**1.3. 如何开发一个Servlet**

**1.3.1. 安装Tomcat**

Servlet是Web服务器功能的补充，要想能够运行必须依赖Servlet容器的管理才可以，为了进行下一步的Servlet的具体开发，则需要先准备用于运行Servlet的环境，在学习过程中我们选择的是主流的Tomcat作为Web服务器及Servlet容器。

下面，简单介绍下Tomcat的安装步骤：

**步骤一：下载并解压安装文件**

在浏览器中输入 tomcat.apache.org，在Download菜单中选择Tomcat7.0后会得到如图- 4 所示的下载链接。

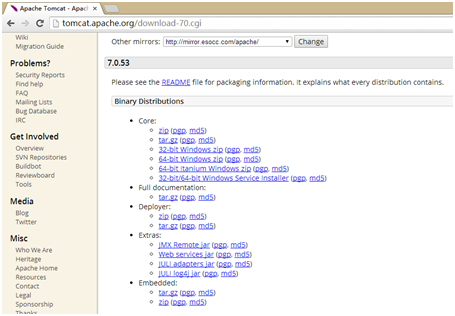


图- 4

选择Core下面符合当前操作系统的版本后下载，下载后的文件为压缩文件，解压到硬盘即可。假定Tomcat解压后的路径为 c:\java\tomcat7.0

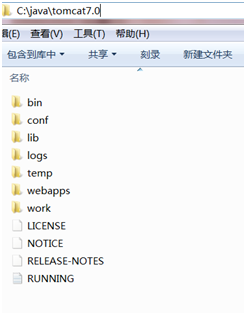


图- 5

**步骤二：配置环境变量**

选择“我的电脑”右键 (“属性”( “高级系统设置” ( “环境变量” 用于新建及修改一些环境变量，以保证系统记录解压的Tomcat软件的位置。如图-6所示：

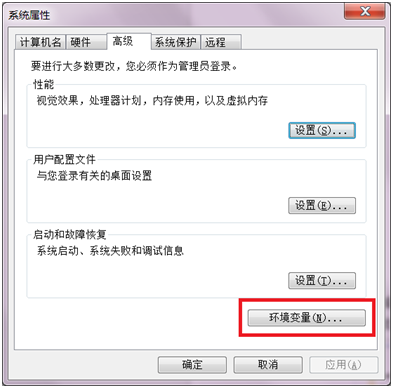


图- 6

点击系统变量下的“新建”按钮，创建“变量名”为 CATALINA\_HOME ， “变量值“ 为 c:\java\tomcat7.0的环境变量后点击确定按钮，如图- 7所示。

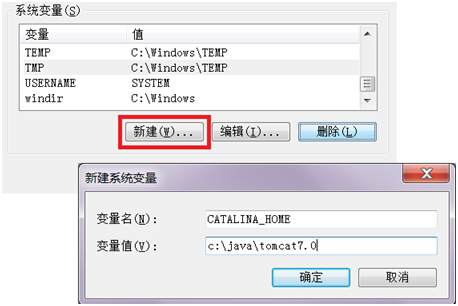


图- 7

选中系统变量中的“Path“后点击编辑，在原有内容的最前面添加 “ %CATALINA\_HOME%\bin; ” 后点击确定按钮。注意：修改值的时候光标放在最前面，并且要用分号将路径与原有Path的值分开。如图- 8所示。

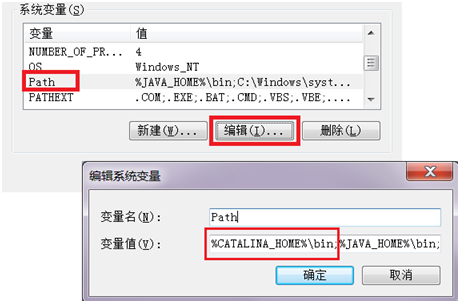


图- 8

若安装jdk时已配置了CLASS\_PATH变量，则选中系统变量中的“CLASS\_PATH“后点击编辑，在原有内容的最前面添加 “ %CATALINA\_HOME%\lib\servlet-api.jar; ” 后点击确定按钮。注意：修改值的时候光标放在最前面，并且要用分号将路径与原有CLASS\_PATH的值分开。如图- 9所示。

