Fiso开发手册

说明：本手册中提到的软件包都可以在ftp上下载到本地电脑安装，ftp地址为ftp://test@192.168.0.222。本手册中提到的ftp地址以及项目测试数据库ip地址如发生变更，请及时联系管理员。下面将详细介绍平台环境的搭建。

文档修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **修改人** | **修改时间** | **修改内容简述** |
| V1.0 | 高超 | 2018-07-19 | 新建 |
| V1.1 | 宋全恒 | 2018-08-20 | * 修改文档格式 * 根据最新勤务系统项目修改配置过程 |
|  |  |  |  |

# 简介

本文档适用于初次部署勤务系统源码的开发人员或者测试人员，文档的主要内容为使用Git或者TortoiseGit克隆勤务系统源代码，并使用集成开发环境Intellij IDEA或者MyEclipse把勤务系统项目部署至Tomcat，通过集成开发环境的配置，并通过浏览器进行验证的文档说明。

# git安装与代码下载

## 安装

从ftp上下载git软件包，然后直接安装（安装时一路next即可），安装文件位于路径

ftp://192.168.0.222/开发环境/Git版本管理

压缩包包含文件：

1. Git-2.18.0-64-bit
2. TortoiseGit-2.6.0.0-64bit
3. TortoiseGit-LanguagePack-2.6.0.0-64bit-zh\_CN

按照次序安装即可

## 配置

在使用Git克隆仓库代码时需要进行配置SSH Keys，配置的过程参见开发文档【Git配置过程】

该文档位于Git仓库

http://192.168.0.223/songquanheng

## 克隆代码

### 使用Git bash克隆代码

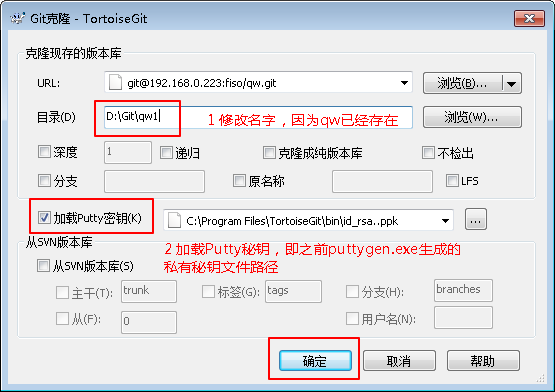
在电脑桌面上空白处右键（也可以在其他位置处右键），选择“git bash here”，然后在弹出的git console界面上执行git clone命令下载项目代码，根据提示，需要输入自己gitlab上的用户名和密码。（注意：在该步骤执行时需要为Git添加SSH keys，并把公钥内容添加到GitLab服务器上）

然后，进入工程目录，执行命令：git checkout develop切换到指定开发分支上

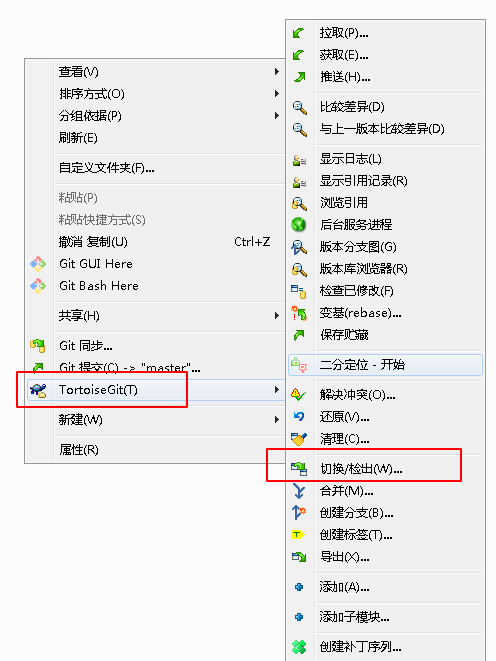


### 使用TortoiseGit克隆代码

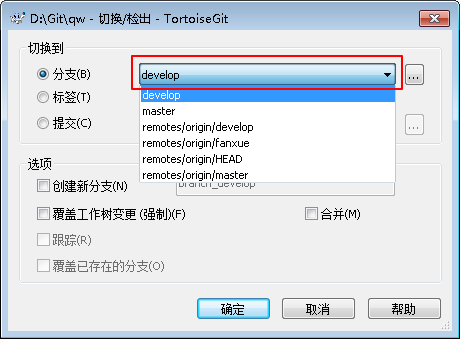
使用之前保存的私钥文件，通过小乌龟TortoiseGit完成项目的克隆，如图：



然后进入项目.git所在目录，右击选择TortoiseGit，切换分支为Develop



选择要切换的分支，在文档中，部署的分支为develop。



# 集成开发环境导入项目

通过第二部分的操作，可以从Git仓库中获取勤务系统的源代码，想要部署该勤务系统到本地，仍需要安装其他的工作环境

## 安装jdk1.8

文件所在位置位于

ftp://192.168.0.222/开发环境 / jdk\_8.0.1310.11\_64.exe

安装时，默认即可，jdk所在目录为

C:\Program Files\Java\jdk1.8.0\_131

### 配置环境变量

在环境变量中新建系统变量JAVA\_HOME值为jdk所在目录

并编辑系统环境变量Path，加入如下内容：

%JAVA\_HOME%\bin

如果该内容位置不在Path的末尾，则需要在bin之后添加英文的分号“;”。

### 验证

在dos窗口下输入命令java –version/javac进行jdk环境配置验证

Microsoft Windows [版本 6.1.7601]

版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\admin>java -version

java version "1.8.0\_131"

Java(TM) SE Runtime Environment (build 1.8.0\_131-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)

C:\Users\admin>javac

用法: javac <options> <source files>

其中, 可能的选项包括:

-g 生成所有调试信息

-g:none 不生成任何调试信息

-g:{lines,vars,source} 只生成某些调试信息

-nowarn 不生成任何警告

-verbose 输出有关编译器正在执行的操作的消息

-deprecation 输出使用已过时的 API 的源位置

-classpath <路径> 指定查找用户类文件和注释处理程序的位置

-cp <路径> 指定查找用户类文件和注释处理程序的位置

-sourcepath <路径> 指定查找输入源文件的位置

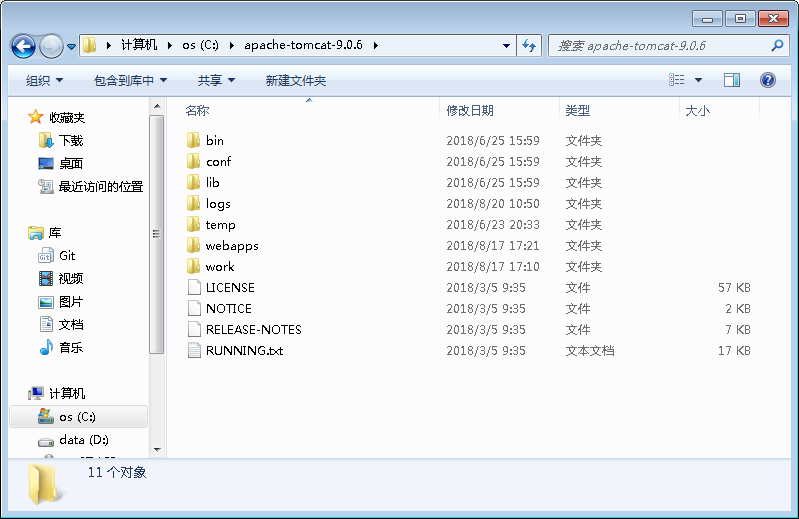
C:\Users\admin>

## 安装Tomcat

Tomcat的安装包位于路径：

ftp://192.168.0.222/开发环境 / apache-tomcat-9.0.6 .rar

直接解压至C:盘即可。截图如下：



## 配置本地maven.

Maven是一个项目管理和综合工具。Maven提供了开发人员构建一个完整的生命周期框架。开发团队可以完成项目的基础工具建设，Maven使用标准的目录结构和默认构建生命周期。

### Maven资源库

Maven位置，中央和远程存储库配置，有两个位置需要理解，一个是Maven本地资源库：

Maven本地资源库是用来存储项目的依赖库，默认的文件目录是.m2目录，可以根据需要将其更改为另一个文件夹。

Maven中央仓库：

Maven中央仓库是Maven用来下载所有项目的依赖库的默认位置。

虽然IDEA自带了Maven，但仍然安装了Maven，依然可以使用IntelliJIDEA自带的maven，自带的配置文件所在目录为：

C:\Users\admin\.m2\settings.xml

因为并没有使用，所以需要进行如下的配置

### 安装

Maven所在目录为：

ftp://192.168.0.222/开发环境/ apache-maven-3.3.9-bin.tar.gz

安装其他版本的Maven，可以访问url：

https://archive.apache.org/dist/maven-3/

解压缩至C：即可

### 配置Maven

Maven的配置依赖于系统环境变量JAVA\_HOME。因此，确保系统变量中存在JAVA\_HOME。

在环境变量中新建系统变量MAVEN\_HOME，值为Maven安装目录

C:\apache-maven-3.3.9

以同样的方式编辑系统变量Path，添加如下内容：

%MAVEN\_HOME%\bin

### 验证

C:\Users\admin>mvn -v

Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5; 2015-11-11T00:41:4

7+08:00)

Maven home: C:\apache-maven-3.3.9\bin\..

