谷粒商城

张 晨

# 一、课程简介

## 1、为什么我们要讲电商？

因为就互联网平台来说，电商网站有很多典型的特征：

* 访问量大
* 数据量大
* 典型互联网业务
* 模块多且独立适合分布式架构
* 涉及支付

## 我们能从这个项目中学到什么？

### 要掌握的知识

|  |
| --- |
| spring springmvc springboot mysql redis nginx linux dubbo zookeeper activemq elasticsearch FastDFS (oss) |

### 需要掌握的解决方案

|  |
| --- |
| 分布式架构 缓存应用 异步通信 分布式事务 分布式存储 单点登录 第三方支付 |

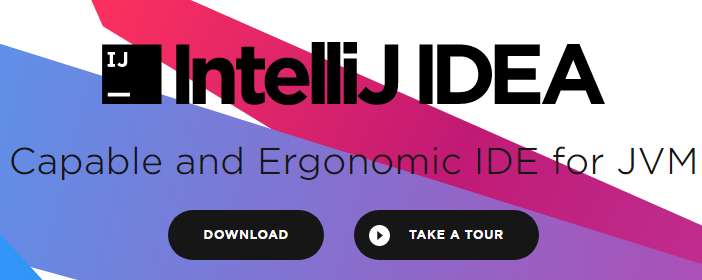
课前说明：

|  |
| --- |
|  |

# 二、 Intellij idea

## 1 介绍

IDEA 全称 IntelliJ IDEA，是java语言开发的集成环境，IntelliJ在业界被公认为最好的java开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE支持、各类版本工具(git、svn、github等)、JUnit、CVS整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的。IDEA是JetBrains公司的产品，这家公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主。它的旗舰版本还支持HTML，CSS，PHP，MySQL，Python等。免费版只支持Java等少数语言



比起Eclipse的好处：

|  |
| --- |
| 1. 快 全程跟踪提示速度快 2. 提示范围广 html js xml properties maven 3. 整合 git、maven、spring 4. 快捷键 比较方便 |

## 2 安装

解压就可以。

第二种方式 需要有网络的情况下才能注册成功

且在注册成功的情况下,没有网络只能打开第一次,如果打开多次,有可能会需要重新联网注册

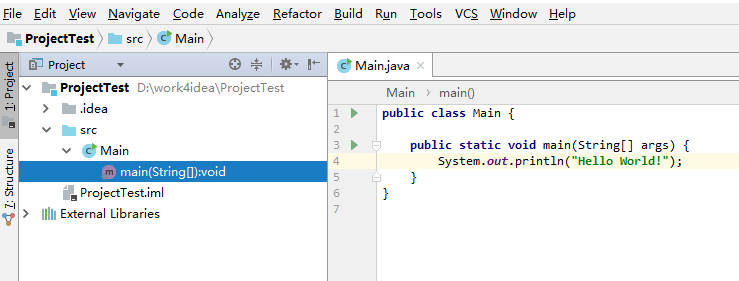
进入ide主页面，help-register-license server,然后输入 http://idea.iteblog.com/key.php

