# 1. Web开发概念

之前学习Java SE时，编写的程序是本地应用程序，符合C/S结构。Web开发就是开发网页应用程序，网页应用不仅包含用于展现网页内容和效果的HTML页面、CSS样式和JS脚本，还包含应用后端的具体功能的实现。而采用Java EE就可编写后端程序，属于B/S结构。

学习Web开发，必须先掌握Java SE知识和基本的前端知识。

## 1.1 服务器和Web服务软件

服务器就是在网络上提供服务的设备，如一台配置较好的计算机就能在网络上提供服务。

Web服务软件运行在服务器上，它的作用就是把本地的资源共享给外部访问。比如有了Web服务软件，用户就能访问服务器上的网页和文件等资源。

进行Java Web开发，也需要用合适的Web服务软件，来将我们开发好的网页应用提供给客户使用。目前常用的Java Web服务器软件有：

（1）Tomcat：Apache组织的产品，开源免费，支持大部分Java EE规范，可满足中小型应用的开发。（Java EE就是用Java开发企业级应用的一系列规范。）

（2）Jetty：可作为一个嵌入的服务器。

（3）JBoss：Redhat公司的产品。

另外，还有Resin、WebLogic和WebSphere等适合Java EE的产品。在目前学习阶段，使用的是Tomcat。

实际上学习了网络编程后，完全可以用Java SE自行实现一个简单的Web服务器软件，比如我们使用TCP协议监听本机8080端口，当有请求过来时，就用流输出一些数据。那么我们使用浏览器访问该端口时，就会得到这些数据。（必须使用可靠的TCP协议，这是显而易见的）

# 2. Tomcat服务器

开发Web应用，首先要在本机搭建一个服务器，以便在本机部署、运行和查看开发的Web应用。我们选用Tomcat。

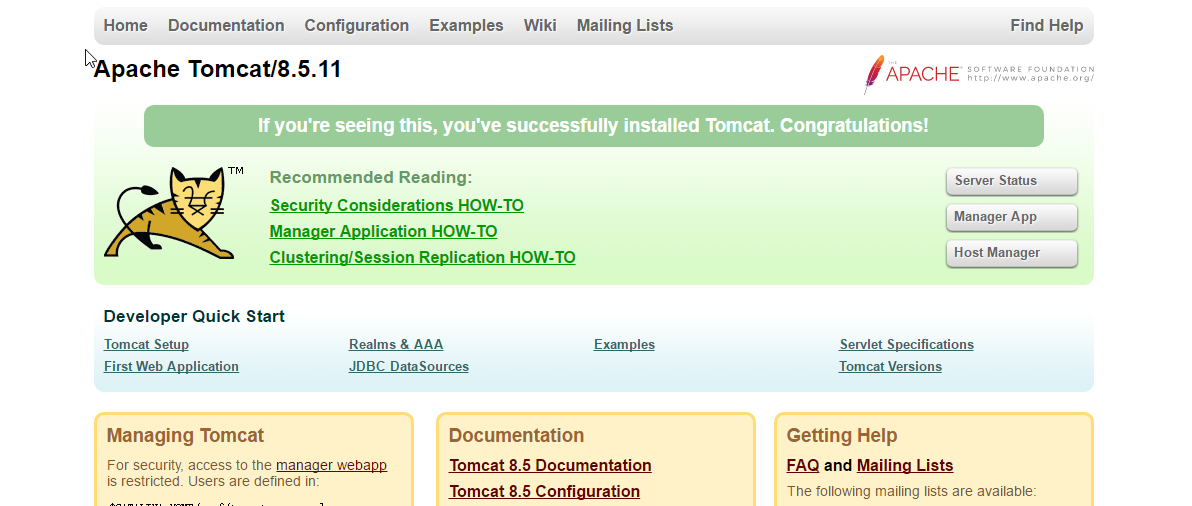
## 2.1 安装和运行

首先到apache的tomcat官网http://tomcat.apache.org/ 下载最新稳定版本的安装包。我们选择Download中的Tomcat 8，目前是8.5.11版本。在Core里面有很多版本，前两个压缩包是各平台通用的，后三个使用于Windows平台。我们选择windows-x64.zip压缩包即可。若项目在Windows平台上投入运行时，可选择最后一个安装版本，方便使用。

