客户名字和logo

餐厅订餐系统—系统详细设计文档

Xu LIN

日期：2015-01-07

文档版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 修订日期 | 修订人 | 审核人 | 变更内容 |
| 1.0 | 2014-11-23 | Xu Lin |  | 初始化文档结构 |
| 1.1 | 2014-12-5 | Xu Lin |  | 新增用户管理模块和数据库 |
|  | 2014-12-8 | Xu Lin |  | 用户管理模块的用户需求更新和新增类说明 |
|  | 2014-12-26 | LUO ZHI |  | 1. 增加章节2 开发环境配置大纲 2. 增加章节3项目目录结构说明大纲 3. 增加章节4用户管理模块大纲 |
|  | 2014/12/30 | Xu Lin |  | 增加章节4的内容 |
|  | 2014/12/31 | Xu Lin |  | 更新章节4 的内容 |
|  | 2015-01-07 | LUO ZHI |  | 增加章节2 |

# 目录

[目录 3](#_Toc408404955)

[1 文档介绍 1](#_Toc408404956)

[1.1 目的 1](#_Toc408404957)

[1.2 范围 1](#_Toc408404958)

[1.3 缩写词列表 1](#_Toc408404959)

[1.4 参考内容 1](#_Toc408404960)

[2 开发环境配置 2](#_Toc408404961)

[2.1 下载及安装JDK 2](#_Toc408404962)

[2.2 下载及安装Eclipse 4](#_Toc408404963)

[2.2.1 设置网络代理 5](#_Toc408404964)

[2.2.2 使用空格缩进 5](#_Toc408404965)

[2.2.3 打开代码行号和80字符标尺 7](#_Toc408404966)

[2.2.4 关闭自动编译 9](#_Toc408404967)

[2.2.5 修改默认工作区编码 9](#_Toc408404968)

[2.3 下载及安装Maven 10](#_Toc408404969)

[2.4 下载及安装MySQL 12](#_Toc408404970)

[2.5 下载及安装Tomcat 18](#_Toc408404971)

[2.6 下载及安装Git 19](#_Toc408404972)

[2.7 导入项目 20](#_Toc408404973)

[2.8 编译项目 20](#_Toc408404974)

[3 项目目录结构说明 21](#_Toc408404975)

[4 权限管理模块 22](#_Toc408404976)

[4.1 用户登录 22](#_Toc408404977)

[4.1.1 时序图 22](#_Toc408404978)

[4.1.2 代码构成 22](#_Toc408404979)

[4.1.3 类说明 23](#_Toc408404980)

[4.1.4 配置文件说明 27](#_Toc408404981)

[4.1 27](#_Toc408404986)

[4.1.5 数据库脚本说明 29](#_Toc408404990)

[5 模块设计 30](#_Toc408404991)

[5.1 功能模板设计说明 30](#_Toc408405001)

[5.2 用户管理 30](#_Toc408405002)

[5.2.1 目标 30](#_Toc408405003)

[5.2.2 用户需求 30](#_Toc408405004)

[5.2.3 类图 33](#_Toc408405005)

[5.2.4 时序图 33](#_Toc408405006)

[5.2.5 组件 35](#_Toc408405007)

[5.2.6 类说明 35](#_Toc408405008)

[5.2.7 其他说明 39](#_Toc408405009)

[6 数据库设计 39](#_Toc408405010)

[6.1 用户表 39](#_Toc408405011)

[6.2 角色表 39](#_Toc408405012)

[6.3 功能表 40](#_Toc408405013)

[6.4 用户角色关联表 40](#_Toc408405014)

[6.5 角色功能关联表 40](#_Toc408405015)

[6.6 菜品分类表 40](#_Toc408405016)

[6.7 菜品表 41](#_Toc408405017)

[6.8 订单表 41](#_Toc408405018)

[6.9 订单条目表 42](#_Toc408405019)

[6.10 会员等级表 42](#_Toc408405020)

[7 接口设计 42](#_Toc408405021)

[7.1 内部接口 42](#_Toc408405027)

[7.2 外部接口 42](#_Toc408405028)

[7.2.1 接口说明 42](#_Toc408405029)

[7.2.2 调用方式 42](#_Toc408405030)

[8 系统认证安全设计 42](#_Toc408405031)

[8.1 访问控制设计 42](#_Toc408405032)

[8.2 访问控制流程图 42](#_Toc408405033)

[8.3 访问控制流程说明 42](#_Toc408405034)

[8.4 系统性能设计 43](#_Toc408405035)

[8.5 概念 43](#_Toc408405036)

[8.6 EnCache配置 43](#_Toc408405037)

[9 系统出错处理 43](#_Toc408405038)

[9.1 出错信息输出 43](#_Toc408405039)

[9.2 出错处理对策 43](#_Toc408405040)

[10 附录 43](#_Toc408405041)

