP_url_parameters_cookie.jsp
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <%@page import="java.util.*" %>
4  <!DOCTYPE html>
5  <html>
6  <head>
7      <meta charset="UTF-8">
8      <title>Insert title here</title>
9  </head>
10 <body>
11 <body>
12 测试 1: 从客户端浏览器的 url 送参数给服务器端的 Tomcat 的 JSP <br/>
13 <%
14     /*****
15      * 下面是纯粹的 Java 代码, 读出url里面的参数
16      *****/
17     String urlStr="";
18     Enumeration<String> all=request.getParameterNames();
19     String key=null;
20     while(all.hasMoreElements()){
21         key=all.nextElement();
22         urlStr=urlStr+"|"+key+"="+request.getParameter(key);
23     }
24     /*****
25      * 下面是纯粹的 Java 代码, 读出cookie,
26      * 如果一个cookie都没有, 我们就自己生成一个
27      *, 然后 加个计数器, 返回去浏览器(response), 下次调用到这里
28      * 就有了。
29      *****/
30     String cookieStr="";
31     Cookie[] cookies=request.getCookies();
32     if (cookies==null||cookies.length==0){
33         cookieStr="目前没有Cookie";
34         Cookie cookie=new Cookie("cookie_key", "1");
35         response.addCookie(cookie);
36     }else{
37         for(Cookie cookie: cookies){
38             cookieStr=cookieStr+"|"+cookie.getName()+"="+cookie.getValue();
39         }
40     }
41
42 %>
43 我们收到 Url参数: <%=urlStr%>, <br/> Cookie : <%=cookieStr %>
44 </body>
45 </html>

```

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

我们试一下不带参数:

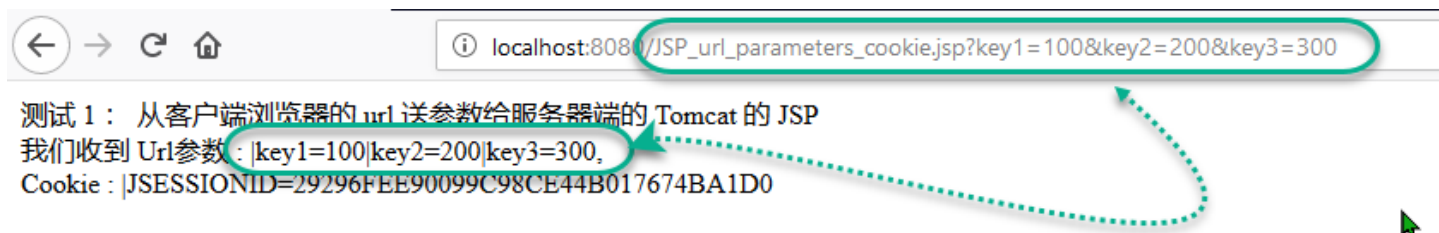
http://localhost:8080/JSP url parameters cookie.jsp



下面我们开始增加参数在 url 上，方法是 ? 后用 & 隔开，这在技术上叫 **get 方法**，因为也可以在浏览器的和浏览里用一个 form 来提交（**post 方法**）：

http://localhost:8080/JSP url parameters cookie.jsp?key1=100&key2=200&key3=300

get 的结果如下：浏览器和远方的 Hello JSP 可以这样交换数据。



## 实验开始

：大家打开自己的电脑，在自己的 Eclipse 上，将上述 JSP\_url\_parameters\_cookie.jsp 中的功能在 **HelloServlet.Java** 中实现起来，运行和调试成功。时间为一堂课里 30 分钟编程，后面 15 分钟总结和答疑。

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

#### 4. JSP/Servlet 要素之 4: 页面跳转（登录，出错，业务分流），页面包含（模块化），上传。

大家要了解互联网的主要目的，是提供给用户使用的一系列功能和服务，所以大量内容由单个页面提供，或整个网站的所有功能由一个网页提供都是不合理的，所以网站就会被设计成多个网页，而且网页之间是有关联的，有逻辑的，这就是网页跳转（Jump）和包含（include）的道理。Jump 包含内部转跳（forward）和外部重定向（redirect），include 则是大页包含小页，一般用于将一个大的页面切割成几个逻辑的小部分，各自有不同的

JSP 实现业务。大家在自己的 Eclipse 实现以下代码，手敲多体验，一分动手一份收获！

JSP\_url\_jump\_include.jsp

```

Hello.jsp  JSP_url_jump_include.jsp  HelloServlet.java  Tomcat-1-8080  server.xml  JSP_url_parameters_cookie.jsp

1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2    pageEncoding="UTF-8"%>