下载完后，直接将文件解压到某个目录中，这个自行确定。这时就完成了“安装”。我们进入Tomcat安装目录下的bin目录，里面提供了一些程序。

双击运行startup.bat即可运行Tomcat服务器软件，注意运行时，不要关闭此窗口。若想关闭Tomcat服务器，则双击运行shutdown.bat即可。

开启Tomcat后，可在浏览器中访问http://localhost:8080地址以验证Tomcat服务开启成功。若开启成功，访问此地址会顺利进入Tomcat页面。即：



因为Tomcat默认监听的端口就是本机的8080端口。

常见的Tomcat不能启动的问题：

（1）Tomcat基于Java开发，因此要确保本机Java环境安装正确，否则就会闪退。

（2）端口占用错误：8080端口可能已经被其他应用占用了。方法一就是找到占用端口的进程，杀掉此进程。方法二是修改Tomcat默认的端口：打开Tomcat安装目录下conf目录下的server.xml文件，修改Connector标签中的port属性，比如改为8090。这样就能修改Tomcat监听端口为8090，启动后，访问地址也为localhost:8090。

（3）查看系统中是否有CATALINA\_HOME这个环境变量，如果有，删除它。因为该变量的作用是设置Tomcat的根目录。如果之前安装过tomcat的Windows安装版程序，该程序会自动设置好该环境变量。这样此时启动Tomcat时，仍然会到CATALINA\_HOME设置的目录中启动Tomcat，因此导致启动不成功。不设置该变量的话，默认就启动当前目录下的tomcat。

## 2.2 Tomcat的目录结构

bin：存放可执行命令。

conf：存放配置文件；其中server.xml是核心配置文件。

lib：存放Tomcat服务器的核心类库。如果需要扩展Tomcat的功能，也可以将第三方类库复制到该路径下。

logs：存放Tomcat产生的日志文件。

temp：用于存放临时文件。

webapps：该路径用于部署Web应用。将Web应用复制在该路径下，Tomcat就会自动部署该应用。应用资源不能直接放在webapps目录下，必须在webapps中新建一个文件夹，将资源放入文件夹中，访问时要带上该文件夹名。比如将1.html放在webapps/mytest下，则访问路径是：localhost:8080/mytest/1.html。

work：tomcat的运行目录。JSP运行时产生的临时文件就存放在这里。

## 2.3 体验Tomcat的作用

我们先写一个静态HTML页面，比如叫index.html。现在目的是通过Tomcat发布此页面，让本机浏览器和本机所在局域网中的其他人能访问该页面。步骤如下：

（1）应用要放在webapps目录下，一定要新建一个文件夹再将这些资源放进去。该文件夹就相当于该应用的“应用名”。比如新建的文件夹是exp。

（2）开启Tomcat服务器，访问地址：[http://localhost:8080/exp/index.html 即可。如果你改过端口号，还要把8080改成你设置的端口。如果局域网其他人想访问此页面，只要将localhost换成你的机器的IP地址即可。](http://localhost:8080/exp/index.html即可。如果你改过端口号，还要把8080改成你设置的端口。如果局域网其他人想访问此页面，只要将localhost换成你的机器的IP地址即可。)

实际上浏览器访问的流程如下：

浏览器先找到系统的hosts文件，看里面有没有域名对应的ip地址。比如hosts里面定义了localhost的ip地址是127.0.0.1，localhost是本地域名。如果有，则直接按照hosts里对应的IP直接访问，如果没有，再向外面请求域名服务器DNS。

# 3. 动态资源开发技术：Servlet

动态资源就是用程序的来控制输出不同内容的HTML页面。Servlet是用Java语言开发动态网站的技术。比如一个页面将用户的登录信息发送给后端Servlet，Servlet中有Java代码连接数据库验证用户信息的合法性，最后根据结果返回不同内容的HTML。这些工作就是后端来做的。

## 3.1 第一个Servlet程序

Oracle只提供了Servlet的规范，具体的由各个服务器厂商实现。Tomcat的lib目录的servlet-api.jar就是实现Servlet的jar包。现在我们在项目中引入此包，完全可以在Java SE项目下开发第一个Servlet程序。