## 3 使用

### 3.1 Project与module

在idea中没有workspace的概念，每一个窗口只能打开一个Project。 对于单一工程的项目，直接建一个Project在其下面开发就好了。

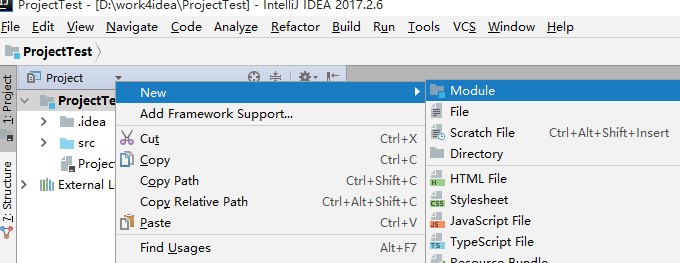
单一工程的项目：

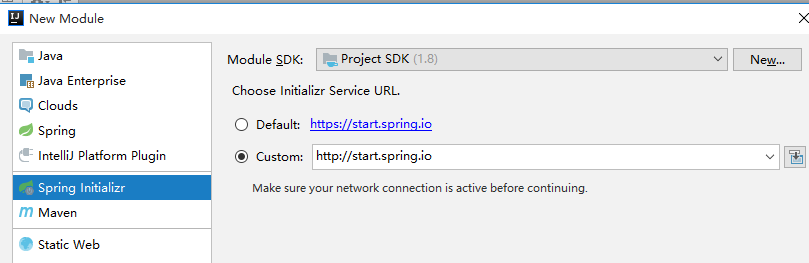


但是我们知道现在稍微大一点的项目都是多项目的分布式部署的，那么岂不是每个子工程都要打开一个窗口？

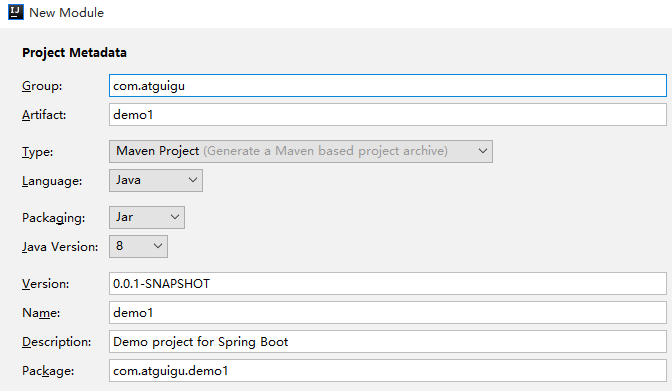
这时候就需要用到Module的概念，Module是项目的子模块，可以独立运行的工程，当一个多项目组成的系统时，Project下本身可以不拥有代码，而是作为一种顶级的管理目录，所有的代码都放到各个module之中。

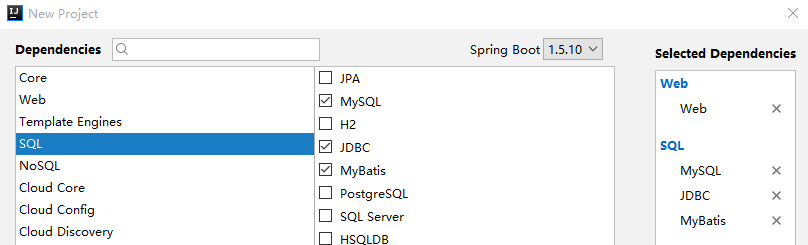
下面我们在这个Project下增加Module，

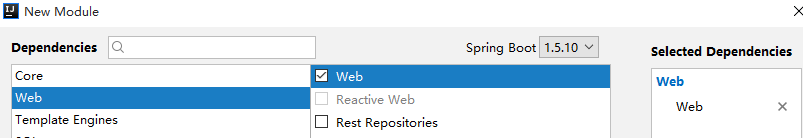


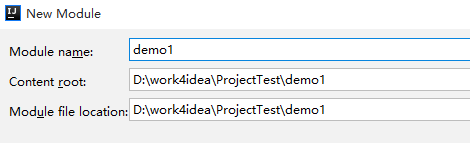


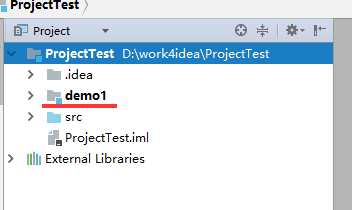
这个时候因为要从网上读取模板所以务必保持联网状态，Spring Initalizr是springboot工程的模板。







