

[shiro& SpringSecurity]

[Shiro&&SpringSecurity学习文档]



2018-10-25

[公司名称]

[公司地址]

目录

[1 Shiro 2](#_Toc528659816)

[1.1是什么 2](#_Toc528659817)

[1.2表的设计 2](#_Toc528659818)

[1.3关键代码 6](#_Toc528659819)

[1.3.1 Shiro配置: 6](#_Toc528659820)

[1.3.2 Shiro登录&认证&授权: 8](#_Toc528659821)

[1.3.3 前台菜单隐藏 14](#_Toc528659822)

[2 SpringSecurity 15](#_Toc528659823)

[2.1为什么要使用Maven？ 15](#_Toc528659824)

[2.2后台工程搭建分析 15](#_Toc528659825)

[2.2.1常见的Maven打包方式 15](#_Toc528659826)

[2.2.2工程依赖关系解析 15](#_Toc528659827)

[2.3搭建taotao-parent工程 16](#_Toc528659828)

[2.3.1工程pom.xml文件 16](#_Toc528659829)

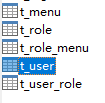
[2.4搭建taotao-common工程 23](#_Toc528659830)

# 1 Shiro

## 1.1是什么

**Apache Shiro™** is a powerful and easy-to-use Java security framework that performs authentication, authorization, cryptography, and session management. With Shiro’s easy-to-understand API, you can quickly and easily secure any application – from the smallest mobile applications to the largest web and enterprise applications.

