## Intellij Maven工程搭建

目录

[1. 新建一个maven](#_Toc28356_WPSOffice_Level1) [1](#_Toc28356_WPSOffice_Level1)

[2. 以上步骤做完后创建的新工程的目录如下](#_Toc11368_WPSOffice_Level1) [3](#_Toc11368_WPSOffice_Level1)

[3. 打开pom.xml修改单元测试版本，要用最新的](#_Toc30389_WPSOffice_Level1) [3](#_Toc30389_WPSOffice_Level1)

[4. 然后配置我们的tomcat](#_Toc7669_WPSOffice_Level1) [4](#_Toc7669_WPSOffice_Level1)

[① 选择Add Configuration](#_Toc11368_WPSOffice_Level2) [5](#_Toc11368_WPSOffice_Level2)

[② 修改配置信息](#_Toc30389_WPSOffice_Level2) [6](#_Toc30389_WPSOffice_Level2)

[③ 点击这个，他在以后的项目里经常要被使用到](#_Toc7669_WPSOffice_Level2) [7](#_Toc7669_WPSOffice_Level2)

[④ 继续配置tomcat](#_Toc25776_WPSOffice_Level2) [8](#_Toc25776_WPSOffice_Level2)

[⑤ 对tomcat进行热效应配置](#_Toc11837_WPSOffice_Level2) [9](#_Toc11837_WPSOffice_Level2)

[⑥ tomcat配置完成，然后点击运行测试是否配置完成](#_Toc16073_WPSOffice_Level2) [9](#_Toc16073_WPSOffice_Level2)

[5. 在pom.xml引入依赖架包](#_Toc25776_WPSOffice_Level1) [10](#_Toc25776_WPSOffice_Level1)

[6. 进行web.xml配置](#_Toc11837_WPSOffice_Level1) [12](#_Toc11837_WPSOffice_Level1)

[7. 新建applicationContext.xml](#_Toc16073_WPSOffice_Level1) [13](#_Toc16073_WPSOffice_Level1)

[① 在applicationContext.xml先写入这几行代码](#_Toc12858_WPSOffice_Level2) [17](#_Toc12858_WPSOffice_Level2)

[8. 新建几个包，这个包很重要](#_Toc12858_WPSOffice_Level1) [17](#_Toc12858_WPSOffice_Level1)

[① 新建如下几个包](#_Toc25239_WPSOffice_Level2) [18](#_Toc25239_WPSOffice_Level2)

[9. 补充applicationContext](#_Toc25239_WPSOffice_Level1) [19](#_Toc25239_WPSOffice_Level1)

[① 数据源，事物控制](#_Toc930_WPSOffice_Level2) [19](#_Toc930_WPSOffice_Level2)

[② 事物控制，连接我们的数据库](#_Toc21383_WPSOffice_Level2) [20](#_Toc21383_WPSOffice_Level2)

[③ 配置和mybatis的整合](#_Toc11897_WPSOffice_Level2) [20](#_Toc11897_WPSOffice_Level2)

[④ 配置扫描器，将mybatis接口加入到ioc容器，指定dao文件存放位置](#_Toc21170_WPSOffice_Level2) [22](#_Toc21170_WPSOffice_Level2)

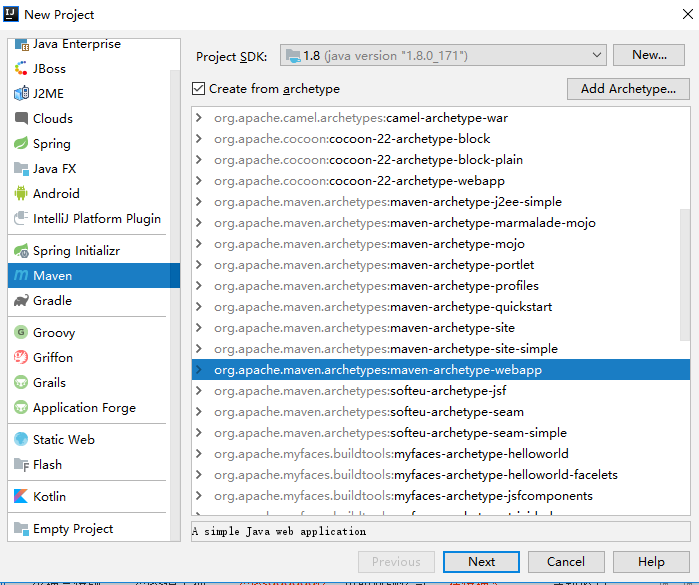
[⑤ 开启基于注释的事物，使用xml配置形式事物](#_Toc18444_WPSOffice_Level2) [22](#_Toc18444_WPSOffice_Level2)

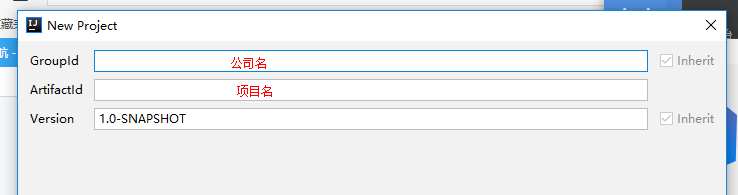
[10. 新建dispatcherServlet-servlet.xml](#_Toc930_WPSOffice_Level1) [23](#_Toc930_WPSOffice_Level1)

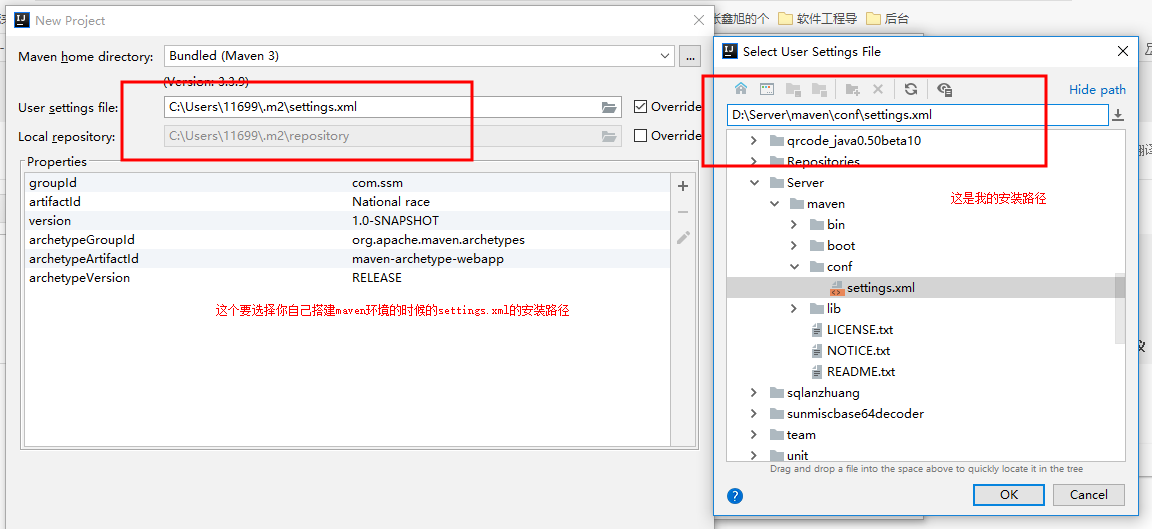
[11. 新建mbg.xml，主要是数据库的反编译](#_Toc21383_WPSOffice_Level1) [24](#_Toc21383_WPSOffice_Level1)