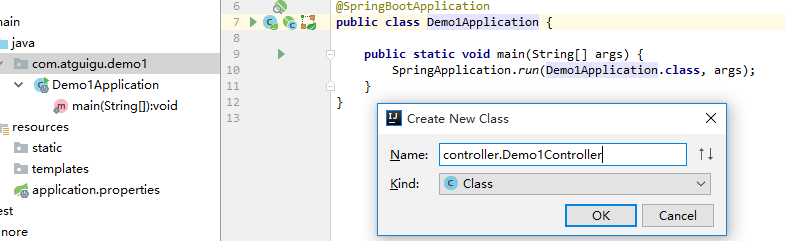




这时候看到Project中多了一个demo1的Module的。

其实这时候Project工程下的src就没什么用了，可以删掉。

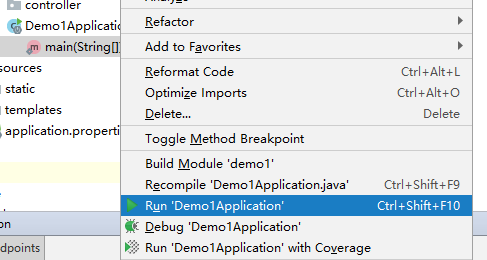
模块建立好了，我们就来用springmvc标签建一个controller看看好不好使。



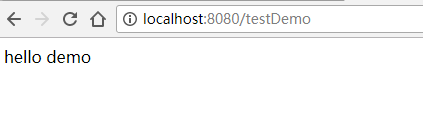
controller代码

|  |
| --- |
| @Controller **public class** Demo1Controller {  @ResponseBody  @RequestMapping(**"testDemo"**)  **public** String testDemo(){  **return "hello demo"**;  } } |