# 文档介绍

## 目的

## 范围

## 缩写词列表

## 参考内容

《餐厅订餐系统\_系统需求文档.docx》

# 开发环境配置

eOrder订餐系统使用Java/Android技术开发前后端应用，所以下面依次介绍在项目说使用到的开发环境和工具的安装配置。

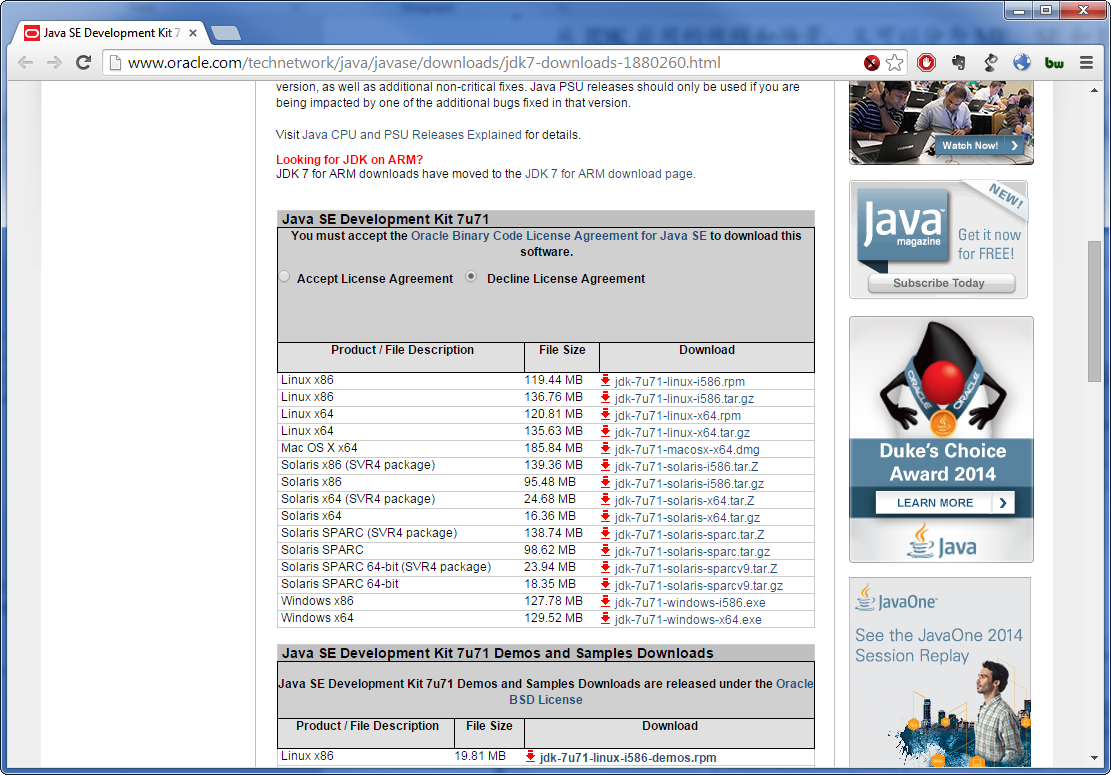
## 下载及安装JDK

JDK是Java Development Kit （Java开发工具箱）的简写，是使用Java进行开发的必备工具。JDK不仅提供了Java核心类库还提供了编译，调试，打包等命令行工具。

很多公司都有JDK产品，标准的JDK是由Sun Microsystem公司拥有，但现在Sun已经被Oracle公司收购；Oracle公司有自有JDK产品的JRocket；另外还有OpenJDK。

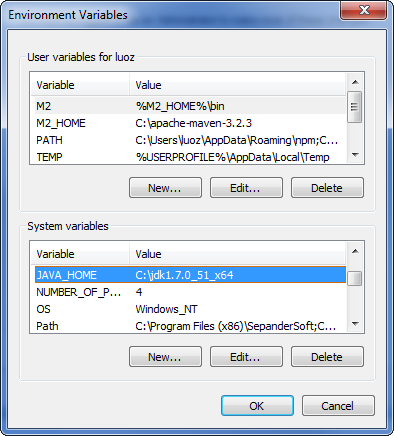
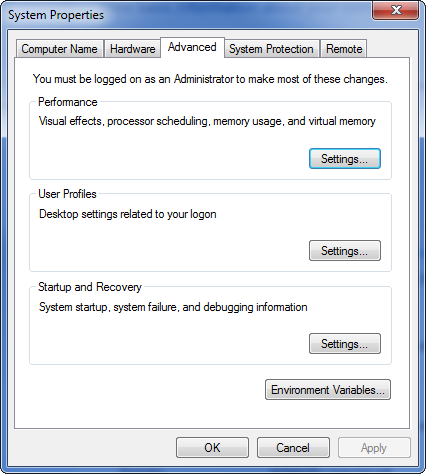
从JDK应用的规模和场景，又可以分为ME，SE和EE三个版本，依次对应于移动程序，标准程序和企业级程序

本项目采用标准JDK SE 7 64位版本，可以从Oracle公司的网站下载(<http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>) 。在下载之前必须选择**Accept License Agreement**选项，才能获取下载链接，然后点击**jdk-7u71-windows-x64.exe**下载