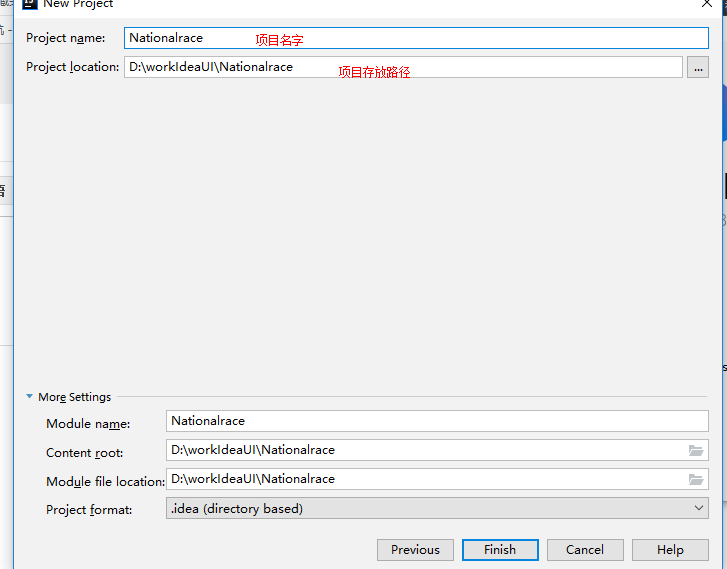
[12. 全部文件](#_Toc11897_WPSOffice_Level1) [25](#_Toc11897_WPSOffice_Level1)