## 1.2表的设计



**CREATE TABLE `t\_user` (**

**`USER\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '用户ID',**

**`USERNAME` varchar(50) NOT NULL COMMENT '用户名',**

**`PASSWORD` varchar(128) NOT NULL COMMENT '密码',**

**`DEPT\_ID` bigint(20) DEFAULT NULL COMMENT '部门ID',**

**`EMAIL` varchar(128) DEFAULT NULL COMMENT '邮箱',**

**`MOBILE` varchar(20) DEFAULT NULL COMMENT '联系电话',**

**`STATUS` char(1) NOT NULL COMMENT '状态 0锁定 1有效',**

**`CRATE\_TIME` datetime NOT NULL COMMENT '创建时间',**

**`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',**

**`LAST\_LOGIN\_TIME` datetime DEFAULT NULL COMMENT '最近访问时间',**

**`SSEX` char(1) DEFAULT NULL COMMENT '性别 0男 1女',**

**`THEME` varchar(10) DEFAULT NULL COMMENT '主题',**

**`AVATAR` varchar(100) DEFAULT NULL COMMENT '头像',**

**`DESCRIPTION` varchar(100) DEFAULT NULL COMMENT '描述',**

**PRIMARY KEY (`USER\_ID`)**

**) ENGINE=InnoDB AUTO\_INCREMENT=168 DEFAULT CHARSET=utf8;**

**CREATE TABLE `t\_user\_role` (**

**`USER\_ID` bigint(20) NOT NULL COMMENT '用户ID',**

**`ROLE\_ID` bigint(20) NOT NULL COMMENT '角色ID'**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8;**

**CREATE TABLE `t\_role` (**

**`ROLE\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '角色ID',**

**`ROLE\_NAME` varchar(100) NOT NULL COMMENT '角色名称',**

**`REMARK` varchar(100) DEFAULT