3  <%@page import="java.util.*"%>
4  <!DOCTYPE html>
5  <html>
6  <head>
7  <meta charset="UTF-8">
8  <title>JSP_url_jump_include.jsp</title>
9  </head>
10 <body>
11 <H1>如果没有带jump=1 或 redirect=1 的参数，本页就包括了<Strong>Hello.jsp</Strong>的内容了。</H1>
12 <%
13  /*****
14   * 内部转跳 (forward) JSP_url_parameters_cookie.jsp, 如果 url 参数有 jump=1
15   *****/
16   String isJump=request.getParameter("forward");
17   if (isJump!=null&& isJump.equals("1")){
18       request.getRequestDispatcher("/JSP_url_parameters_cookie.jsp").forward(request, response);
19   }
20  /*****
21   * 外部重定向 (redirect) 到 JSP_url_parameters_cookie.jsp, 如果 url 参数有 redirect=1
22   * redirect 和 forward 的最大不同点，redirect 将新的url传回浏览器，浏览器据此再发
23   * 一次http请求到 tomcat 服务器，而 forward只是在内部转发，所以大家思考一下性能上的差别？
24   * 当然redirect 也可以发一个第三方外部的url, 这是redirect最重要的应用！
25   *****/
26   String isRedirect=request.getParameter("redirect");
27   if (isRedirect!=null&& isRedirect.equals("1")){
28       response.sendRedirect("http://www.bitzh.edu.cn");
29   }
30  /*****
31   * 包含网页 (include):
32   *
33   *****/
34  %>
35 <jsp:include page="/Hello.jsp" flush="true"/>
36
37 </body>
38 </html>

```

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，  
使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

- 然后用 1) [http://localhost:8080/JSP\\_url\\_jump\\_include.jsp](http://localhost:8080/JSP_url_jump_include.jsp)
- 2) [http://localhost:8080/JSP\\_url\\_jump\\_include.jsp?forward=1](http://localhost:8080/JSP_url_jump_include.jsp?forward=1)
- 3) [http://localhost:8080/JSP\\_url\\_jump\\_include.jsp?redirect=1](http://localhost:8080/JSP_url_jump_include.jsp?redirect=1)



测试和检验结果。

**下面学习文件上传**, 上传文件是个主要功能, 例如因为很多互联网应用允许和需要用户上传图片等。这个可以简单地用 html 的 form 来实现, 当然很酷, 很炫的上传一定会用到 JavaScript 及其框架来一起配合完成, 大家可以[百度“jquery 的上传插件”](#)而去进一步学习。

由于 JSP 和 Servlet 的代码越来越复杂和行数增加, 所以大家最好尽快学会

**“Tomcat 上的 JSP/Servlet 的断点调试”** 正如中国古人所说“工欲善其事, 必先利其器”:

```

10  /*****
11  * 定义上传文件的最大字节
12  *****/
13  int MAX_SIZE = 102400 * 102400;
14  /*****
15  * 创建根路径的保存变量
16  *****/
17  String rootPath;
18
19  /*****
20  * 声明文件读入类
21  *****/
22
23  DataInputStream in = null;
24  FileOutputStream fileOut = null;
25  /*****
26  * 取得互联网程序的绝对地址
27  *****/
28
29  String realPath = request.getSession().getServletContext().getRealPath("/");
30  realPath = realPath.substring(0, realPath.indexOf("\\out"));
31
32
33  /*****
34  * 创建文件的保存目录

```

如何在Eclipse 中断点调试 JSP/Servlet?