1. 新建一个maven

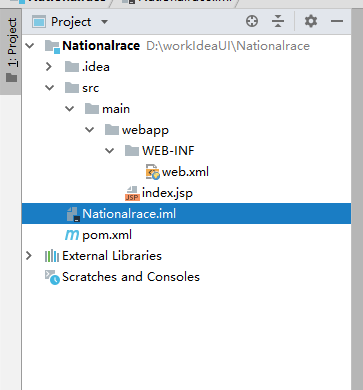








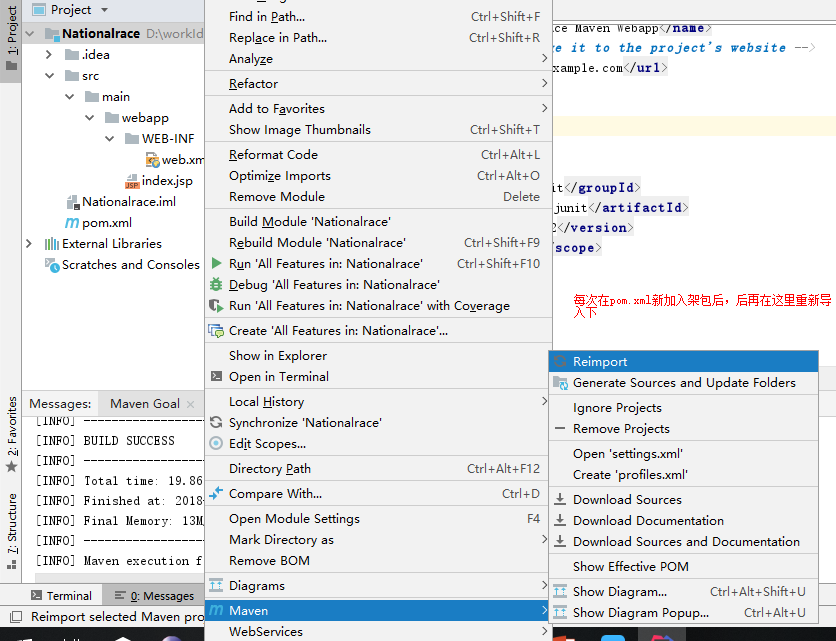
1. 以上步骤做完后创建的新工程的目录如下



1. 打开pom.xml修改单元测试版本，要用最新的



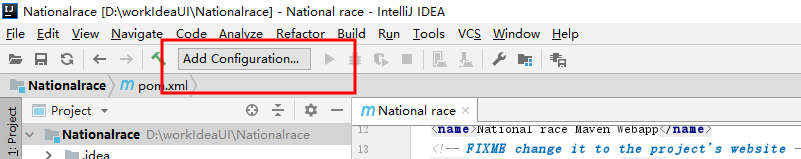
修改完成后：对整个工程进行reimport

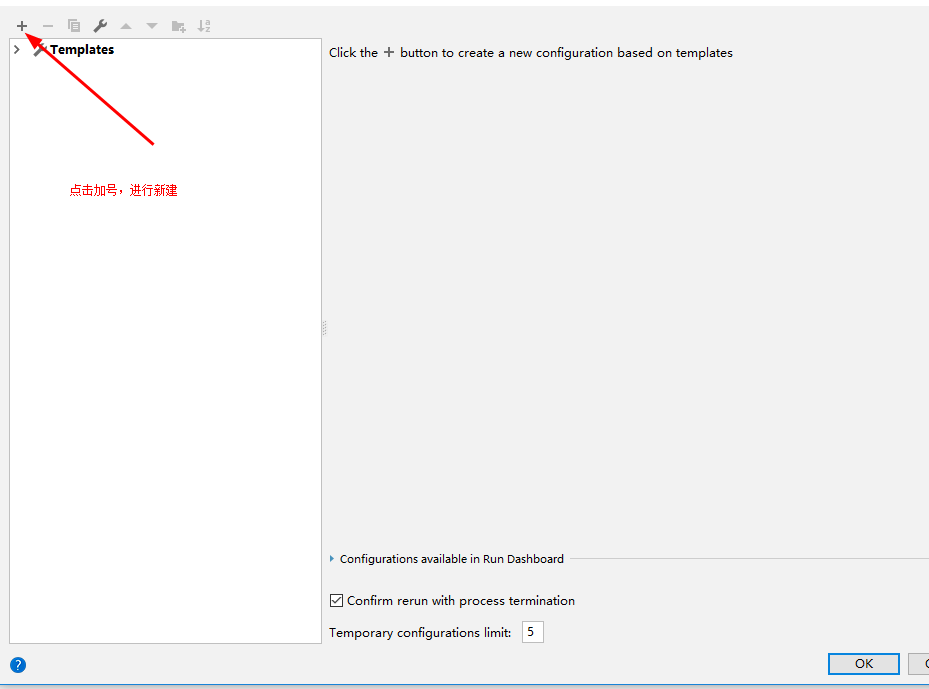


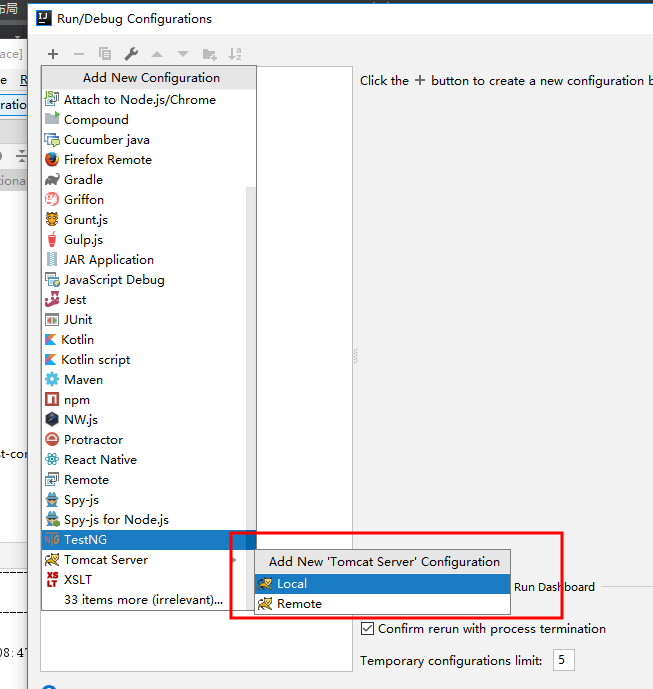
可以发现这里架包就多了个



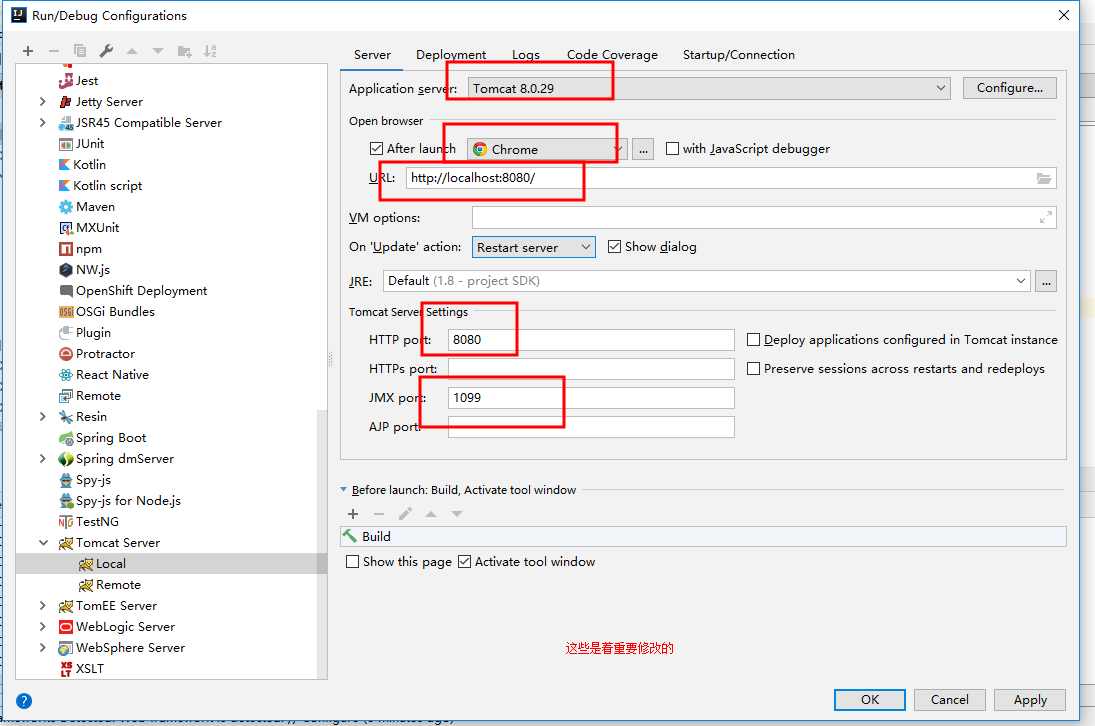
1. 然后配置我们的tomcat
2. 选择Add Configuration



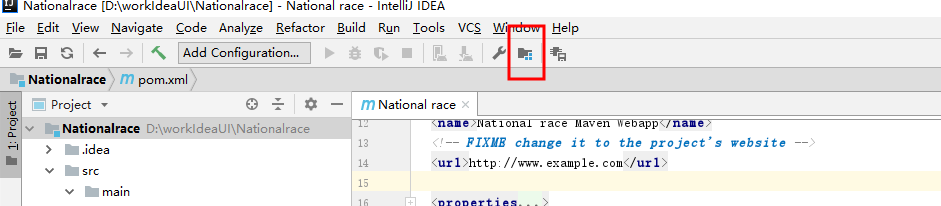


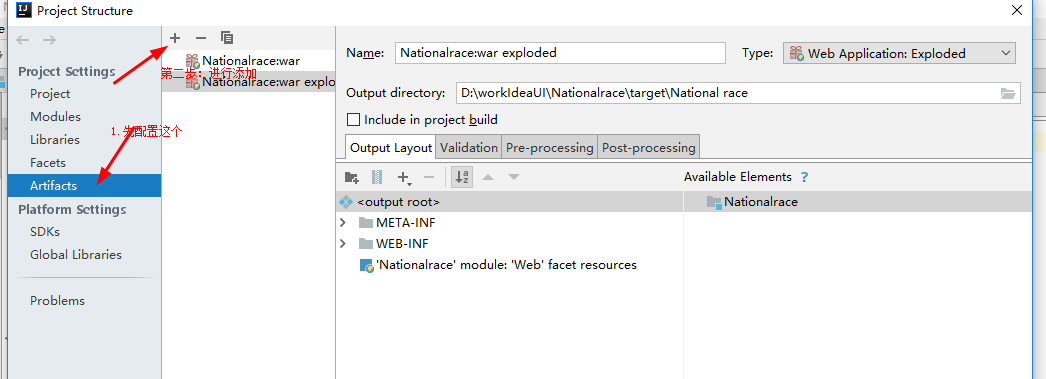


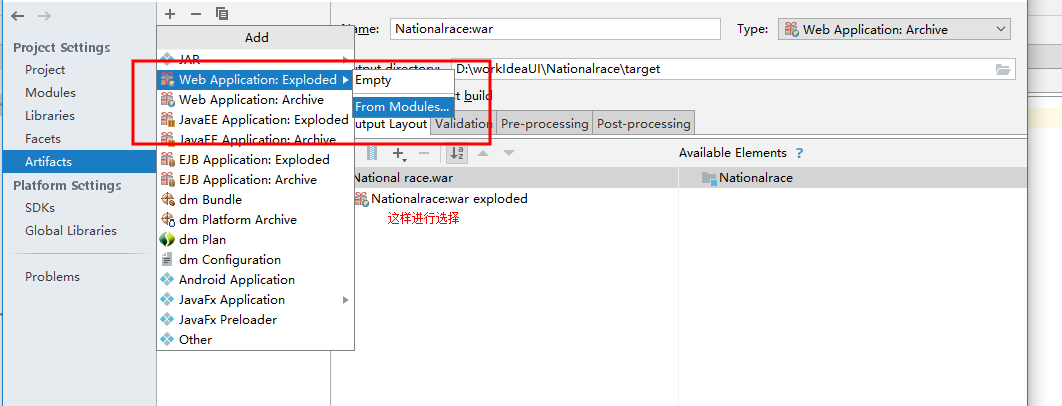
