# 尚马教育 JAVA WEB课程

# Servlet入门

文档编号：B11

创建日期： 2017-06-13

最后修改日期：2019-09-23

版 本 号：V2.0

电子版文件名：尚马教育-第二阶段-1.servlet入门.docx

**文档修改记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新作者 | 更新说明 | 版本号 |
| 2018-08-01 | 王绍成 | 初始版本 | V1.0 |
| 2019-09-13 | 郭雪岩 | 增加http协议分析 | V2.0 |

**主讲人：**

**郭雪岩**

目录

[尚马教育 JAVA WEB课程 1](#_Toc14193)

[Servlet入门 1](#_Toc25425)

[1. Web服务器介绍 3](#_Toc20537)

[1.1. 什么是web服务器 3](#_Toc469)

[1.2. 常见名词解释 4](#_Toc26168)

[1.3. 开发web应用的语言 5](#_Toc26329)

[2. Tomcat下载和使用 6](#_Toc20298)

[2.1. Tomcat介绍 6](#_Toc15338)

[2.2. Tomcat目录结构 6](#_Toc5475)

[2.3. Tomcat在idea中配置 7](#_Toc14175)

[2.4. Tomcat在eclipse中配置 13](#_Toc24804)

[2.5. 其他web服务器 15](#_Toc9684)

[3. Servlet 16](#_Toc698)

[3.1. Servlet介绍 16](#_Toc22617)

[3.2. 编写servlet 16](#_Toc1080)

[3.3. Servlet实现方式 17](#_Toc25938)

[3.4. Servlet生命周期 18](#_Toc4882)

[3.5. Servlet其他配置参数 18](#_Toc22808)

[3.6. Servlet中处理乱码 19](#_Toc4581)

## Web服务器介绍

### 什么是web服务器

* 网络应用一般分为C/S和B/S两种

C/S结构是Client/Server（客户机/服务器）的简称，桌面应用程序采用的多是这种结构；

例：QQ 微信

B/S结构是Browser/Server（浏览器/服务器）的简称，特点是客户端无需安装特定的软件，只需要安装一个浏览器就可以与系统进行交互。

例：访问百度 访问搜狐

* Web服务器是基于B/S结构，通过http协议进行数据交互（收发）

|  |
| --- |
| 浏览器---------http协议---------服务器  http协议规定了发送数据的格式（报文格式）  浏览器中发送和读取的格式已固定  Web服务器需要能够接收和解析浏览器发来的符合http协议的请求报文，并调用对应的代码处理，再向浏览器发送符合http协议规范响应报文。  请求报文格式：  IMG_256  响应报文格式：  IMG_256 |

报文实例

|  |
| --- |
| 请求：  GET /mytest HTTP/1.1  Host: www.cnblogs.com  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; zh-CN; rv:1.8.1) Gecko/20061010 Firefox/2.0  .......  响应：  HTTP/1.1 200 OK  Content-Type:text/html;charset=utf-8    <html>  <body>  Hello http!  </body>  </html> |

### 常见名词解释

* Web应用、HTTP服务器、Web应用服务器
* Web应用被称为Web Application，通过浏览器访问的应用程序，从而为用户提供相关的服务，例如查询、购物等；淘宝，途牛旅游等提供服务的网站；也包含很多公司内部使用的软件，如oa系统，医院内使用的系统，企业内部业务管理系统等；
* HTTP服务器：也称为Web服务器，主要功能是提供网上信息浏览服务，例如Apache、Nginx、IIS是比较常用的HTTP服务器；使用浏览器访问Web站点或者Web应用，则必须通过HTTP服务器；
* Web应用服务器：Web应用服务器能够运行服务器上的应用程序，并将结果返回给客户端浏览器；例如，Tomcat就是一种Web应用服务器；通常情况下，Web应用服务器兼具HTTP服务器的部分功能；

注意：Web应用需要HTTP服务器及Web应用服务器，因为不仅需要浏览信息，还需要运行应用程序；但是tomcat已兼具了http服务器的部分功能，所以运行web应用可以直接使用tomcat。