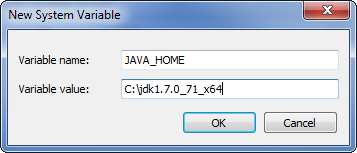


下载完成后以管理员身份运行安装包进行安装，在选择安装路径的时候建议选择安装在某个分区的根目录，例如**C:\jdk1.7.0\_71\_x64**，方便接下来的配置。

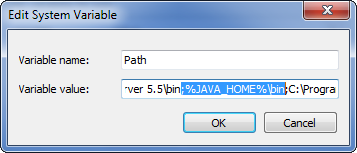
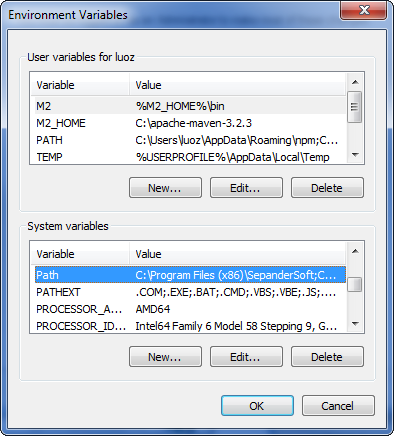
安装完成之后，需要为JDK配置环境变量，打开**控制面板->系统->高级->环境变量…**



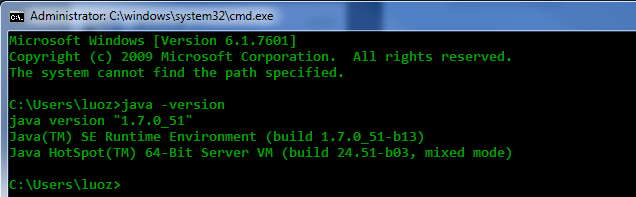
点击**新建…**按钮，在变量名中输入**JAVA\_HOME**，在变量值中填入JDK的安装路径，点击**确定**按钮



编辑系统变量中的**Path**，在变量值输入框中添加**;%JAVA\_HOME%\bin**，点击确定按钮。



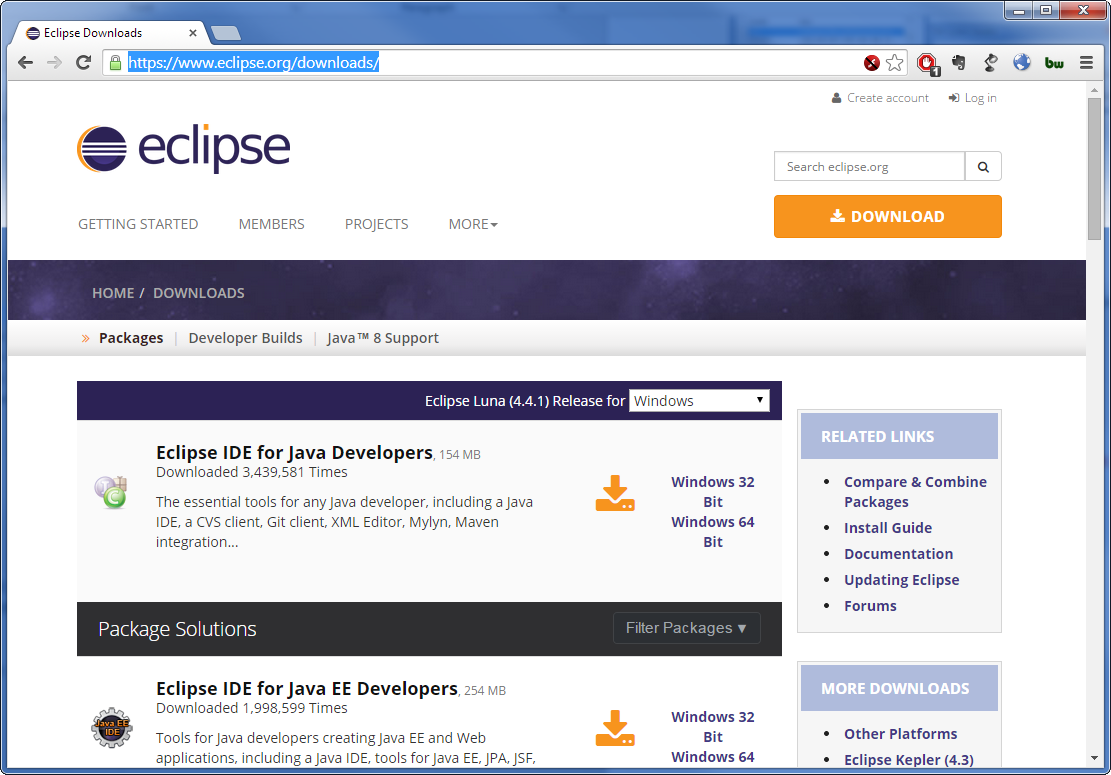
打开命令行窗口，输入**java –version**得到如图所示的文本信息，则说明JDK安装以及环境变量设定正确