运行Demo1Application中的main方法

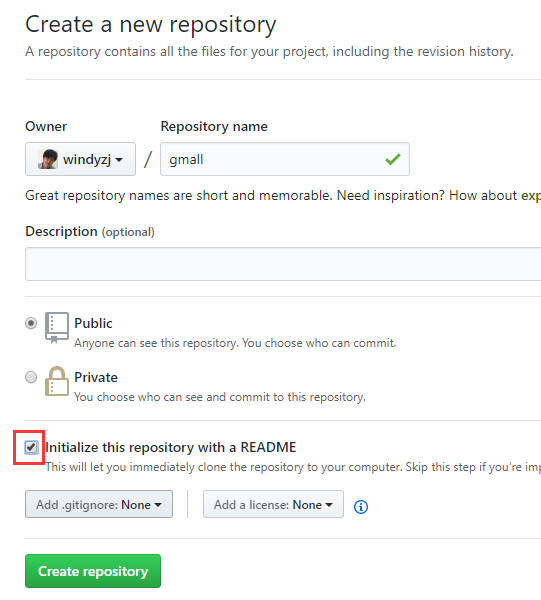


用浏览器测试：

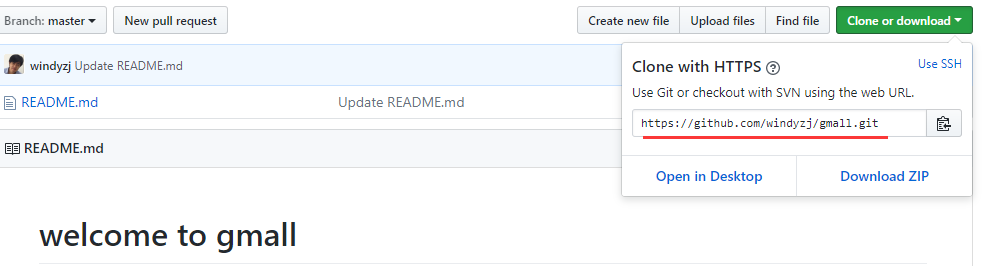


### 3.2 从Git中clone项目

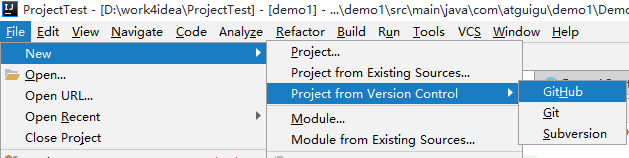
首先要去GitHub上创建一个项目

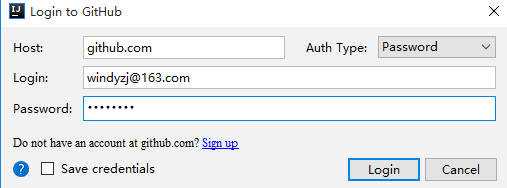


注意的地方是记得加一个README，这样clone下来的工程就不是空的了。

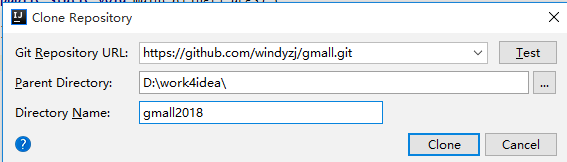


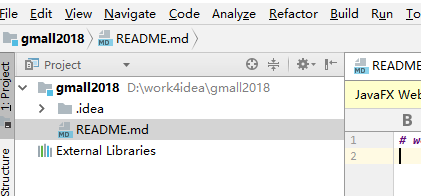
这个就是咱们的仓库地址，咱们来进行第一次复制



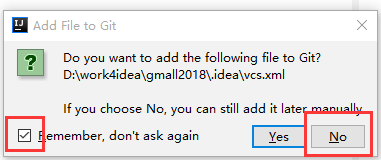


在github.com 网站中新建一个空项目



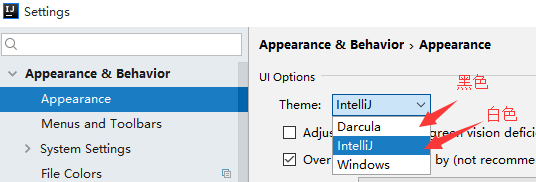


如果弹出提示框如下，问你是否要自动提交某些文件，请一律选NO,且不再提醒。否则系统会自动提交一些不必要的文件。



## 4、界面颜色风格

setting->Appearance



## 5 idea的快捷键

### 5.1 常用快捷键

智能补全 引包 **alt+Enter**

由方法自动生成返回值变量 **ctrl+alt+v**

跳到方法的实现类 **ctrl+alt+b | ctrl+alt+鼠标左键**

查看方法参数 **ctrl+p**

查看方法文档 ctrl+q

复制行 ctrl+D

删除行 ctrl+Y

跳转到上一个/下一个位置 ctrl+alt+左右

大小写切换 ctrl+shift+u

打开最近编辑过的文档 ctrl+E

### 5.2 Debug:

F8 执行下一行 (相当于eclipse的 F6)

F7 跳入内部 (相当于eclipse的 F5)

F9 继续执行 (相当于eclipse的 F8)

