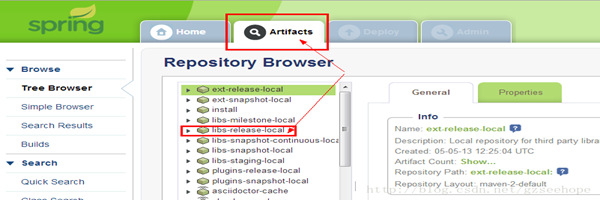
# Spring下载

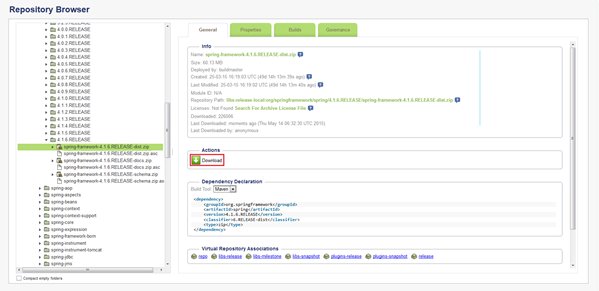
（1）进入官网http://repo.spring.io

（2）在资源库界面点击[Artifacts](http://repo.spring.io/webapp/browserepo.html?3)标签，然后点击**libs-release-local**，展开后依次点击org -> springframework -> spring。



然后，就可以看到发布的各个版本，选一个自己需要的版本，点击**Download**。

（我这里下载的是当前最新版本spring-framework-4.1.6.RELEASE-dist）



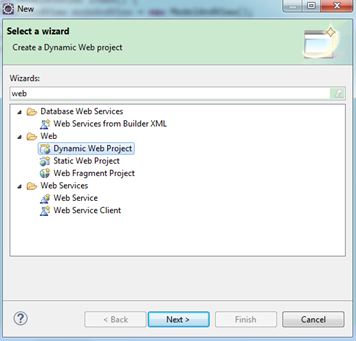
# Hello World

SpringMVC 配置过程中很容易出现各式各样的问题，很是费劲。

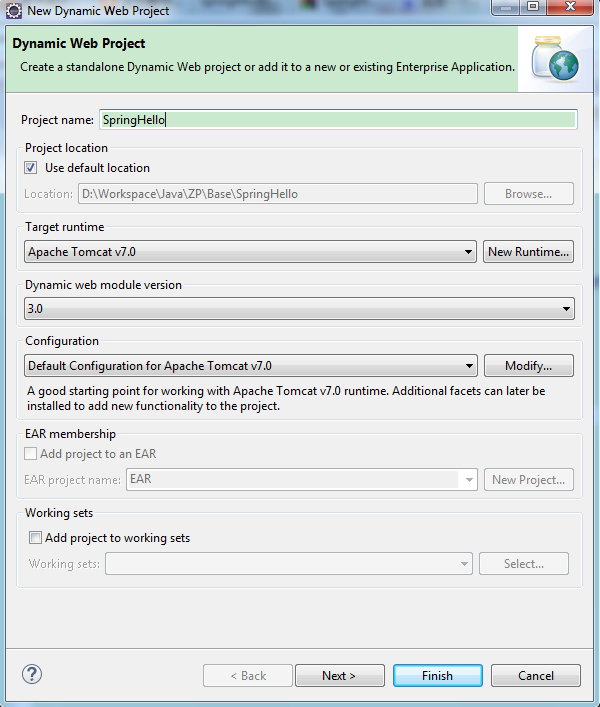
下面提供傻瓜式创建HelloWorld工程的方式，希望大家少走弯路。

**（1）创建一个Dynamic Web Project**

点击File -> New -> Other, 输入web， 可以找到**Dynamic Web Project**。



新建一个名为SpringHello的工程。可参考下图的配置 。



**（2）导入spring的jar包**

在WEB-INF目录下新建一个lib目录（如果没有lib目录）。

将下载的**spring-framework-4.1.6.RELEASE-dist\spring-framework-4.1.6.RELEASE\libs**中的jar包拷贝到**WEB-INF\lib** 目录下

为图省事，本人将所有jar都拷贝进来了。

此外，spring框架还依赖一个外部jar包：**commons-logging-1.2.jar**，需要下载并导入。

**注**：顺便提一下，spring-framework-4.1.6.RELEASE-dist\spring-framework-4.1.6.RELEASE\libs下的jar包分为普通jar包、sources jar包和javadoc jar包。sources 和javadoc分别包含了普通jar包的源码和java文档。