1. 修改配置信息

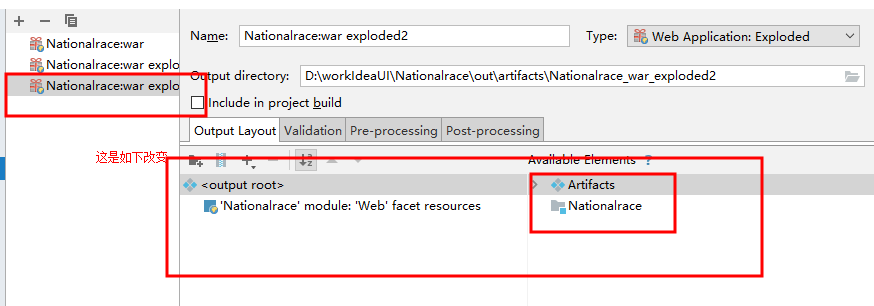


1. 点击这个，他在以后的项目里经常要被使用到



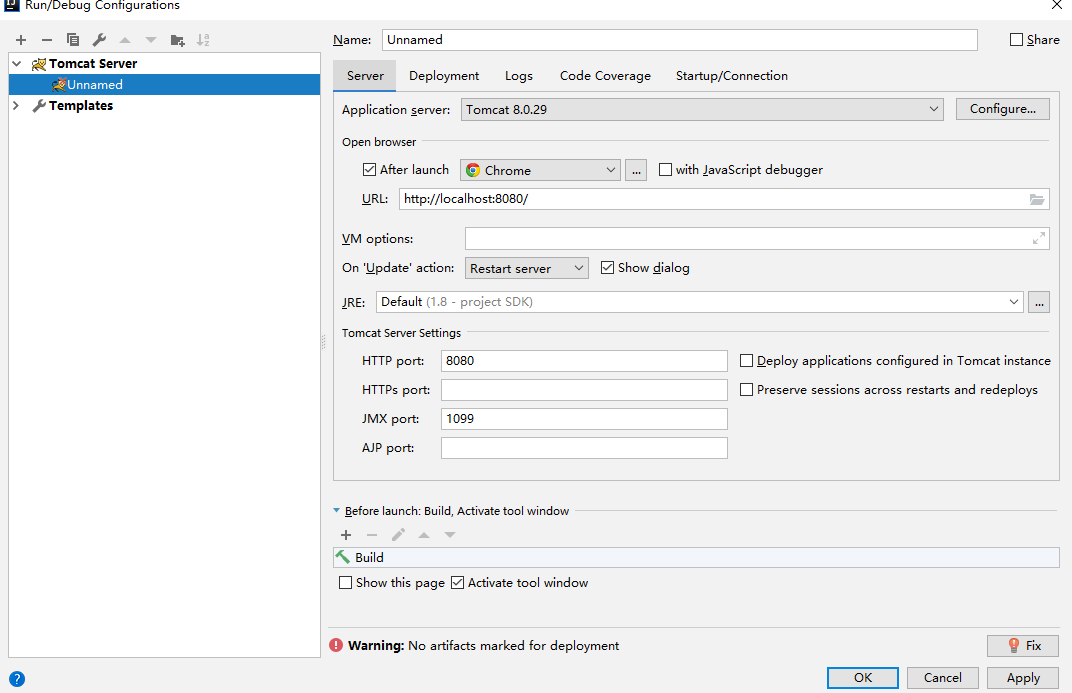




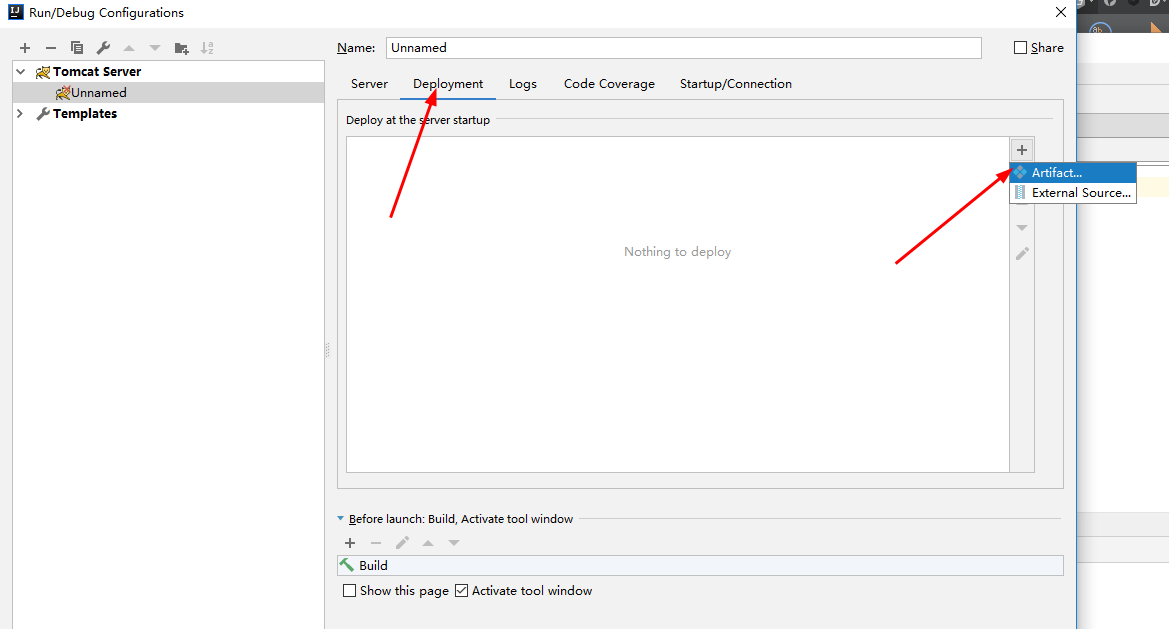


1. 继续配置tomcat

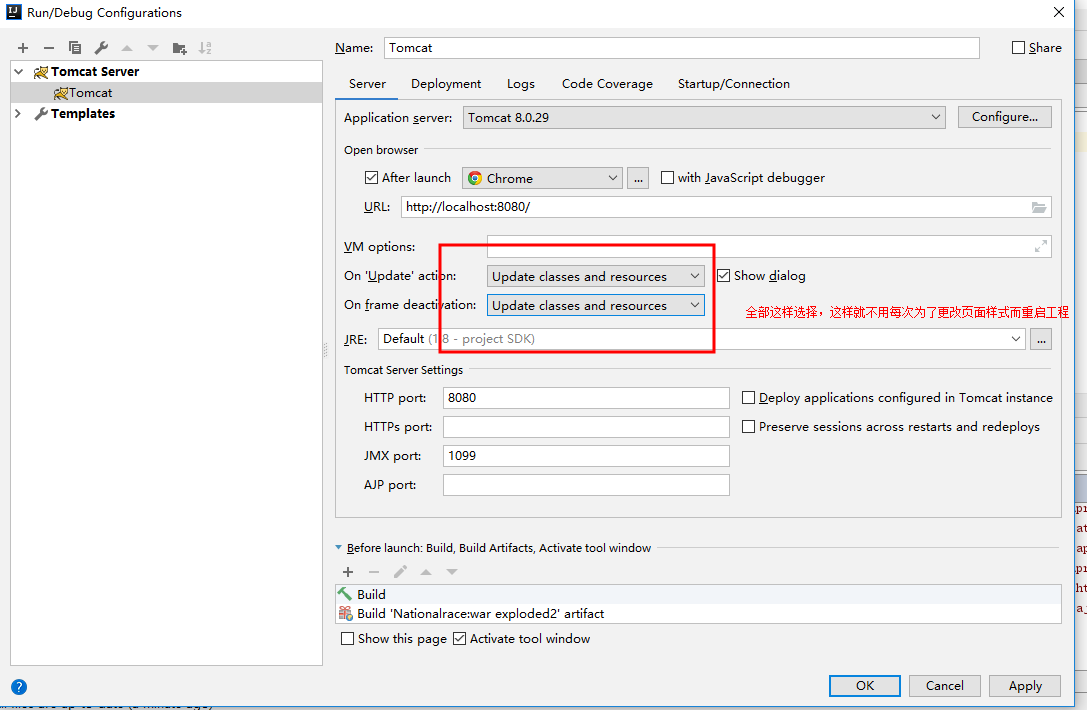
回到这个界面



选择这两项

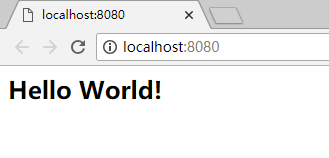


1. 对tomcat进行热效应配置



1. tomcat配置完成，然后点击运行测试是否配置完成

当浏览器出现这个就代表配置完成



1. 在pom.xml引入依赖架包

（Spring\_mvc Spring JDBC Spring面向切面编程 mybatis mybatis整合Spring的适配包 数据库连接池 mysql驱动 jstl servlet api 反编译数据库用的架包 Spring单元测试模块 ）