NULL COMMENT '角色描述',**

**`CREATE\_TIME` datetime NOT NULL COMMENT '创建时间',**

**`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',**

**PRIMARY KEY (`ROLE\_ID`)**

**) ENGINE=InnoDB AUTO\_INCREMENT=66 DEFAULT CHARSET=utf8;**

**CREATE TABLE `t\_role\_menu` (**

**`ROLE\_ID` bigint(20) NOT NULL COMMENT '角色ID',**

**`MENU\_ID` bigint(20) NOT NULL COMMENT '菜单/按钮ID'**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8;**

**CREATE TABLE `t\_menu` (**

**`MENU\_ID` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '菜单/按钮ID',**

**`PARENT\_ID` bigint(20) NOT NULL COMMENT '上级菜单ID',**

**`MENU\_NAME` varchar(50) NOT NULL COMMENT '菜单/按钮名称',**

**`URL` varchar(100) DEFAULT NULL COMMENT '菜单URL',**

**`PERMS` text COMMENT '权限标识',**

**`ICON` varchar(50) DEFAULT NULL COMMENT '图标',**

**`TYPE` char(2) NOT NULL COMMENT '类型 0菜单 1按钮',**

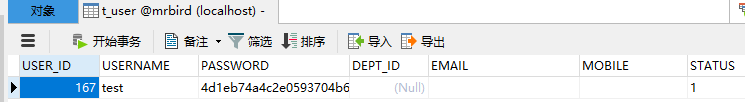
**`ORDER\_NUM` bigint(20) DEFAULT NULL COMMENT '排序',**

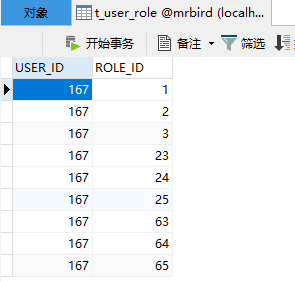
**`CREATE\_TIME` datetime NOT NULL COMMENT '创建时间',**

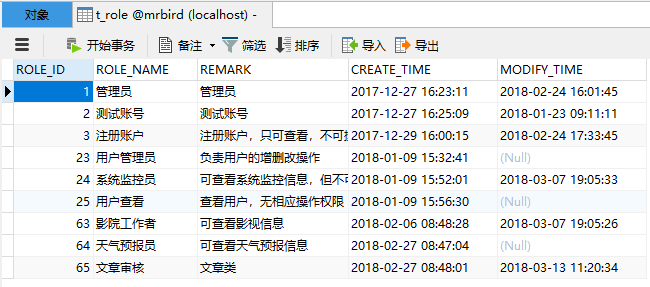
**`MODIFY\_TIME` datetime DEFAULT NULL COMMENT '修改时间',**

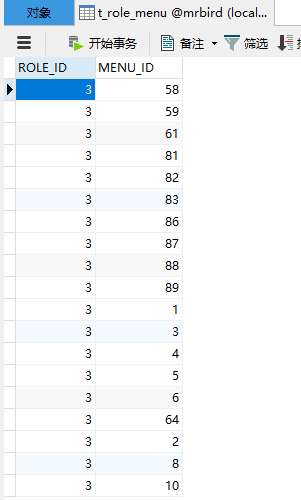
**PRIMARY KEY (`MENU\_ID`)**

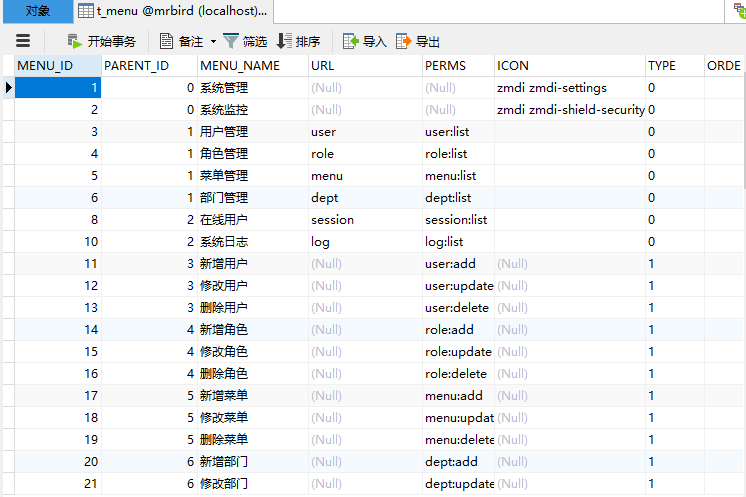
**) ENGINE=InnoDB AUTO\_INCREMENT=115 DEFAULT CHARSET=utf8;**