热部署 ctrl+shift+F9 （仅debug模式）

### 5.3 搜索

全文搜索文本 ctrl+shift+f

全文替换文本 ctrl+shift+r

搜索类 ctrl+n

任何地方搜索 **双击shift**

### 5.4 快速录入

查看快速录入列表 ctrl+j

foreach iter

普通for循环 fori

循环数组 itar

迭代器遍历 itco

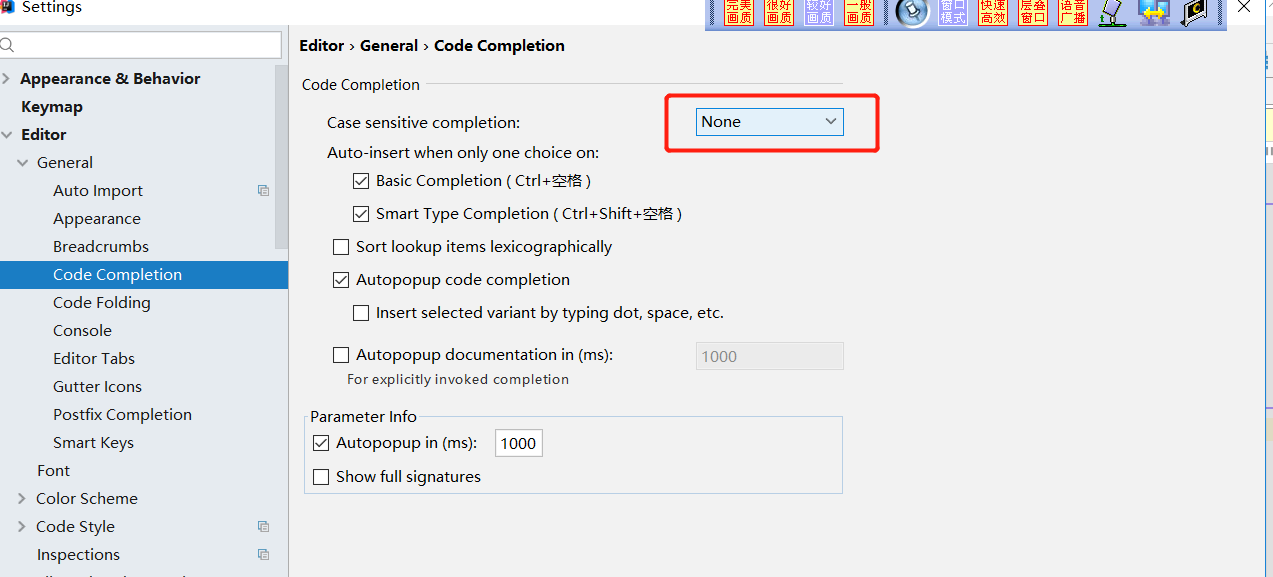
psvm 主函数

pfs 常量

生成代码块： try/ if / for/ while/ synchronized

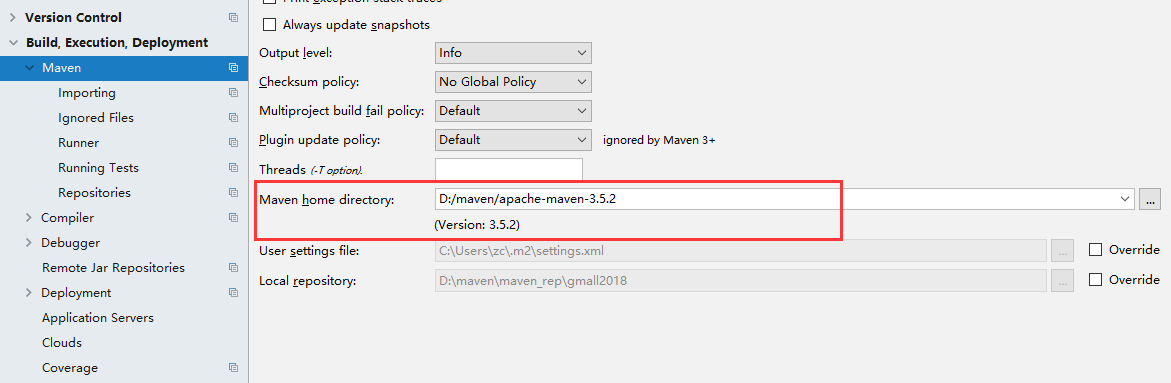
ctrl+alt+t

### 5.5 设置大小写



## 7 配置maven

设置默认配置！



# 三、EZDML工具 类似于数据库设计工具

配置： 工具—>修改ini配置

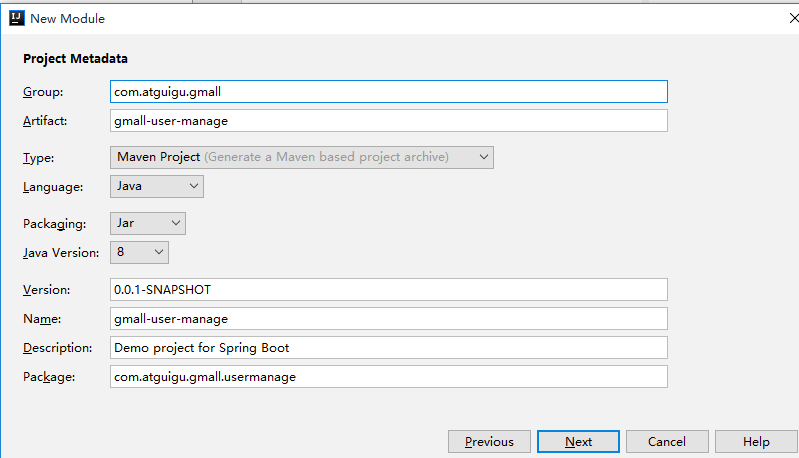
|  |
| --- |
| [DefaultFieldTypes]  [CustFieldTypes]  1=bigint  2=decimal  [DbConn] |