### 开发web应用的语言

Java 、.net、php等语言均可开发web应用；

java开发web应用最广泛；

官方在JavaEE中提供了Servlet、JSP组件来开发Web应用；

## Tomcat下载和使用

### Tomcat介绍

Tomcat开源、免费，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试Servlet/JSP程序的首选；

下载地址：http://tomcat.apache.org/；

安装：下载到的是一个压缩包，解压缩即可（路径中不要有中文）；

### Tomcat目录结构

|  |  |
| --- | --- |
| **/bin** | **存放Windows或Linux平台上用于启动和停止Tomcat的脚本文件** |
| **/conf** | **存放Tomcat服务器的各种配置文件，其中最重要的是server.xml** |
| **/lib** | **存放Tomcat服务器所需的各种JAR文件** |
| **/logs** | **存放Tomcat运行中产生的日志文件** |
| **/temp** | **存放Tomcat服务器在运行中产生的临时缓存** |
| **/webapps** | **存放Tomcat发布的Web应用** |
| **/work** | **Tomcat把由JSP生成的Servlet放于此目录下** |

* 启动和停止Tomcat服务器

可通过apache解压目录中的startup.bat/shutdown.bat来启动或停止tomcat

在浏览器地址栏中输入：http://127.0.0.1:端口号或者http://localhost:端口号，页面进入到Tomcat启动成功界面

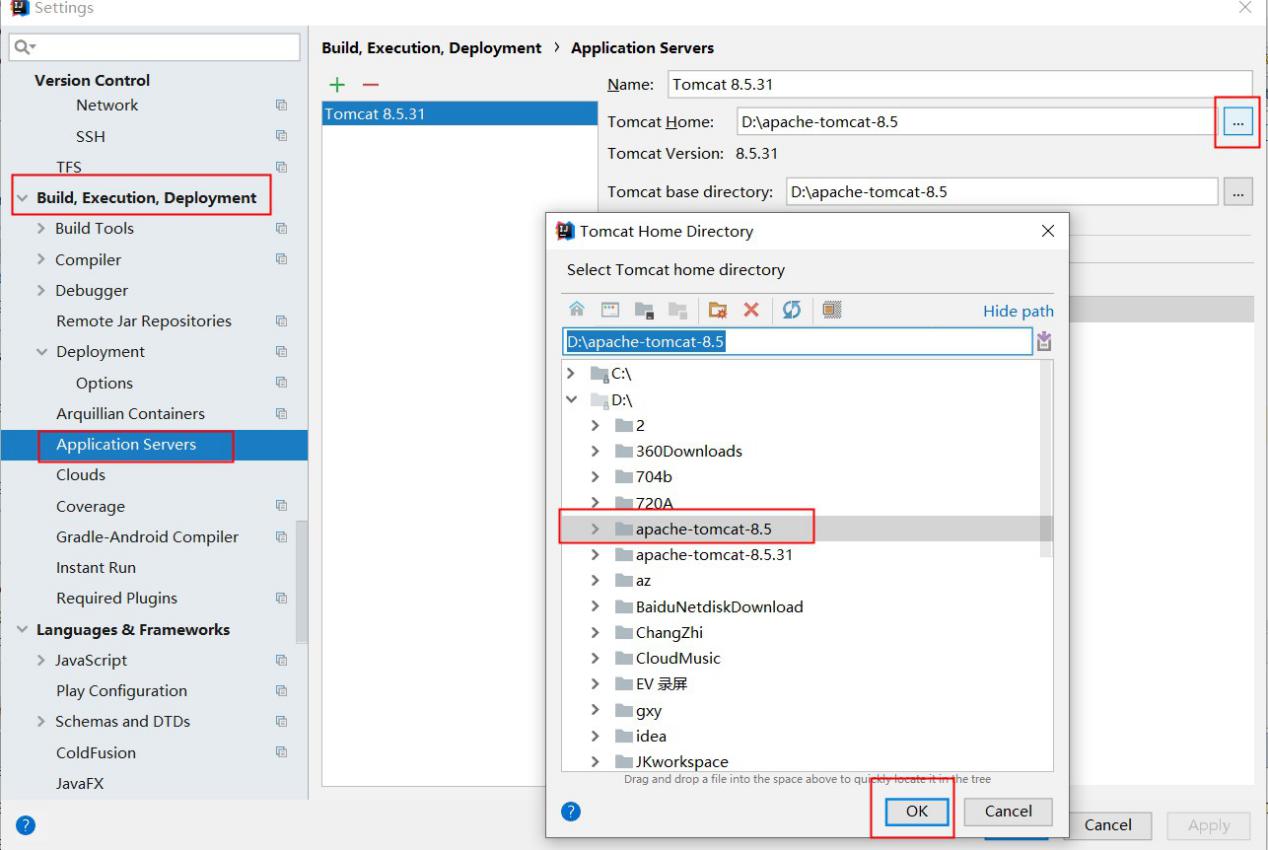
Tomcat默认端口号为-8080

* 可以在conf中的server.xml中修改tomcat端口号

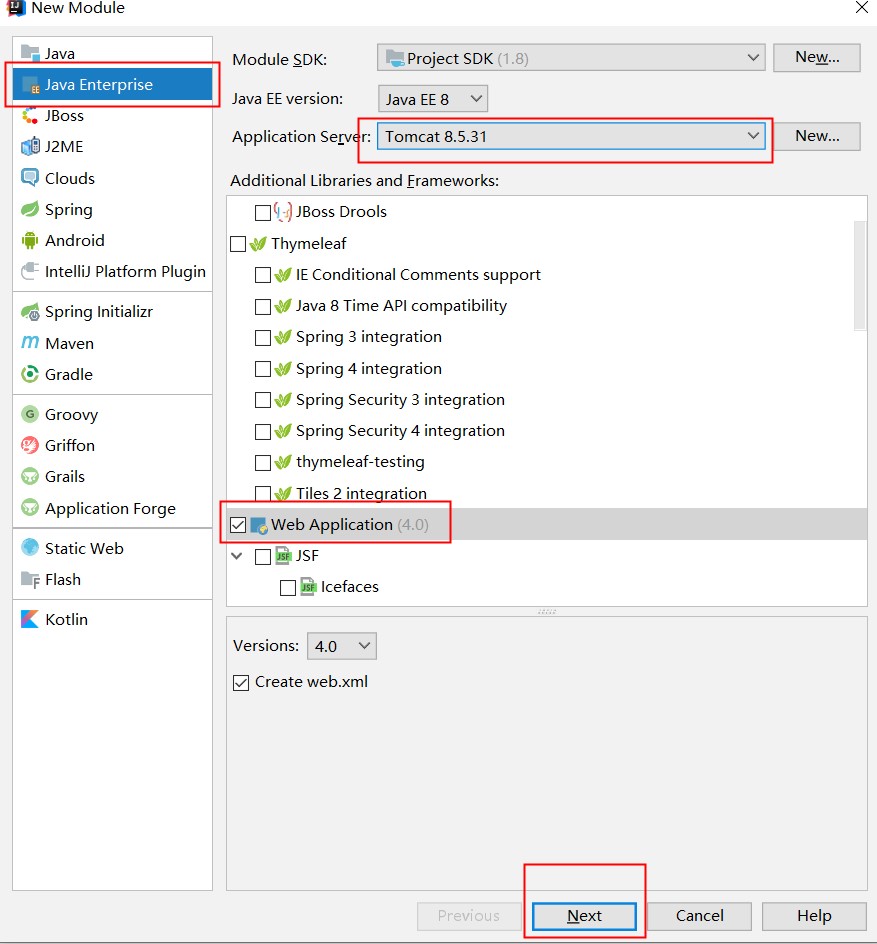
|  |
| --- |
| <Connector port="8080" protocol="HTTP/1.1 "  connectionTimeout="20000" redirectPort="8443 "  /> |