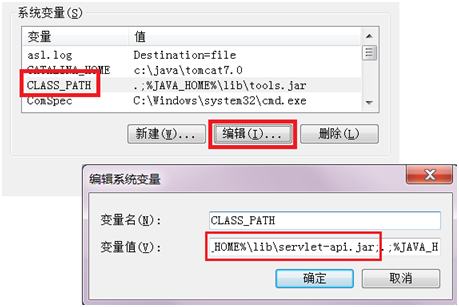


图- 9

至此，环境变量配置完毕。

**步骤三：启动Tomcat**

为了检测环境变量是否配置成功，以及Tomcat是否能够成功启动，首先进入命令行，windows系统下可以使用 “cmd“命令进入。在命令行窗口下输入“startup”命令，会得到如图 – 10 所示结果。

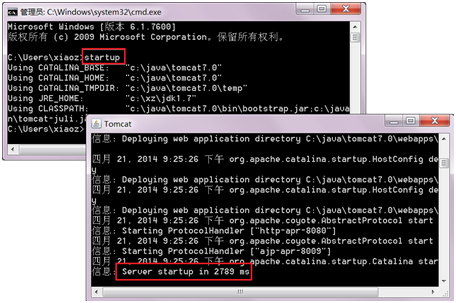


图- 10

打开浏览器，输入 “ http://localhost:8080 ”地址后，看到如图 – 11 所示结果。

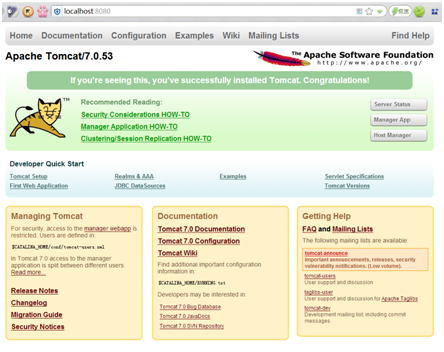


图- 11

至此，看到该页面即代表Tomcat启动成功。

**步骤四：关闭Tomcat**

在命令提示符下输入“shutdown”命令后，Tomcat的命令提示符窗口会关闭，在浏览器中再次输入http://localhost:8080后，看不到 图- 11，代表关闭Tomcat

**1.3.2. 开发Servlet的步骤**

**步骤一：编写一个实现Servlet接口或继承HttpServlet的Java类**

在硬盘上新建一个文件名为“HelloServlet.java”的源文件，并编写代码如下：

[copytextpop-up](http://pdf7.tarena.com.cn/tts8_source/ttsPage/JAVA/JSD_N_V06/SERVLETJSP/DAY01/SUPERDOC/01/index.html)

1. package web;
3. import java.io.IOException;
4. import java.io.PrintWriter;
6. import javax.servlet.ServletException;
7. import javax.servlet.http.HttpServlet;
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;

12. public class HelloServlet extends HttpServlet {
13. protected void service(HttpServletRequest req, HttpServletResponse resp)throws ServletException, IOException {
14. resp.setContentType("text/html");
15. PrintWriter out = resp.getWriter();
16. out.write("<h1>Hello Servlet");
17. out.close();
18. }
19. }

package web;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class HelloServlet extends HttpServlet {

protected void service(HttpServletRequest req, HttpServletResponse resp)throws ServletException, IOException {

resp.setContentType("text/html");

PrintWriter out = resp.getWriter();

out.write("<h1>Hello Servlet");

out.close();

}

}

**步骤二：使用javac命令编译源文件为字节码文件**

在命令行提示符下，进入到HelloServlet.java文件所在的位置，输入如下命令：

[copytextpop-up](http://pdf7.tarena.com.cn/tts8_source/ttsPage/JAVA/JSD_N_V06/SERVLETJSP/DAY01/SUPERDOC/01/index.html)

1. javac HelloServlet.java –cp c:\java\tomcat7.0\lib\servlet-api.jar

javac HelloServlet.java –cp c:\java\tomcat7.0\lib\servlet-api.jar

在没有报错的情况下会在java源文件的同级目录内出现HelloServlet.class的文件，代表编译成功。

**步骤三：将编译完的组件打包**

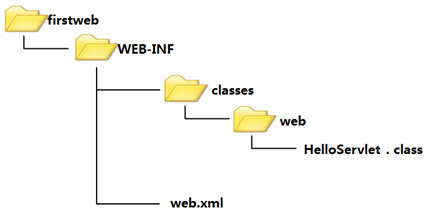


图- 12

按照如图 – 12 所示的结构将组建打包，并新建web.xml文件，内容如下：

[copytextpop-up](http://pdf7.tarena.com.cn/tts8_source/ttsPage/JAVA/JSD_N_V06/SERVLETJSP/DAY01/SUPERDOC/01/index.html)