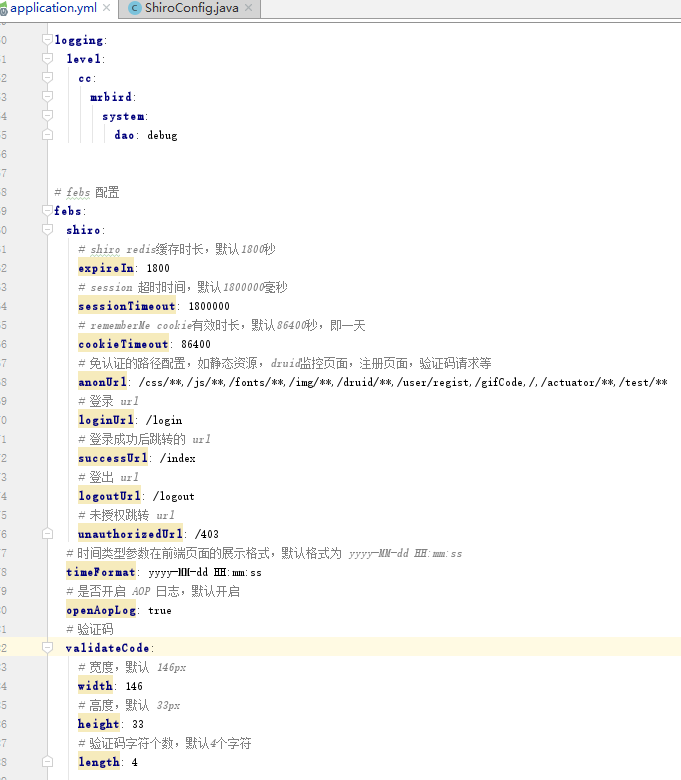


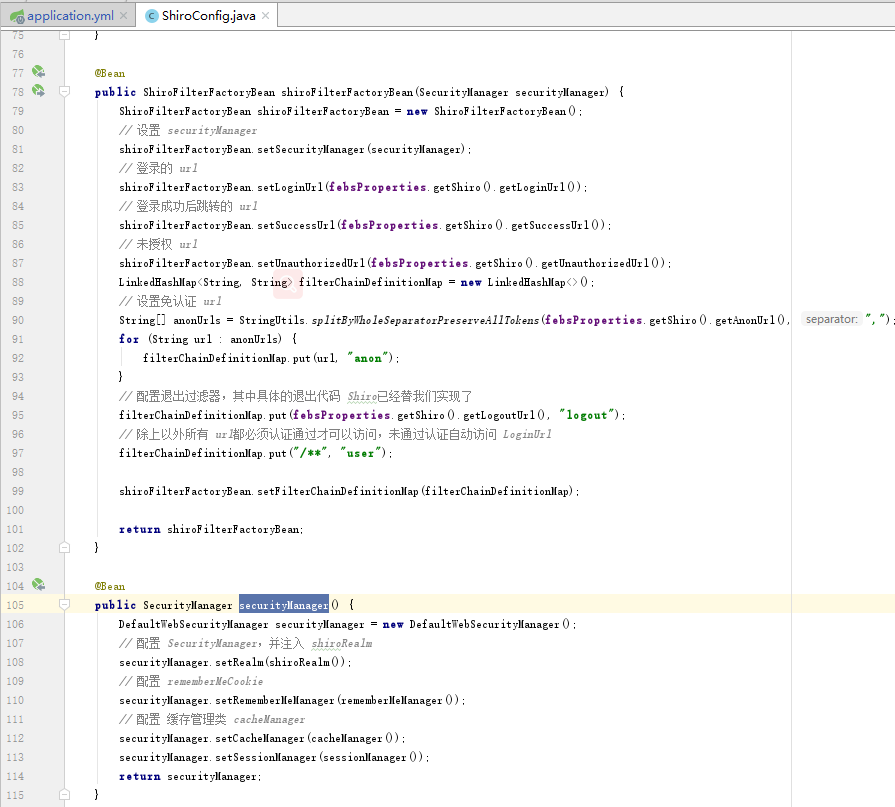




## 1.3关键代码

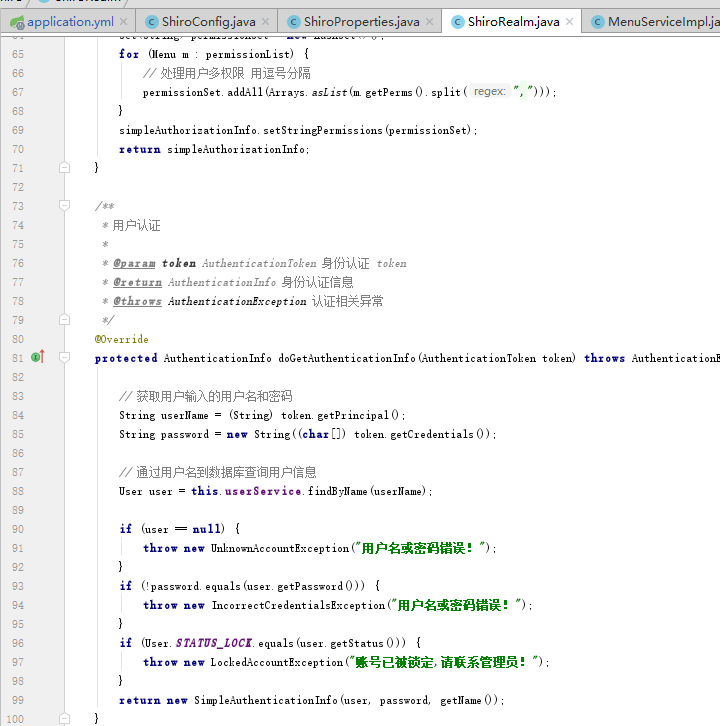