### Tomcat在idea中配置

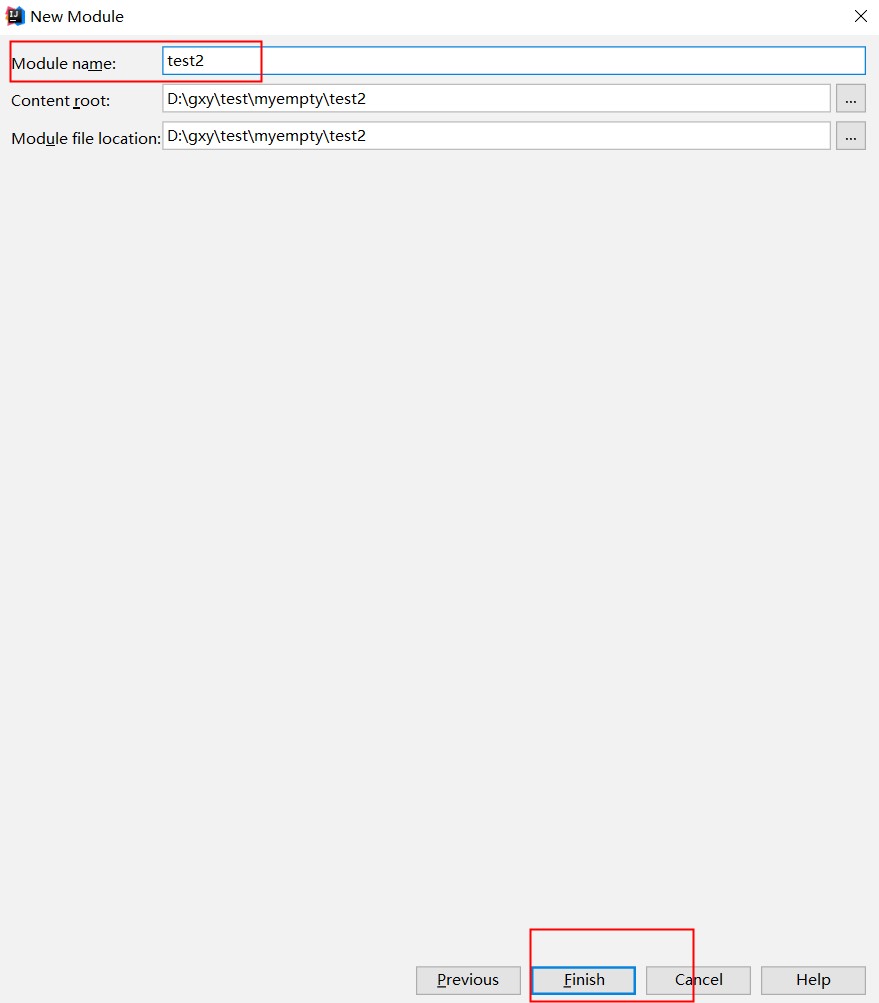
Step1：在idea的setting中配置tomcat路径



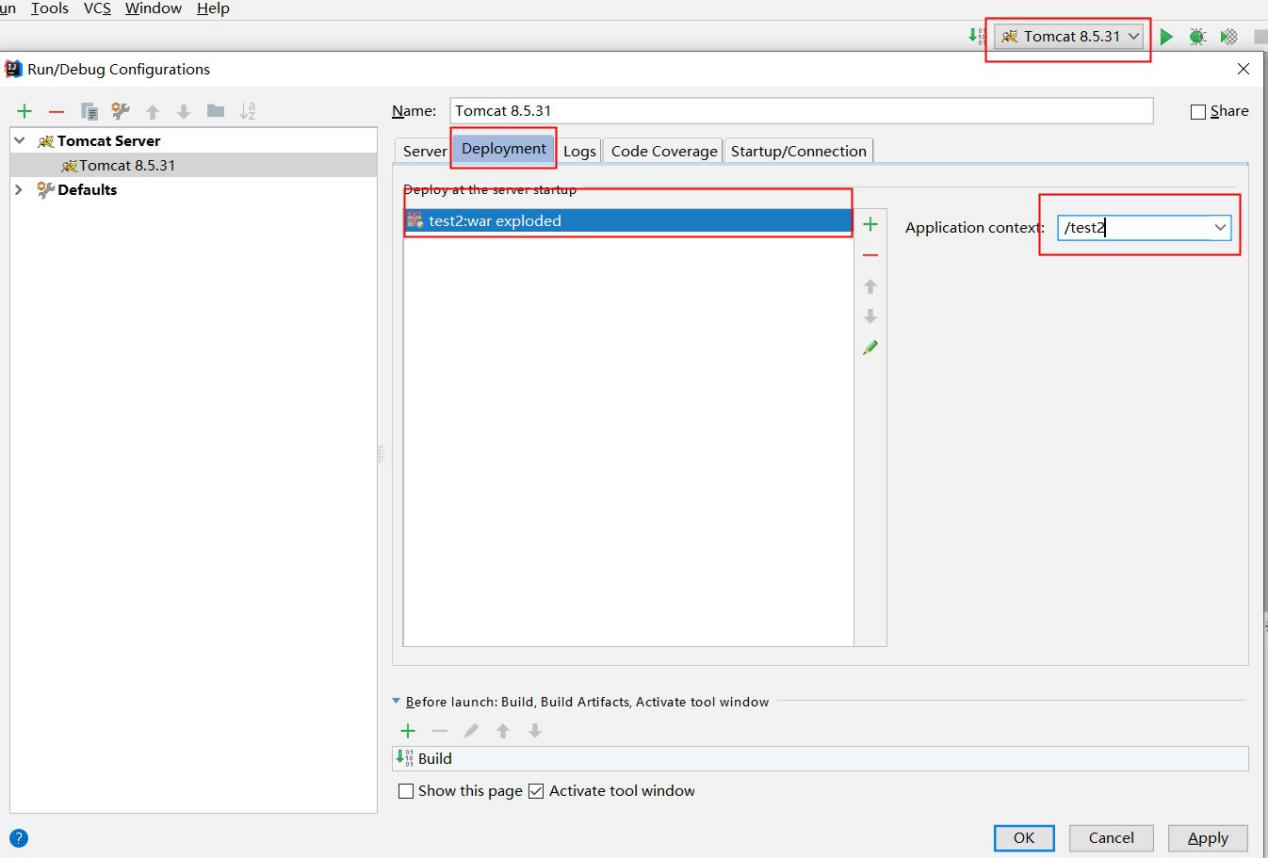
Step2：创建项目或模块时，选择java enterprise（也可以叫javaEE 企业应用），检查Tomcat服务器设置是否正确，并勾上webApplication 自动创建web.xml文件（web项目核心文件，必须创建）



