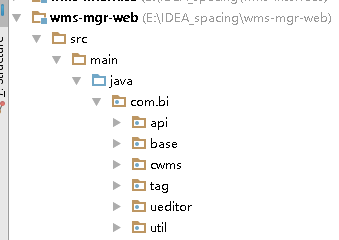
一.创建maven项目

1. GroupID是项目组织唯一的标识符，实际对应**[Java](http://lib.csdn.net/base/java" \o "Java 知识库" \t "http://blog.csdn.net/limm33/article/details/_blank)**的包的结构是main目录里java后面的目录结构。
2. ArtifactID就是项目的唯一的标识符，实际对应项目的名称，就是项目根目录的名称。

如下所示:



在pom.xml中也就是

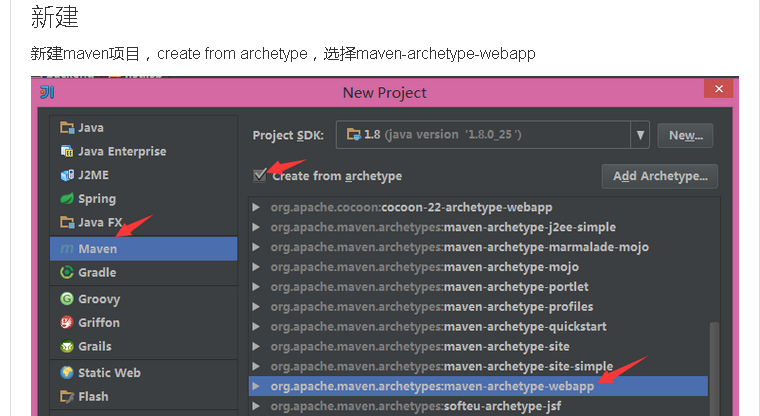


其中,groupId对应main/java下的目录结构

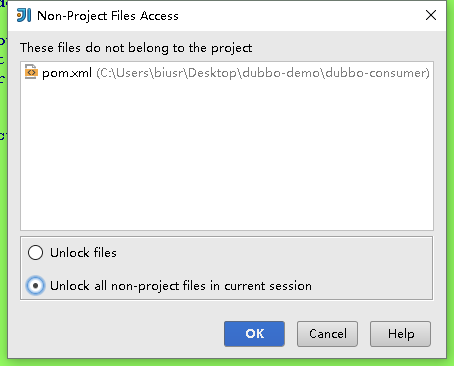
artifactId对应本工程名

注意：pom作为项目对象模型。通过xml表示maven项目，使用pom.xml来实现。主要描述了项目：包括配置文件；开发者需要遵循的规则，缺陷管理系统，组织和licenses，项目的url，项目的依赖性，以及其他所有的项目相关因素

3.使用idea创建一个web工程,如下所示:

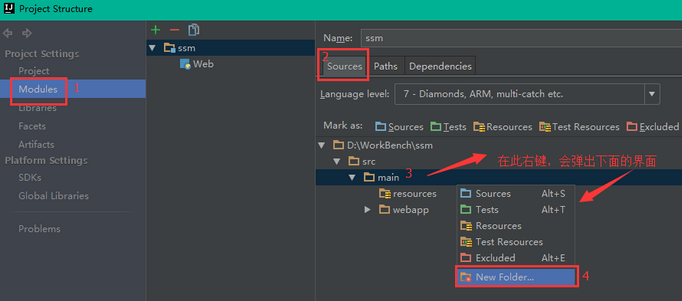


最后一步:出现这里选最下面的

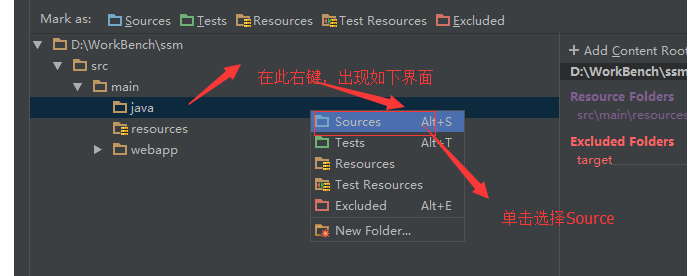


4.[解决Idea创建maven-archetype-webapp项目无java目录的问题](http://blog.csdn.net/u014756827/article/details/52497572)