Java version: 1.8.0\_131, vendor: Oracle Corporation

Java home: C:\Program Files\Java\jdk1.8.0\_131\jre

Default locale: zh\_CN, platform encoding: GBK

OS name: "windows 7", version: "6.1", arch: "amd64", family: "dos"

如果执行上述命令，可以看到相应的输出，即完成了Maven的配置。

### Intellij IDEA配置自定义Maven

在目录

C:\apache-maven-3.3.9\conf

比较关键的是settings.xml和目录repository。settings.xml的文件内容如下：

<?xml version="1.0" encoding="UTF-8"?>

<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">

<!-- localRepository

| The path to the local repository maven will use to store artifacts.

|

| Default: ${user.home}/.m2/repository

<localRepository>/path/to/local/repo</localRepository>

-->

<localRepository>C:\apache-maven-3.3.9\conf\repository</localRepository>

<mirrors>

<mirror>

<id>alimaven</id>

<name>aliyun maven</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

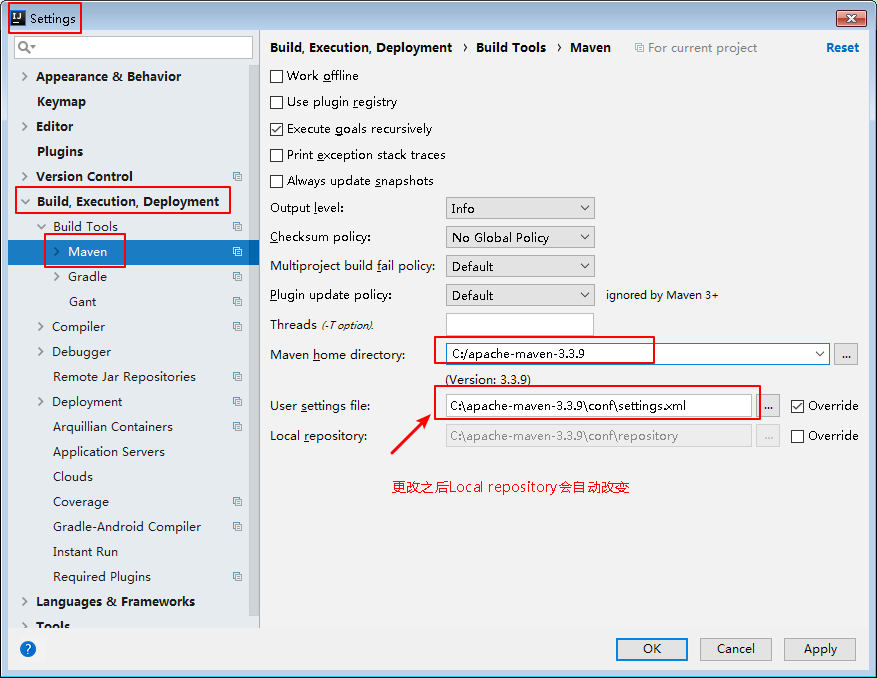
</mirror>

</mirrors>

</settings>

其中localRepository仓库表示的maven的本地仓库所在位置，在工作时放置于settings.xml同级目录。因为在内网开发时无法使用中央仓库下载，因此需要从外网下载好对应的依赖，并传入内网中settings.xml对应的本地仓库目录。