注：截图中安装的JDK SE 7小版本为51，文中给出下载链接为71，这是没有关系的，用户安装和配置的时候自行将51字样改为71即可

## 下载及安装Eclipse

Eclipse是一个集成开发环境框架，在这个集成开发框架下提供了对C++、Java等众多开发语言的支持。项目需要用到Eclipse IDE for Java Developers，可以到Eclipse的官方网站下载（https://www.eclipse.org/downloads/），最新的Eclipse版本名称是Luna，下载**Windows 64 Bit** 版本



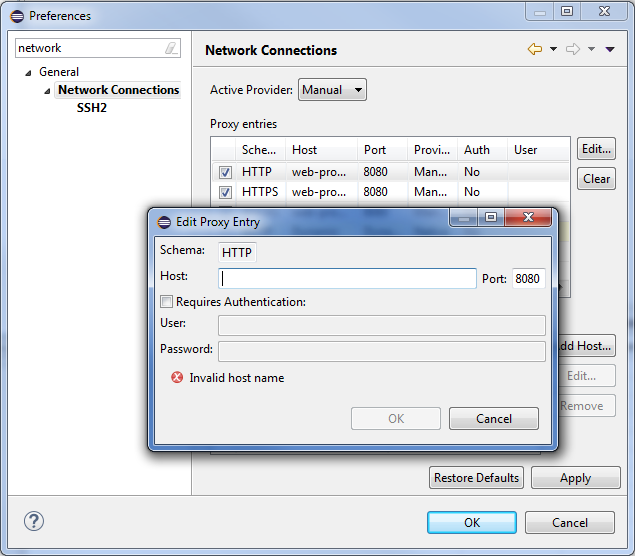
因为没有用到JavaEE的高级特性，诸如JPA，JSF等，所以下载Java Developers版本即可，这个版本已经内置了Git，Maven支持，很好的满足项目开发的需求。下载的安装包为zip格式，使用解压缩软件，比如7Zip（http://www.7-zip.org/）将压缩包解压，建议解压至某个分区的根目录，例如**C:\eclipse**，运行安装目录下的**eclipse.exe**即可打开Eclipse集成开发工具。

第一次运行eclipse会提示设定工作区（workspace），将分区上的任一目录设置为默认工作区即可，例如：**C:\MyWorkspace**

在使用eclipse进行项目开发之前，还需要对eclipse进行一些设定

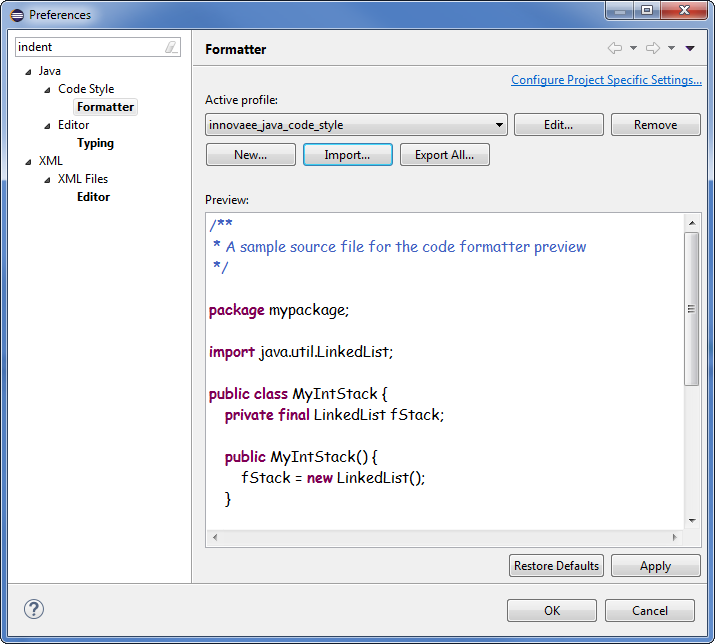
### 设置网络代理

很多时候eclipse需要访问网络来获取最新的依赖或者插件更新，但是如果需要通过代理服务器才能访问网络时，则需要进行代理设置，点击**Window->Preferences**，在左上角的输入框中输入**network**，选中**Network Connections**，在右边面板中的**Active Provider**下拉框中选择**Manual**，勾选要配置的代理类型，点击**Edit…**按钮，在弹出的对话框中输入代理服务器地址和端口，点击**OK**按钮即可