- 1) 在JSP 或 Servlet的 Java 代码中设断点。
- 2) 在Tomcat的服务器启动时选择 **Restart in Debug** (调试启动)。



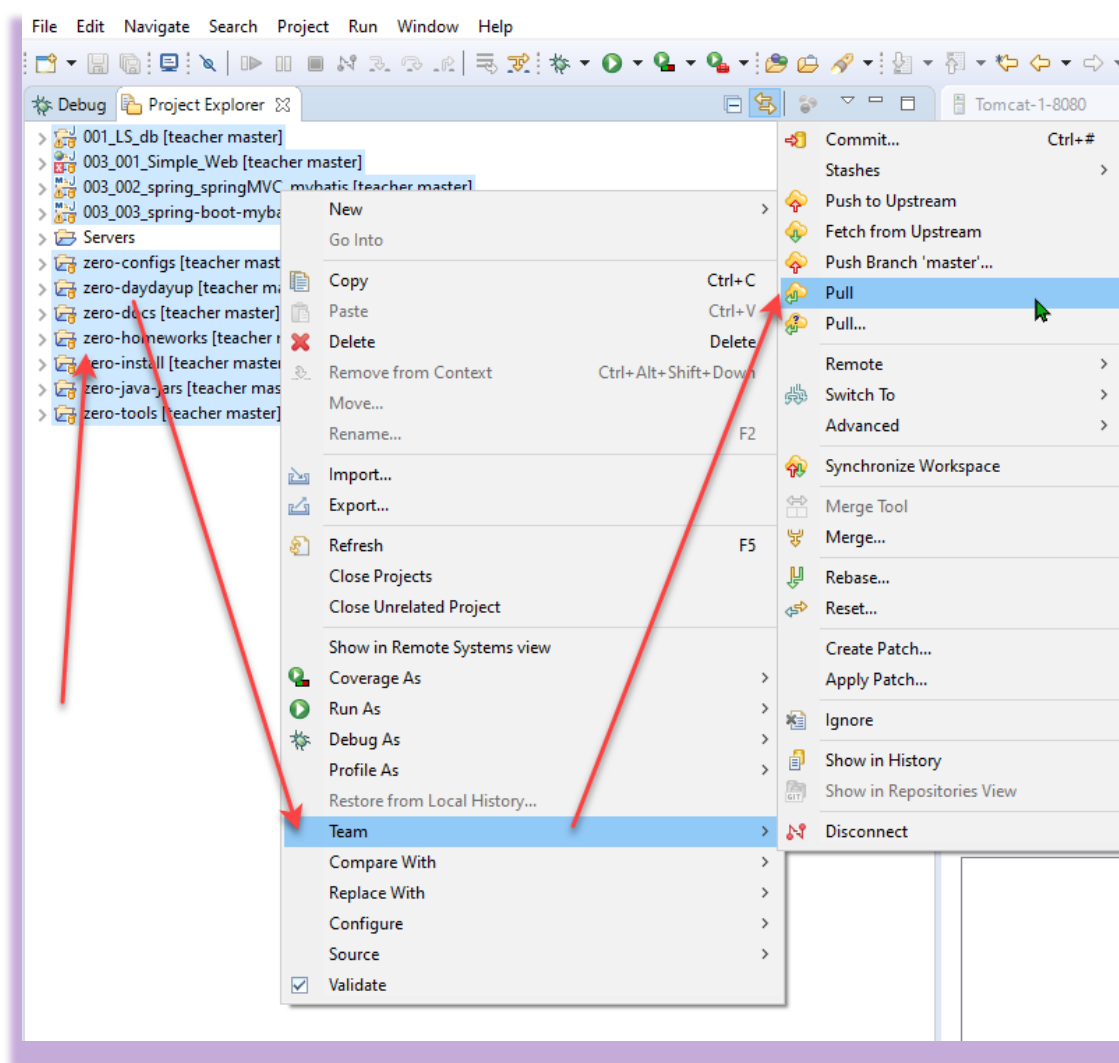
Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

如何获得老师 Gitee 上的最新的手册或代码

在你自己的 Eclipse 上：

步骤 1) 选上单个或全部项目（注意必须是有类似  标志的，需要有那个小仓库，为什么？因为这个小仓库标志表明这个项目是有 git 的）

步骤 2) 然后一步步按下图操作：



郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册（3）

声明：版权属于郭鹏（Peng Guo），允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

## 作业 2： 综合练习 （JSP/Servlet+Ajax+JQuery+Session+JDBC+IO）

需求：

1) 描述：在自己建立的 Homework02Proj 的 Dynamic Web Project 上从 send.jsp，有一个 html div (Id=data) 和一个 button (“读远方的文件”) 每按一次就用 Ajax+JQuery 发送一个指令给 daoServlet，要求从 Data.txt 文件里读取一行，然后加进 Session 里，必须允许至少发 5 条记录，每次执行完毕将 Session 中的记录送回 send.jsp，并显示在 htm div (ID=data) 中。

另为一个 list.jsp 则有一个 button (“写数据库并清零 Session”)，按完后，可以发指令到 daoServlet，让 daoServlet 从 Session 中获取前面存入的记录，并将这一批记录全部一条一条地用 JDBC 写入 single Mysql 数据库的表 Data。