在Intellij IDEA中，设置中搜索Maven更改配置到自定义Maven如下：

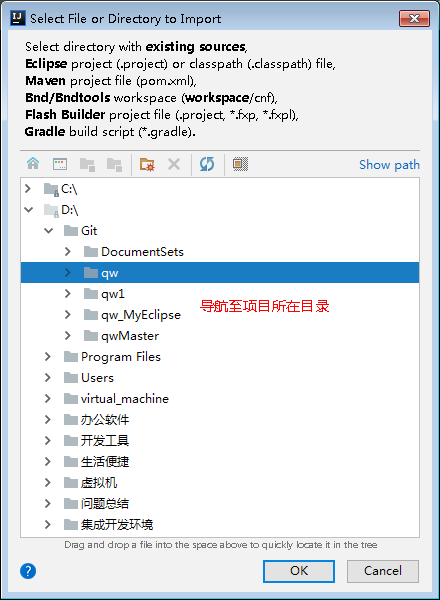


## 勤务系统项目导入IDEA

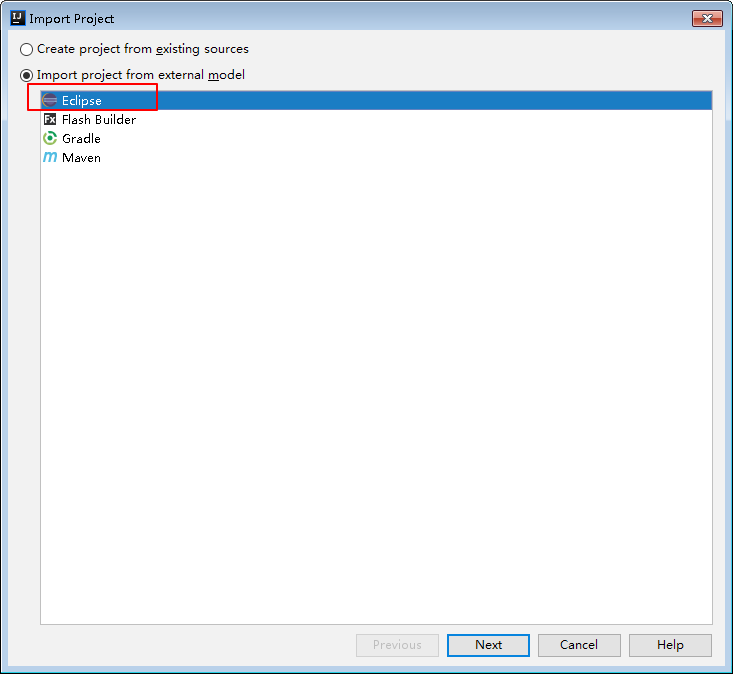
### 导入Intellij IDEA

第一步：打开IDEA，点击“Import Project”,然后选择相应的工程目录即可。使用Open，在文件对话框中导航到工程目录也可。

在文件对话框中导航到项目所在目录：

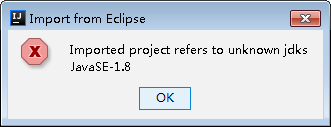


默认选择从已有模型Eclipse导入，直接next即可

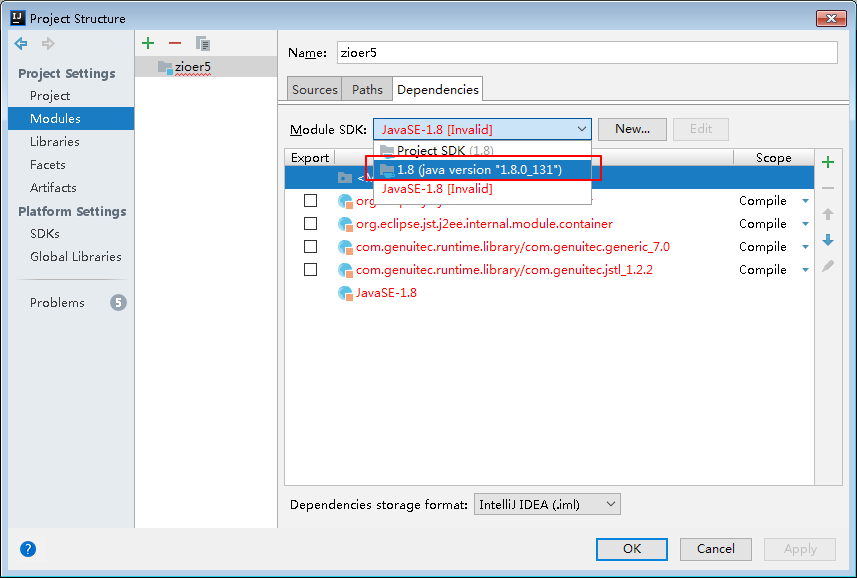


无须改变，选择默认然后next即可。最后选择Finish

注意：在点击Finish之后，若弹出如下对话框，表明JDK未找到，需要进行配置



在设置中搜索Project Structure在该界面设置Module SDK，点击下三角即可：

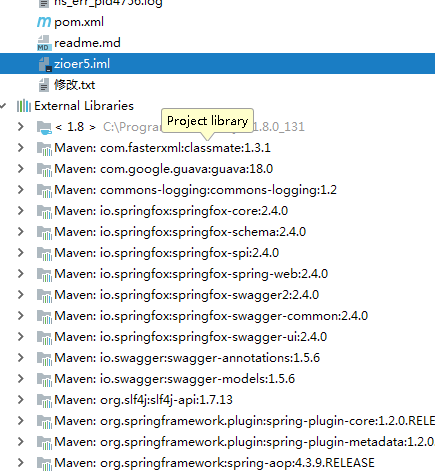


### 添加依赖

因为勤务项目的依赖包由maven管理一部分，也有一些jar包未有maven管理。

#### Maven依赖

首先，打开pom.xml文件，右击，选择Add as Maven Project在添加为Maven项目之后，配置本地Maven，右击项目选择Maven-->Reimport,此时可以看到项目的Externa Libraries变化如下：



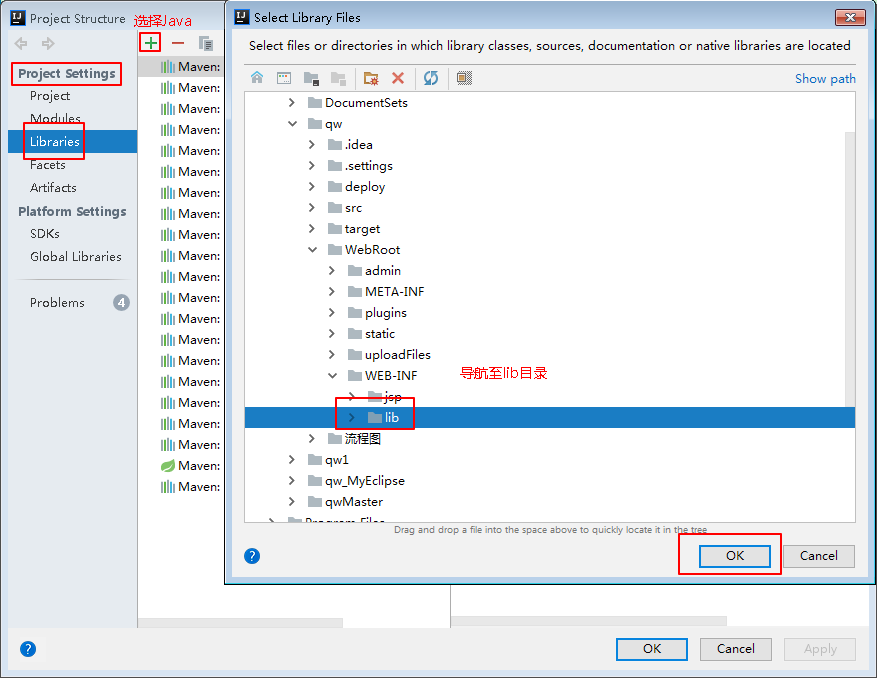
除了jdk之外，可以看到由maven管理的依赖已经存在。

#### lib导入

此时，打开com.zioer.business.GoodsBusiness，可以看到文件出现了标红的显示，其中导入语句与activiti、shiro、lang3相关的语句均有红色提示，表示该jar包IDEA未找到。因此要在IDEA中添加本地jar包依赖。本地jar包依赖位于目录：

\qw\WebRoot\WEB-INF\lib

打开IDEA，进入设置，选择Project Structure，在左侧Project Settings下选择Libraries，然后选择“+” New Project Library。选择Java，弹出如下界面：



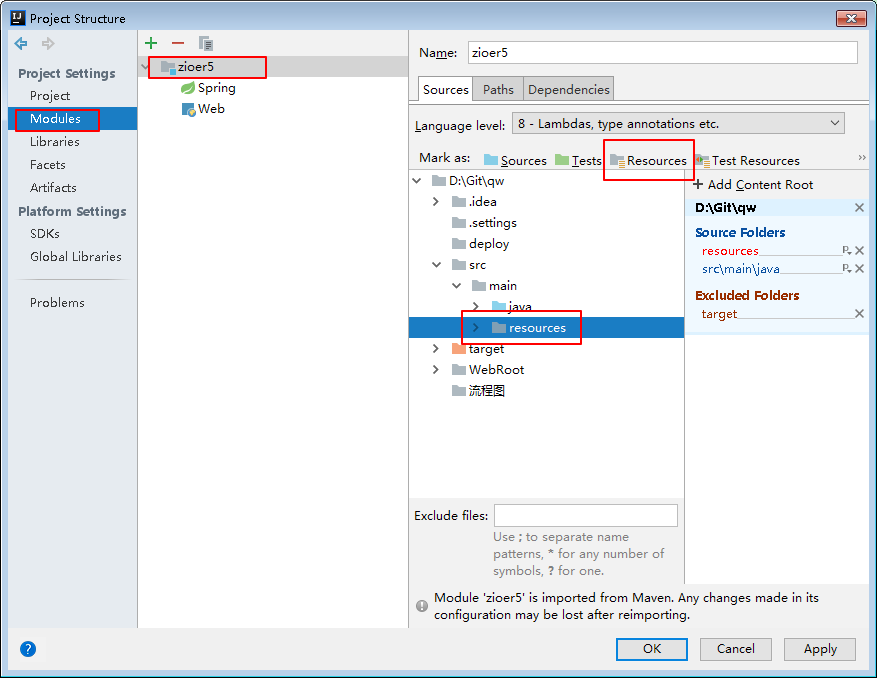
在弹出的Choose Modules对话框中选择zioer5，即当前勤务系统项目。待IDEA scanning files之后可以看到之前标红处已经变黑。表示IDEA已经可以找到相关jar包了。

### 配置Project Structure

#### 标记资源文件夹

第一步：点击'file"->settings->"Project Structure"[快捷键为Ctrl + Shift + Alt + S】

在Modules项，标记resouces目录为为资源类型文件夹



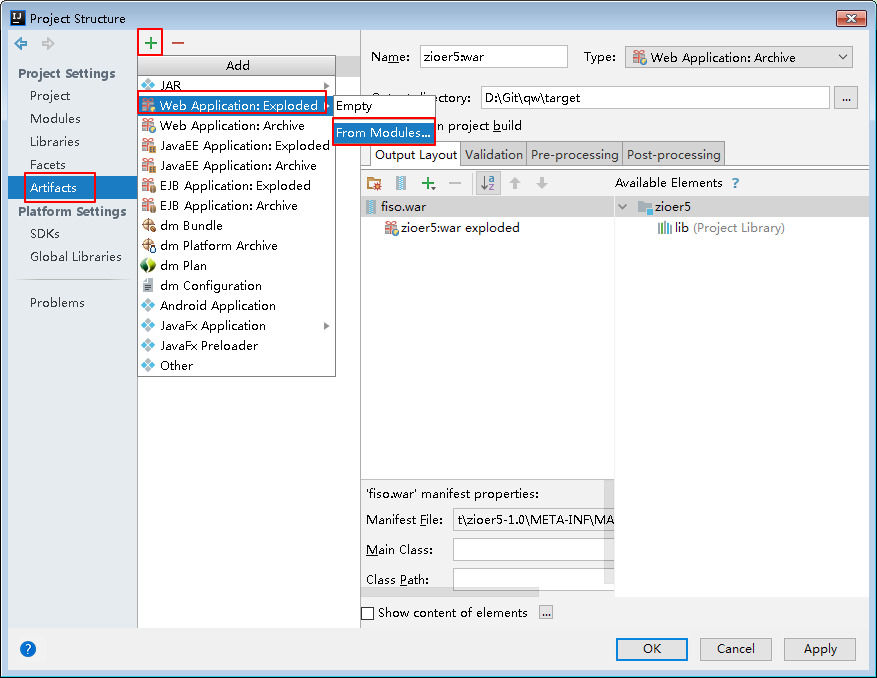
注：简单的方式直接在项目结构中，右击resources，选择Mark Directory as，在弹出的对话框中选择Resources Root