### 1.3.1 Shiro配置:

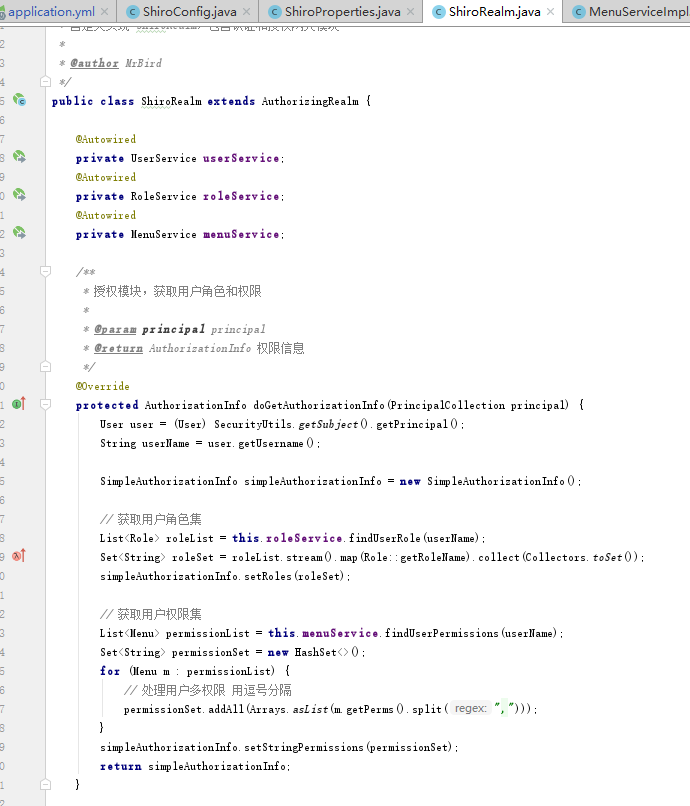




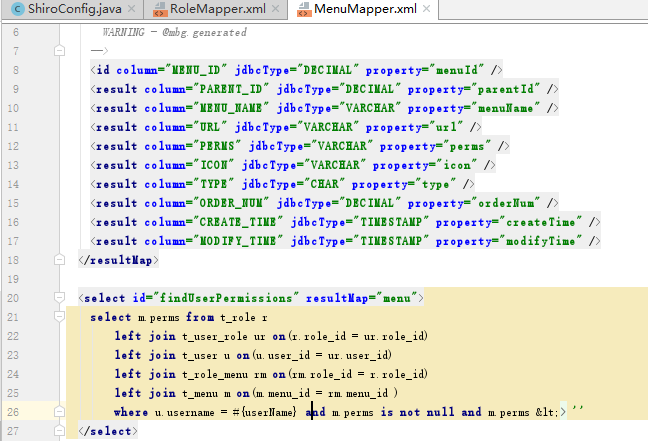
### 1.3.2 Shiro登录&认证&授权:

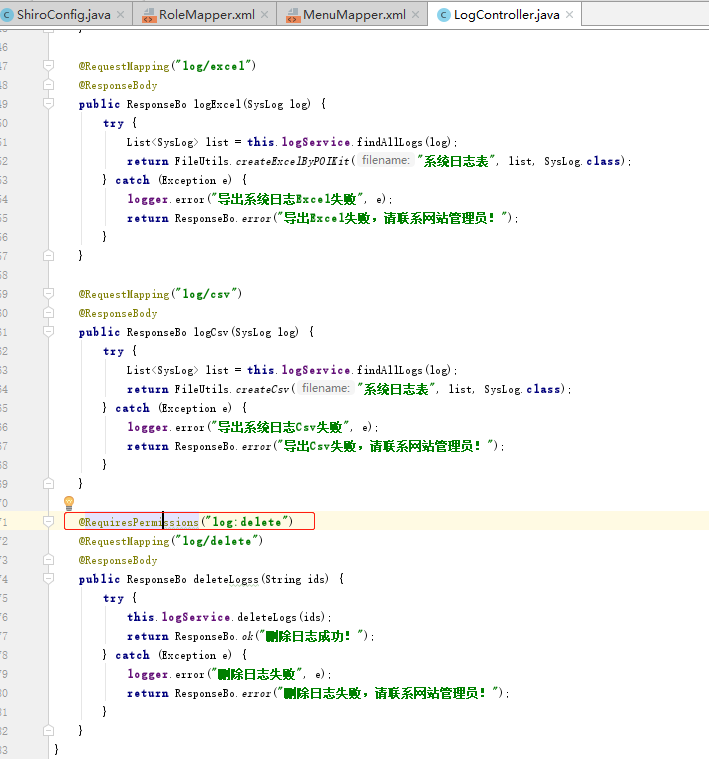




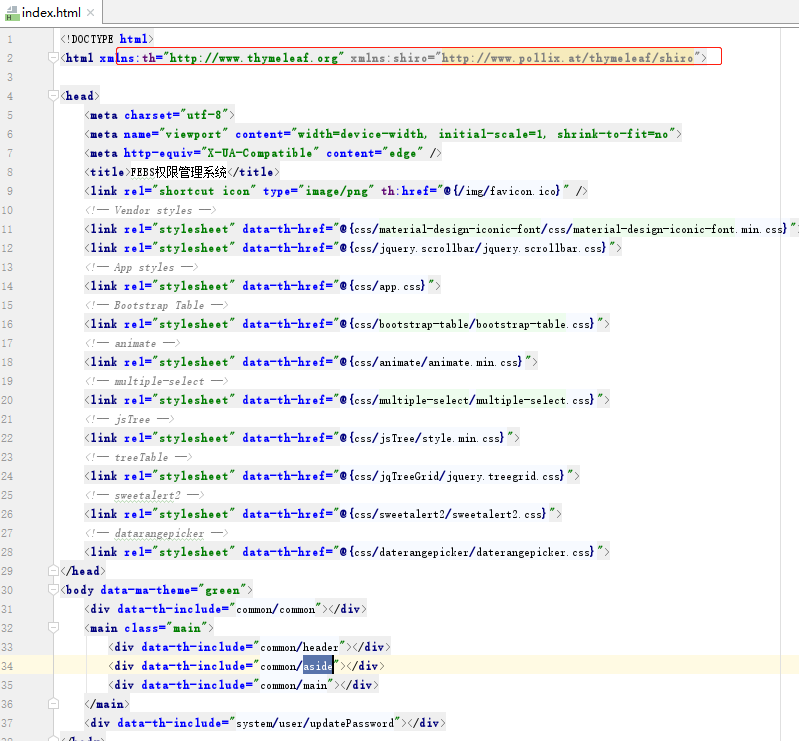








### 1.3.3 前台菜单隐藏





# 2 SpringSecurity

## 2.1为什么要使用Maven？

1. 项目构建。Maven定义了软件开发的整套流程体系，并进行了封装，开发人员只需要指定项目的构建流程，无需针对每个流程编写自己的构建脚本。
2. 依赖管理。除了项目构建，Maven最核心的功能是软件包的依赖管理，能够自动分析项目所需要的依赖软件包，并到Maven中心仓库去下载。
   * 管理Jar包的依赖。
   * 管理工程之间的依赖关系，即可使用Maven依赖其他的工程。

## 2.2后台工程搭建分析

### 2.2.1常见的Maven打包方式

Maven常见的打包方式有：

* jar   
  jar工程，很显然就是pom工程的子工程，由pom工程来管理。
* war   
  war工程是一个web工程，是可以直接放到tomcat下运行的工程。
* pom   
  pom工程一般都是父工程，管理jar包的版本、maven插件的版本、统一的依赖管理，它是一个聚合工程。其实说白了它只有一个pom.xml文件，一般是作为父工程出现的，只是定义了一些依赖、插件、还有一些版本号等等

### 2.2.2工程依赖关系解析

taotao-parent：父工程，其打包方式是pom，主要管理jar包的版本号。项目中的所有工程都应该继承父工程。

* taotao-common：存放一些通用的工具类、通用的pojo。其打包方式为jar，被其他的工程所依赖。
* taotao-manager：服务层工程，又称之为聚合工程或pom工程。
  + taotao-manager-pojo：这是一个独立的模块，其打包方式是jar。
  + taotao-manager-dao：这是一个独立的模块，其打包方式是jar。
  + taotao-manager-interface：打包方式是jar。
  + taotao-manager-service：打包方式是war，主要为了方便部署。

## 2.3搭建taotao-parent工程

### 2.3.1工程pom.xml文件

可以看出这个pom文件内容很简单，就这么几个——当前这个工程的坐标以及其打包方式。现在我们需要在这个父工程里面管理jar包的版本号，怎样管理jar包的版本号呢？那就需要在pom文件中进行配置了。下面我提供一个比较全面的pom.xml文件，文件内容可能有点多，初次接触这种商城大项目的人是不可能一蹰而就写完的，肯定是在编写项目的过程中，需要用到哪个jar包，就在pom.xml文件进行配置。

* ：

servlet-api和javax-servlet-api的区别：

servlet.jar 是servlet 3.1.0 版本之前的地址   
javax.servlet-api.jar 是servlet 3.1.0 版本之后的地址

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.taotao</groupId>

<artifactId>taotao-parent</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>pom</packaging>