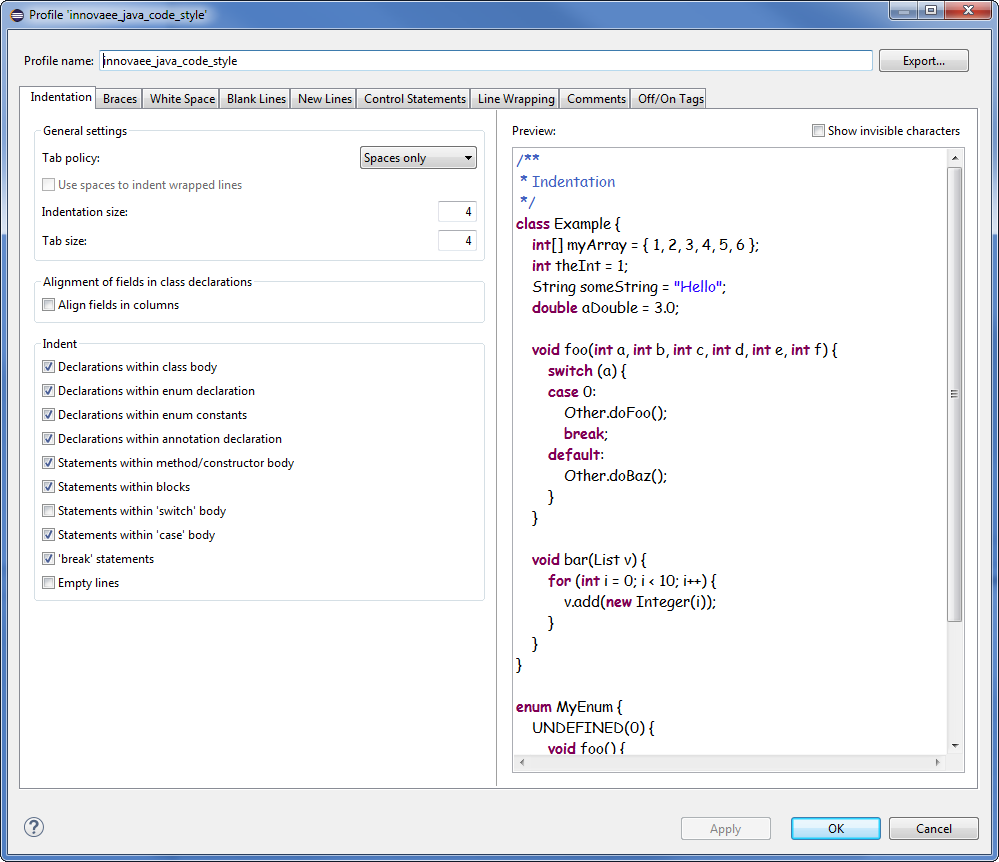
****

### 使用空格缩进

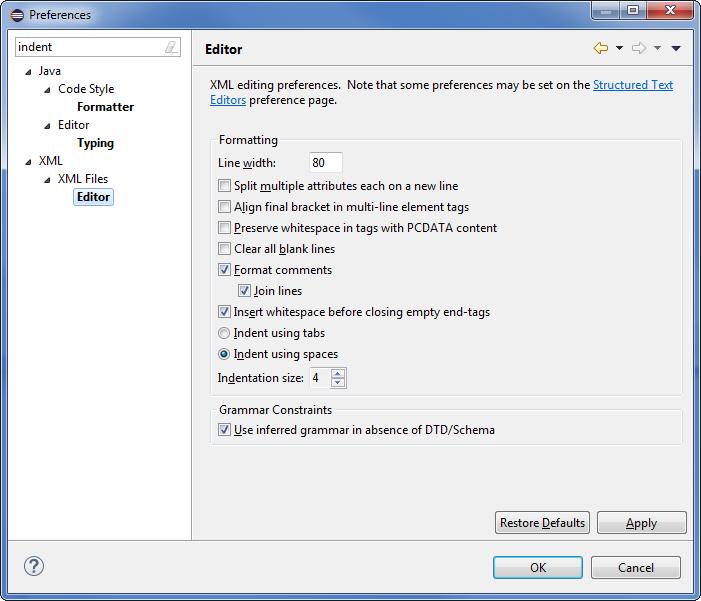
因为不同系统对缩进的处理不同，所以使用空格而不是制表符可以让代码在不同的系统上的表现形式一致。点击**Window->Preferences**，在左上角的输入框中输入**indent**，在左边菜单中选择**Code Style->Formatter**，在右边面板中选择**Import…**按钮，导入项目编码风格定义文件**innovaee\_java\_code\_style.xml**



导入完成后，在**Active profile**下拉框中选择导入的自定义编码风格，点击**Edit…**按钮，确认在**General settings**下的**Tab Policy**下拉框值为**Spaces only**，且**Indentation size**和**Tab size**的值都为4

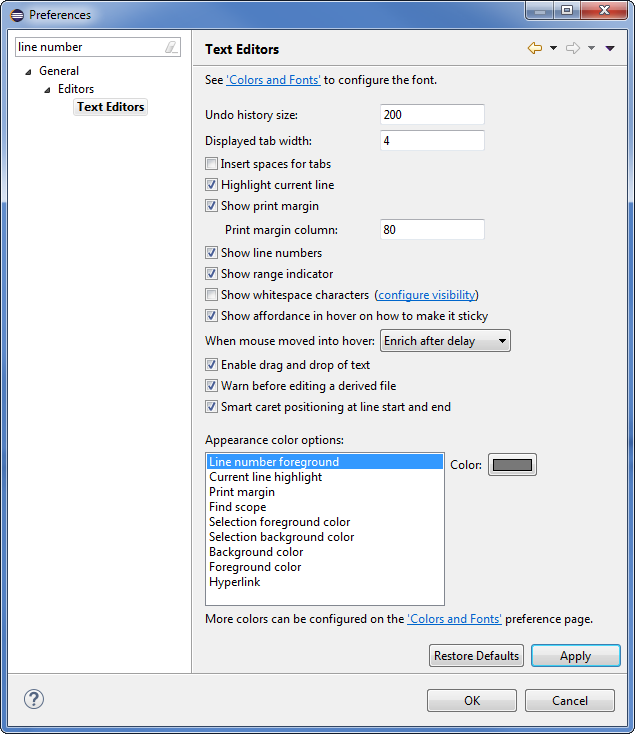


选择**XML->XML Files->Editors**，在左边面板中选择**Indent using spaces**，**Indentation size**设置为**4**