Resources Roots are for resource files used in your application (images, various configuration XML and properties files, etc). During the build process, all the contents of the resources are copied to the output folder as is. Similarily to sources, you can specify that your resources are generated. You can also specify which folder within the ourput folder you resources should be copied to.

注意：默认情况下，工程编译后，resources中的文件会被放置在和源码编译后相同的文件夹中，所以如果在源码中以相对路径读取resources中的配置文件时，可以视为src中的java文件夹和resources文件夹没有区别。

#### 配置Artifacts：

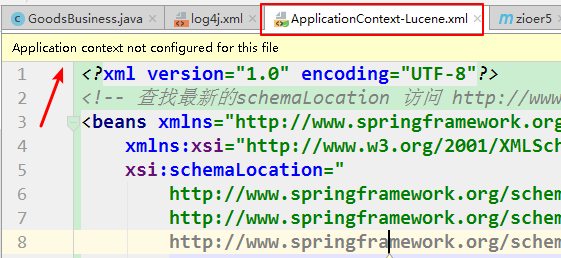
点击"Artifacts"，选择“+”号-->"Web Application Explode"-->"from module",选择默认的“zioer5”modules即可。



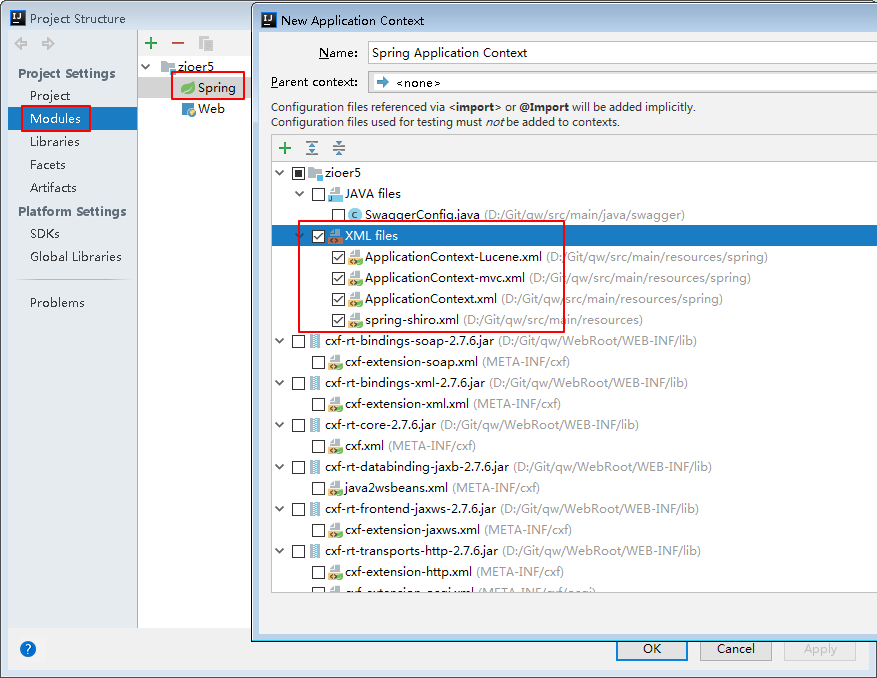
#### 配置XML配置文件检测

在Intellij IDEA运行时需要读取许多XML配置文件，Intellij IDEA会出现Unmapped Spring configuration files: \*\*\*.xml，为了让IDEA能够自动识别xml文件，可以在Spring检测时加入Configure application context：

根据idea提示增加对其他的xml文件配置，如下所示：



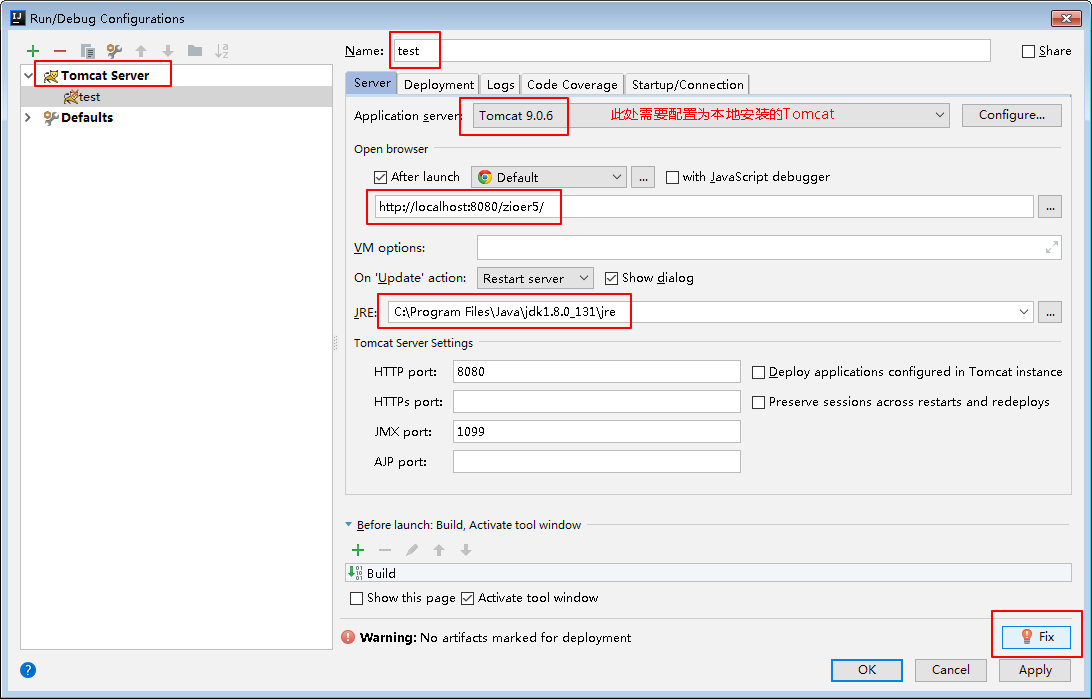
在Project Structure对话框中，Project Settings下的Modules项，Spring中添加新的应用上下文Application Context。



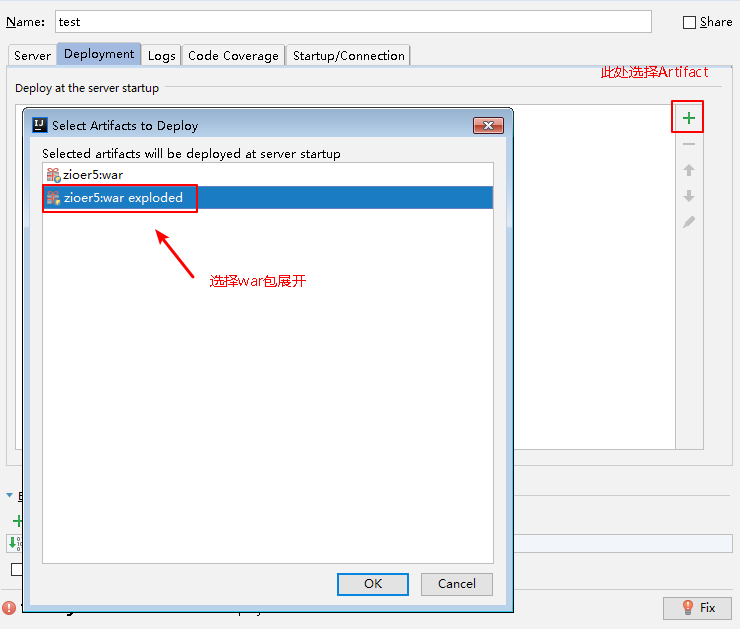
### 添加Tomcat

Tomcat服务器是一个免费的开放源代码的Web应用服务器，因为它运行时占用系统资源少，扩展性好，支持负载均衡与邮件服务器等开发应用系统中常用的功能，而且一直处于不断的版本完善之中，因此受到开发者的喜爱，而且也是企业开发应用中常用的轻量级应用服务器。

在IDEA的菜单栏中选择Run->Edit Configuration，然后选择“+”Add New Configuration，选择Tomcat Server，在弹出的Local和Remote中选择Local，配置如下：



然后需要此对话框同时配置Deployment，点击右下角Fix即可自动定位，或者主动点击Deployment项：



在点击OK之后，在tomcat配置页面中，Application Context勾选为值/zioer5.