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app version="2.5"
3. xmlns="http://java.sun.com/xml/ns/javaee"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
6. http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd">
7. <servlet>
8. <servlet-name>helloServlet</servlet-name>
9. <servlet-class>web.HelloServlet</servlet-class>
10. </servlet>
11. <servlet-mapping>
12. <servlet-name>helloServlet</servlet-name>
13. <url-pattern>/sayHi</url-pattern>
14. </servlet-mapping>
15. </web-app>

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

<web-app version="2.5"

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\_2\_5.xsd">

<servlet>

<servlet-name>helloServlet</servlet-name>

<servlet-class>web.HelloServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>helloServlet</servlet-name>

<url-pattern>/sayHi</url-pattern>

</servlet-mapping>

</web-app>

**步骤四：部署**

将整个firstweb文件夹拷贝到c:\java\tomcat7.0\webapps

**步骤五：启动Tomcat，并访问Servlet**

在命令提示符下输入“startup”命令启动Tomcat，启动成功后，打开浏览器，在地址栏中输入 “ http://localhost:8080/firstweb/sayHi “回车，得到第一个Web应用程序的运行结果。如图 – 13所示。



图- 13

**1.4. 常见错误及解决方法**

**1.4.1. 404错误产生原因及解决方法**

在刚开始进行Web应用开发的时候，经常看见页面出现404这个数字，我们一般会称之为运行产生了404错误。类似于404这个数字，还有可能在页面上看到405、500这两个数字，他们都是服务器执行完客户端的请求以后，返回给客户端的一个关于执行结果的状态编码说明。 如果在运行结果页面中没有看到期待的页面，却看到了404、405、500这样的数字，那么代表着服务器告诉客户端运行产生了错误，掌握何种错误情况产生对应的数字将有利于问题的解决。

404产生的原因为Web服务器（容器）根据请求地址找不到对应资源，以下情况都会出现404的错误提示：

1. 输入的地址有误（应用名大小写不正确，名称拼写不正确）
2. 在web.xml文件中<servlet>和<servlet-mapping>中的两个<servlet-name>不一致
3. 没有将工程部署到Tomcat的webapps下面
4. Web工程部署时的结构没有遵循Servlet的规范

具体的解决办法就是根据上面提到的4种情况，逐条进行检查。

**1.4.2. 容器如何找到service（）方法**

当在浏览器中输入 http://localhost:8080/firstweb/sayHi 这个地址后，容器是如何找到 HelloServlet.class这个文件并执行的呢？

首先容器会根据firstweb这个应用名找到位于webapps下面对应的文件夹，然后根据地址中的“/sayHi”到web.xml文件中寻找与之匹配的<url-pattern>节点，找到匹配的节点后会找到与该节点紧邻的<servlet-name>节点，获取名称并在此寻找与该名称相等的<servlet-name>节点，找到相等的节点后，搜寻该节点下面紧邻的<servlet-class>节点，于是获取到了与该地址相对应的类名，借助于ClassLoader加载该类文件，创建对象并调用service（）方法，客户端即看到了与该地址匹配的运行结果。

**1.4.3. 405错误产生原因及解决方法**

405这个错误的产生原因是容器找不到service方法来处理请求。以下情况容器都将视为没有找到service（）方法

1. service方法名写错，字母顺序不对或大小写错误
2. service方法参数类型与标准不一致，或参数顺序与标准不一致。一定是先HttpServletRequest，然后是HttpServletResponse类型
3. service方法异常、返回值类型与标准不一致，一定是抛出ServletException、IOException。返回值类型是void。

解决405错误的方法即检查service方法是否存在，签名（方法名、参数、返回值、异常类型）是否与覆盖的父类中的方法一致。

**1.4.4. 500错误产生原因及解决方法**

500的错误原因是请求的Servlet在执行service方法的过程中出现了错误，程序不能继续运行了。以下情况容器都将视为500的一种情况，而返回给客户端这个错误说明。

1. Servlet没有继承HttpServlet或实现Servlet接口
2. web.xml文件中的<servlet-class>写错，与真正的包名类名不符
3. service方法中的代码运行时抛出异常

解决500的方法为依据上面三种情况依次进行检测，逐条排除。