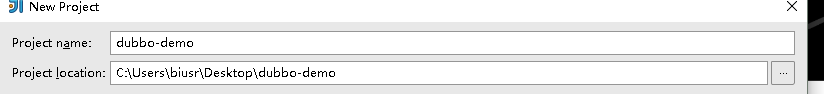
1. 选择File->Project Structure...
2. 选择Modules选项卡下面的Sources项，在main文件夹上右键，选择New Folder...并点击OK



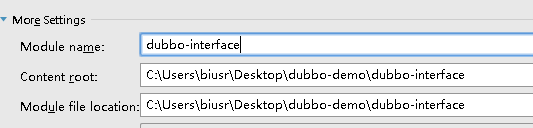
(3)在创建好的java文件夹上右键选择Sources项将该文件夹标记为源文件夹



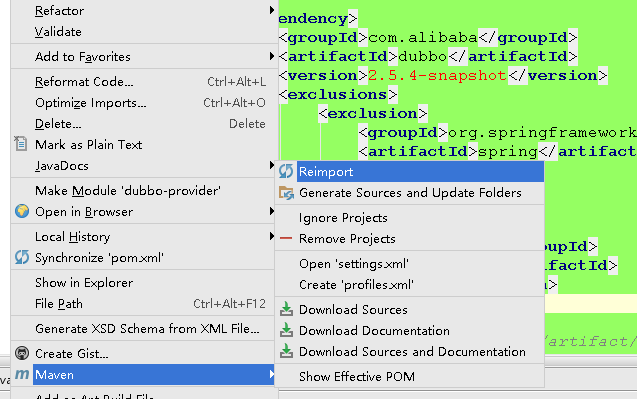
5.新建项目时,project name为总的项目名,也就是工作空间所在位置



Module name为每一个分项目名



6.向pom.xml添加jar包依赖的时候,如果报红,则



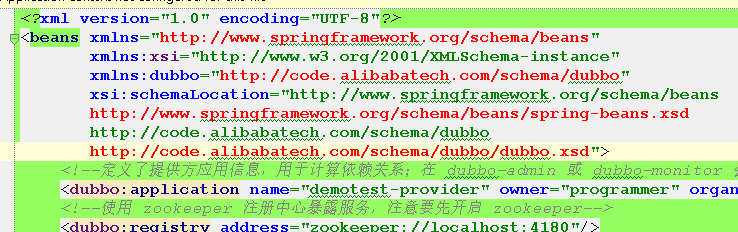
7.启动Main方法的时候报错:

**c: 通配符的匹配很全面, 但无法找到元素 'dubbo:application' 的声明。**

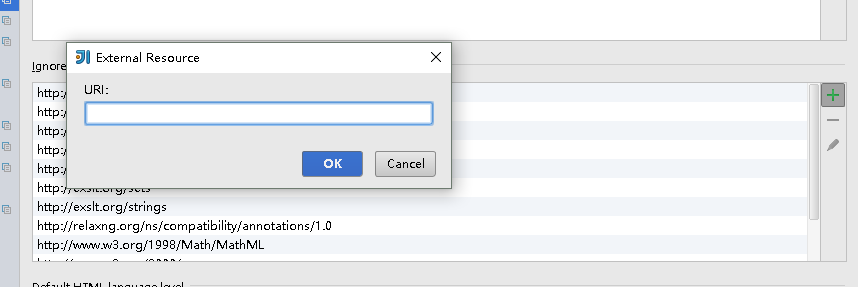
**--解决方法为:idea工具clean一下,然后maven--reimport就可以(在这之前要导入需要的schema和jar),每当导入完成,就要reimport一下**