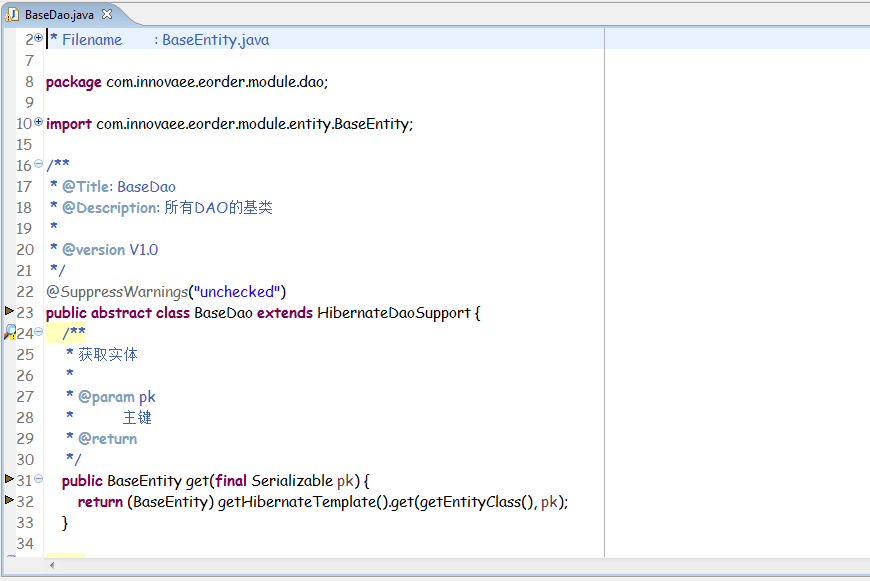


### 打开代码行号和80字符标尺

在编辑区显示行号有利于调试定位，而80字符标尺有利于在编写长语句的时候进行有效换行。点击**Window->Preferences**，在左上角的输入框中输入**line number**，点击左边菜单中的**General->Editors->Text Editors**，在右边面板中勾选**Show print margin**，在**Print margin column**输入框输入80；勾选**Show line numbers**



