JAVA 达摩班

Servlet & JSP

SERVLET 1

Servlet概念

一个Servlet就是一个类，用于处理request，返回response
例如，获得来自表单的数据，从数据库查询数据，动态创建并返回页面

Servlet也是一个规范，基于该规范的类需要运行在Servlet Container容器中。
JSP, Spring等技术都是基于Servlet规范的抽象实现，而且不仅限于HTTP协议。
常说的Tomcat, Jetty, JBoss等服务器就是servlet容器。

Servlet生命周期

1. init：仅执行一次

加载并实例化servlet类，通过init方法初始化该实例

```
public void init() throws ServletException {
}
```

2. service：只有在init成功之后被调用

用于处理请求，重载doGet，doPost，doPut和doDelete方法

```
public void service(ServletRequest request, ServletResponse response)
throws ServletException, IOException {
}
```

3. destroy：只会执行一次

当服务结束或者超时，容器删除该实例

```
public void destroy() {
}
```

Servlet注释

@WebServlet在JavaEE 6被引入，在部署阶段被容器解析处理，用于替代web.xml中的配置。

```
@WebServlet(name = "GoServlet", urlPatterns = "/go")
```

↓

```
<servlet>
   <servlet-name>GoServlet</servlet-name>
   <servlet-class>goServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>GoServlet</servlet-name>
    <url-pattern>/go</url-pattern>
</servlet-mapping>
```

Java Server Page (JSP)

2

JSP概念

JSP是Java Server Page的简称，它通过Java和Servlet将动态内容插入到静态内容当中
JSP传递java特定的数据到jsp视图，被客户端消费

JSP = html + <jsp syntax>

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

JSP语法提供两种方式插入java代码

1. 脚本语法 **<% Java code here %>**
2. XML语法 **<jsp:scriptlet>Java code here</jsp:scriptlet>**

JSP核心

静态内容是固定的，独立于请求操作和用户交互的内容；**动态内容**是根据请求返回而变化的，会被用户修改的内容

JSP清晰的实现**静态内容和动态内容相分离**，servlet管理动态内容，jsp管理静态内容和动态内容的嵌入

隐式对象(implicit object)是由JSP引擎在编译时动态生成的对象，包含HttpRequest，HttpResponse，用于servlet的运行在服务端的对象和用户交互对象，列表如下：

request(javax.servlet.http.HttpServletRequest), response(javax.servlet.http.HttpServletResponse)，out(javax.servlet.jsp.JspWriter), session(javax.servlet.http.HttpSession), application(javax.servlet.ServletContext), exception(javax.servlet.jsp.JspException), page(java.lang.Object) 当前servlet, pageContext(javax.servlet.jsp.PageContext), config(javax.servlet.ServletConfig)

JSP提供了大量可以直接使用的**指令(directive)**，实现JSP页面核心功能。

分为三类：<%@ **page** attribute=”value” %> 定义JSP依赖和属性, <%@ **include** … %> 用于导入文件，<%@ **taglib** …%> 自定义tag.

例如：<%@ page session=”value”%>是否开启session

JSP实现方式

1. **Servlet在服务器端渲染Html内容，通过out直接返回给客户端**

```
PrintWriter out = response.getWriter();
out.println();
```

2. **业务逻辑（java代码）直接写在JSP页面中，不和servlet交互**

```
<%  String[] arr = {"What's up?", "Hello", "It's a nice day today!"};
    String greetings = arr[(int)(Math.random() * arr.length)];
%>
```

3. **Servlet处理请求和业务逻辑，将请求转发回JSP页面**

```
request.setAttribute("text", message);
request.getRequestDispatcher("/binding.jsp").forward(request, response);
```

Tomcat

3

Tomcat安装 -手动安装

1. 下载Tomcat

https://tomcat.apache.org/download-80.cgi 或者 https://tomcat.apache.org/download-90.cgi

下载页面选择*tar.gz*包，默认下载到*Downloads*目录下，解压出相应文件目录

2. 拷贝tomcat到/usr/local目录，建立链接到/Library/Tomcat

```
sudo mkdir -p /usr/local
sudo mv ~/Downloads/apache-tomcat-9.0.8 /usr/local
sudo rm -f /Library/Tomcat
sudo ln -s /usr/local/apache-tomcat-9.0.7 /Library/Tomcat
```

3. 修改/Library/Tomcat权限

```
sudo chown -R <your_username> /Library/Tomcat
sudo chmod +x /Library/Tomcat/bin/*.sh
```

4. 启动和关闭Tomcat

```
/Library/Tomcat/bin/startup.sh
/Library/Tomcat/bin/shutdown.sh
```

