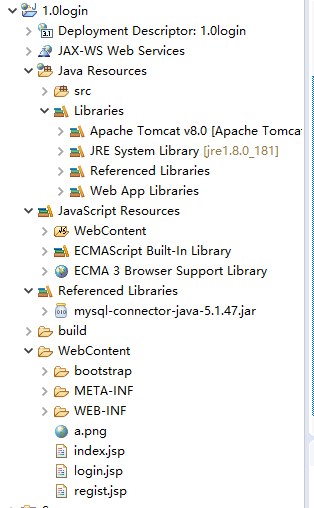
**Eclipse web项目目录结构（来源于网络，可供参考）**

注：此文档说明了eclipse下web项目目录结构的大致意思，可以对web项目有更深一层的理解，方便今后对eclipse的使用。（可能会因为eclipse版本、或者电脑操作系统的不同，目录结构会有少许差异，但是大体内容相似，且将就一看吧！）

刚开始使用eclipse新建web项目默认的目录结构：

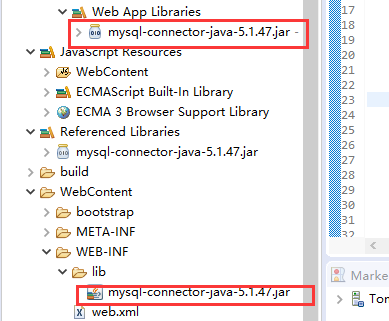


（1）Src： 里面放的是 java 源程序，也就是自己写的 java 代码。

JRE System Library: 指Java SE 的常用库文件集合，也就是 jar 包，建立普通的Java项目均会使用;

（2） Apache Tomcat v8.0： 是指这个项目所依赖的服务器（Tomcat）的目录。

（3）Web App Libraries ：是自己导入的项目依赖 jar 包，Web App Libraries 下的所有 jar 包都可以在本地的项目名\WebContent\WEB-INF\lib 中找到。如果你直接在WEB-INF/lib文件夹下，copy 一个 jar 包，刷新一下，会自动编译到 Web App Library 中。



（4）Referenced Libraries ：是编译环境下使用的 jar 包, 所谓编译环境下使用的 jar 包, 就是说你在 Eclipse 中进行源文件的编写的时候, 所需要引用到的类都从Referenced Libraries 这个集合中的 jar 包中拿。这里导入和配置 jar包，都是为了让 web 工程能够在 eclipse 开发环境可以正常编译，不报错而已。

（5）build：eclipse新建的 Dynamic web project 默认是将类编译在 build 文件夹下。可以在本地的项目名\build\classes下查看。

（6）WebContent：一般我们用 Eclipse 的时候创建一个 Web Project，就会生成WebContent 文件夹。 WebContent 用来存放 JSP，JS，CSS，图片等文件，是项目访问的默认路径，也是工程的发布文件夹，发布时会把该文件夹发布到 tomcat 的webapps 里。

（7）META-INF：存放一些 meta information 相关的文件的这么一个文件夹, 一般来说尽量不要自己手工放置文件到这个文件夹。

（8）WEB-INF：WEB-INF 目录是一个专用区域， 容器不能把此目录中的内容提供给用户。这个目录下的文件只供容器使用，里面包含不应该由客户直接下载的资源。Web 容器要求在你的应用程序中必须有 WEB-INF 目录。WEB-INF 中包含着发布描述符（也就是 web.xml 文件）, 一个 classes 目录和一个 lib目录, 以及其它内容。注意： 如果你的 Web 应用程序中没有包含这个目录, 它可能将无法工作 。

（9）web.xml：发布描述符(deployment descriptors)是 J2EE Web 应用程序不可分割的一部分(也就是说是它的最小部分, 必不可缺的一部分)。它们在应用程序发布之后帮助管理 Web 应用程序的配置。

（10）WEB-INF/classes 目录，编译后的 Java类，这个文件夹是可选的。有一些配置文件需要放到 WEB-INF 的 classes 文件夹下，所以，通常的做法是手动在工程的 WEB-INF 文件夹下建立 classes 文件夹。如果不在工程的 WEB-INF 下手动建立 classes，发布到 tomcat 时 tomcat 里的WEB-INF 中也会有 classes 文件夹。

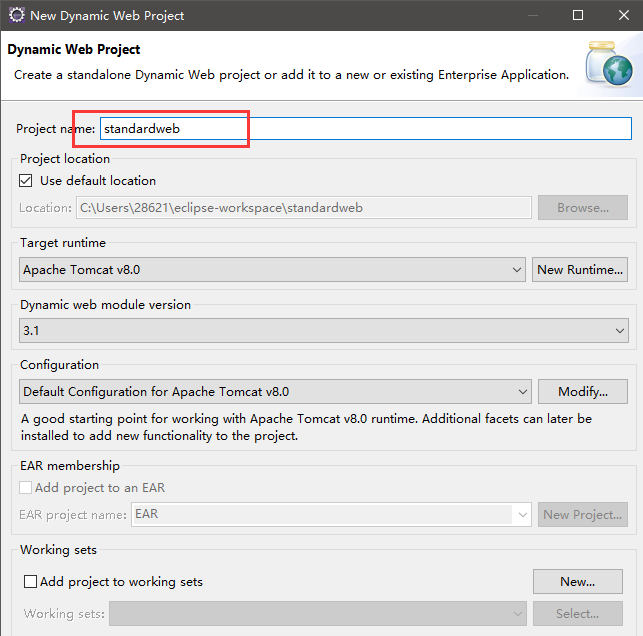
（11）WEB-INF/lib 目录，该目录中的 jar 包是运行时环境下使用的 jar 包, 所谓运行时环境下使用的 jar 包,就是说你在运行你的项目的时候所需要使用的 jar 包的集合。