|  |
| --- |
| <**dependencies**>  *<!--做单元测试用的-->  <!-- https://mvnrepository.com/artifact/junit/junit -->* <**dependency**>  <**groupId**>junit</**groupId**>  <**artifactId**>junit</**artifactId**>  <**version**>4.12</**version**>   </**dependency**>    *<!--引入项目依赖包-->  <!--spring spring\_mvc -->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->* <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-webmvc</**artifactId**>  <**version**>5.0.8.RELEASE</**version**>  </**dependency**>  *<!--spring JDBC-->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->* <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-jdbc</**artifactId**>  <**version**>5.0.8.RELEASE</**version**>  </**dependency**>  *<!-- spring 面向切面编程-->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-aspects -->* <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-aspects</**artifactId**>  <**version**>5.0.8.RELEASE</**version**>  </**dependency**>  *<!-- mybatis-->  <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->* <**dependency**>  <**groupId**>org.mybatis</**groupId**>  <**artifactId**>mybatis</**artifactId**>  <**version**>3.4.2</**version**>  </**dependency**>  *<!--mybatis整合spring适配包-->  <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->* <**dependency**>  <**groupId**>org.mybatis</**groupId**>  <**artifactId**>mybatis-spring</**artifactId**>  <**version**>1.3.1</**version**>  </**dependency**>  *<!--数据库连接池-->  <!-- https://mvnrepository.com/artifact/c3p0/c3p0 -->* <**dependency**>  <**groupId**>c3p0</**groupId**>  <**artifactId**>c3p0</**artifactId**>  <**version**>0.9.1.2</**version**>  </**dependency**>  *<!--mysql驱动-->  <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->* <**dependency**>  <**groupId**>mysql</**groupId**>  <**artifactId**>mysql-connector-java</**artifactId**>  <**version**>5.1.41</**version**>  </**dependency**> *<!--jstl-->  <!-- https://mvnrepository.com/artifact/jstl/jstl -->* <**dependency**>  <**groupId**>jstl</**groupId**>  <**artifactId**>jstl</**artifactId**>  <**version**>1.2</**version**>  </**dependency**> *<!--servlet api-->  <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->* <**dependency**>  <**groupId**>javax.servlet</**groupId**>  <**artifactId**>javax.servlet-api</**artifactId**>  <**version**>3.1.0</**version**>  <**scope**>provided</**scope**>  </**dependency**> *<!--反编译数据库用的架包-->  <!-- https://mvnrepository.com/artifact/org.mybatis.generator/mybatis-generator-core -->* <**dependency**>  <**groupId**>org.mybatis.generator</**groupId**>  <**artifactId**>mybatis-generator-core</**artifactId**>  <**version**>1.3.5</**version**>  </**dependency**>  *<!--Spring单元测试模块-->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->* <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-test</**artifactId**>  <**version**>5.0.8.RELEASE</**version**>   </**dependency**>   </**dependencies**> |

1. 进行web.xml配置



1. 启动spring的容器

|  |
| --- |
| *<!--1.启动spring的容器-->* <**context-param**>  <**param-name**>contextConfigLocation</**param-name**>  <**param-value**>classpath:applicationContext.xml</**param-value**> </**context-param**> <**listener**>  <**listener-class**>org.springframework.web.context.ContextLoaderListener</**listener-class**> </**listener**> |

1. 启动spring的前端控制器，用它来拦截前端的所有请求

|  |
| --- |
| *<!--2.spring的前端控制器，用它来拦截前端的所有请求-->* <**servlet**>  <**servlet-name**>dispatcherServlet</**servlet-name**>  <**servlet-class**>org.springframework.web.servlet.DispatcherServlet</**servlet-class**>  <**load-on-startup**>1</**load-on-startup**> </**servlet**> <**servlet-mapping**>  <**servlet-name**>dispatcherServlet</**servlet-name**>  <**url-pattern**>/</**url-pattern**> </**servlet-mapping**> |

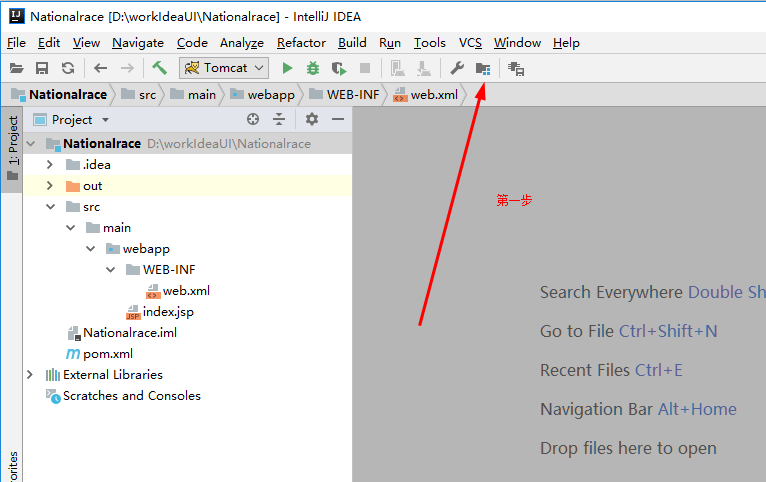
1. 字符编码过滤器，必须放在所有过滤器之前

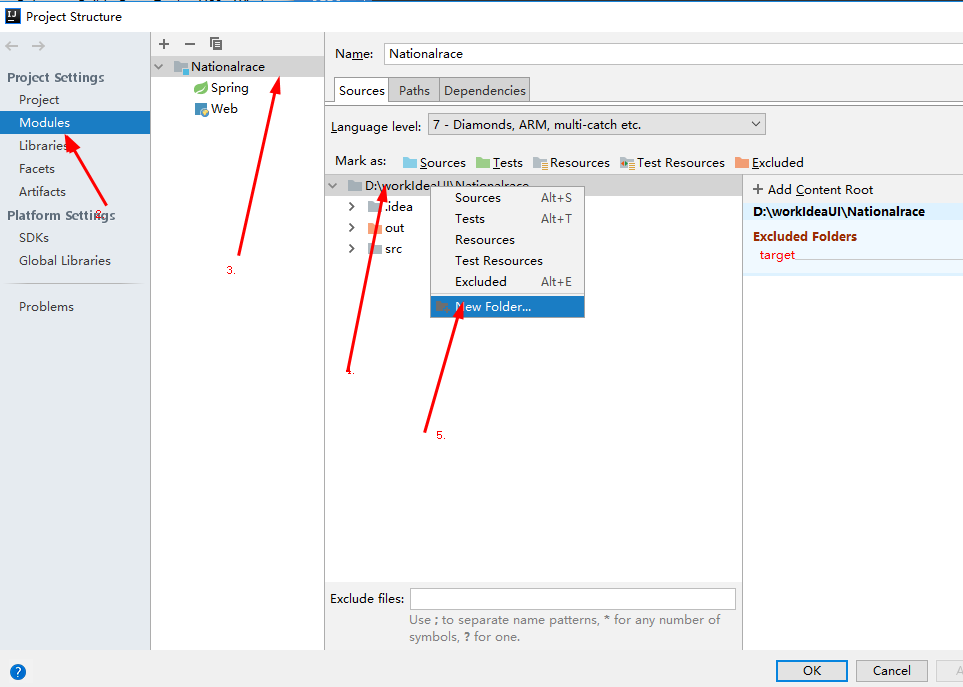
|  |
| --- |
| *<!--3.字符编码过滤器 必须放在所有过滤器之前-->* <**filter**>  <**filter-name**>CharacterEncodingFilter</**filter-name**>  <**filter-class**>org.springframework.web.filter.CharacterEncodingFilter</**filter-class**>  <**init-param**>  <**param-name**>encoding</**param-name**>  <**param-value**>utf-8</**param-value**>  </**init-param**>  <**init-param**>  <**param-name**>forceRequestEncoding</**param-name**>  <**param-value**>true</**param-value**>  </**init-param**>  <**init-param**>  <**param-name**>forceResponseEncoding</**param-name**>  <**param-value**>true</**param-value**>  </**init-param**> </**filter**> <**filter-mapping**>  <**filter-name**>CharacterEncodingFilter</**filter-name**>  <**url-pattern**>/</**url-pattern**> </**filter-mapping**> |