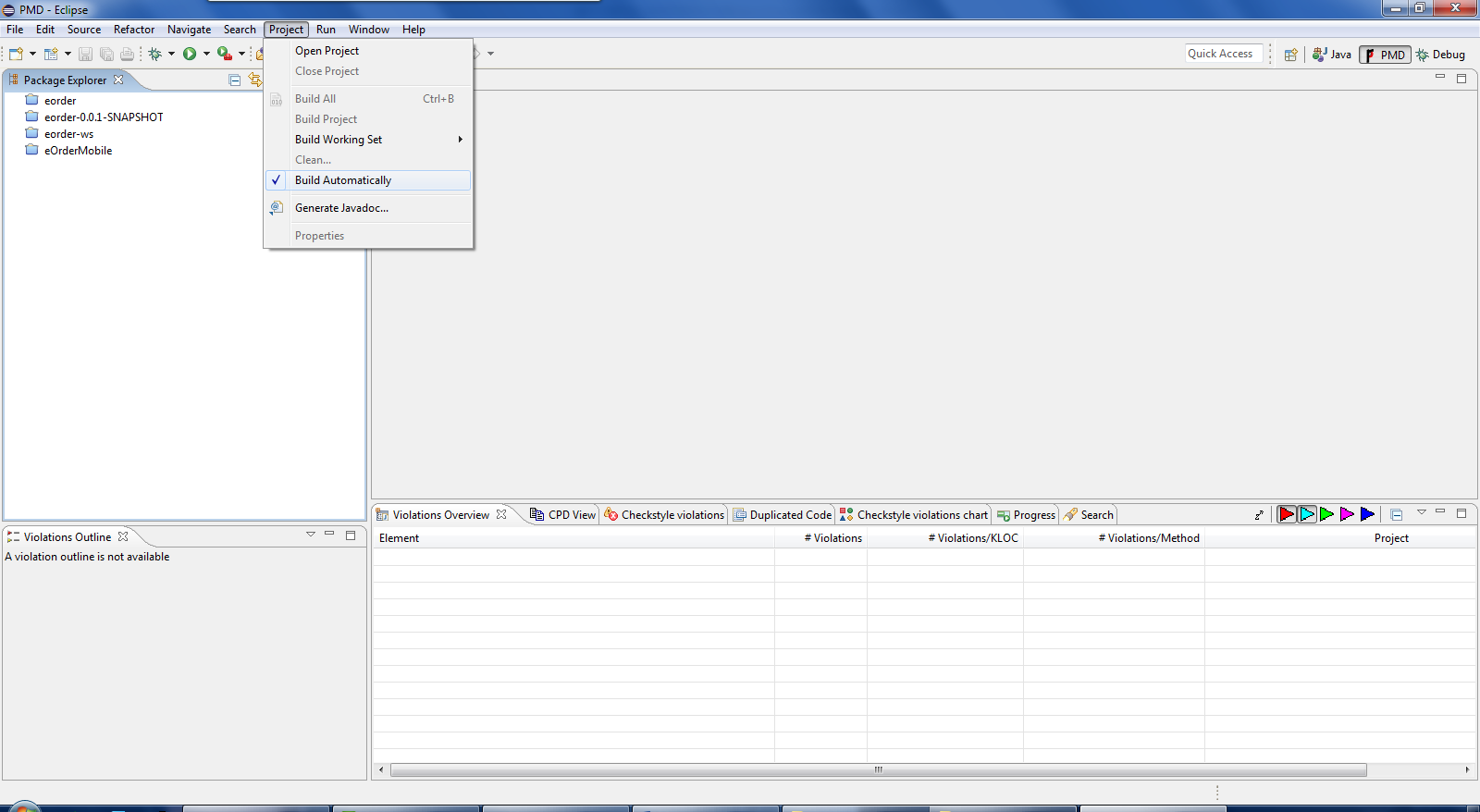
点击OK按钮后，打开Java文件会看到如下效果，



### 关闭自动编译

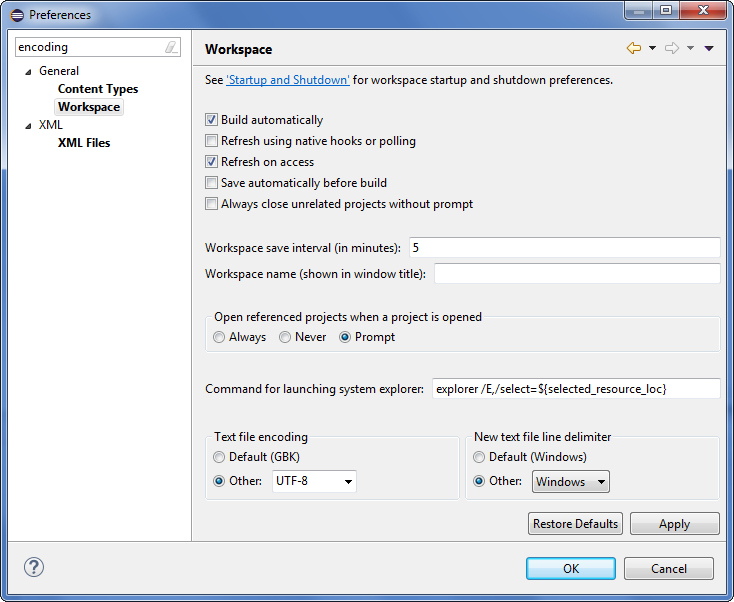
默认情况下，eclipse在代码变动被保存的时候自动重新编译工程，但是在工程代码很大或者需要网络验证XML文件合法性的时候会导致整个编译过程十分缓慢，这里建议关闭eclipse的自动编译特性

点击**Project**菜单，点击取消**Build Automatically**的选中标志



### 修改默认工作区编码

Windows操作系统默认的编码方式是GB1252，但在项目中通常会用到国际化这一特性，所以文件编码方式要修改为UTF-8，点击**Window->Preferences**，在左上角的输入框中输入**encoding**，选中**Workspace**，在右边的面板下方找到**Text file encoding**，选择**Other**，并在下拉框中选择**UTF-8**



## 下载及安装Maven

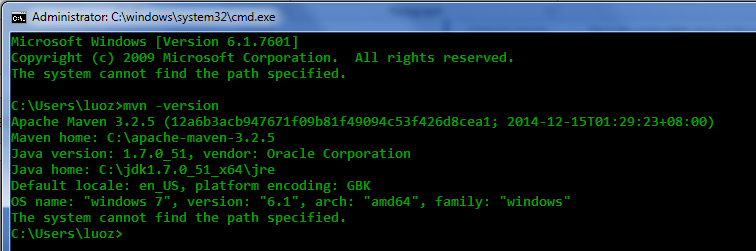
Maven是一个开源的项目构建管理工具，与耳熟能详的Ant相比，Maven提供了软件开发周期更为全面的管理。

* 从项目创建，开发测试，打包发布，覆盖软件开发整个生命周期，而Ant更多的被用作编译工具。
* Maven的另外一个特点是对Java相关项目的依赖组件Jar包的管理，通过在POM.xml文件中的依赖定义，开发团队成员能保持统一的组件库，而且无须将Jar包纳入代码的配置管理库，节约团队对Jar包的管理工作
* Maven的项目创建工件，可以新建众多类型的Java项目，例如web application，JSF application。在创建项目的同时，会为开发团队自动创建标准的目录结构以及默认的配置文件
* 围绕Maven，有众多的插件可以使用，最为常见的是Maven-compiler插件，在很多时候，Maven的默认编译参数往往不能满足项目的要求，通过Maven-compiler插件，可以定义项目需要的编译参数