list.jsp 有另一个 button (“访问数据库”)，可以访问 daoServlet，再访问数据库 single mysql 的表 Data 而获取当前最新的所有记录并显示在 list.jsp 中。

2) 数据格式：

Data 表的格式：

ID	Name	数目
K01	手机	100
K02	手表	200
K03	电视	300

Data.txt 的格式



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

K04 硬盘 400

K05 屏幕 500

## 要求提交的格式：

将整个过程的 1) 每一步的断点调试 debug 的屏幕拷贝，2) 以及各个 JSP 的显示结果的屏幕拷贝，3) Eclipse 中的 Homework02Proj 的屏幕拷贝，3) 所有 JSP 和 Servlet 的代码，压缩打包成提交到网络教学平台。

Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

## 9) JSP/Servlet 要素之 5: Cookie, Session 状态信息在哪? 存在的意义, Http 连接的无状态性。

从客户端发送 Http 协议到服务器端 (B/S) 模式, 有一个最大的特点就是无状态, 也就是面对成千上万个来自客户端的请求, 服务器端是被动地接受而且不能访问客户端, 在 cookie 和 session 被引入之前, 服务器端不能识别请求是否是同一客户。

cookie 是一串字符串, 表示一对键值, key 和 value, 最大特点是存放在客户端, 例如浏览器, 每次客户端访问服务器端时携带这些 cookie, 从而可以一定程度上与服务器端保持一定的状态联系, 尤其当这些 cookie 是服务器设置的和标记的, 那么服务器就可以凭此来记住客户端, 但是记住, 谁能修改 cookie? 不但服务器端可以, 连客户端可以轻易篡改 cookie 的内容 (Java Script, 浏览器内嵌的工具, 我们接着需要学习这种手段, 因为调试时可以使用), 所以有的应用就将 cookie 进行一定的加密, 提高篡改的难度。

Session 是什么? session 是特殊的 cookie, 为什么特殊?