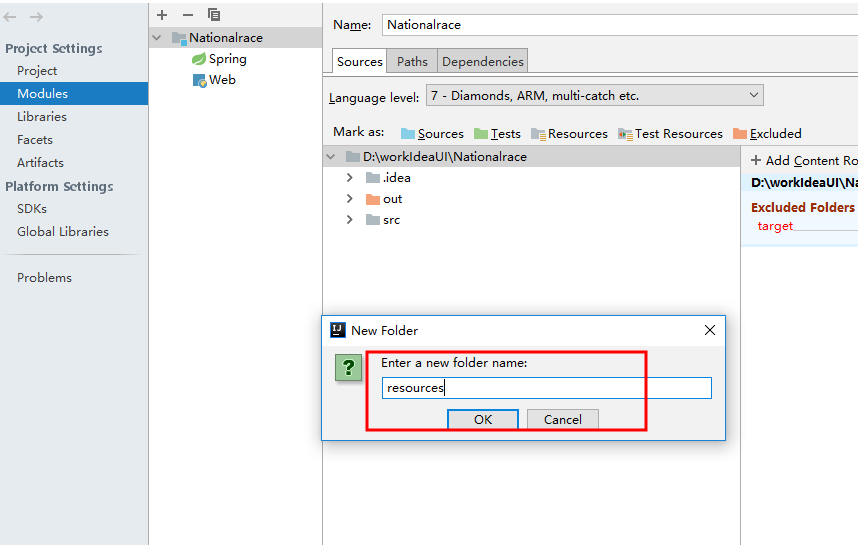
1. 使用rest分割的url,将页面普通的post请求转化为指定的delete或者put请求

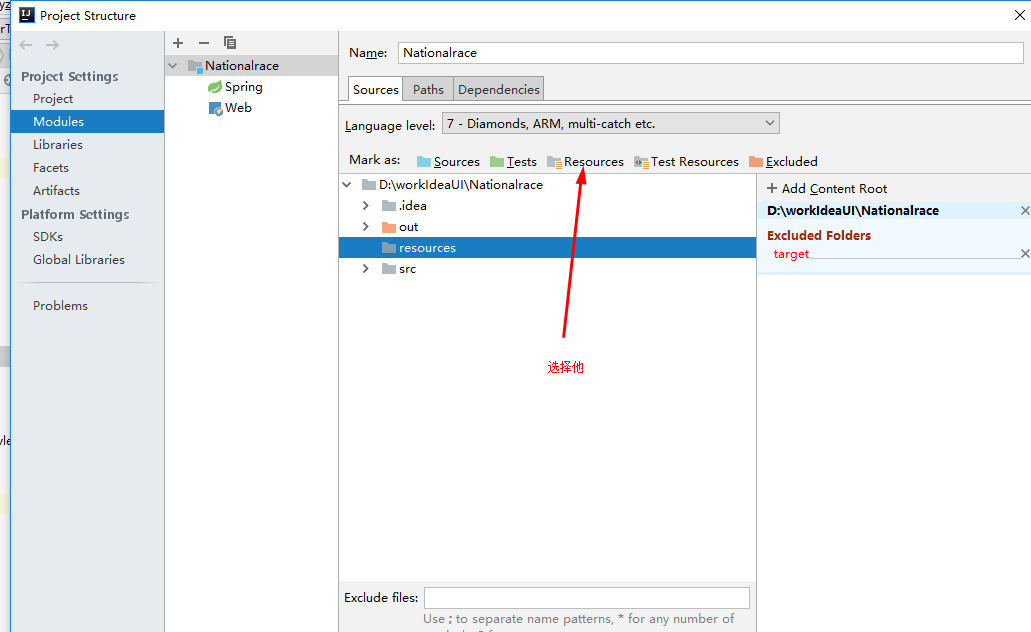
|  |
| --- |
| *<!-- 4使用rest分隔的url,将页面普通的post请求转为指定的delete或者put请求-->* <**filter**>  <**filter-name**>HiddenHttpMethodFilter</**filter-name**>  <**filter-class**>org.springframework.web.filter.HiddenHttpMethodFilter</**filter-class**> </**filter**> <**filter-mapping**>  <**filter-name**>HiddenHttpMethodFilter</**filter-name**>  <**url-pattern**>/\*</**url-pattern**> </**filter-mapping**> |