Tomcat安装 -通过Homebrew

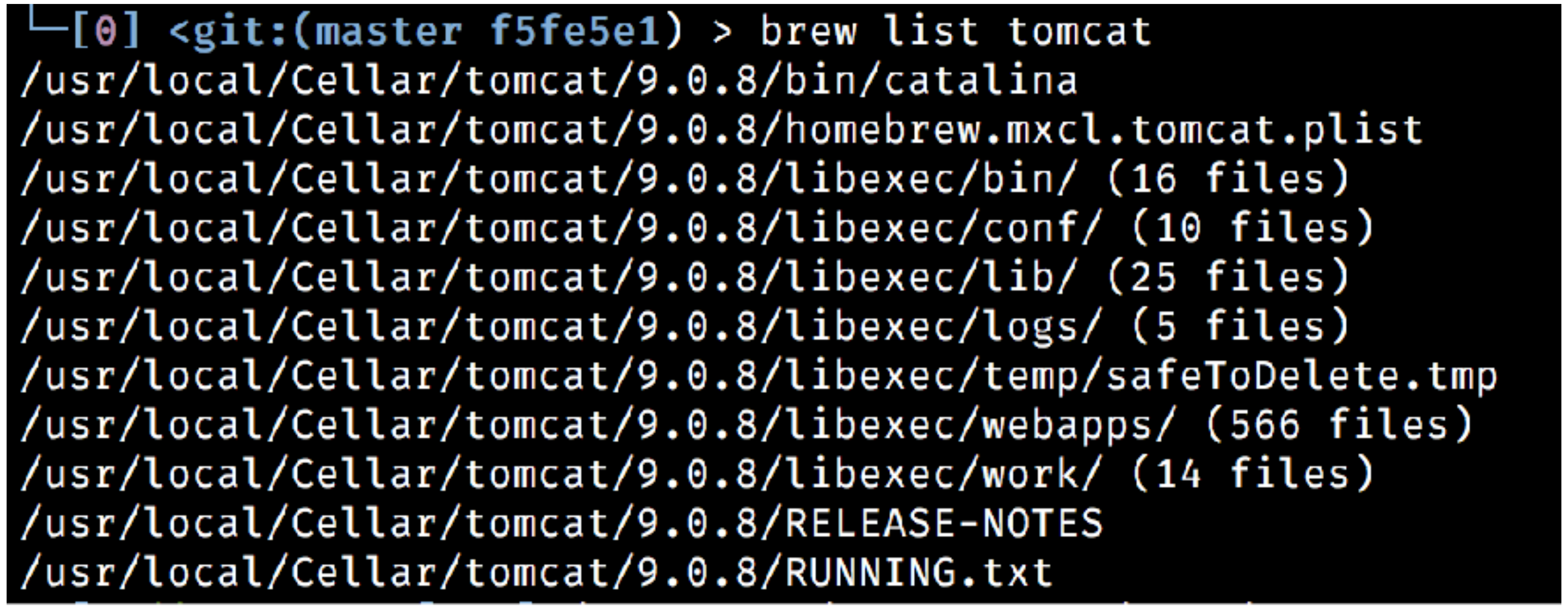
1. **使用brew安装**

   brew install tomcat

2. **查看tomcat安装情况**

   ls /usr/local/Cellar

   brew list tomcat



```
└─[0] <git:(master f5fe5e1) > brew list tomcat
/usr/local/Cellar/tomcat/9.0.8/bin/catalina
/usr/local/Cellar/tomcat/9.0.8/homebrew.mxcl.tomcat.plist
/usr/local/Cellar/tomcat/9.0.8/libexec/bin/ (16 files)
/usr/local/Cellar/tomcat/9.0.8/libexec/conf/ (10 files)
/usr/local/Cellar/tomcat/9.0.8/libexec/lib/ (25 files)
/usr/local/Cellar/tomcat/9.0.8/libexec/logs/ (5 files)
/usr/local/Cellar/tomcat/9.0.8/libexec/temp/safeToDelete.tmp
/usr/local/Cellar/tomcat/9.0.8/libexec/webapps/ (566 files)
/usr/local/Cellar/tomcat/9.0.8/libexec/work/ (14 files)
/usr/local/Cellar/tomcat/9.0.8/RELEASE-NOTES
/usr/local/Cellar/tomcat/9.0.8/RUNNING.txt
```

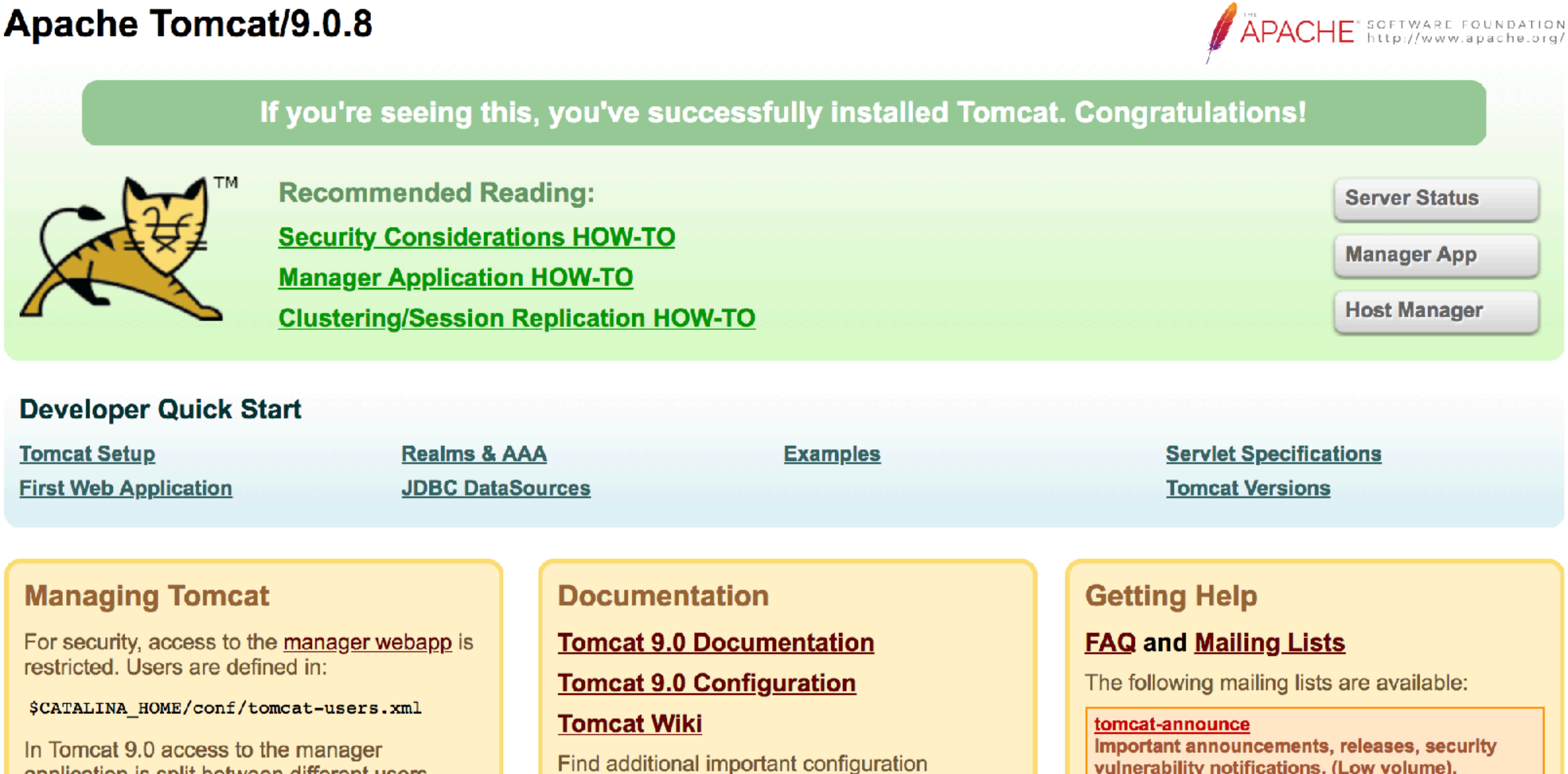
3. **启动和关闭Tomcat**

   catalina run &

   catalina stop

Tomcat安装

在浏览器输入 http://localhost:8080, 如果能看到如下页面说明安装成功



Apache Tomcat/9.0.8

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:

Security Considerations HOW-TO

Manager Application HOW-TO

Clustering/Session Replication HOW-TO

Developer Quick Start

Tomcat Setup

First Web Application

Realms & AAA

JDBC DataSources

Examples

Servlet Specifications

Tomcat Versions

Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager application is split between different users.