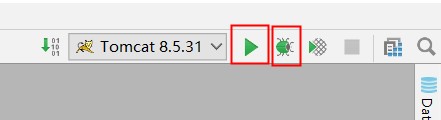
写模块名



Step3:项目会自动配置完成 再设置项目根路径（建议设置）

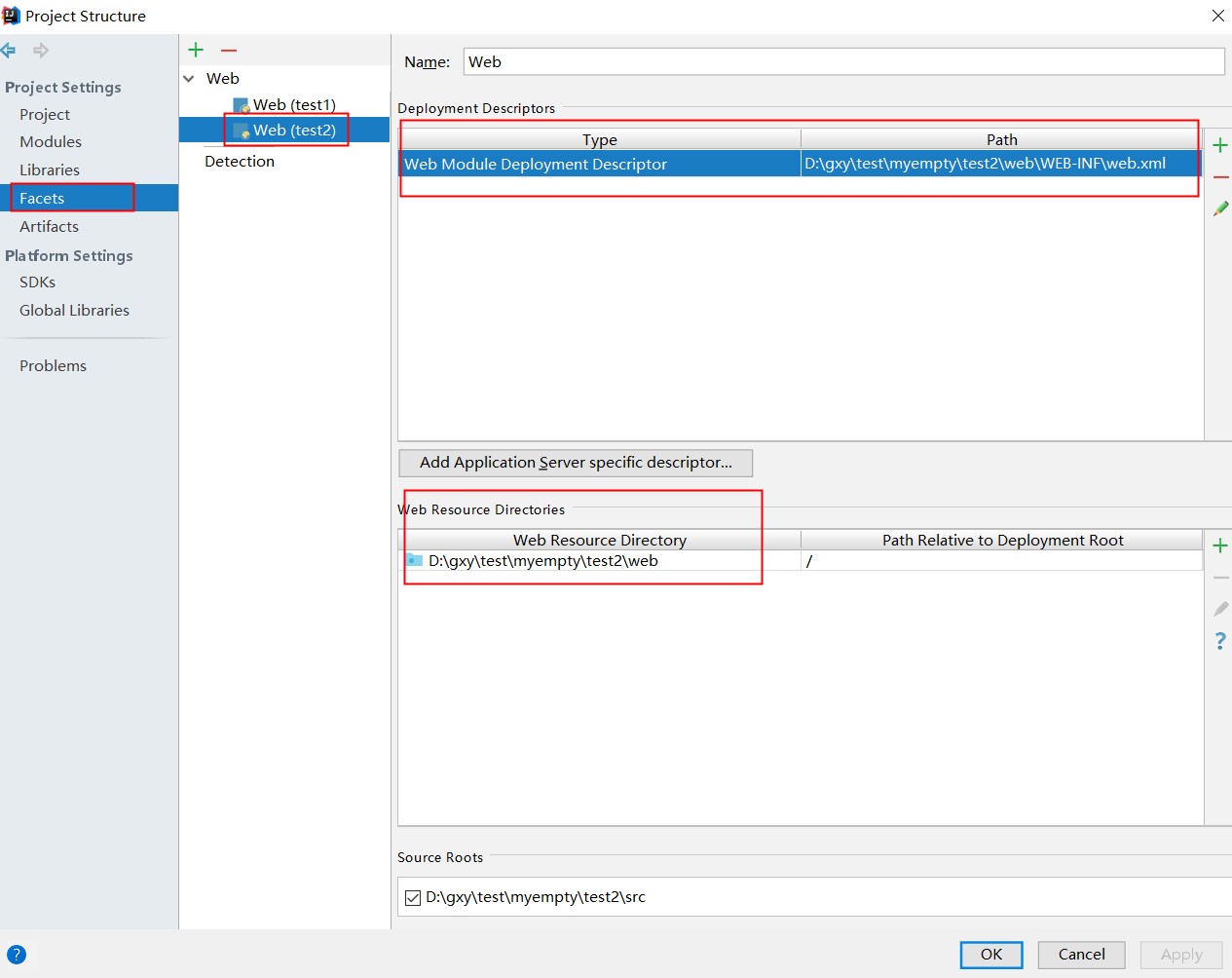


完成以上配置可以直接启动服务器

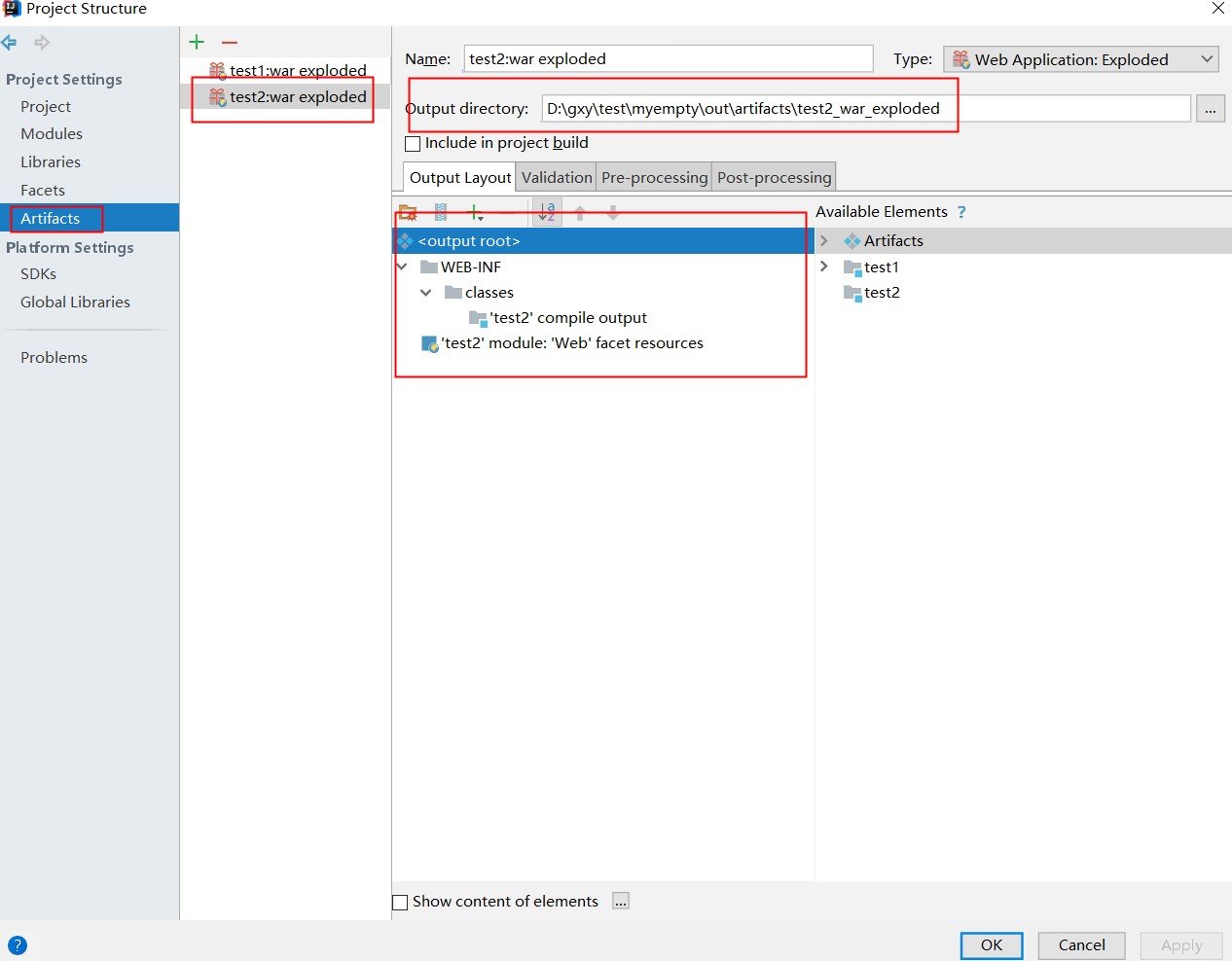


备注：由于idea中相关配置为自动配置 如果服务器启动异常 可以检查一下下图中的配置

1读取web目录的路径 web.xml文件路径是否正确



2项目打包策略是否正确 编译文件路径是否正确（WEB-INF/classes）



### Tomcat在eclipse中配置

Step1：

|  |
| --- |
| 右上角切换到javaEE视图 下方找到servers选项 点击蓝色文字配置tomcat服务器 |

Step2：

|  |
| --- |
| 弹出的窗口中选择对应的tomcat版本 再点击下一步 |

Step3：

|  |
| --- |
| 选择到tomcat解压目录，把jre选成自己配置的java运行环境，之后点击finish完成 |

Step4：

|  |
| --- |
| 双击配置好的服务器，进行参数设置，之后保存设置ctrl+s |

Step5：

|  |
| --- |
| 配置好的服务器上点右键测试启动 |

注意：配置好tomcat之后，会多出一个servers项目

|  |
| --- |
| 其中是tomcat的主要配置文件，每次tomcat通过eclipse启动时 会把servers中的配置文件复制到tomcat安装目录（eclipse中此项目不可删除），所以如果调试项目时需要修改配置文件，在servers项目下的配置文件修改 |