**前提必须在mysql 数据库中有 一个叫 gmall 的数据库！**

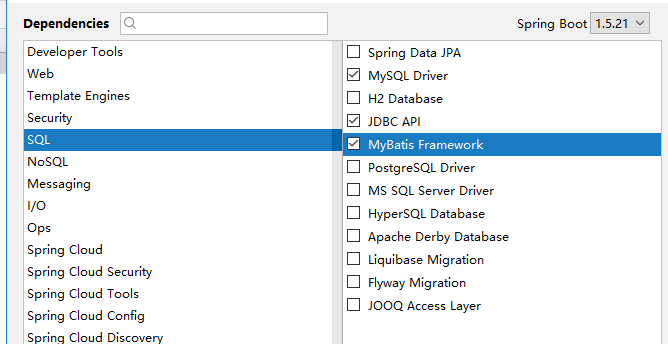
导入 gmall.sql

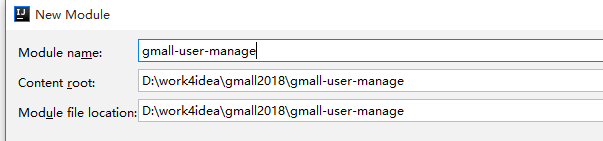
# 四 通用Mapper的使用

## 1、搭建module



依赖选Web和Mysql, Jdbc,MyBatis





注意Module位置要在Project路径下面

## 2、配置通用Mapper

在pom.xml文件中，加入

|  |
| --- |
| *<!-- 通用mapper -->* <**dependency**>  <**groupId**>tk.mybatis</**groupId**>  <**artifactId**>mapper-spring-boot-starter</**artifactId**>  <**version**>1.2.3</**version**>  <**exclusions**>  <**exclusion**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-jdbc</**artifactId**>  </**exclusion**>  </**exclusions**> </**dependency**>  <**dependency**>  <**groupId**>org.projectlombok</**groupId**>  <**artifactId**>lombok</**artifactId**> </**dependency**> |

GmallUserManageApplication.java 中增加注解

|  |
| --- |
| @SpringBootApplication @MapperScan(basePackages = **"com.atguigu.gmall.usermanage.mapper"**) **public class** GmallOrderServiceApplication {   **public static void** main(String[] args) {  SpringApplication.*run*(GmallOrderServiceApplication.**class**, args);  } }  **注意通用mapper是tk.mybatis.mapper** |

## 3、配置数据源

在application.properties中

|  |
| --- |
| **spring.datasource.url**=**jdbc:mysql://localhost:3306/gmall?characterEncoding=UTF-8**  **spring.datasource.username**=**root spring.datasource.password**=**123123** |

表结构

|  |
| --- |
| CREATE TABLE `user\_info` (  `id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '编号',  `login\_name` varchar(200) DEFAULT NULL COMMENT '用户名称',  `nick\_name` varchar(200) DEFAULT NULL COMMENT '用户昵称',  `passwd` varchar(200) DEFAULT NULL COMMENT '用户密码',  `name` varchar(200) DEFAULT NULL COMMENT '用户姓名',  `phone\_num` varchar(200) DEFAULT NULL COMMENT '手机号',  `email` varchar(200) DEFAULT NULL COMMENT '邮箱',  `head\_img` varchar(200) DEFAULT NULL COMMENT '头像',  `user\_level` varchar(200) DEFAULT NULL COMMENT '用户级别',  PRIMARY KEY (`id`)  ) ENGINE=InnoDB AUTO\_INCREMENT=1000 DEFAULT CHARSET=utf8 COMMENT='用户表' |

## 4、代码开发

|  |  |  |
| --- | --- | --- |
| 包 | 类 | 说明 |
| controller | UserManageController | web |
| service | UserManageService | 接口 |
| service.impl | UserManageServiceImpl | 实现类 |
| bean | UserInfo | 实体bean |
| mapper | UserInfoMapper | mapper接口 |
|  |  |  |

