## Servlet简介

Servlet是sun公司提供的一门用于开发动态web资源的技术。

Sun公司在其API中提供了一个servlet接口，用户若想开发一个动态web资源（即开发一个Java程序向浏览器输出数据），需要完成以下2个步骤：

1. 编写一个Java类，实现servlet接口
2. 把开发好的Java类部署到web服务器中。

按照一种约定俗称的称呼习惯，通常我们也把实现了servlet接口的java程序，称之为Servlet。

## Servlet的运行过程

Servlet程序是由WEB服务器调用，web服务器收到客户端的Servlet访问请求后：

1. Web服务器首先检查是否已经装载并创建了该Servlet的实例对象。如果是，则直接执行第4步，否则执行第2步。
2. 装载并创建改Servlet的一个实例对象。
3. 调用Servlet实例对象的init()方法。
4. 创建一个用于封装HTTP请求消息的HttpServletRequest对象和一个代表HTTP响应消息的HttpServletResponse对象，然后调用Servlet的service()方法并将请求和响应对象作为参数传递进去。
5. WEB应用程序被停止或重新启动之前，Servlet引擎将卸载Servlet，并在卸载之前调用Servlet的destroy()方法。

## 在Eclipse中开发Servlet

在eclipse中新建一个maven project，如图3-1.在Filter中输入web，并选择筛选出的选项，

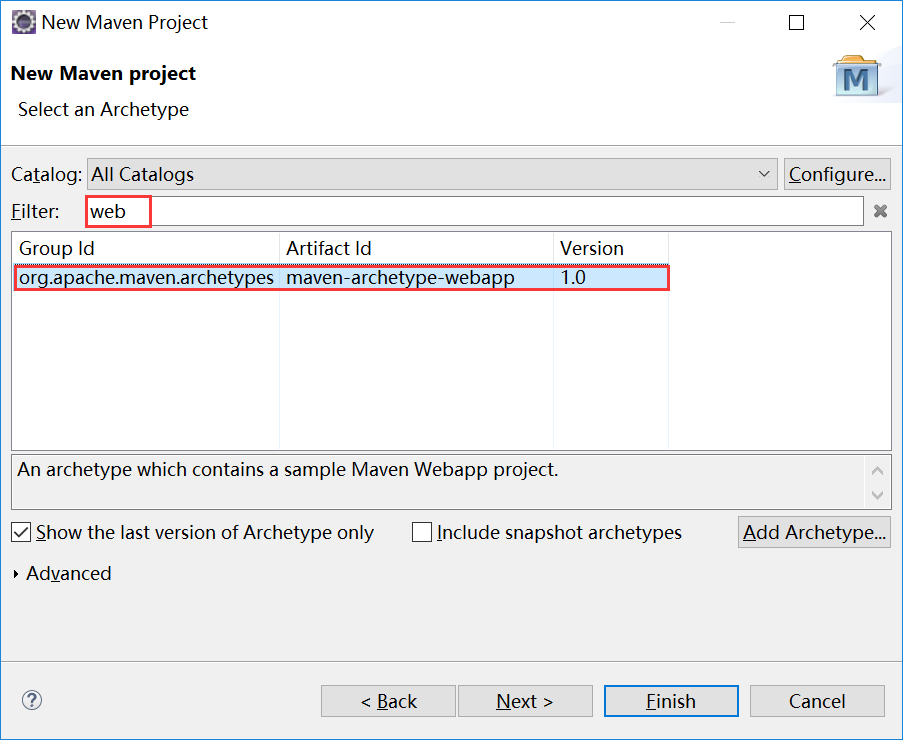


图3-1