Documentation

Tomcat 9.0 Documentation

Tomcat 9.0 Configuration

Tomcat Wiki

Find additional important configuration

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

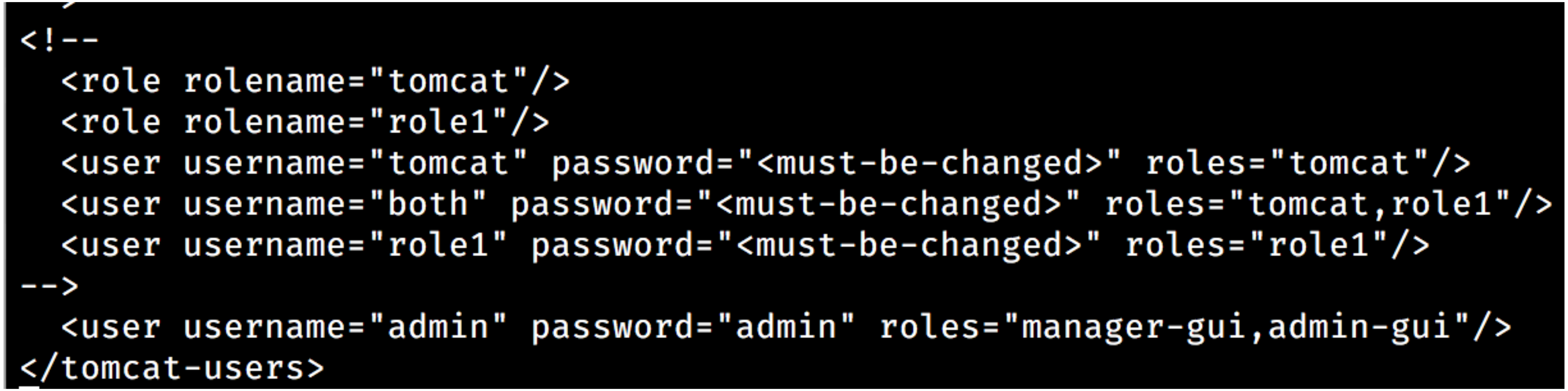
tomcat-announce
Important announcements, releases, security vulnerability notifications. (Low volume).