### 修改配置文件

修改qw\resources\shiro.properties文件，修改成如下内容：

#localhost

shiro.loginUrl=http://192.168.0.221:8080/cas/login?service=http://localhost:8080/zioer5/shiro-cas  
shiro.logoutUrl=http://192.168.0.221:8080/cas/logout?service=http://localhost:8080/zioer5/shiro-cas  
shiro.cas.serverUrlPrefix=http://192.168.0.221:8080/cas  
shiro.cas.service=http://localhost:8080/zioer5/shiro-cas  
shiro.failureUrl=/logoutsys  
shiro.successUrl=/login\_toLogin

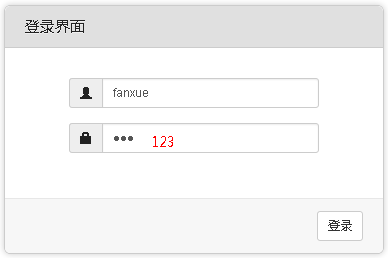
注：192.168.0.221表示勤务系统项目及其相关的mysql、monogodb数据库所在的ip地址

### 运行tomcat启动项目

在IDEA中，以Debug方式启动，等待Building artifact进度条到达终点，然后出现successfully compiled。访问url：

http://localhost:8080/zioer5

出现登录界面，表示项目部署成功



至此完成Intellij IDEA对勤务系统的部署。

## MyEclipse部署勤务系统

### 在myeclipse上打开项目工程

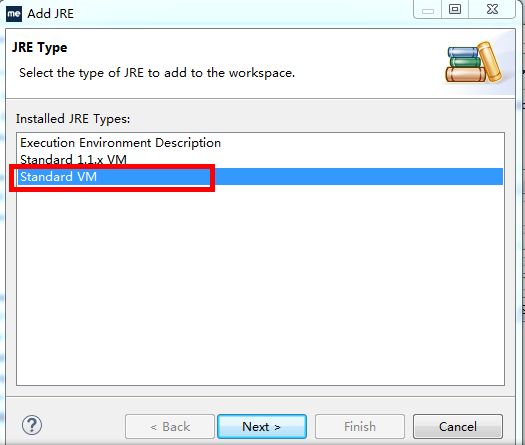
指定workspace的路径后直接打开项目工程目录即可，打开MyEclipse之后，通过选择File->Open Projects from File System…，导航到对应的工程目录，直接打开即可。

亦可以使用File->Import->General->Existing Project into Workspace，然后导航到相应的工程目录也可以把勤务系统导入到MyEclipse。

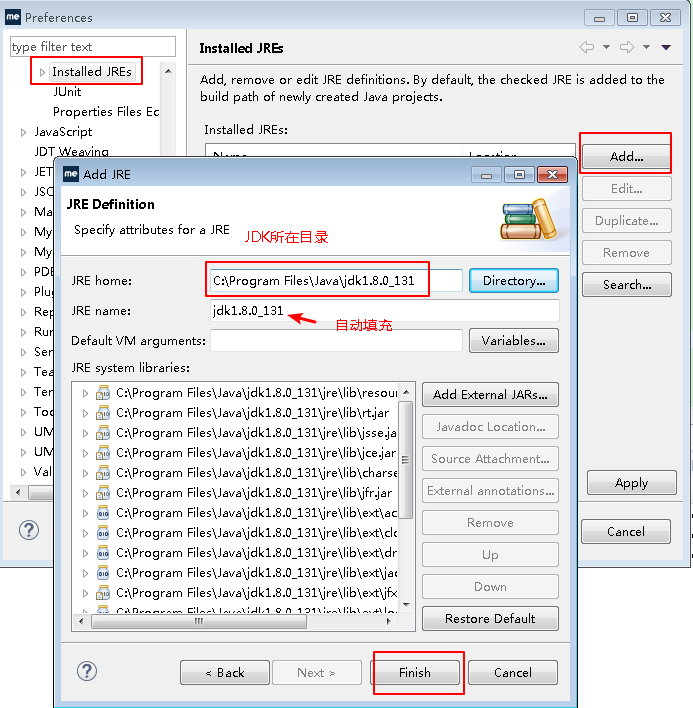
### 配置项目

#### 配置jre运行环境

点击选择"window"->"Preferences"->"java"->"Installed JREs",点击"add",选择jre的类型为"Standard VM",然后设置"jre home"等参数，这里的"jre home"参数要配置成jdk的安装目录，如下所示：

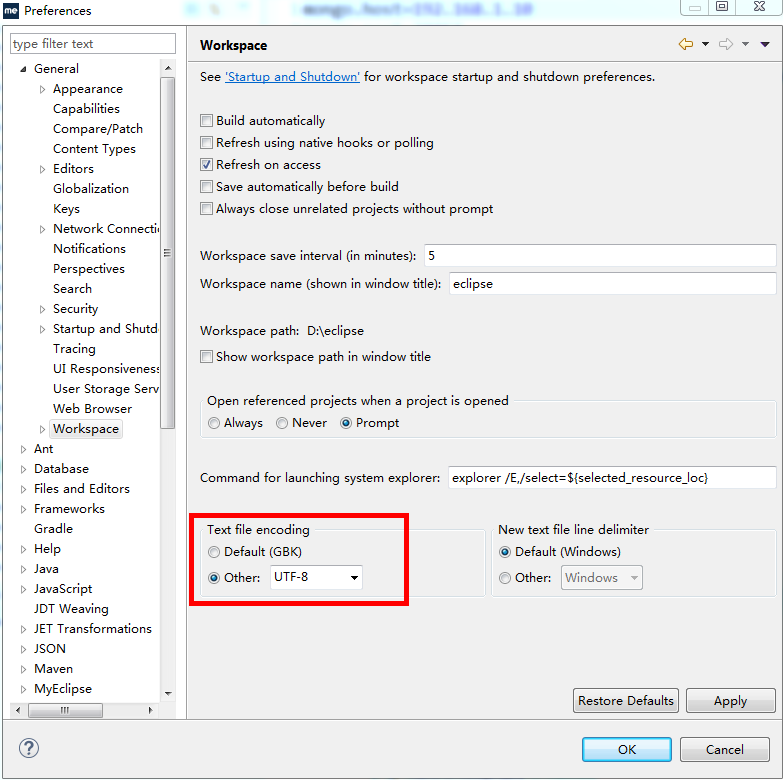


添加安装的JDK所在的目录：JRE home。



#### 设置workspace的编码环境

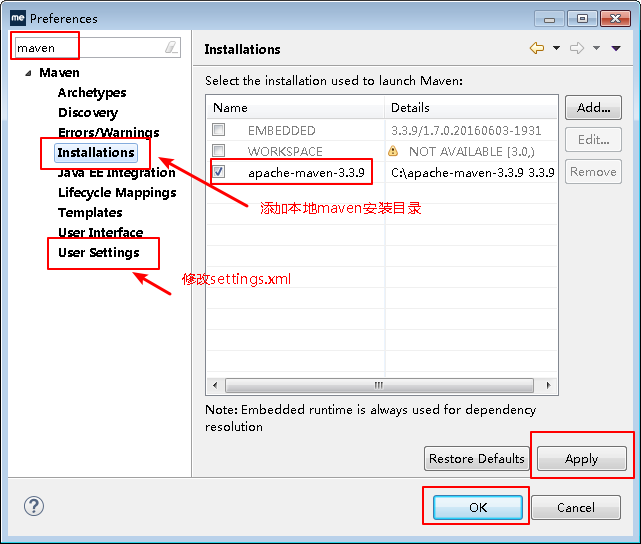
点击选择"window"->"Preferences"->"General"->"Workspace",设置"Text file encoding"为"UTF-8"，如下图所示：