同上6

8.Idea头文件报红如何解决:



File-settings-schema and dtd



9.idea介绍

Project 和 Module 介绍

这两个概念是 IntelliJ IDEA 的必懂知识点之一，请务必要学会。

如 果你是 Eclipse 用户，那 IntelliJ IDEA 首先告诉你一个非常重要的事情：IntelliJ IDEA 没有类似 Eclipse 工作空间（workspace）的概念的。很多从 Eclipse 转过来的人总是下意识地要再同一个窗口管理 n 个项目，这在 IntelliJ IDEA 是无法得到。IntelliJ IDEA 提供的体验是：一个 Project 打开一个 Window 窗口。

通 过上面的介绍我们知道，在 IntelliJ IDEA 中 Project 是最顶级的级别，次级别是 Module。一个 Project 可以有多个 Module。目前主流的大型项目结构都是类似这种多 Module 结构，这类项目一般是这样划分的，比如：core Module、web Module、plugin Module、solr Module 等等，模块之间彼此可以相互依赖。通过这些 Module 的命名也可以看出，他们之间应该都是处于同一个项目业务情况下的模块，彼此之间是有不可分割的业务关系的。

所以我们现在总结：一个 Project 是由一个或多个 Module 组成，模块之间尽量是处在同一个项目业务的的情况下，彼此之间互相依赖关联。这里用的是 尽量，因为 IntelliJ IDEA 的 Project 是一个没有具备任何编码设置、构建等开发功能的，主要起到一个项目定义、范围约束、规范等类型的效果，也许我们可以简单地理解为就是一个单纯的目录，只是这个目录命名上必须有其代表性的意义。

总结:

　　IntelliJ系中的 Project 相当于Eclipse系中的 Workspace；  
　　IntelliJ系中的 Module 相当于Eclipse系中的 Project；  
　　IntelliJ中一个 Project 可以包括多个 Module；  
　　Eclipse中一个 Workspace 可以包括多个 Project

1. 具体步骤
2. Interface

|  |
| --- |
| (1)接口  public interface GreetingService {  String hello(String name);  } |