Tomcat 管理员

1. 新增用户（以homebrew创建的为例）

nano /usr/local/Cellar/tomcat/8.5.15/libexec/conf/tomcat-users.xml

添加*user <user username=”someUser” password=”somePassword” roles=”manager-gui”/>*

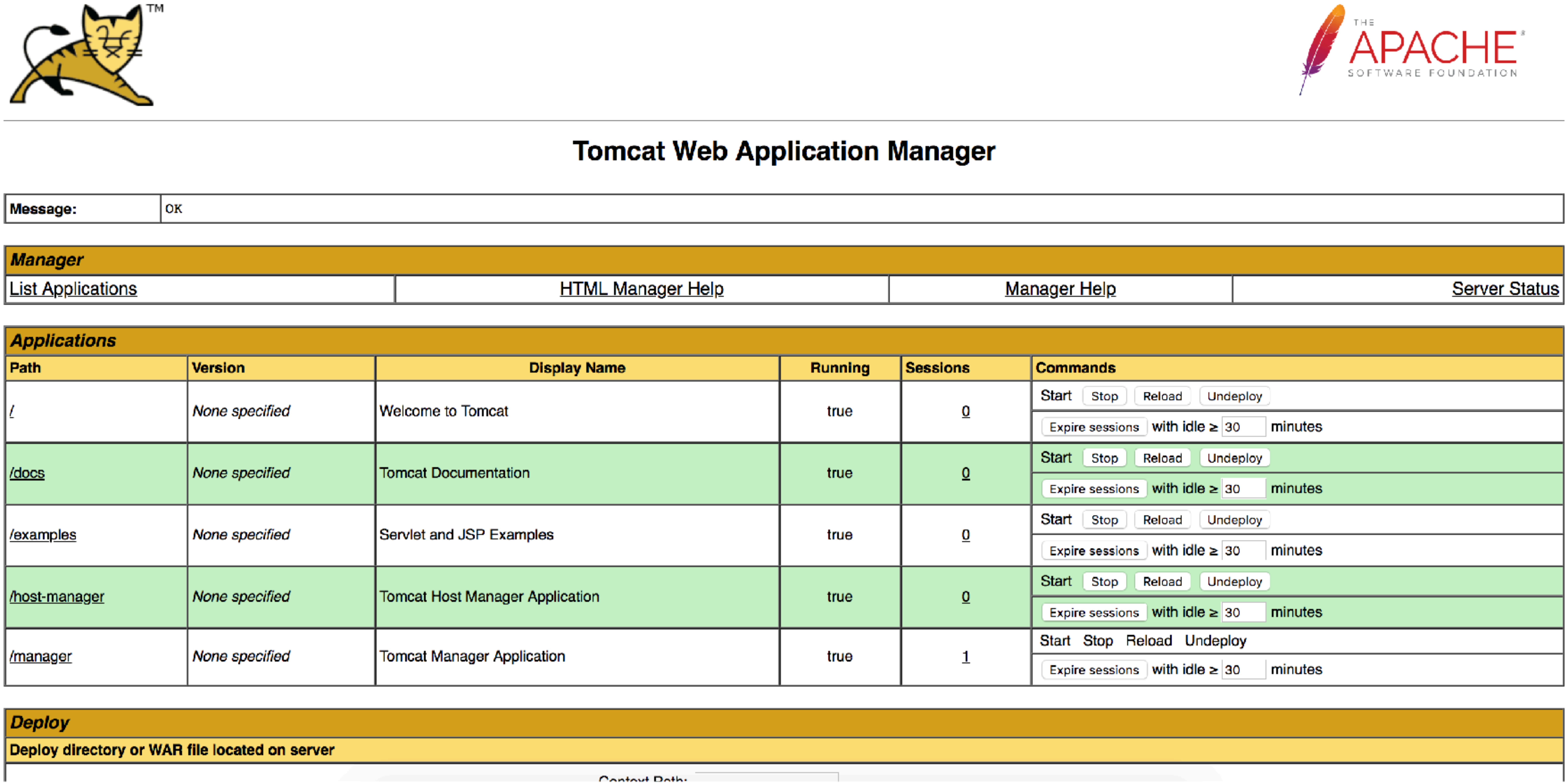


```
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
-->
  <user username="admin" password="admin" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

关闭并重启*tomcat*服务器

2. 在浏览器输入 http://localhost:8080/manager, 如果看到如下页面说明配置成功

Tomcat 管理员



Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	*None specified*	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	*None specified*	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	*None specified*	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

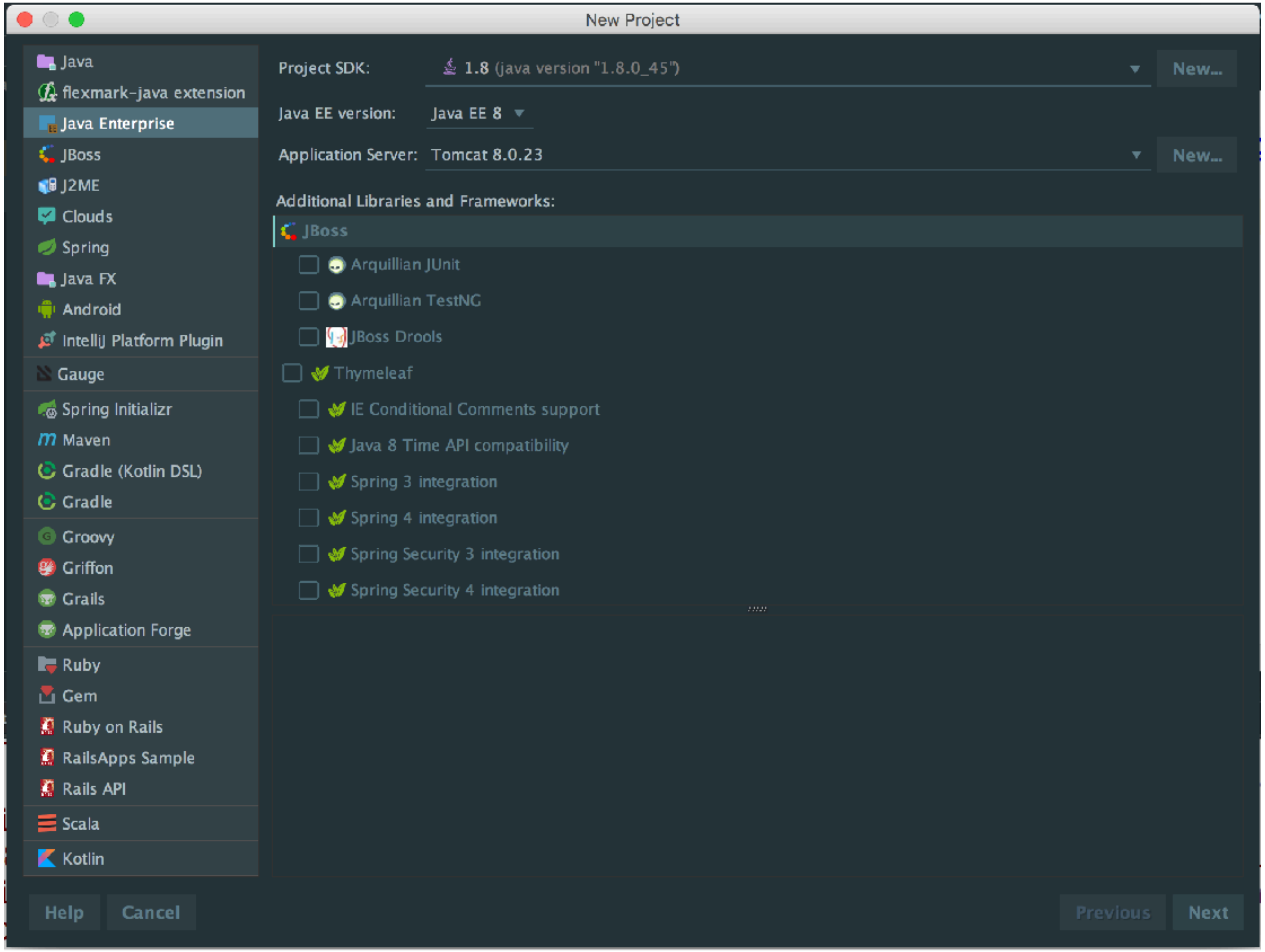
Deploy directory or WAR file located on server

Context Path:

Servlet和 JSP 演示

4

基于Intellij IDEA JavaEE模版



Project SDK: 1.8 (java version "1.8.0_45")

Java EE version: Java EE 8

Application Server: Tomcat 8.0.23

Additional Libraries and Frameworks:

JBoss
- ☐ Arquillian JUnit
- ☐ Arquillian TestNG
- ☐ JBoss Drools