引入jar包后，新建一个Servlet类，名叫MyServlet，要求Servlet继承自HttpServlet。Servlet中可以重写doGet()和doPost()方法，分别用于处理网页的GET和POST请求。示例如下：

|  |
| --- |
| **public class** MyServlet **extends** HttpServlet {  @Override  **protected void** doGet(HttpServletRequest req, HttpServletResponse resp) {    }   @Override  **protected void** doPost(HttpServletRequest req, HttpServletResponse resp) {    } } |

由于Servlet是在Tomcat服务器中运行的，因此并不要写main主方法。方法中有两个参数，分别是HttpServletRequest和HttpServletResponse对象，其中就用于封装网络请求和响应。

现在我们正式写好第一个Servlet类：

|  |
| --- |
| **package** com.zhang.test;  **import** javax.servlet.http.HttpServlet; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** java.io.IOException; **import** java.util.Date;  **public class** MyServlet **extends** HttpServlet {  @Override  **protected void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** IOException {  *// 现在只处理GET请求  // 用response对象设置返回的响应数据   // 设置响应编码为utf-8，防止中文乱码* response.setContentType(**"text/html;charset=UTF-8"**);  *// 想浏览器输出内容* response.getWriter().write(**"你好，当前时间为："** + **new** Date());  } } |

该Servlet处理类返回了当前服务器时间。现在，需要把该类放到Tomcat中运行，并能接收浏览器请求返回相应数据。步骤如下：

（1）首先在webapps目录下新建一个文件夹，比如叫DynamicWeb，即我们的“网站”就叫DynamicWeb，然后我们需要将资源全部放在此目录下。

（2）Java Web应用的资源是有规范的，首先应该在DynamicWeb下创建一个WEB-INF文件夹，该文件夹下创建一个web.xml文件和一个classes目录。该web.xml用于配置该网络应用，而classes目录下专门用于存放Java代码生成的字节码文件。

（3）首先拿到我们MyServlet的字节码文件MyServlet.class，因为IDE会自动编译保存的Java文件，所以在IDE中很容易拿到。然后将MyServlet.class放入classes目录下，注意的是，由于MyServlet类是在包com.zhang.test中的，因此必须要在classes下新建com/zhang/test目录，然后把字节码放入test目录中，否则无效。

（4）配置web.xml文件，主要就是配置该Servlet的访问路径（映射）。例子：

|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"***?> <!-- Schema约束 -->* <**web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee"  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"  id="WebApp\_ID" version="3.1"**>  *<!-- servlet标签用于配置程序中的一个servlet -->* <**servlet**>  *<!-- 定义该Servlet的名称 -->* <**servlet-name**>MyFirstServlet</**servlet-name**>  *<!-- 指定Servlet所在的类 -->* <**servlet-class**>com.zhang.test.MyServlet</**servlet-class**>  </**servlet**>  *<!-- servlet-mapping用于配置一个Servlet的访问路径 -->* <**servlet-mapping**>  *<!-- servlet的名称，和上面的一致 -->* <**servlet-name**>MyFirstServlet</**servlet-name**>  *<!-- 指定该servlet的访问路径，就是浏览器访问的路径。一定要写上斜杠 -->* <**url-pattern**>/hello</**url-pattern**>  </**servlet-mapping**> </**web-app**> |

web.xml中指明了可以用/hello路径来访问此Servlet。

（5）最后，我们启动Tomcat，在浏览器中输入：localhost:8080/DynamicWeb/hello 即可得到响应结果。至此我们完成了第一个Servlet的开发。

## 3.2 在Eclipse中开发Servlet

### 3.2.1 在Eclipse配置Tomcat

上述的过程只是为了简单了解Servlet是如何运作的，当然实际开发中不可能这么做。IDE中对Java Web开发提供了很好的支持，能够提升开发效率。下面讲如何在Eclipse中进行开发。当然，你选择的版本一定要是Eclipse IDE For Java EE Developers版本，而不是“For SE”版本。