1. Provider

|  |
| --- |
| (1)实现类  public class GreetingServiceImpl implements GreetingService {  public String hello(String name) {  System.out.println("Hello Service is calling : " + new Date());  String greetMessage = "Hello, " + name;  return greetMessage;  }  }   1. 启动类   public class Main {  @SuppressWarnings("resource")  public static void main(String[] args) throws IOException {  ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext(  "applicationContext.xml");  context.start();  System.in.read();  }  }   1. pom.xml(属于maven)   <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <spring.framework.version>4.3.6.RELEASE</spring.framework.version>  </properties>  <dependencies>  <dependency>  <groupId>com.xxx.dubbo.demo</groupId>  <artifactId>dubbo-demo-interface</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <!--引入Spring依赖包 -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>${spring.framework.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>${spring.framework.version}</version>  </dependency>  <!-- Dubbo依赖包 -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>dubbo</artifactId>  <version>2.5.4-snapshot</version>  <exclusions>  <exclusion>  <groupId>org.springframework</groupId>  <artifactId>spring</artifactId>  </exclusion>  </exclusions>  </dependency>  <dependency>  <groupId>org.javassist</groupId>  <artifactId>javassist</artifactId>  <version>3.18.1-GA</version>  </dependency>  <!-- https://mvnrepository.com/artifact/io.netty/netty -->  <dependency>  <groupId>io.netty</groupId>  <artifactId>netty</artifactId>  <version>3.10.6.Final</version>  </dependency>  <!-- ZK-client(zookeeper包) -->  <dependency>  <groupId>com.github.sgroschupf</groupId>  <artifactId>zkclient</artifactId>  <version>0.1</version>  </dependency>  </dependencies>   1. spring-dubbo-provider.xml(属于dubbo)   <?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:dubbo="http://code.alibabatech.com/schema/dubbo"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  ">  <description>Dubbo Demo Service</description>  <!-- 提供方应用信息，用于计算依赖关系 -->  <dubbo:application name="dubbo-demo-provider" />  <!-- 使用zookeeper注册中心暴露服务地址 -->  <dubbo:registry protocol="zookeeper" client="zkclient" address="zookeeper://localhost:2181"/>  <!-- dubbo缓存 -->  <dubbo:protocol id="dubbo" name="dubbo" port="20880"  threadpool="cached" threads="100" />  <!-- Greeting服务 -->  <bean id="greetingService" class="com.xxx.dubbo.demo.service.impl.GreetingServiceImpl" />  <dubbo:service protocol="dubbo"  interface="com.xxx.dubbo.demo.service.GreetingService" ref="greetingService/>  </beans>   1. applicationContext.xml   <?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"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd  ">  <!--服务层配置 -->  <import resource="classpath:spring-dubbo-provider.xml" /></beans>   1. log4j.properties   # Rules reminder: # DEBUG < INFO < WARN < ERROR < FATAL  # Global logging configuration log4j.rootLogger=INFO,stdout  ## Console output... log4j.appender.stdout=org.apache.log4j.ConsoleAppender log4j.appender.stdout.layout=org.apache.log4j.PatternLayout log4j.appender.stdout.layout.ConversionPattern=[%-5p][%d{yyyy-MM-dd HH:mm:ss,SSS}] %l - %m%n |

1. Consumer

|  |
| --- |
| 1. pom.xml   <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <spring.framework.version>4.3.6.RELEASE</spring.framework.version>  </properties>  <dependencies>  <dependency>  <groupId>com.xxx.dubbo.demo</groupId>  <artifactId>dubbo-demo-interface</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <!--引入Spring依赖包 -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>${spring.framework.version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>${spring.framework.version}</version>  </dependency>  <!-- Dubbo -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>dubbo</artifactId>  <version>2.5.4-snapshot</version>  <exclusions>  <exclusion>  <groupId>org.springframework</groupId>  <artifactId>spring</artifactId>  </exclusion>  </exclusions>  </dependency>  <dependency>  <groupId>org.javassist</groupId>  <artifactId>javassist</artifactId>  <version>3.18.1-GA</version>  </dependency>  <!-- https://mvnrepository.com/artifact/io.netty/netty -->  <dependency>  <groupId>io.netty</groupId>  <artifactId>netty</artifactId>  <version>3.10.6.Final</version>  </dependency>    <!-- ZK-client -->  <dependency>  <groupId>com.github.sgroschupf</groupId>  <artifactId>zkclient</artifactId>  <version>0.1</version>  </dependency>  </dependencies>   1. spring-dubbo-consumer.xml   <?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:dubbo="http://code.alibabatech.com/schema/dubbo"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  ">  <description>Dubbo Demo Service</description>  <!-- 提供方应用信息，用于计算依赖关系 -->  <dubbo:application name="DemoProvider" />  <!-- 使用zookeeper注册中心暴露服务地址 -->  <dubbo:registry protocol="zookeeper" client="zkclient" address="zookeeper://localhost:2181"/>  <!-- dubbo缓存 -->  <dubbo:protocol id="dubbo" name="dubbo" port="20880"  threadpool="cached" threads="100" />  <!-- 生成远程服务代理，可以和本地bean一样使用demoService -->  <dubbo:reference id="greetingService"  interface="com.xxx.dubbo.demo.service.GreetingService" />  </beans>   1. applicationContext.xml   <?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"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd  ">  <!--服务层配置 -->  <import resource="classpath:spring-dubbo-consumer.xml" /></beans> |