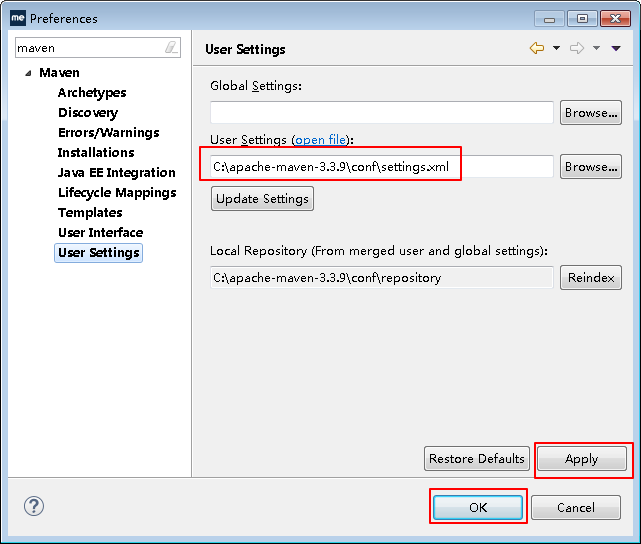
#### 添加依赖

* 添加Maven控制

在MyEclipse中菜单栏中点击Windows->Preference，搜索Maven，在Installations更改Maven的安装目录如下：



在user Settings中更改setting.xml



settings.xml内容如下：

<?xml version="1.0" encoding="UTF-8"?>

<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">

<!-- localRepository

| The path to the local repository maven will use to store artifacts.

|

| Default: ${user.home}/.m2/repository

<localRepository>/path/to/local/repo</localRepository>

-->

<localRepository>C:\apache-maven-3.3.9\conf\repository</localRepository>

<mirrors>

</mirrors>

</settings>

确保settings.xml中对应的本地仓库中的内容为项目所需的完整jar包。

在MyEclipse中右击项目中pom.xml，选择Run as->mvn install，若输出如下，表示项目maven配置成功：

[INFO] Processing war project

[INFO] Copying webapp resources [D:\Git\qw\_MyEclipse\WebRoot]

[INFO] Webapp assembled in [72932 msecs]

[INFO] Building war: D:\Git\qw\_MyEclipse\target\fiso.war

[INFO]

[INFO] --- maven-install-plugin:2.4:install (default-install) @ zioer5 ---

[INFO] Installing D:\Git\qw\_MyEclipse\target\fiso.war to C:\apache-maven-3.3.9\conf\repository\zonghouqin\zioer5\1.0\zioer5-1.0.war

[INFO] Installing D:\Git\qw\_MyEclipse\pom.xml to C:\apache-maven-3.3.9\conf\repository\zonghouqin\zioer5\1.0\zioer5-1.0.pom

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

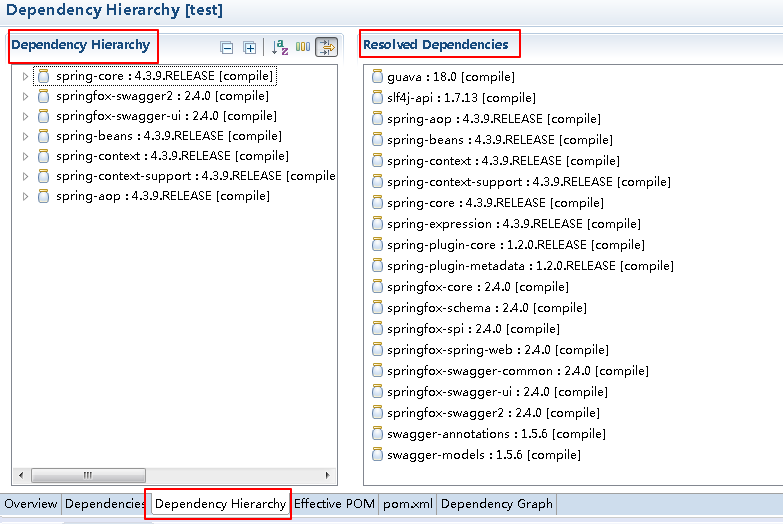
[INFO] Total time: 02:43 min

[INFO] Finished at: 2018-08-21T16:03:55+08:00

[INFO] Final Memory: 45M/427M

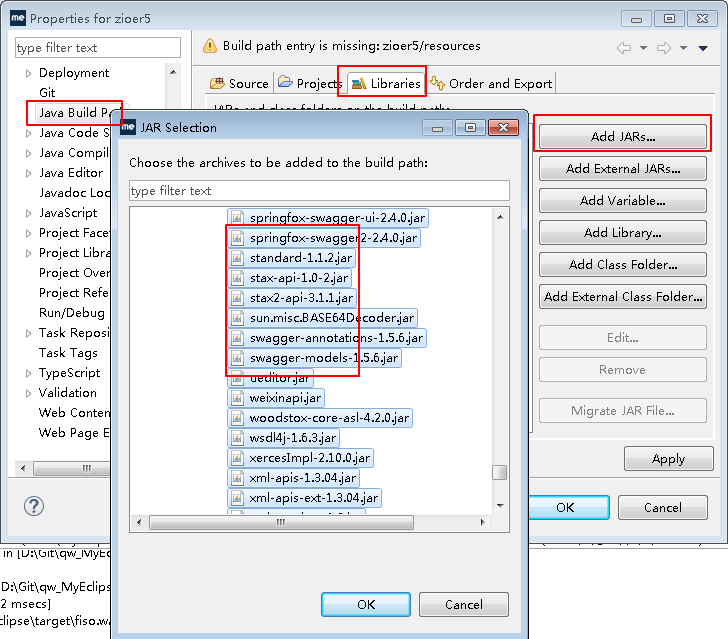
[INFO] ------------------------------------------------------------------------

点击Dependency Hierarchy可以看到Maven依赖如下



* 添加本地依赖

右击项目选择Build Path->Configure Build Path->Add JARs->选择zioer5->导航到zioer5\WebRoot\WEB-INF\lib，勾选lib目录下所有的jar包。



#### 解决Access restriction

在src\main\java\com\zioer\util\Watermark.java中import语句中含有如下语句：

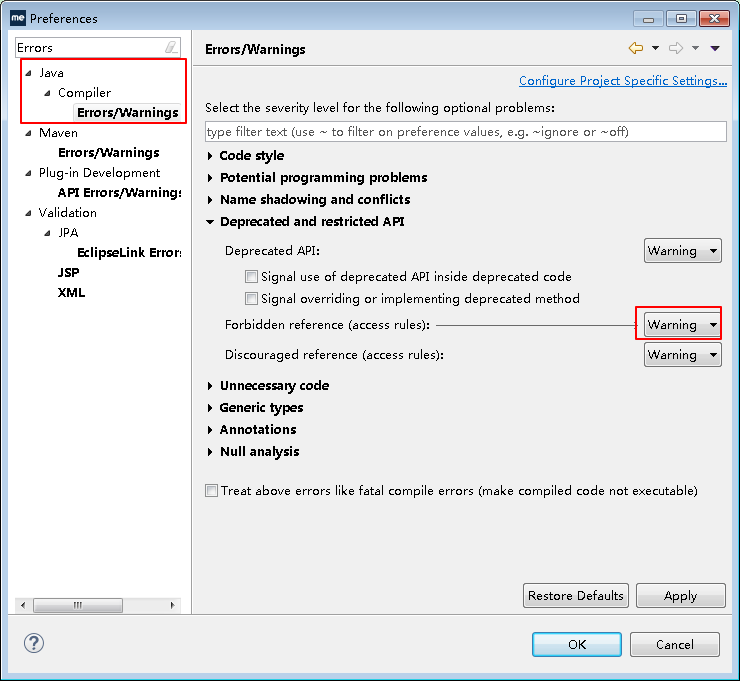
**import** com.sun.image.codec.jpeg.JPEGCodec;