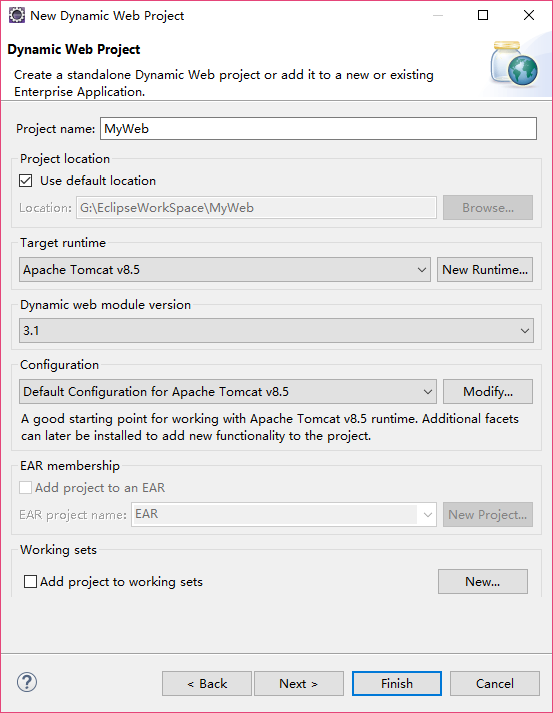
首先在Eclipse中配置好Tomcat。点击Window -> Proferences ，选择左边树状菜单中Server中的“Runtime Environments”一项。点击右侧界面的“Add...”按钮，选择Apache下的对应的Tomcat版本（这里选择v8.5），随后点击“Next”，浏览选择我们Tomcat的安装目录。最后Finish完成。

随后在Eclipse中新建一个Tomcat服务器实例。点击Window -> Show View -> Servers，显示服务器选项卡。点击Servers中的蓝色链接新建一个服务器，就选择新建一个Tomcat v8.5，其他保持默认，点击Finish完成。

最后，双击新建完成的服务器，进行如下配置：在Server Locations中选择“Use Tomcat installation”一项，并且修改Deploy Path为webapps。这样才算完成Tomcat配置。如果不进行最后这项配置，也能运行项目，但是默认不会把项目部署到Tomcat的安装路径。为了直观就这样配置，方便查看。如果以后采用了Maven管理项目，就无所谓了。

### 3.2.2 新建Java Web项目

首先新建一个Project，选择Web下的Dynamic Web Project，点击Next，输入项目名，下面各项选择和Tomcat有关的最新版本，例如：



点击Next，不变，再点击Next，把“Generate web.xml deployment description”勾选上，这样会自动帮我们创建web.xml文件。此页面的Content directory可以不改，不过以前人们喜欢改成“WebRoot”。最后点击Finish完成。

提示：如果发现点击Finish后，Eclipse卡在“installing dynamic web module facet”，则先断开网络连接，重新新建项目即可。

建好项目以后，发现项目的WebContent文件夹下和我们需要复制到Tomcat目录下的文件结构基本一致。实际上项目运行时，Eclipse会自动把工作空间中WebContent目录下的内容复制到Tomcat对应的我们的应用文件夹中。这里的WebContent就相当于一个“虚拟目录”。这样的话，进行开发部署调试就很方便。因为IDE自动把项目部署到本地服务器上。

WebContent目录中可新建资源文件和文件夹，比如HTML等，因为运行项目时，此目录下资源会被复制到Tomcat/webapps/我们的应用目录下。其中，WEB-INF/lib文件夹中专门用于存放项目中需要使用的第三方jar包。Web项目中，直接把第三方jar包复制到此目录中即可完成jar包导入，是自动的。

新建好项目，即可编写Servlet了。在项目上右键New可直接新建Servlet等，但是不推荐这样用（设置较繁杂且生成垃圾代码多），还是自己新建Class类自行编写配置。

可以按照第一个Servlet示例编写Servlet和web.xml文件。编写完毕后，选中Servers选项卡中我们新建好的服务器，右键“Add and Remove...”，把项目添加到右边中，这样就完成部署了，最后右键服务器点击“Start”即启动项目。启动完毕我们即可在浏览器中浏览了。

## 3.3 使用Servlet注解