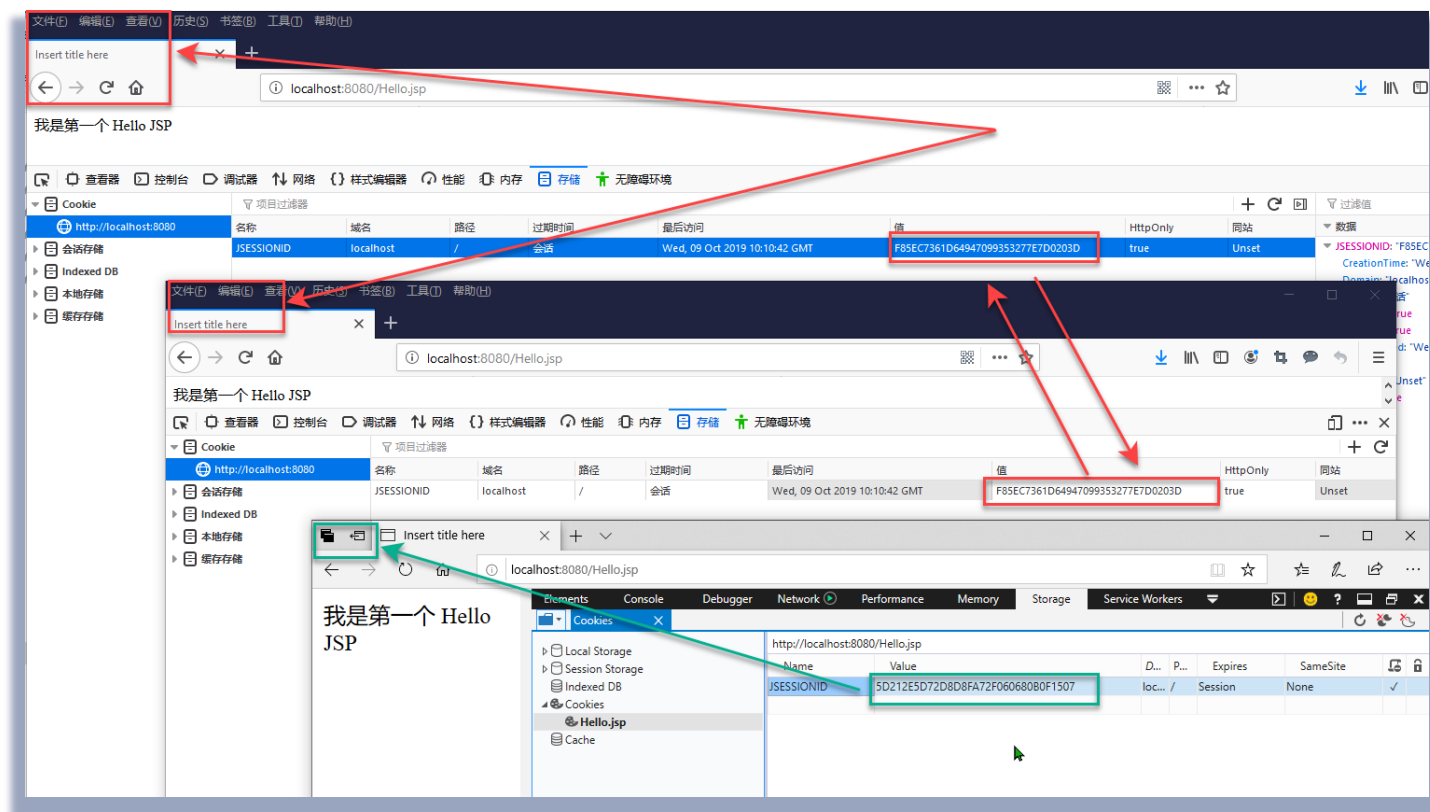
- 1) session 的 ID 是 JSESSIONID, 由 Tomcat 容器 (服务器端) 产生的, 而且内容是唯一的。
- 2) session 产生后, 存放在 Tomcat 的内存里。
- 3) 同一种浏览器的不同 Tab (标签) 或不同窗口访问时携带着同一个 session, 这是浏览器厂家遵从的工业标准。
- 4) session 在关掉浏览器后, 浏览器清除这个特殊的 cookie, 以便下次打开访问同一个网站时, 可以产生新的 session。

以下是一个 session 在浏览器中的样子 :

名称是 JSESSIONID, 值是 F85EC7361D64947099353277E7D0203D



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！



以上是红箭头所指向是 Firefox 的浏览器，两个 Session 是相同的，而绿色的则是 IE 浏览器，明显两个 session 是不同值得。

F85EC7361D64947099353277E7D0203D

5D212E5D72D8D8FA72F060680B0F1507

所以我们可以用测试时，采用不同的浏览器而模仿不同的用户，因为 session 的不同。

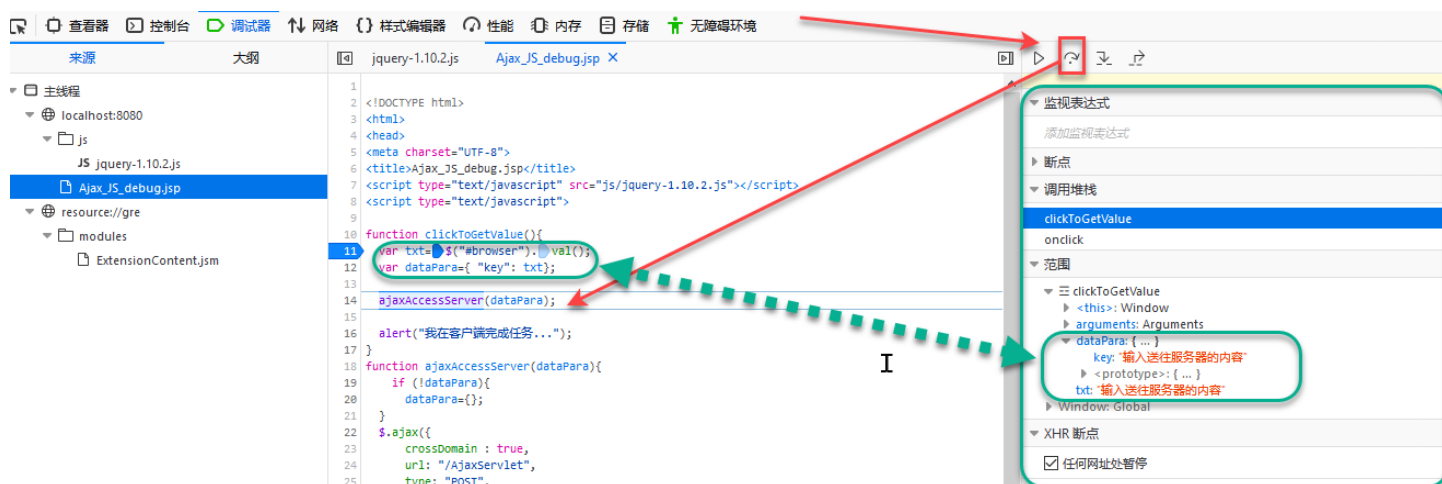
一步步跟踪数据的变化，从而早出错误！

http://localhost:8080/Ajax\_JS\_debug.jsp



Why to learn? 集群已经是高并发, 大网站, 云计算以及高性能服务的标准配置和基本架构!

步骤 2) 激发后, 右面的窗口是每一步 Java Script 所带来的当前数据变化。



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

## Ajax 和 Servlet 的互动调试：

步骤 1) 请在以下的客户端设置断点：



```

9
10 function clickToGetValue(){
11     var txt=$("#browser").val();