☐ Thymeleaf
- ☐ IE Conditional Comments support
- ☐ Java 8 Time API compatibility
- ☐ Spring 3 integration
- ☐ Spring 4 integration
- ☐ Spring Security 3 integration
- ☐ Spring Security 4 integration

1. 新建工程，选择Java Enterprise模版

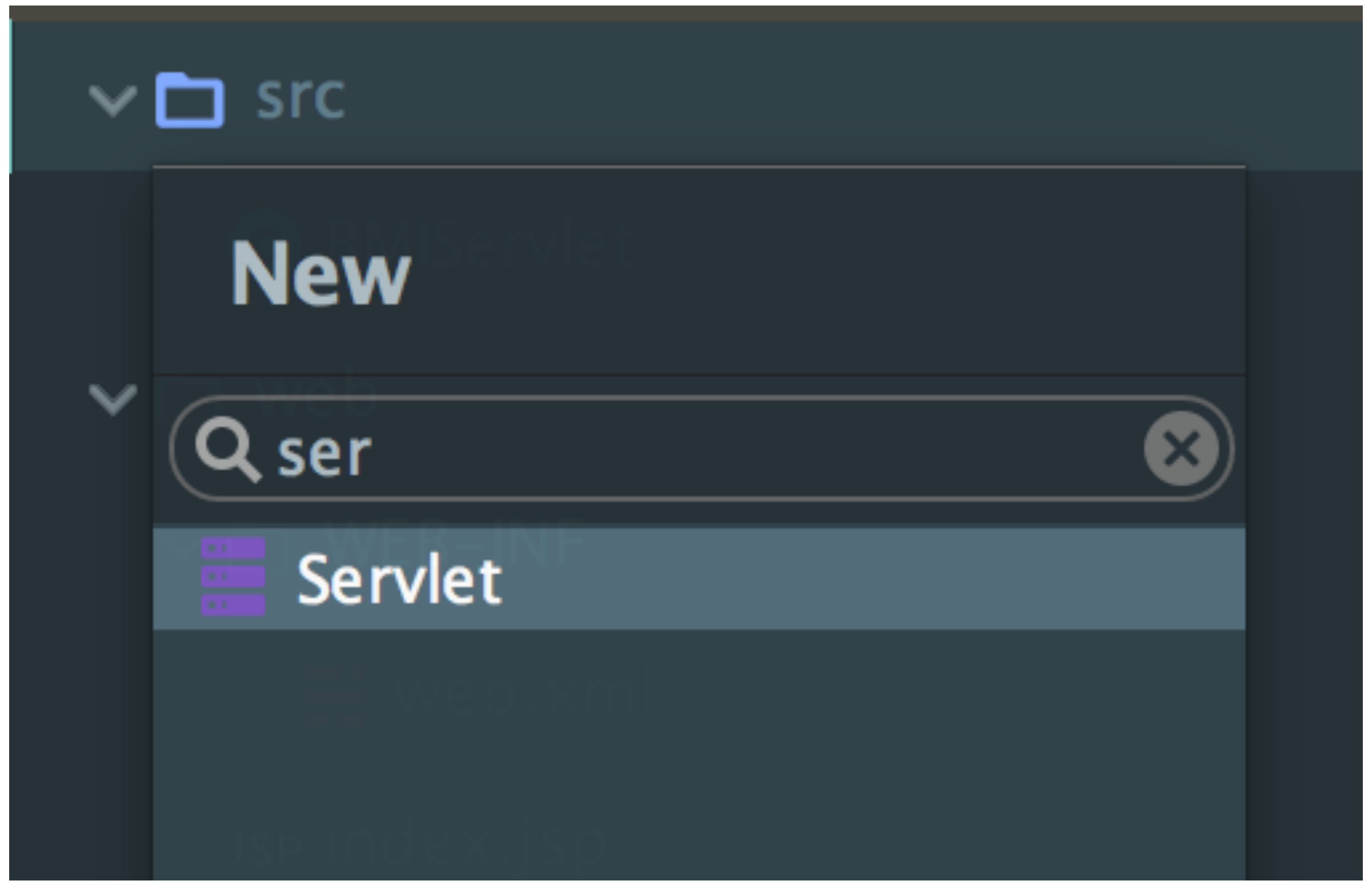
2. 新建Servlet类文件

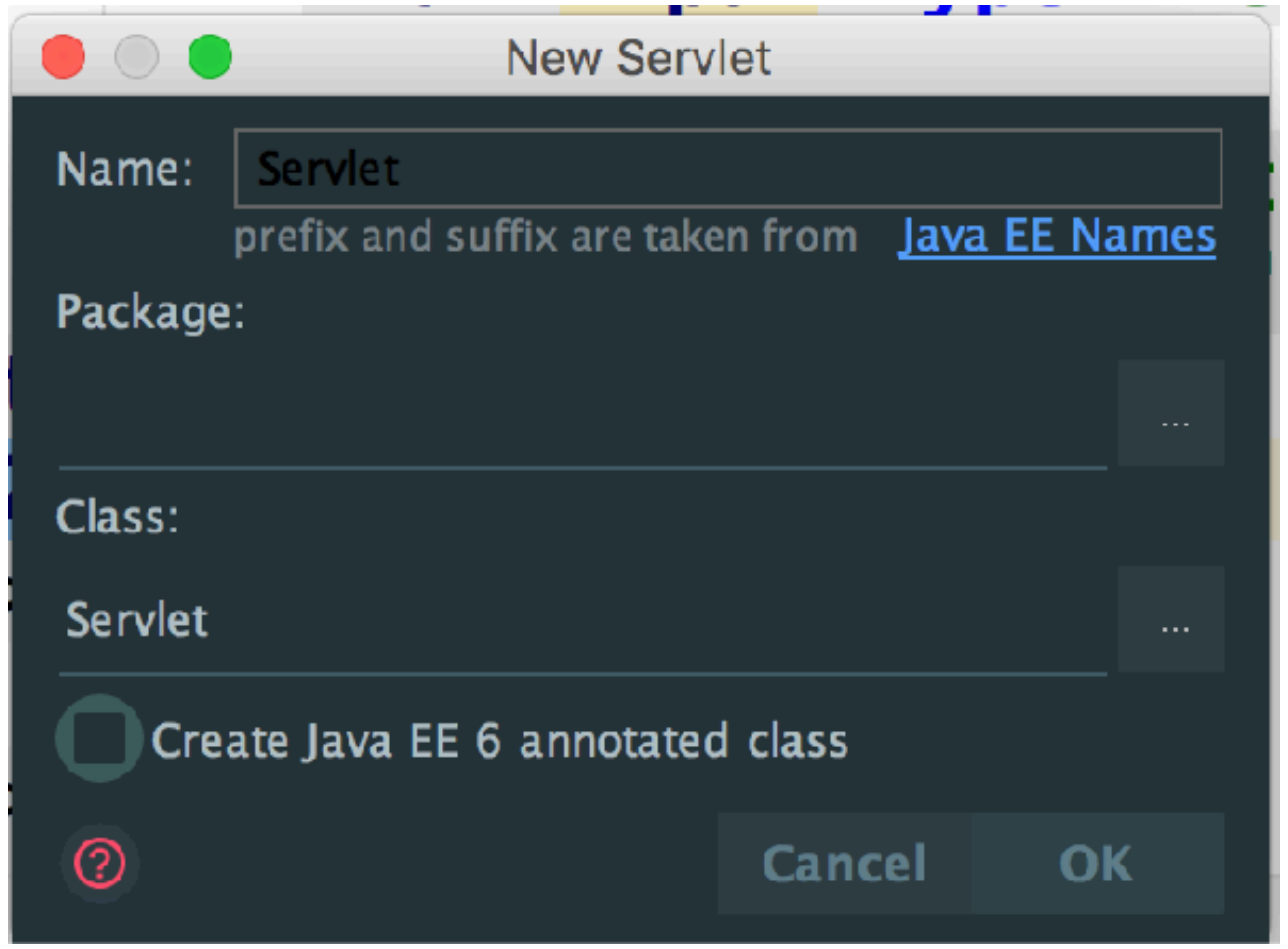
不勾选*“Create Java EE 6 anotation class“*

3. 修改web.xml注册Servlet

4. 实现JSP页面

基于Intellij IDEA JavaEE模版




New Servlet

Name: Servlet

prefix and suffix are taken from Java EE Names

Package:

Class: Servlet

□ Create Java EE 6 annotated class

1. 新建工程，选择Java Enterprise模版

2. 新建Servlet类文件

不勾选*“Create Java EE 6 anotation class“*

3. 修改web.xml注册Servlet

4. 实现JSP页面

基于Intellij IDEA JavaEE模版

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
         http://java.sun.com/xml/ns/javaee/web-app_3_1.xsd"
          version="3.1">

    <servlet>
        <servlet-name>BMIServlet</servlet-name>
        <servlet-class>BMIServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>BMIServlet</servlet-name>
        <url-pattern>/bmi</url-pattern>
    </servlet-mapping>

</web-app>
```