<!-- 集中定义依赖版本号 -->

<properties>

<junit.version>4.12</junit.version>

<spring.version>4.2.4.RELEASE</spring.version>

<mybatis.version>3.2.8</mybatis.version>

<mybatis.spring.version>1.2.2</mybatis.spring.version>

<mybatis.paginator.version>1.2.15</mybatis.paginator.version>

<mysql.version>5.1.32</mysql.version>

<slf4j.version>1.6.4</slf4j.version>

<jackson.version>2.4.2</jackson.version>

<druid.version>1.0.9</druid.version>

<httpclient.version>4.3.5</httpclient.version>

<jstl.version>1.2</jstl.version>

<!--<servlet-api.version>2.5</servlet-api.version>-->

<servlet-api.version>3.1.0</servlet-api.version>

<jsp-api.version>2.0</jsp-api.version>

<joda-time.version>2.5</joda-time.version>

<commons-lang3.version>3.3.2</commons-lang3.version>

<commons-io.version>1.3.2</commons-io.version>

<commons-net.version>3.3</commons-net.version>

<pagehelper.version>3.4.2-fix</pagehelper.version>

<jsqlparser.version>0.9.1</jsqlparser.version>

<commons-fileupload.version>1.3.1</commons-fileupload.version>

<jedis.version>2.7.2</jedis.version>

<solrj.version>4.10.3</solrj.version>

<dubbo.version>2.5.3</dubbo.version>

<zookeeper.version>3.4.7</zookeeper.version>

<zkclient.version>0.1</zkclient.version>

<activemq.version>5.11.2</activemq.version>

<freemarker.version>2.3.23</freemarker.version>

<quartz.version>2.2.2</quartz.version>

</properties>

<dependencyManagement>

<dependencies>

<!-- 时间操作组件 -->

<dependency>

<groupId>joda-time</groupId>

<artifactId>joda-time</artifactId>

<version>${joda-time.version}</version>

</dependency>

<!-- Apache工具组件 -->

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-lang3</artifactId>

<version>${commons-lang3.version}</version>

</dependency>

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-io</artifactId>

<version>${commons-io.version}</version>

</dependency>

<dependency>

<groupId>commons-net</groupId>

<artifactId>commons-net</artifactId>

<version>${commons-net.version}</version>

</dependency>

<!-- Jackson Json处理工具包 -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>${jackson.version}</version>

</dependency>

<!-- httpclient -->

<dependency>

<groupId>org.apache.httpcomponents</groupId>

<artifactId>httpclient</artifactId>

<version>${httpclient.version}</version>

</dependency>

<!-- quartz任务调度框架 -->

<dependency>

<groupId>org.quartz-scheduler</groupId>

<artifactId>quartz</artifactId>

<version>${quartz.version}</version>

</dependency>

<!-- 单元测试 -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

<!-- 日志处理 -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!-- Mybatis的分页插件 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>${mybatis.version}</version>

</dependency>

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-spring</artifactId>

<version>${mybatis.spring.version}</version>

</dependency>

<dependency>

<groupId>com.github.miemiedev</groupId>

<artifactId>mybatis-paginator</artifactId>

<version>${mybatis.paginator.version}</version>

</dependency>

<dependency>

<groupId>com.github.pagehelper</groupId>

<artifactId>pagehelper</artifactId>

<version>${pagehelper.version}</version>

</dependency>

<!-- MySql -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>${mysql.version}</version>

</dependency>

<!-- 连接池 -->

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>druid</artifactId>

<version>${druid.version}</version>

</dependency>

<!-- Spring -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aspects</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jms</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context-support</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- JSP相关 -->

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>${jstl.version}</version>

</dependency>

<!-- <dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>${servlet-api.version}</version>

<scope>provided</scope>

</dependency>-->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>${servlet-api.version}</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jsp-api</artifactId>

<version>${jsp-api.version}</version>

<scope>provided</scope>

</dependency>

<!-- 文件上传组件 -->

<dependency>

<groupId>commons-fileupload</groupId>

<artifactId>commons-fileupload</artifactId>

<version>${commons-fileupload.version}</version>

</dependency>

<!-- Redis客户端 -->

<dependency>

<groupId>redis.clients</groupId>

<artifactId>jedis</artifactId>

<version>${jedis.version}</version>

</dependency>

<!-- solr客户端 -->

<dependency>

<groupId>org.apache.solr</groupId>

<artifactId>solr-solrj</artifactId>

<version>${solrj.version}</version>

</dependency>

<!-- dubbo相关 -->

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>dubbo</artifactId>

<version>${dubbo.version}</version>

</dependency>

<dependency>

<groupId>org.apache.zookeeper</groupId>

<artifactId>zookeeper</artifactId>

<version>${zookeeper.version}</version>

</dependency>

<dependency>

<groupId>com.github.sgroschupf</groupId>

<artifactId>zkclient</artifactId>

<version>${zkclient.version}</version>

</dependency>

<dependency>

<groupId>org.apache.activemq</groupId>

<artifactId>activemq-all</artifactId>

<version>${activemq.version}</version>

</dependency>

<dependency>

<groupId>org.freemarker</groupId>

<artifactId>freemarker</artifactId>

<version>${freemarker.version}</version>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<finalName>${project.artifactId}</finalName>