12     var dataPara={ "key": txt};
13
14     ajaxAccessServer(dataPara);
15
16     alert("我在客户端完成任务...");
17 }
18 function ajaxAccessServer(dataPara){
19     if (!dataPara){
20         dataPara={};
21     }
22     $.ajax({
23         crossDomain : true,
24         url: "/AjaxServlet",
25         type: "POST",
26         data: {"AjaxData":JSON.stringify(dataPara)} ,
27         dataType: "json",
28         success: function (data) {
29             listenToServer(data);
30         },
31         complete : function (){
32             alert("我(Ajax)完成了向服务器发送数据 :"+dataPara);
33         },
34         error: function (xhr, ajaxOptions, thrownError) {
35             alert("Encounter Error in ajaxAccessServer(), Detail is : "+thrownError);
36         }
37     });
38 }
39 function listenToServer(data){
40     $("#server").html(data);
41 }
42 </script>

```

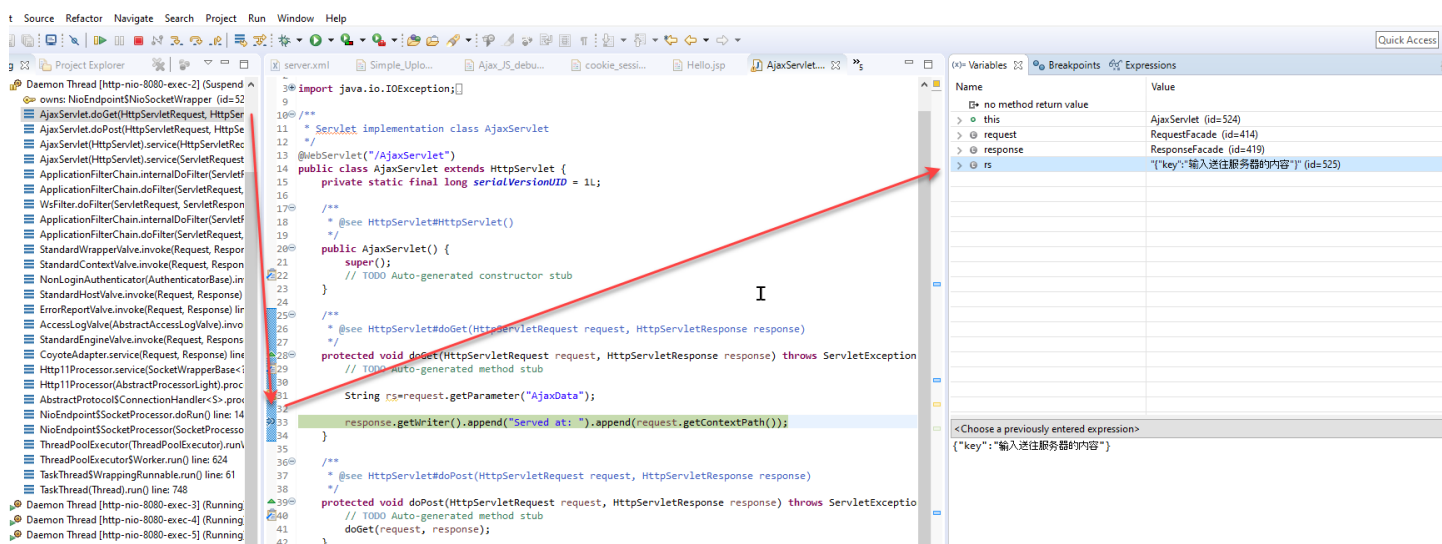
郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，  