1. 新建工程，选择Java Enterprise模版

2. 新建Servlet类文件
   不勾选*“Create Java EE 6 anotation class“*

3. **修改web.xml注册Servlet**

4. 实现JSP页面

基于Intellij IDEA JavaEE模版

```
<body>

<form name="bmiForm" action="bmi" method="POST">
    <table>
        <tr>
            <td>Your Weight (kg) :</td>
            <td><input type="text" name="weight"/></td>
        </tr>
        <tr>
            <td>Your Height (m) :</td>
            <td><input type="text" name="height"/></td>
        </tr>
        <th><input type="submit" value="Submit" name="find"/></th>
        <th><input type="reset" value="Reset" name="reset"/></th>
    </table>
    <h2>${bmi}</h2>
</form>

</body>
```

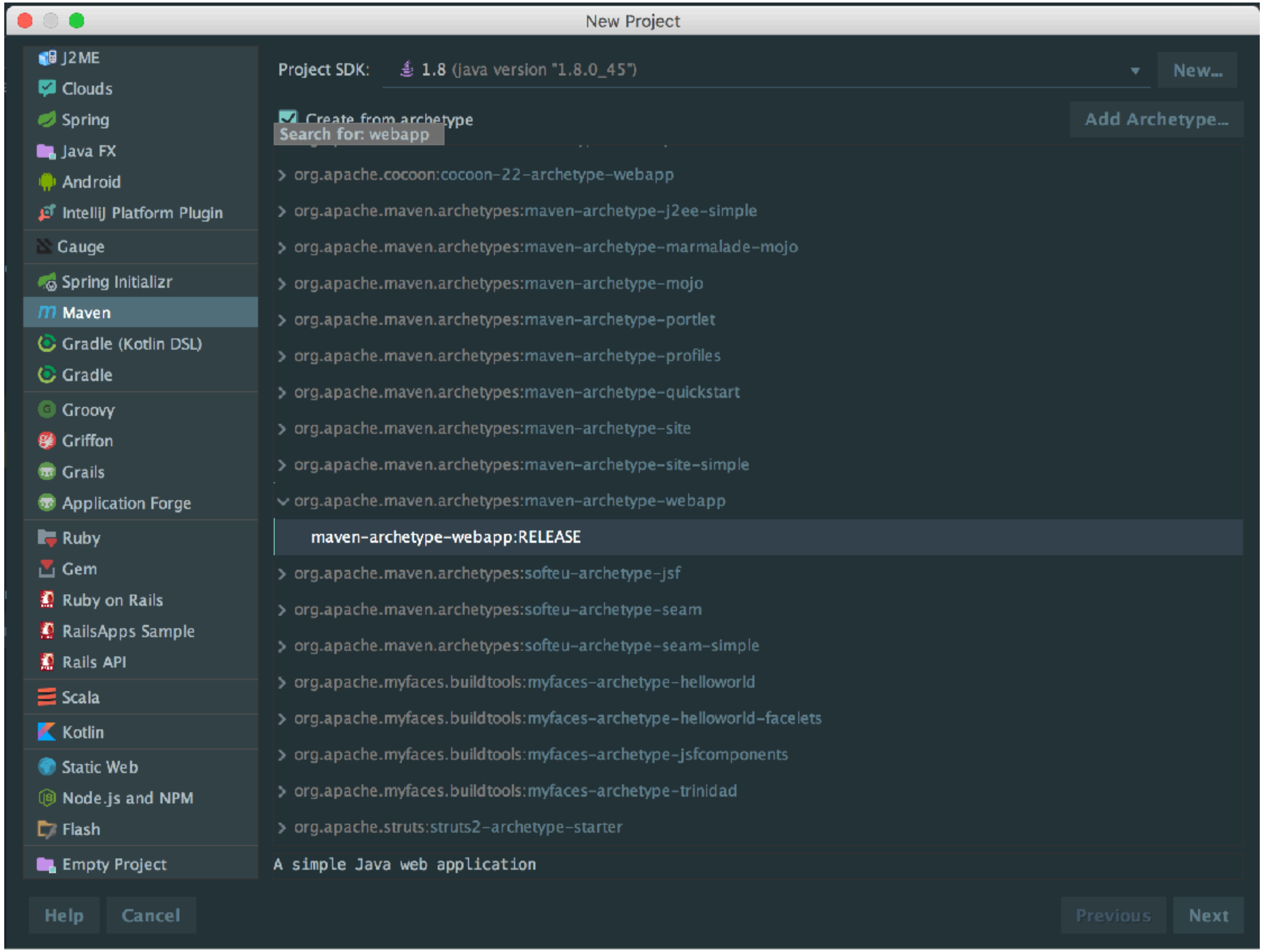
1. 新建工程，选择Java Enterprise模版

2. 新建Servlet类文件
不勾选*“Create Java EE 6 anotation class“*

3. 修改web.xml注册Servlet

4. **实现JSP页面**

基于Maven archetype原型



Project SDK: 1.8 (java version "1.8.0_45")

☑ Create from archetype

Search for: webapp

- org.apache.cocoon:cocoon-22-archetype-webapp
- org.apache.maven.archetypes:maven-archetype-j2ee-simple
- org.apache.maven.archetypes:maven-archetype-marmalade-mojo
- org.apache.maven.archetypes:maven-archetype-mojo
- org.apache.maven.archetypes:maven-archetype-portlet
- org.apache.maven.archetypes:maven-archetype-profiles
- org.apache.maven.archetypes:maven-archetype-quickstart
- org.apache.maven.archetypes:maven-archetype-site
- org.apache.maven.archetypes:maven-archetype-site-simple
- org.apache.maven.archetypes:maven-archetype-webapp
  - maven-archetype-webapp:RELEASE
- org.apache.maven.archetypes:softeu-archetype-jsf
- org.apache.maven.archetypes:softeu-archetype-seam
- org.apache.maven.archetypes:softeu-archetype-seam-simple
- org.apache.myfaces.buildtools:myfaces-archetype-helloworld
- org.apache.myfaces.buildtools:myfaces-archetype-helloworld-facelets
- org.apache.myfaces.buildtools:myfaces-archetype-jsfcomponents
- org.apache.myfaces.buildtools:myfaces-archetype-trinidad
- org.apache.struts:struts2-archetype-starter

```
A simple Java web application
```

1. 新建Maven工程，选择原型
maven-archetype-webapp

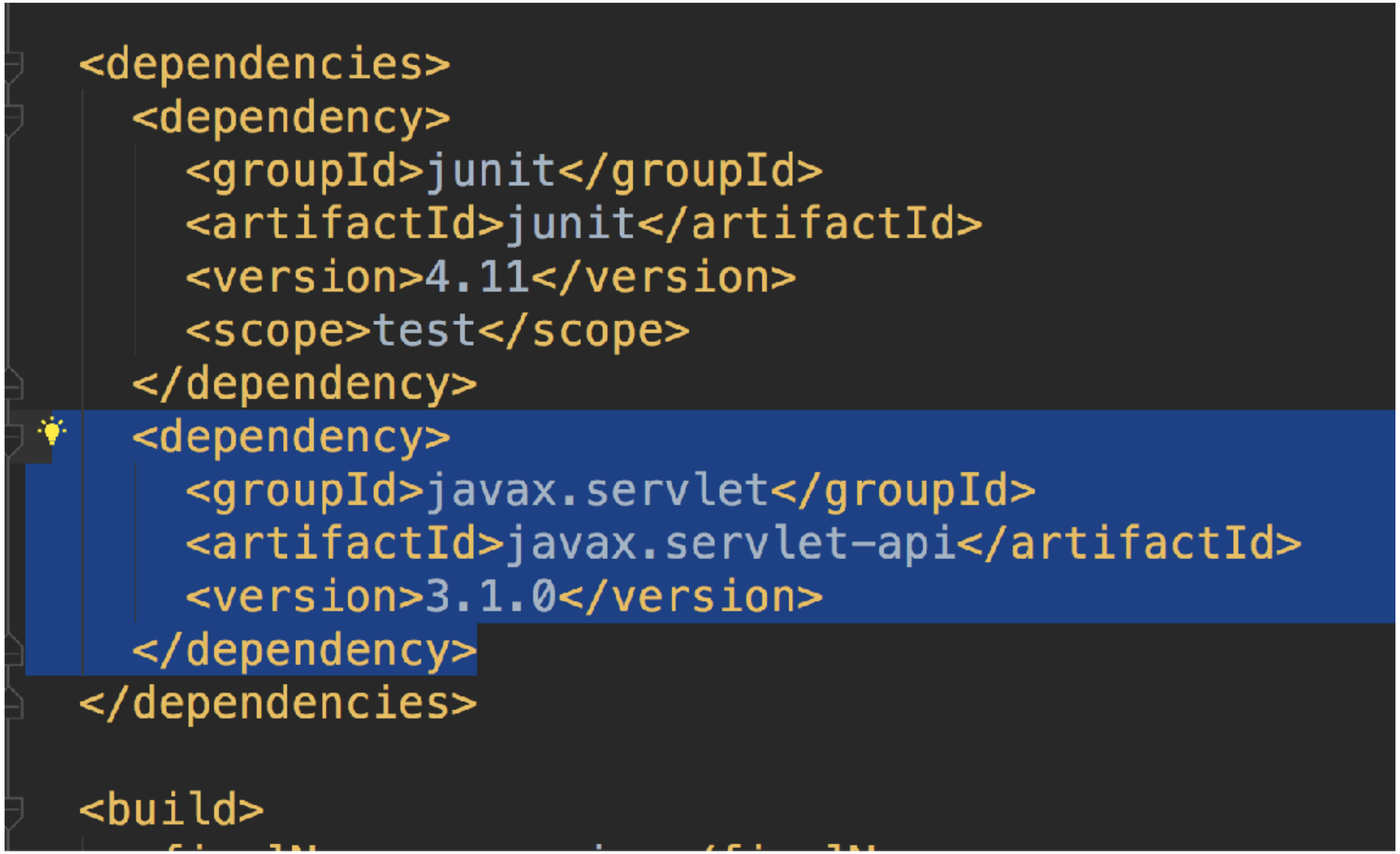
2. 添加依赖
javax.servlet:javax.servlet-api:3.1.0