（12）index.jsp：首页页面

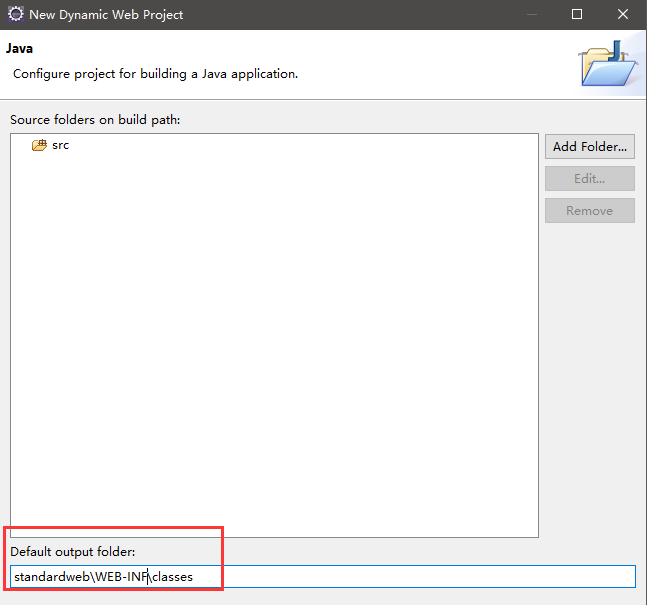
虽然运行没有错，还是有一些瑕疵，标准的web应用一般不采用默认的这种结构，因此总结一下，标准Web应用推荐的目录结构。

**1、如何在Eclipse中创建一个标准的web应用：(假设项目名称为：standardweb)**

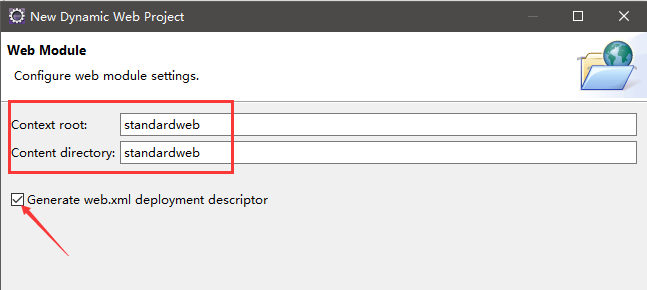
**1)**



PS：projectname一般全小写字母，最好不要出现数字和大写字母。

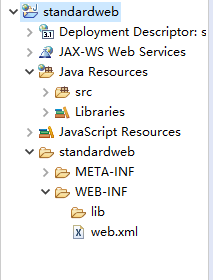
2）

PS：default output flolder要根据project\_name修改。

3）

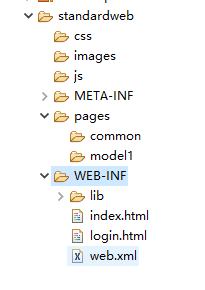
注意：context directory改成项目名，并勾选Generata web.xml deployment descriptor。

4）点击finish后，生成的目录结构如下：



**2、标准Web应用推荐的目录结构。**

**1）web端**



* css：用来存放系统共用的css文件。
* images：用来存放系统共用的图片。
* js：用来存放系统共用的js文件。
* META-INF：用来存放war包元数据信息。
* pages：用来存放业务系统系统的页面文件，比如：html文件，jsp文件，或者JS文件（ExtJS系列）。

       -common：用来存放一些非业务页面，比如top，bottom等。

       -model1：业务模块名称。

* WEB-INF：系统自生成。

        -classes：存放java文件的编译文件

        -lib:存放lib。

        -web.xml：系统自生成，配置web的基础属性。

* index.html：首页页面
* login.html:登陆页面

2）服务器端src文档目录

com.【projectname】

projectName：当前项目的项目名称

com.projectname:

common：用来存放一些共用文件，比如GlobalNames.java等

model1：业务模块 1

       -controller：controller类

        -service：service类

model2：业务模块 2

util：util类。

说明：

1.controller目录和service目录分离的原因（可参考MVC模式分层原理）

controller层：用来编写代码控制层的逻辑，每个方法对应前台一个请求。

Service层：

a.真正的业务层逻辑，颗粒度小，会出现共用模块，实现代码共用。

b.在Service类中控制事务和MySQL。

c.便于实现客户化代码。

2.所有的配置文件放到Src目录中，简化配置难度。