下一步，输入Group Id，Artifact Id，Finish

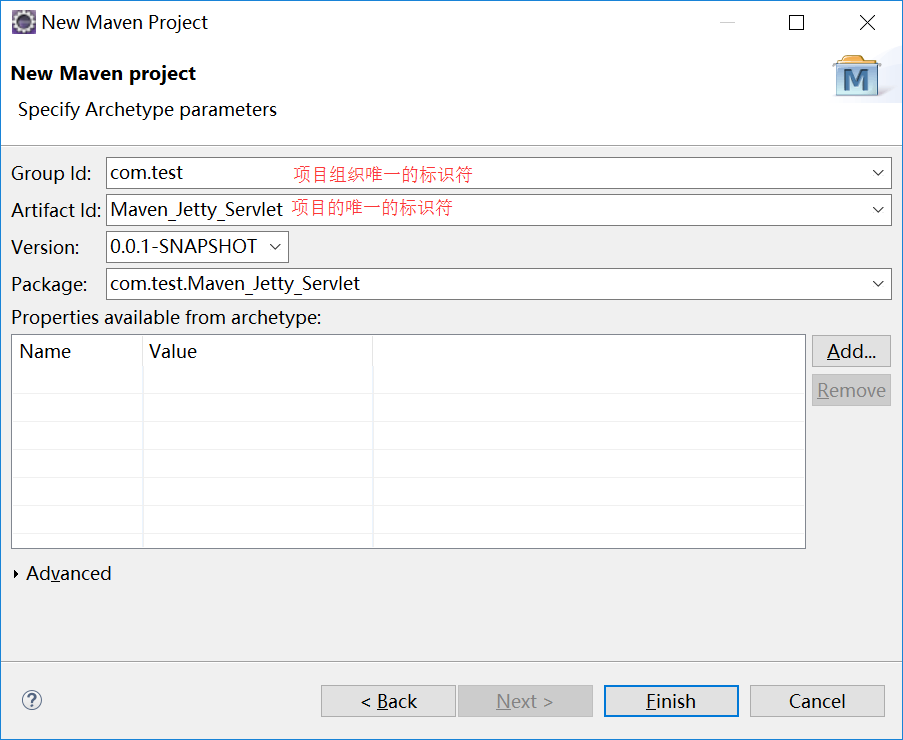
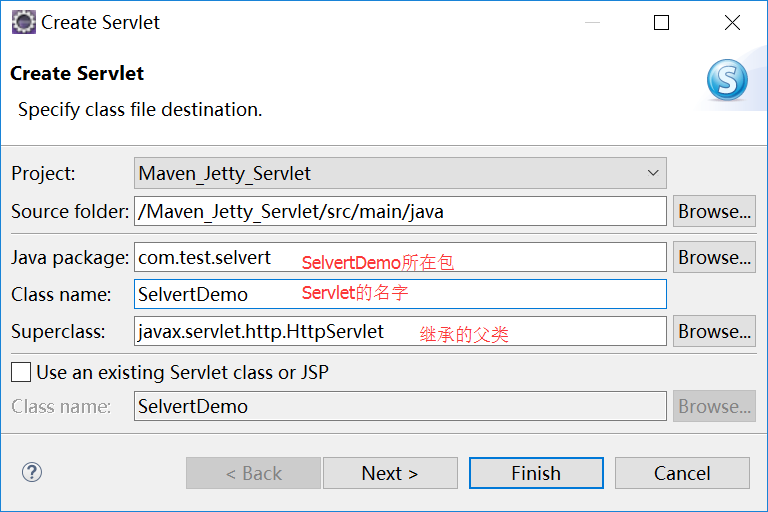


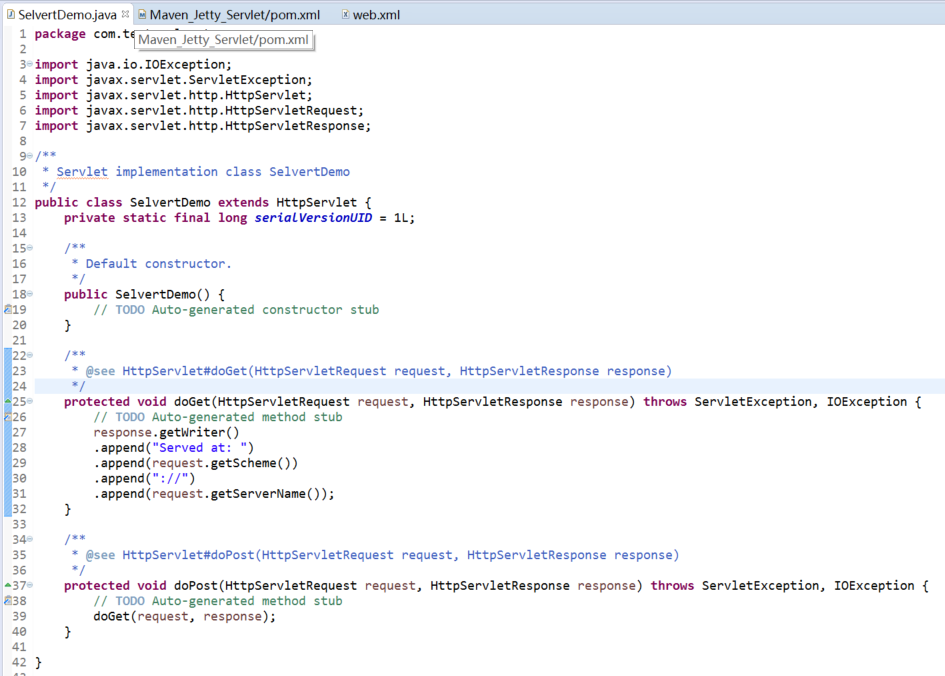
图3-2

eclipse会生成如图3-3所示目录结构：

选中com.test.selvert新建一个Selvert，



生成的代码如图所示（doGet稍作修改），



缺少javax.servlet包需要在pom.xml中加入相对应的依赖：

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

<scope>provided</scope>

</dependency>

## Servlet访问url映射配置

由于客户端是通过URL地址访问web服务器中的资源，所以Servlet程序若想被外界访问，必须把servlet程序映射到一个URL地址上，这个工作在web.xml文件中使用<servlet>元素和<servlet-mapping>元素完成。  
　　<servlet>元素用于注册Servlet，它包含有两个主要的子元素：<servlet-name>和<servlet-class>，分别用于设置Servlet的注册名称和Servlet的完整类名。   
一个<servlet-mapping>元素用于映射一个已注册的Servlet的一个对外访问路径，它包含有两个子元素：<servlet-name>和<url-pattern>，分别用于指定Servlet的注册名称和Servlet的对外访问路径。默认配置：