有兴趣想看spring源码的朋友可以导入，导入方法网上有很多介绍，这里不赘述。

**（3）添加web.xml**

web.xml文件中声明了servlet的清单。

|  |
| --- |
| <?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"*>  <display-name>SpringHello</display-name>  <!-- 声明前端控制器 -->  <servlet>  <servlet-name>spring</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <!-- 如果不设置的话，默认的servlet配置文件名为xxx-servlet.xml -->  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/spring.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <!-- 声明DispatcherServlet处理哪些URL -->  <servlet-mapping>  <servlet-name>spring</servlet-name>  <url-pattern>/</url-pattern><!-- 映射到/表示会处理所有的请求 -->  </servlet-mapping>  </web-app> |

SpringMVC的核心是**DispatcherServlet**，这个servlet的角色是前端控制器。所有的servlet都是在**web.xml**中配置的，DispatcherServlet当然也不例外。

如果不指定**contextConfigLocation**，默认情况下servlet配置文件的名字形式应该为**servlet名-servlet.xml**, 这个servlet名就是<servlet-name>标签中的名字。

**（4）添加servlet——spring.xml**

新建一个名为spring.xml的文件（这个文件名必须和web.xml中的servlet对应上），内容如下：

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans  xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns:p=*"http://www.springframework.org/schema/p"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xmlns:util=*"http://www.springframework.org/schema/util"*  xmlns:mvc=*"http://www.springframework.org/schema/mvc"*  xsi:schemaLocation=*"*  *http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-3.0.xsd*  *http://www.springframework.org/schema/context*  *http://www.springframework.org/schema/context/spring-context-3.0.xsd*  *http://www.springframework.org/schema/util*  *http://www.springframework.org/schema/util/spring-util-3.0.xsd*  *http://www.springframework.org/schema/mvc*  *http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd"*>  <!-- 默认的注解映射的支持 -->  <mvc:annotation-driven />  <!-- 如果当前请求为“/”时，则转发到“/login/index” -->  <mvc:view-controller path=*"/"* view-name=*"forward:/login/index"* />  <!-- 设置后，将查找使用@Component(@Controller是它的一种具体化)注解的类并将其注册为Bean -->  <context:component-scan base-package=*"com.demo.web.controllers"* />  <!-- 配置视图解析器 (该配置的是jstl解析器的一个扩展解析类) -->  <bean class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>  <property name=*"prefix"* value=*"/WEB-INF/view/"* />  <property name=*"suffix"* value=*".jsp"* />  </bean>  </beans> |

**（5）添加Controller——LoginController.java**

Controller是SpringMVC的控制器，用于处理页面请求。

新建一个名为LoginController的java文件，内容如下：

|  |
| --- |
| @Controller  **public** **class** LoginController {  @RequestMapping(value = "/login/index", method = RequestMethod.***GET***)  **public** ModelAndView index() {  ModelAndView mav = **new** ModelAndView();  mav.addObject("message", "Hello World! First program!");  mav.setViewName("index"); // 设置视图名称  **return** mav;  }  } |

Spring2.5版本引入了注解方式，所以，我们可以很方便的使用@Controller来将一个类标注为控制器。

在spring.xml中，我们定义了<context:component-scan>标签，这样LoginController会自动被发现并注册为Bean。需要注意的是pacake要相互对应。

@RequestMapping可以将一个URL映射到一个实体类或一个特殊的handler方法上。RequestMethod.***GET***表示只接受get请求。

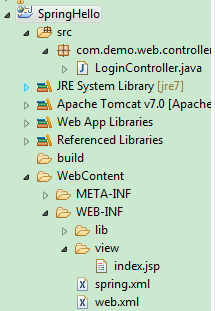
**（6）添加视图——index.jsp**

在/WEB-INF文件夹下新建一个view文件夹，并添加一个名为index的jsp文件，内容如下：

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>  <title>Spring Hello World</title>  </head>  <body>  <h1>${message}</h1>  </body>  </html> |

**（7）运行项目**

经过上面的步骤，项目文件结构如下



运行项目，选择Run As -> Run On Server结果如下:



# 参考资料

Spring实战（第3版）

http://www.cnblogs.com/liukemng/p/3725582.html