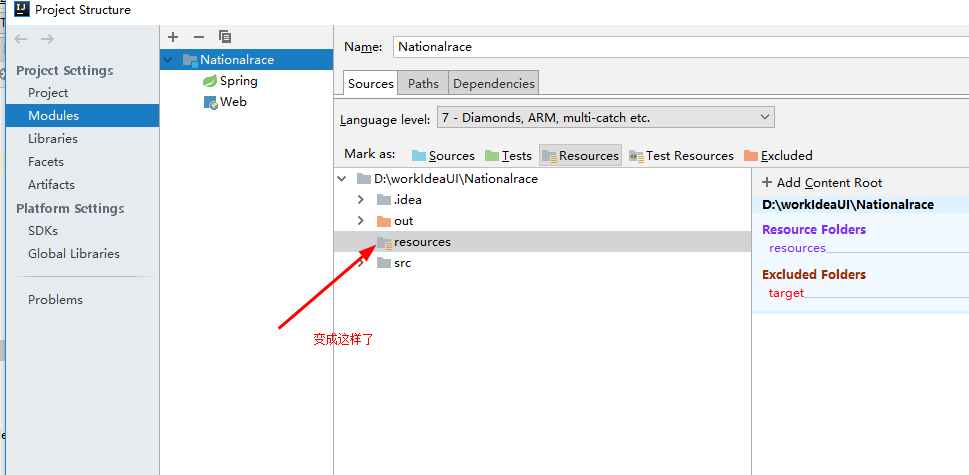
1. 新建applicationContext.xml

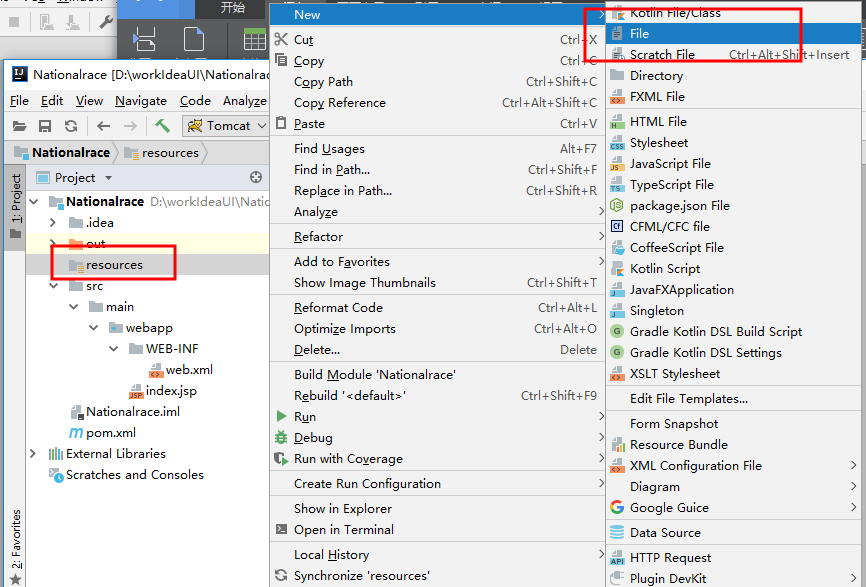


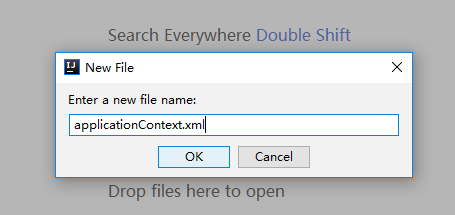








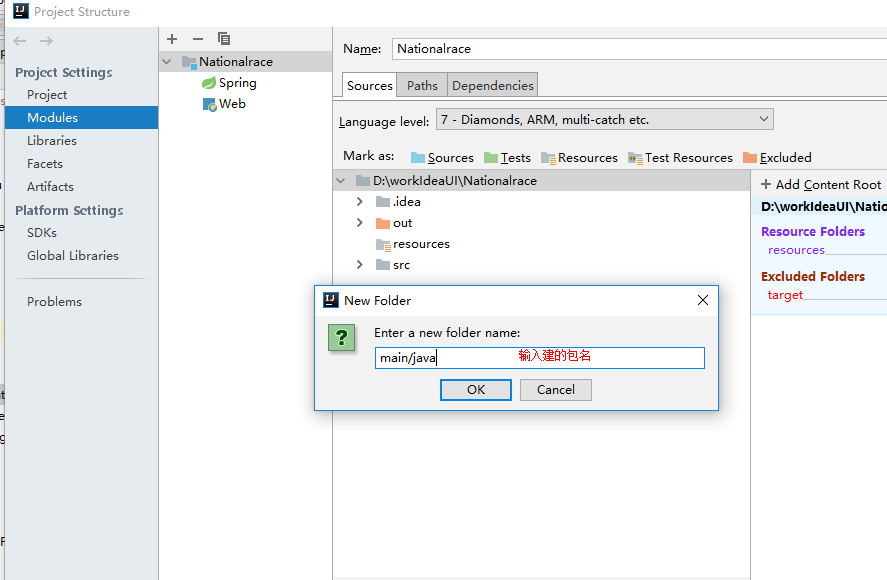


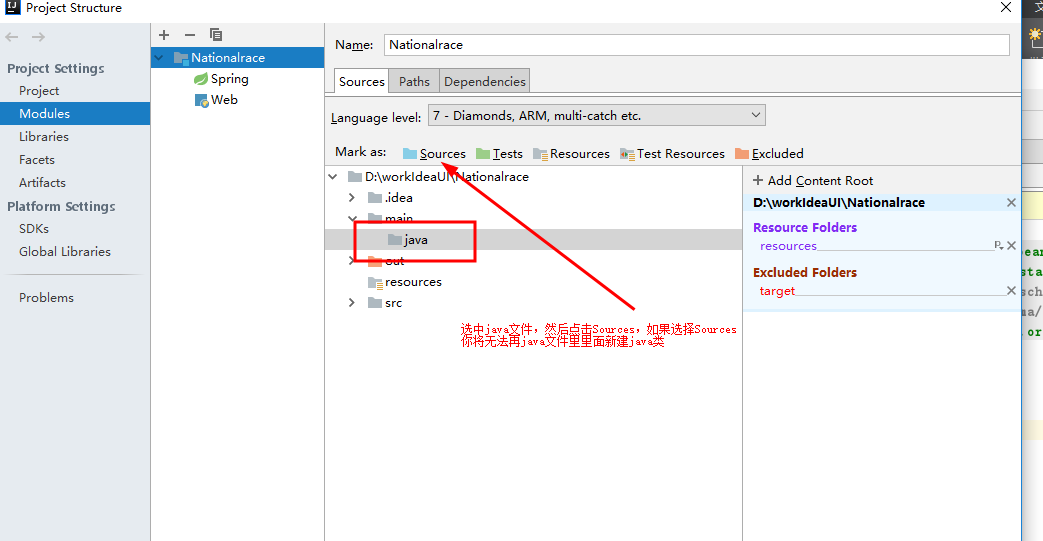


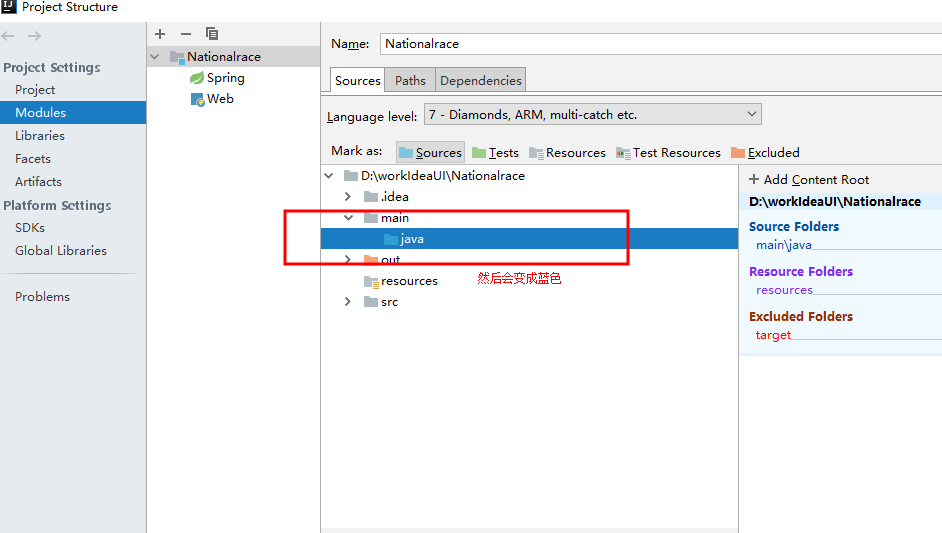
1. 在applicationContext.xml先写入这几行代码

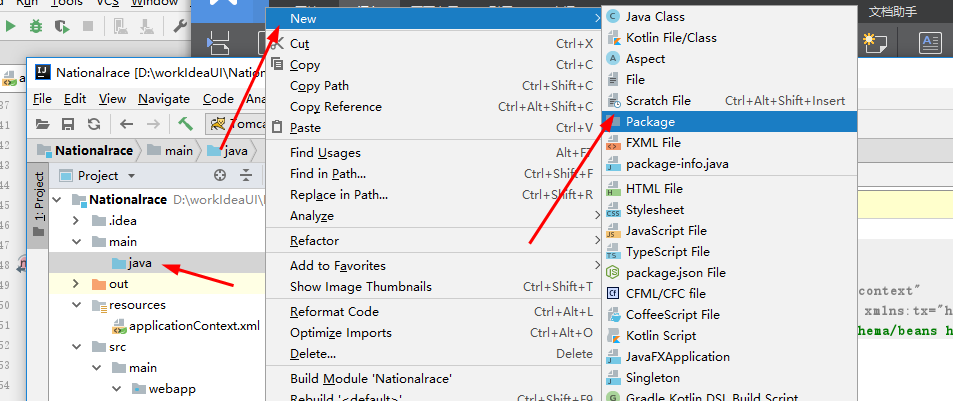
|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"***?>* <**beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:context="http://www.springframework.org/schema/context"  xmlns:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://www.springframework.org/schema/tx"  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 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx.xsd"**> </**beans**> |