<servlet>

<servlet-name>SelvertDemo</servlet-name>

<display-name>SelvertDemo</display-name>

<description></description>

<servlet-class>com.test.selvert.SelvertDemo</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SelvertDemo</servlet-name>

<url-pattern>/SelvertDemo</url-pattern>

</servlet-mapping>

同一个Servlet可以被映射到多个URL上，即多个<servlet-mapping>元素的<servlet-name>子元素的设置值可以是同一个Servlet的注册名。 例如：

<servlet>

<servlet-name>SelvertDemo</servlet-name>

<display-name>SelvertDemo</display-name>

<description></description>

<servlet-class>com.test.selvert.SelvertDemo</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SelvertDemo</servlet-name>

<url-pattern>/SelvertDemo</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>SelvertDemo</servlet-name>

<url-pattern>/slevert/SelvertDemo</url-pattern>

</servlet-mapping>

## 用jetty服务器启动web应用

jetty插件安装

首先打开项目的pom.xml文件，然后找到build节点，在其中添加plugins节点，然后添加如下插件：

<plugin>

<groupId>org.mortbay.jetty</groupId>

<artifactId>maven-jetty-plugin</artifactId>

<version>6.1.5</version>

<configuration>

<contextPath>/</contextPath>

<systemProperties>

<systemProperty>

<name>MAVEN\_OPTS</name>

<value>"-Xms512m -Xmx1024m -XX:MaxPermSize=384m"</value>

</systemProperty>

<systemProperty>

<name>JVM-ARG</name>

<value>"-XX:MaxPermSize=386m"</value>

</systemProperty>

<systemProperty>

<name>JAVA\_OPTS</name>

<value>"-Xms512m -Xmx1024m"</value>

</systemProperty>

</systemProperties>

<webDefaultXml>src/main/resources/webdefault.xml</webDefaultXml>

</configuration>

</plugin>

在src/main/resources/目录创建webdefault.xml文件。

## 配置maven私服

在maven项目的pom.xml中配置远程仓库地址，不配置的话，会从国外的服务器下载jar包，非常浪费时间：

<repositories>

<repository>

<id>idohoo-nexus</id>

<name>idohoo-nexus</name>

<url>http://119.29.64.126:8081/nexus/content/groups/public</url>

<layout>default</layout>

<releases>

<updatePolicy>always</updatePolicy>

<checksumPolicy>warn</checksumPolicy>

<enabled>true</enabled>

</releases>

<snapshots>

<updatePolicy>always</updatePolicy>

<checksumPolicy>warn</checksumPolicy>

<enabled>true</enabled>

</snapshots>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>idohoo-nexus</id>

<name>idohoo-nexus</name>

<url>http://119.29.64.126:8081/nexus/content/groups/public</url>

<layout>default</layout>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</pluginRepository>

</pluginRepositories>

## 配置Debug Configurations

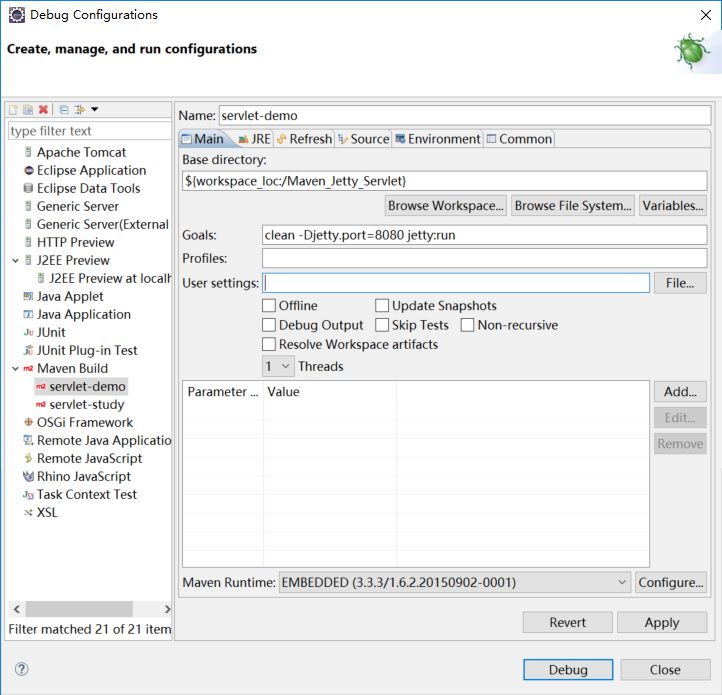
Name：该Debug Configurations的名称

Base directory：该项目路径

Goals：maven命令，这里指定jetty端口：clean -Djetty.port=8080 jetty:run

配置完之后点击Debug，启动该项目

如下图所示：



浏览器访问<http://localhost:8080/SelvertDemo>，如下图：