### 其他web服务器

* Jetty 更轻量级，更灵活，被一些框架集成，免费
* JBoss JBoss支持JavaEE标准，能够作为JavaEE应用服务器使用，Servlet容器只是其

中一部分，免费

* Weblogic 市场上最好的J2EE工具之一，功能大而全，收费

## Servlet

### Servlet介绍

* Servlet是JavaEE规范中的Web开发组件；
* Servlet运行在服务器端，需要Servlet容器的支持，例如Tomcat（运行在tomcat中）；
* 可以通过浏览器访问Servlet，Servlet可以生成动态页面返回给浏览器

|  |
| --- |
|  |

### 编写servlet

|  |
| --- |
| public class TestServlet extends HttpServlet {//继承httpservlet 重写doGet doPost方法  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  //处理代码  }  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  doGet(request, response);  }  }  Web.xml中配置  <servlet>  <servlet-name> TestServlet </servlet-name>  <servlet-class>com.javasm.servlet. TestServlet </servlet-class>  </servlet>  <servlet-mapping>  <servlet-name> TestServlet </servlet-name>  <url-pattern>/TestServlet </url-pattern>  </servlet-mapping>  配置中servlet-name相同作为同一组配置  servlet-class配置完整包名类名  url-pattern配置开放的访问路径 该路径为虚拟的路径  配置时也可以使用注解进行配置  @WebServlet(**"/配置访问路径"**)  注意：配置文件和用注解配置作用相同 只选用一种方式 不建议两种同时使用 |

### Servlet实现方式

* 实现Servlet的三种方式：一个实现，两个继承

1.实现Servlet接口

2.继承GenericServlet抽象类 重写service()方法

3.继承HttpServlet抽象类 HttpServlet继承于GenericServlet

重写doGet()，doPost()方法

### Servlet生命周期

|  |
| --- |
|  |

### Servlet其他配置参数

* servlet中配置初始化参数，可在servlet中直接读取

|  |
| --- |
| 配置中：  <servlet>  <servlet-name>HelloServlet</servlet-name>  <servlet-class>com.javasm.servlet.HelloServlet</servlet-class>  <init-param>  <param-name>testParam</param-name>  <param-value>Hello Servlet</param-value>  </init-param>  </servlet>  代码中：  String initParam = getInitParameter("testParam"); |

* 配置servlet全局参数，与servlet标签同级，所有servlet都可以读取

|  |
| --- |
| 配置中：  <context-param>  <param-name>contextParam</param-name>  <param-value>Hello Servlet</param-value>  </context-param>  代码中：  String contextParam = getServletContext().getInitParameter("contextParam"); |

* 配置servlet启动顺序load-on-startup

|  |
| --- |
| <servlet>  <servlet-name>HelloServlet</servlet-name>  <servlet-class>com.javasm.servlet.HelloServlet</servlet-class>  <load-on-startup>1</load-on-startup>  </servlet> |

注意：如果配置了load-on-startup，servlet实例会在项目启动时就加载，并且按照load-on-startup中配置的顺序加载，如果不加load-on-startup，实例会在有请求访问到该servlet时实例化

### Servlet中处理乱码

Servlet运行在web容器（tomcat）中，tomcat默认对传递的数据使用iso-8859-1进行编码，传递中文无法正常显示。

|  |
| --- |
| 请求乱码：  如果数据在传递时是正常中文，但是在servlet中取出是乱码，需要在请求对象中设置编码格式，对请求数据进行重新编码。  request.setCharacterEncoding("UTF-8");  响应乱码：  如果数据在servlet中是正常中文，到页面中显示乱码，则需要在响应对象中设置响应头，让页面指定以何种编码格式解析响应数据。  response.setContentType("text/html;charset=UTF-8"); |