**import** com.sun.image.codec.jpeg.JPEGImageEncoder;

在默认的MyEclipse配置下，这两行输入会报出如下的问题：

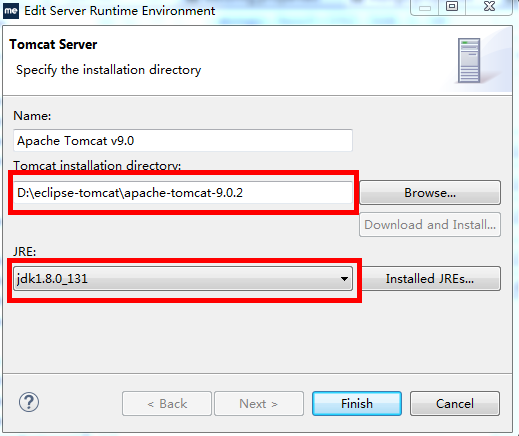
Access restriction: The type 'JPEGCodec' is not API (restriction on required library 'C:\Users\admin\AppData\Local\MyEclipse 2017 CI\binary\com.sun.java.jdk8.win32.x86\_64\_1.8.0.v112\jre\lib\rt.jar')

修改配置如下：



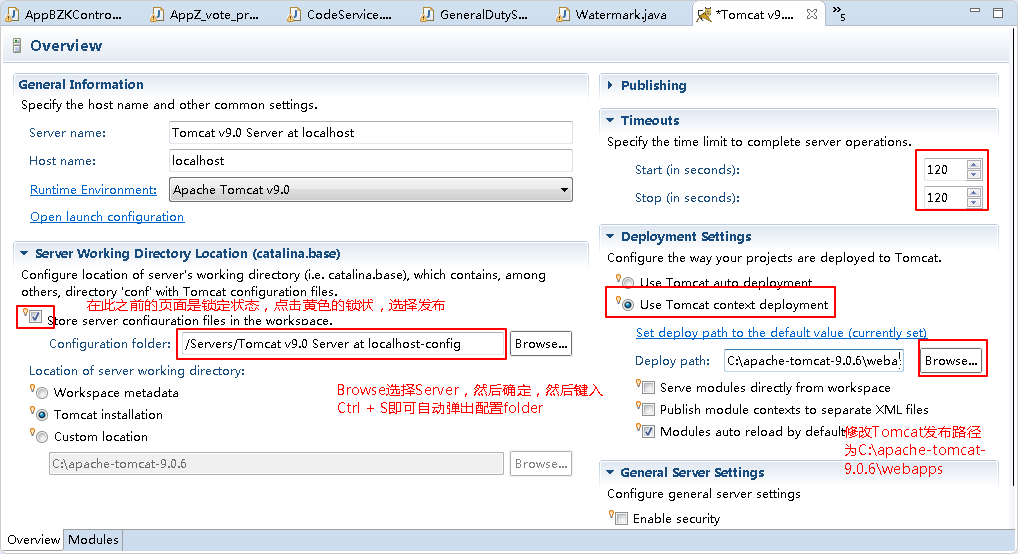
#### 添加Tomcat服务器

点击选择window->Preferences->Servers->Runtime Environments,点击"add",添加tomcat目录及jre目录。



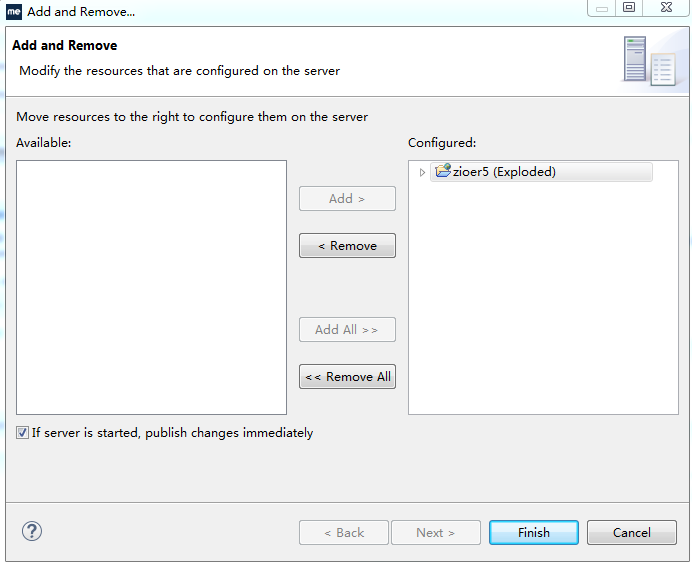
* 修改tomcat的配置

在Servers栏，双击Tomcat V9.0 Server at localhost，修改项目发布路径和超时。



* 添加发布资源

右键点击"Apache Tomcat v9.0 at localhost"->"Add/remove deployments",然后进行如下配置：



#### 修改配置文件

修改qw\resources\shiro.properties文件，修改成如下内容：

#localhost

shiro.loginUrl=http://192.168.0.221:8080/cas/login?service=http://localhost:8080/zioer5/shiro-cas  
shiro.logoutUrl=http://192.168.0.221:8080/cas/logout?service=http://localhost:8080/zioer5/shiro-cas  
shiro.cas.serverUrlPrefix=http://192.168.0.221:8080/cas  
shiro.cas.service=http://localhost:8080/zioer5/shiro-cas  
shiro.failureUrl=/logoutsys  
shiro.successUrl=/login\_toLogin

注：192.168.0.221表示勤务系统项目及其相关的mysql、monogodb数据库所在的ip地址

#### 修改MyEclipse配置内存分配值上限

在MyEclipse.exe同级目录下，有文件MyEclipse.ini，其中-Xmx表示虚拟机占用的最大内存，256m太小，而MyEclipse又太耗内存，因此直接修改为2048m。-Xms表示虚拟机占用系统的最小内存。

C:\Users\admin\AppData\Local\MyEclipse 2017 CI

-vmargs

-Xms256m

-Xmx2048m

-XX:+UseG1GC

-XX:+UseStringDeduplication

-Dosgi.requiredJavaVersion=1.8

-Dosgi.nls.warnings=ignore

### Tomcat启动项目

点击Servers栏下的甲壳虫，即Start the server in the debug mode.以Debug的形式启动项目即可。

### 验证

如果控制台Console输出如下，表示Tomcat已经正常启动

信息: OpenSSL successfully initialized [OpenSSL 1.0.2m 2 Nov 2017]

八月 21, 2018 7:02:39 下午 org.apache.coyote.AbstractProtocol init

信息: Initializing ProtocolHandler ["http-nio-8080"]

八月 21, 2018 7:02:40 下午 org.apache.tomcat.util.net.NioSelectorPool getSharedSelector

信息: Using a shared selector for servlet write/read

八月 21, 2018 7:02:40 下午 org.apache.coyote.AbstractProtocol init

信息: Initializing ProtocolHandler ["ajp-nio-8009"]

八月 21, 2018 7:02:40 下午 org.apache.tomcat.util.net.NioSelectorPool getSharedSelector

信息: Using a shared selector for servlet write/read

八月 21, 2018 7:02:40 下午 org.apache.catalina.startup.Catalina load

信息: Initialization processed in 3591 ms

八月 21, 2018 7:02:40 下午 org.apache.catalina.core.StandardService startInternal

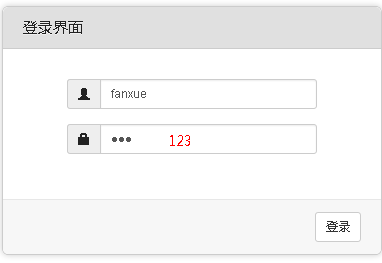
信息: Starting service [Catalina]