使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

## 步骤 2) 请在以下的服务器端 AjaxServlet 设置断点：



最后可以执行的代码如下：

```

25  /**
26   * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
27   */
28  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException
29  // TODO Auto-generated method stub
30
31  String rs=request.getParameter("AjaxData");
32
33  rs=rs.replace("key", "ServerKey");
34  response.getWriter().write(rs);
35  }
36
37  /**

```

郭鹏著《服务器集群技术》JSP/Servlet 速成教学手册 (3)

声明：版权属于郭鹏 (Peng Guo)，允许北理珠学生自由抄传直到被版权拥有人郭鹏通知中止，使用时不得擅自改动，不得商用牟利。

联系：[3164801047@qq.com](mailto:3164801047@qq.com) 或 [guopeng6869@yeah.net](mailto:guopeng6869@yeah.net)

Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

未完待续



Why to learn? 集群已经是高并发，大网站，云计算以及高性能服务的标准配置和基本架构！

大家思考一下这么的数据库设计有什么好处？

《本手册结束》

**后话**：将来工作后（估计 1-5 年内，有人快，有人慢，看各自的悟性，专业上的，有识人上的），如果觉得老师这种教法和这种知识技能确实有较大的帮助，可以将反馈意见送到老师的个人信箱或你的大学母校。

最后，谢谢同学们，我们师生一起努力，将计算机软件水平提高上去吧！

时间是最好的考试！

时间会考出理念优劣，时间会考出水平高低，时间会考出悟性好与差！