### 4.1 bean

|  |
| --- |
| @Data @NoArgsConstructor  **public class** UserInfo **implements** Serializable{  @Id  @Column  @GeneratedValue(strategy = GenerationType.IDENTITY)  **private** String **id**;  @Column  **private** String **loginName**;  @Column  **private** String **nickName**;  @Column  **private** String **passwd**;  @Column  **private** String **name**;  @Column  **private** String **phoneNum**;  @Column  **private** String **email**;  @Column  **private** String **headImg**;  @Column  **private** String **userLevel**;  } |

注意：@Column 和@Id 都是javax.persistence包中的

**技巧 idea快捷键：alt+insert可以快速插入getter 和setter**

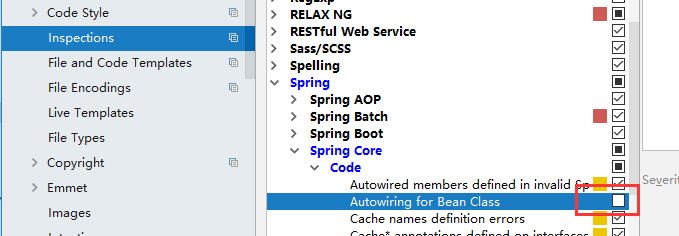
### 4.2 Mapper

|  |
| --- |
| **public interface** UserInfoMapper **extends** Mapper<UserInfo> { } |

注意： Mapper也是引用tk.mybatis.mapper.common.Mapper包中的

Idea 有的时候校验@Autowired不准 可以把校验关闭

settings -> Inspections -> spring->spring core -> code-> Autowiring for Bean class



### 4.4 接口声明UserService

|  |
| --- |
| **public interface** UserService {   List<UserInfo> getUserInfoListAll();   **void** addUser(UserInfo userInfo);   **void** updateUser(UserInfo userInfo);   **void** updateUserByName(String name,UserInfo userInfo);   **void** delUser(UserInfo userInfo);  } |

### 4.5 接口实现UserServiceImpl

|  |
| --- |
| @Service **public class** UserServiceImpl **implements** UserService {  @Autowired UserMapper **userMapper**;  @Override **public** List<UserInfo> getUserInfoListAll() {  List<UserInfo> userInfoList = **userMapper**.selectAll();  **return** userInfoList; }  **public** UserInfo getUserInfo(String id){  UserInfo userInfo = **userMapper**.selectByPrimaryKey(id);  **return** userInfo; }  @Override **public void** addUser(UserInfo userInfo) {  **userMapper**.insert(userInfo); }  @Override **public void** updateUser(UserInfo userInfo) {  **userMapper**.updateByPrimaryKeySelective(userInfo); }  @Override **public void** updateUserByName(String name, UserInfo userInfo) {   Example example = **new** Example(UserInfo.**class**);  example.createCriteria().andEqualTo(**"name"**,name);  **userMapper**.updateByExample(userInfo,example);  }  @Override **public void** delUser( String id) {  **userMapper**.deleteByPrimaryKey(id); }  } |

### 4.6 Controller

|  |
| --- |
| @RestController **public class** UserController {   @Autowired  UserService **userService**;   @PostMapping(**"addUser"**)  **public** String addUser( UserInfo userInfo){  **userService**.addUser(userInfo);  **return "success"**;  }   @GetMapping(**"allUser"**)  **public** List<UserInfo> getAllUser(){  **return userService**.getUserInfoListAll();  }   @GetMapping(**"userInfo"**)  **public** UserInfo getUser(String id){  **return userService**.getUserInfo(id);  }   @PostMapping(**"delUser"**)  **public** String deleteUser(String id){  **userService**.delUser(id);  **return "success"**;  }   @PostMapping(**"updateUser"**)  **public** String updateUser(UserInfo userInfo){  *// userService.updateUser(userInfo);* **userService**.updateUserByName(userInfo.getName(),userInfo);  **return "success"**;  }  } |

# 五 hosts工具

用来修改：

C:\Windows\System32\drivers\etc\hosts文件



如果系统中没有该文件

New 创建