<plugins>

<!-- 资源文件拷贝插件 -->

<plugin>

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

<artifactId>maven-resources-plugin</artifactId>

<version>2.7</version>

<configuration>

<encoding>UTF-8</encoding>

</configuration>

</plugin>

<!-- java编译插件 -->

<plugin>

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

<artifactId>maven-compiler-plugin</artifactId>

<version>3.2</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

<encoding>UTF-8</encoding>

</configuration>

</plugin>

</plugins>

<pluginManagement>

<plugins>

<!-- 配置Tomcat插件 -->

<plugin>

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

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

<version>2.2</version>

</plugin>

</plugins>

</pluginManagement>

</build>

</project>

大家一看是不是傻眼了，我靠！这么多，大家心都凉了一截，这可能是大家对使用maven构建工程感到陌生，说实话，我本人也是第一次使用maven来构建工程，我也很难受，不抱怨了，下面对一些我感到特别陌生的点讲解一下。

* <dependencyManagement>...</dependencyManagement>：依赖管理，管理的就是jar包的版本号，它里面并不是真正的依赖，如果你继承com.taotao这个工程，会发现什么jar包也没有依赖过来。工程中用到的jar包都应该在这里有定义。
* httpclient：也是一个工具包，相当于使用Java API模拟浏览器向服务器发送请求，然后再回送响应。浏览器有的功能，httpclient也有。
* <scope>provided</scope>：除了打包时不打包，别的时候都会全程参与，编译、测试、运行时候都会用到，部署时也不会用它，因为我们要把这个war包部署到tomcat里面去，tomcat里面都会提供servlet和jsp的API，如果把它也打包到war包里面，可能会有冲突。
* <pluginManagement>...</pluginManagement>：pluginManagemen也是管理插件的版本，这里面并不是对插件实际的依赖，而只是管理插件的版本。

## 2.4搭建taotao-common工程

taotao-common工程主要存放通用的工具类和pojo，它本身是一个子工程，所以它需要继承父工程，并且它的打包方式是jar。注意，taotao-common工程并不是一个独立的模块。

可以看到新建的taotao-common工程的jre的版本默认便是jdk1.7，这是由于我们在taotao-parent父工程中定义了JDK的编译版本，taotao-parent父工程关于JDK编译版本的依赖配置如下：

<!-- java编译插件 -->

<plugin>

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

<artifactId>maven-compiler-plugin</artifactId>

<version>3.2</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

<encoding>UTF-8</encoding>

</configuration>

</plugin>

即代表父工程，我们还可以看见<groupId>com.taotao</groupId>和<version>0.0.1-SNAPSHOT</version>下面有黄色的下划线，这意味着可以删掉它们，因为我们在父工程里面已经定义了组织名和版本号了，当然了你不删除掉也是没关系的。   
接下来，我们来给taotao-common工程的pom.xml文件添加依赖，common顾名思义，就是大家都会调用的工具类，就像公共组件。我们从taotao-parent工程中复制公共组件的依赖到taotao-common的pom.xml文件当中，taotao-common的pom.xml文件的内容如下，由于taotao-parent工程里面已经统一管理版本号了，因此我们在taotao-common工程中就不用再定义组件的版本了。保存后，maven会自动下载相关依赖。

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>com.taotao</groupId>

<artifactId>taotao-parent</artifactId>

<version>0.0.1-SNAPSHOT</version>

</parent>

<groupId>com.taotao</groupId>

<artifactId>taotao-common</artifactId>

<version>0.0.1-SNAPSHOT</version>

<!-- 依赖工具类的jar包 -->

<dependencies>

<!-- 时间操作组件 -->

<dependency>

<groupId>joda-time</groupId>

<artifactId>joda-time</artifactId>

</dependency>

<!-- Apache工具组件 -->

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-lang3</artifactId>