八月 21, 2018 7:02:40 下午 org.apache.catalina.core.StandardEngine startInternal

信息: Starting Servlet Engine: Apache Tomcat/9.0.6

在浏览器中输入如下url：

localhost:8080/zioer5



至此，即在MyEclipse中成功部署了勤务系统项目。

# 问题定位和解决

## 本地fiso或者zioer5路径修改

之前项目运行时，使用本地运行路径为zioer5，由于开发人员在提交时，未注意，把自己修改过的zioer5的完整代码提交到远端仓库，因此在此拉取下来的代码无法正常部署和运行，对于该问题解决方式如下：

### shiro.properties

#localhost

shiro.loginUrl=http://192.168.0.221:8081/cas/login?service=http://localhost:8080/fiso/shiro-cas

shiro.logoutUrl=http://192.168.0.221:8081/cas/logout?service=http://localhost:8080/fiso/shiro-cas

shiro.cas.serverUrlPrefix=http://192.168.0.221:8081/cas

shiro.cas.service=http://localhost:8080/fiso/shiro-cas

shiro.failureUrl=/logoutsys

shiro.successUrl=/login\_toLogin

在上述的文件中分别配置了cas端和本地localhost端，其中cas端修改为端口8081，localhost仍为8080端。

注意：上述文件中localhost:8080/fiso片段中的fiso即为要修改的部分，如果想要在本地运行时，使用url包含fiso则在下述的修改中均修改为fiso，否则均修改为zioer5

在本次修改中，修改路径为fiso与远端保持一致，同时要修改以下文件

### header.jsp

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%

String sitePath = "http://localhost:8080/fiso/";

String backPath = "http://localhost:8080/fiso/";

basePath = "http://localhost:8080/fiso/";

%>

上述代码片段为header.jsp中开始的内容，修改本问题时需要同时修改sitePath, backPath， basePath中url中8080/后的内容，上述代码片段已经均修改为fiso，若为zioer5，修改即可，保持一致。

### pom.xml

POM是项目对象模型Project Object Model的简称，是Maven项目中的文件，该文件用于管理源代码、配置文件、开发者的信息和角色、问题追踪信息、组织信息、项目授权，项目的url和依赖关系。因此，如果修改了本地项目运行的url需要同时修改如下的内容：

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat7-maven-plugin</artifactId>

<version>2.1</version>

<configuration>

<path>/fiso</path>

<port>8080</port>

<uriEncoding>UTF-8</uriEncoding>

<server>tomcat7</server>

<warSourceDirectory>WebRoot</warSourceDirectory>

</configuration>

<executions>

<execution>

<id>executable</id>

<goals>

<goal>exec-war-only</goal>

<goal>exec-war</goal>

</goals>

<phase>package</phase>

</execution>

</executions>

</plugin>

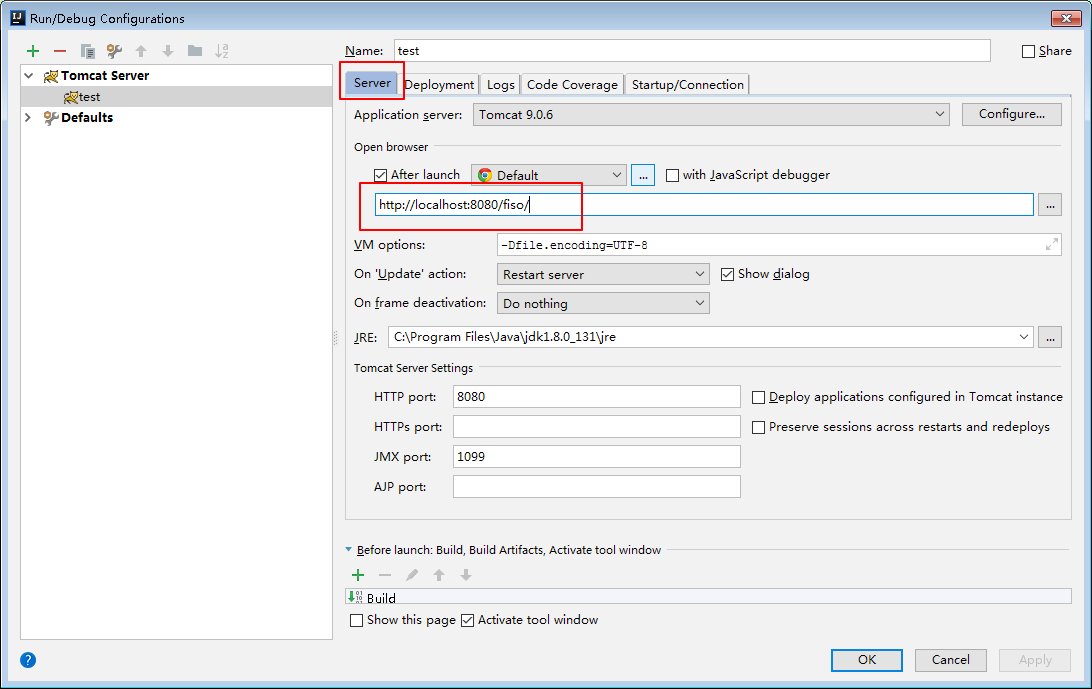
</plugins>

修改上述配置中<path>标记中的内容为/fiso，与修改保持一致。

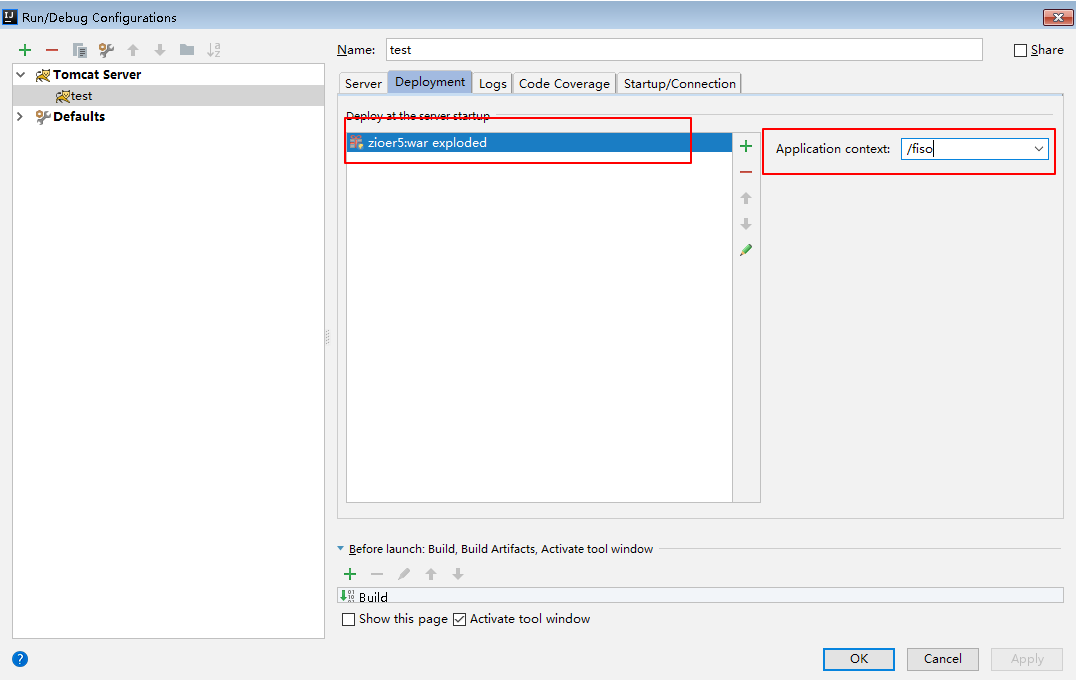
### Tomcat配置

修改完文件中的url部分之后，需要同时修改Tomcat的配置，点击IDEA菜单栏中的Run->Edit Configuration，修改Tomcat配置，如下图：

修改Tomcat Server页面的Server项的Open browser，url的内容



同时修改Tomcat的Deployment项



注意：以上所有位置修改为fiso，均可以修改为zioer5.保持一致即可

# 结束语

本文档涵盖了使用IDEA和MyEclipse部署勤务系统的过程，其中包括JDK的安装，Maven的配置、Tomcat的布置以及IDEA和MyEclipse的属性配置，通过文档指南，可以在快速部署勤务系统的过程中熟悉开发工具的使用以及运行时的整体架构。