3. 创建java目录，建立Servlet文件

4. 修改web.xml

5. 创建JSP文件内容，并移动到WEB-INF目录

基于Maven archetype原型



```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.11</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <version>3.1.0</version>
  </dependency>
</dependencies>

<build>
```

1. 新建Maven工程，选择原型
maven-archetype-webapp

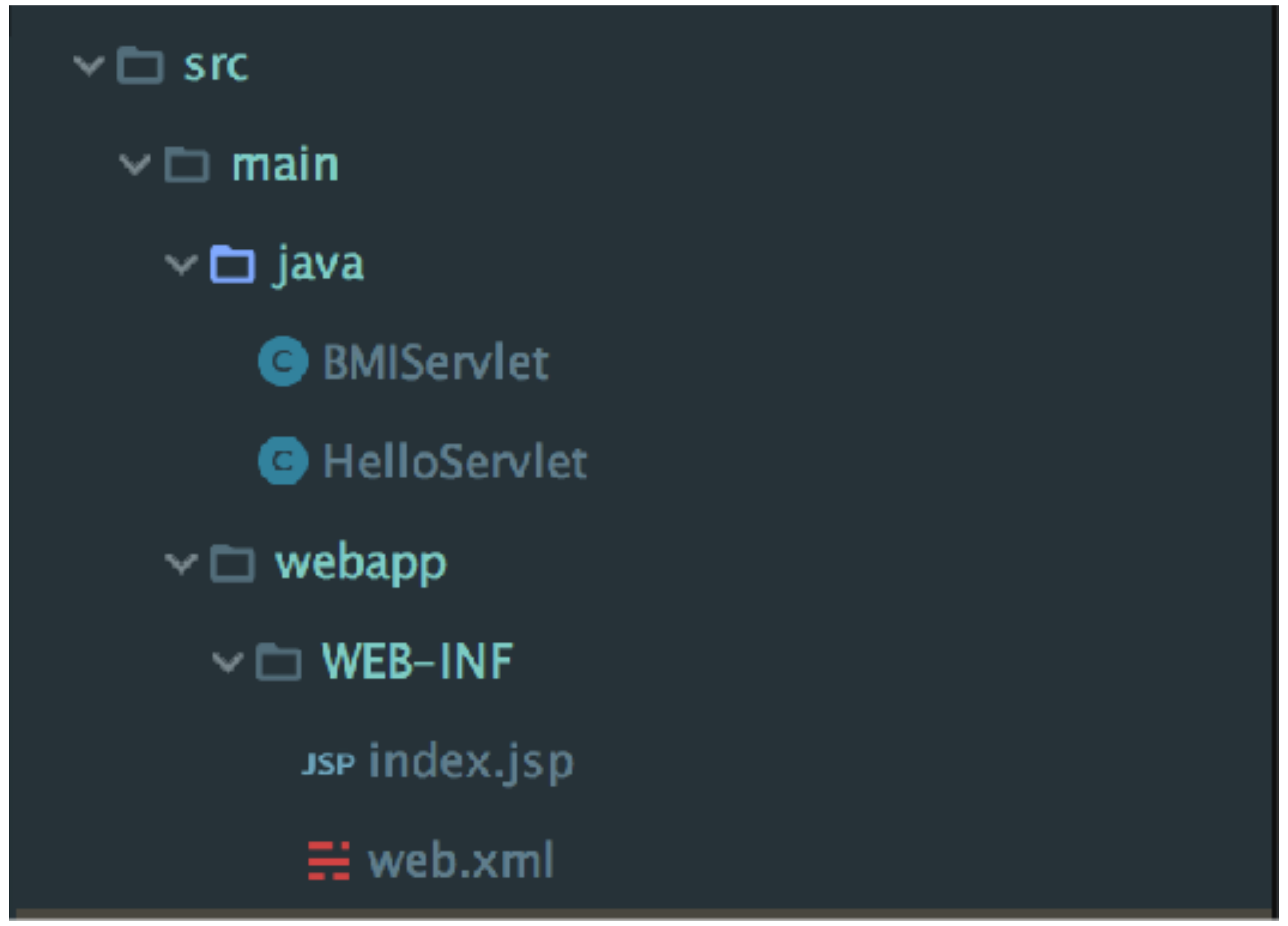
2. 添加依赖
javax.servlet:javax.servlet-api:3.1.0