1. 新建几个包，这个包很重要

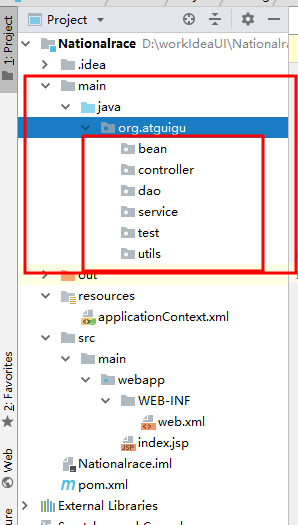








1. 新建如下几个包



1. 补充applicationContext
2. 数据源，事物控制

|  |  |
| --- | --- |
| *<!--数据源 事物控制 -->* <**context:property-placeholder location="classpath:dbconfig.properties"**></**context:property-placeholder**>     |  | | --- | | 新建**dbconfig.properties文件，**这个文件是对数据库连接做的描述    写入内容    **jdbc.jdbcUrl**=**jdbc:mysql://localhost:3306/ssm?characterEncoding=utf-8 jdbc.driverClass**=**com.mysql.jdbc.Driver jdbc.user**=**root jdbc.password**=**root** | |

1. 事物控制，连接我们的数据库

|  |
| --- |
| *<!--事物控制-->* <**bean id="dataSourceManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager"**>  *<!--控制住数据源-->* <**property name="dataSource" ref="pooledDataSource"**></**property**> </**bean**> <**bean id="pooledDataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"**>  <**property name="jdbcUrl" value="${jdbc.jdbcUrl}"**></**property**>  <**property name="driverClass" value="${jdbc.driverClass}"**></**property**>  <**property name="user" value="${jdbc.user}"**></**property**>  <**property name="password" value="${jdbc.password}"**></**property**> </**bean**> |

1. 配置和mybatis的整合

|  |
| --- |
| *<!--配置和mybatis的整合-->* <**bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean"**>  *<!--指定mybatis的全局配置文件的位置-->* <**property name="configLocation" value="classpath:mybaties-config.xml"**></**property**>  *<!--数据源-->* <**property name="dataSource" ref="pooledDataSource"**></**property**>  *<!--指定mapper文件指定位置-->* <**property name="mapperLocations" value="classpath:mapper/\*.xml"**></**property**> </**bean**>    新建**mybaties-config.xml和mapper mybaties.xml也是数据库连接需要配置的**      新建mapper |

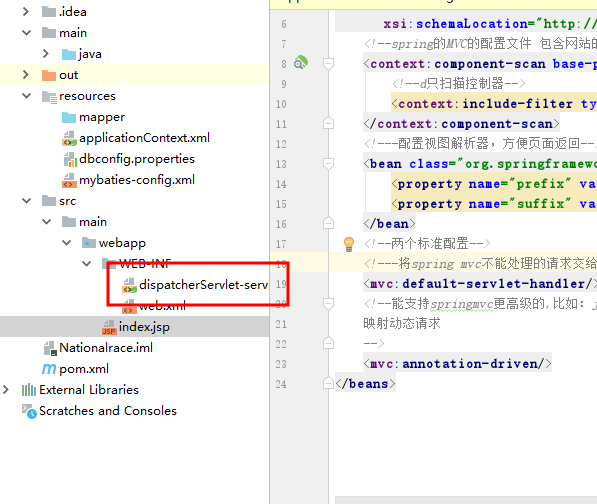
1. 配置扫描器，将mybatis接口加入到ioc容器，指定dao文件存放位置

|  |
| --- |
| *<!--配置扫描器，将mybatis接口加入到ioc容器中-->* <**bean class="org.mybatis.spring.mapper.MapperScannerConfigurer"**>  *<!--扫描所有的dao接口实现加入到ioc容器中-->* <**property name="basePackage" value="org.atguigu.dao"**></**property**> </**bean**> |

1. 开启基于注释的事物，使用xml配置形式事物

|  |
| --- |
| *<!--开启基于注释的事物，使用xml配置形式事物-->* <**aop:config**>  *<!---切入点表达式-->* <**aop:pointcut id="txPoint" expression="execution(\* org.atguigu.service..\*(..))"**></**aop:pointcut**>  *<!--事物增强-->* <**aop:advisor advice-ref="txAdvice" pointcut-ref="txPoint"**></**aop:advisor**> </**aop:config**> <**tx:advice id="txAdvice" transaction-manager="dataSourceManager"**>  <**tx:attributes**>  *<!--所有方法都是事物方法-->* <**tx:method name="\*"**/>  *<!--以get开始的所有方法-->* <**tx:method name="get\*" read-only="true"**></**tx:method**>  </**tx:attributes**>  </**tx:advice**> *<!---Spring配置文件的核心点，（数据源，与mybatis的整合，事物控制）-->* |

1. 新建dispatcherServlet-servlet.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:context="http://www.springframework.org/schema/context"  xmlns:mvc="http://www.springframework.org/schema/mvc"  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 http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd"**>  *<!--spring的MVC的配置文件 包含网站的跳转逻辑控制。配置-->* <**context:component-scan base-package="org.atguigu" use-default-filters="false"**>  *<!--d只扫描控制器-->* <**context:include-filter type="annotation" expression="org.springframework.stereotype.Controller"**></**context:include-filter**>  </**context:component-scan**>  *<!---配置视图解析器，方便页面返回-->* <**bean class="org.springframework.web.servlet.view.InternalResourceViewResolver"**>  <**property name="prefix" value="/WEB-INF/views/"**></**property**>  <**property name="suffix" value=".jsp"**></**property**>  </**bean**>  *<!--两个标准配置-->  <!---将spring mvc不能处理的请求交给tomcat-->* <**mvc:default-servlet-handler**/>  *<!--能支持springmvc更高级的,比如：jsr30校验，快捷的ajax  映射动态请求  -->* <**mvc:annotation-driven**/> </**beans**> |

1. 新建mbg.xml，主要是数据库的反编译

|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"***?>* **<!DOCTYPE generatorConfiguration  PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"  "http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd"*>*** <**generatorConfiguration**>   <**context id="DB2Tables" targetRuntime="MyBatis3"**>  *<!--生成没有注释-->* <**commentGenerator**>  <**property name="suppressAllComments" value="true"**></**property**>  </**commentGenerator**>  *<!--配置数据库连接信息-->* <**jdbcConnection  driverClass="com.mysql.jdbc.Driver"  connectionURL="jdbc:mysql://localhost:3306/ssm"  userId="root"  password="root"**>  </**jdbcConnection**>   <**javaTypeResolver** >  <**property name="forceBigDecimals" value="false"** />  </**javaTypeResolver**>  *<!--指定javabean生成的位置-->* <**javaModelGenerator  targetPackage="org.atguigu.bean"  targetProject=".\main\java"**>  <**property name="enableSubPackages" value="true"** />  <**property name="trimStrings" value="true"** />  </**javaModelGenerator**>  *<!--指定sql映射文件位置-->* <**sqlMapGenerator  targetPackage="mapper"  targetProject=".\resources"**>  <**property name="enableSubPackages" value="true"** />  </**sqlMapGenerator**>  *<!--指定dao接口生成的位置 mapper接口-->* <**javaClientGenerator  type="XMLMAPPER"  targetPackage="org.atguigu.dao"  targetProject=".\main\java"**>  <**property name="enableSubPackages" value="true"** />  </**javaClientGenerator**>  *<!--table指定每个表 的生成策略-->* <**table tableName="tb\_emp" domainObjectName="Employee"**></**table**>  <**table tableName="tnl\_dept" domainObjectName="Department"**></**table**>  </**context**> </**generatorConfiguration**> |

1. 全部文件