3. 创建java目录，建立Servlet文件

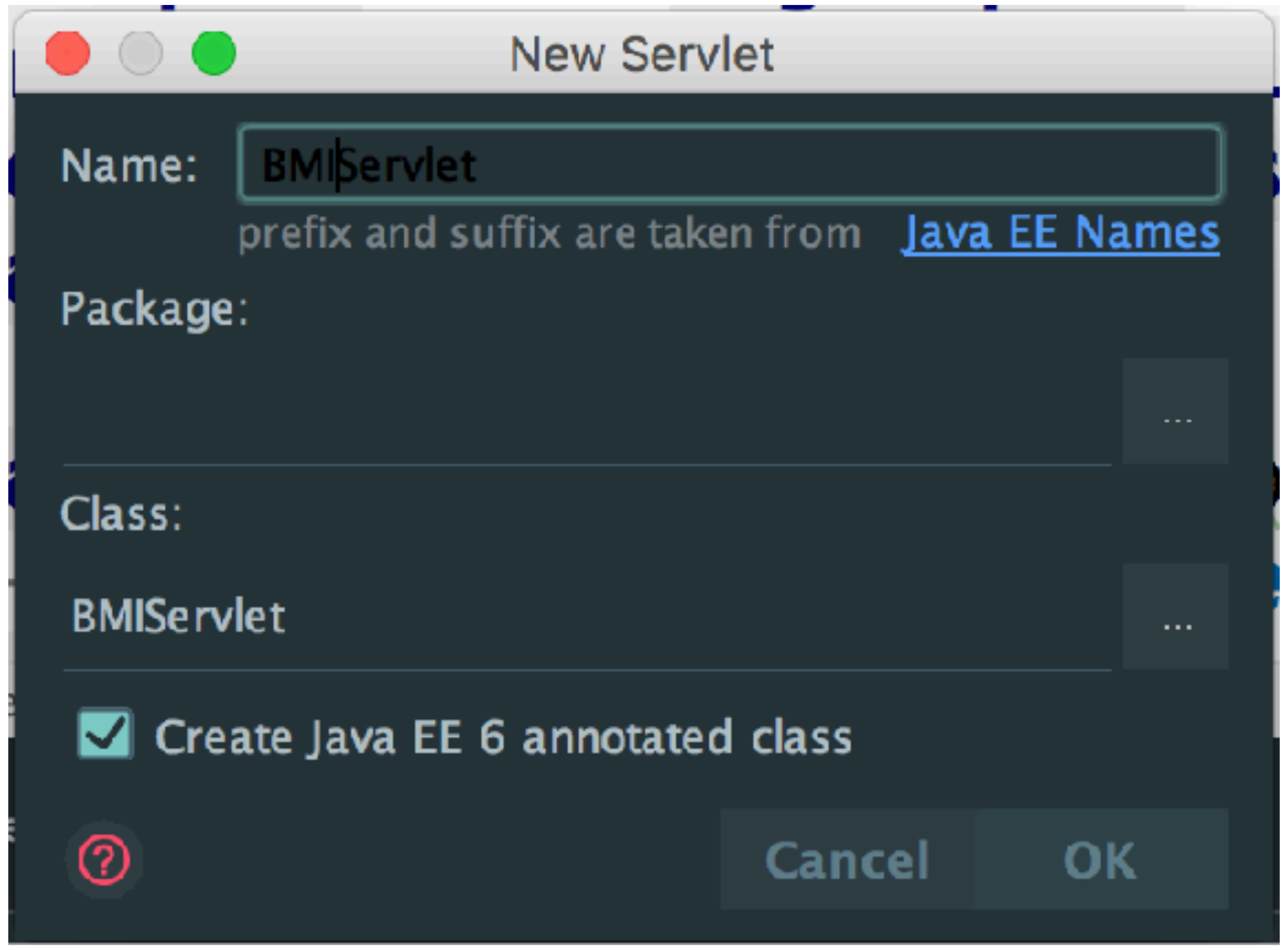
4. 修改web.xml

5. 创建JSP文件内容，并移动到
WEB-INF目录

基于Maven archetype原型



```
src
  main
    java
      BMIServlet
      HelloServlet
    webapp
      WEB-INF
        index.jsp
        web.xml
```



New Servlet

Name: BMIServlet

prefix and suffix are taken from Java EE Names

Package:

Class: BMIServlet

☑ Create Java EE 6 annotated class

1. 新建Maven工程，选择原型
maven-archetype-webapp

2. 添加依赖
javax.servlet:javax.servlet-api:3.1.0

3. 创建java目录，建立Servlet文件

4. 修改web.xml

5. 创建JSP文件内容，并移动到WEB-INF目录

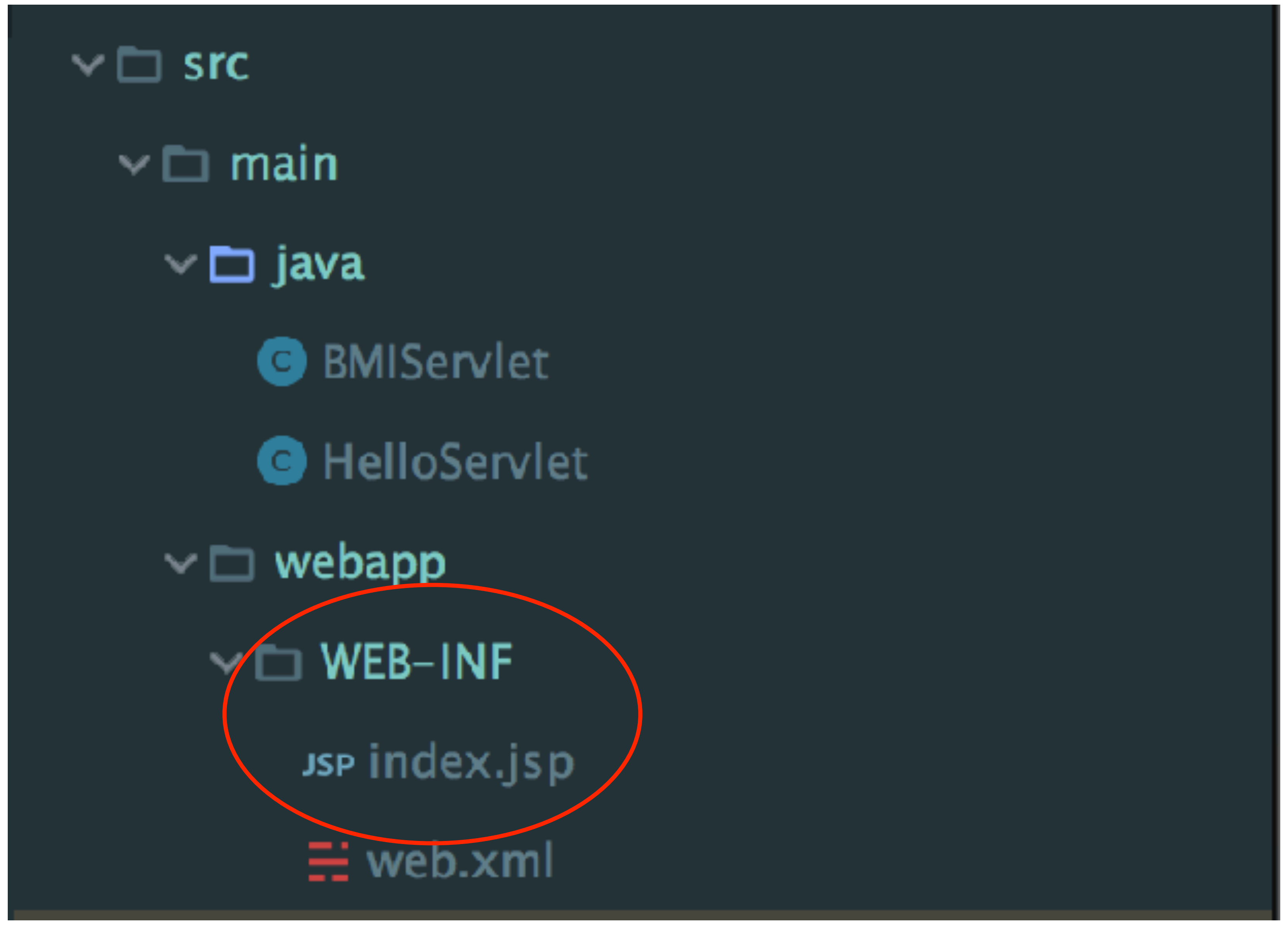
基于Maven archetype原型

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://
java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        id="WebApp_ID" version=“3.0">

  <display-name>Archetype Created Web Application</display-name>

</web-app>
```



```
src
  main
    java
      BMIServlet
      HelloServlet
    webapp
      WEB-INF
        index.jsp
      web.xml
```

1. 新建Maven工程，选择原型 *maven-archetype-webapp*

2. 添加依赖 *javax.servlet:javax.servlet-api:3.1.0*

3. 创建java目录，建立Servlet文件

4. **修改web.xml**

5. **创建JSP文件内容，并移动到WEB-INF目录**

作业

1. 通过Tomcat自带examples代码学习JSP

Thanks!

Any questions?