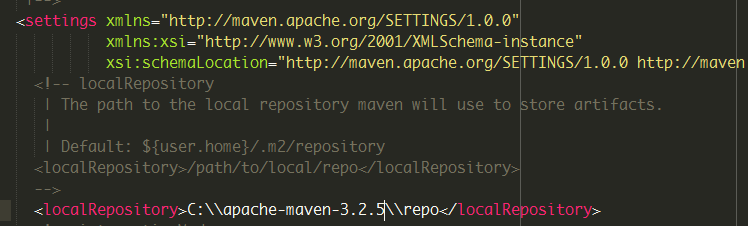
前往Maven的官方网站下载当前最稳定版本（3.2.25）<http://maven.apache.org/download.cgi> ，点击**apache-maven-3.2.5-bin.zip**进行下载，将下载后的zip文件解压缩到任一分区的任一位置，建议放入某个分区的根目录，便于接下来的配置，例如**C:\apache-maven-3.2.5**



按照[2.1](#_下载及安装JDK) 章节中设置JDK环境变量的方法，新建环境变量**M2\_HOME**，值为Maven安装的目录，例如**C:\apache-maven-3.2.5**，在系统环境变量**Path**后添加**;%M2\_HOME%\bin**, 打开命令行窗口，输入**mvn –version，**显示如下图类似文本，则Maven安装完毕



在使用Maven进行项目构建管理之前，需对Maven中的2个配置进行修改，打开**%M2\_HOME%\conf\settings.xml**，找到<localRepository>节点，添加如下图所示节点，设定Maven下载Jar的本地仓库路径



找到<proxies>节点，添加如下图所示的节点，设定Maven使用的代理服务器（如果需要使用代理服务器访问网络的情况下）

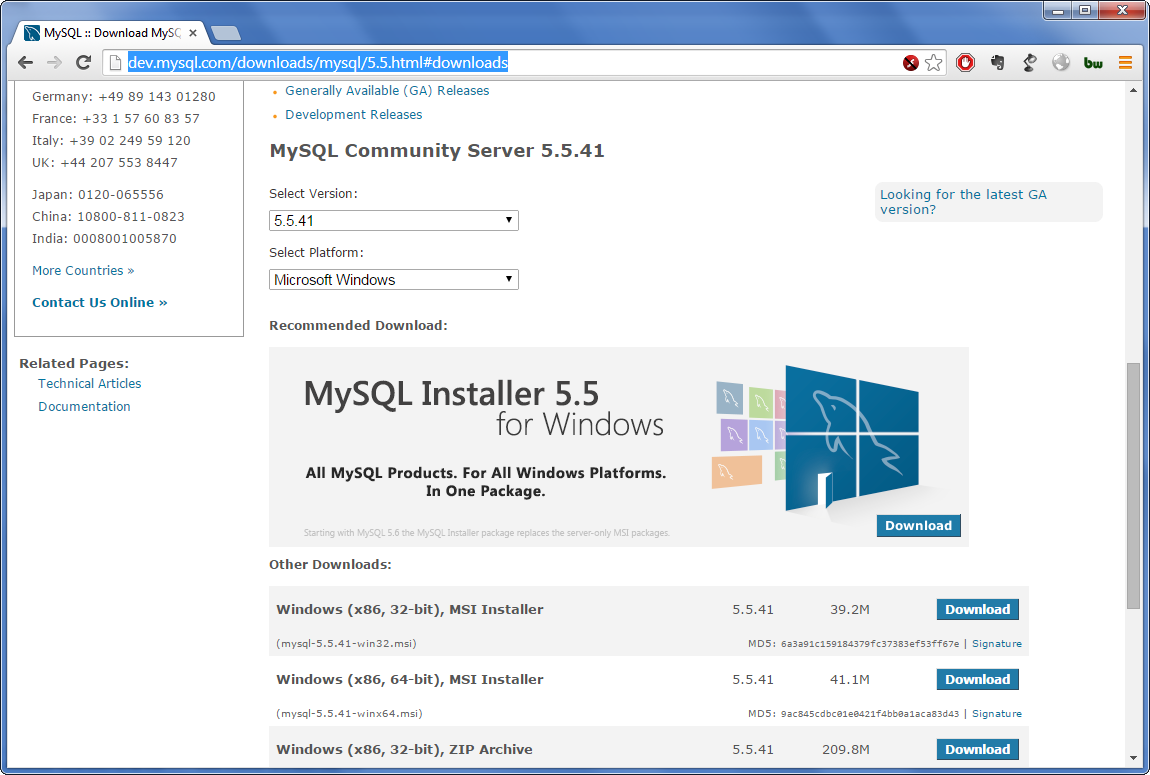


* id : 代理服务器配置标识符
* active : 是否激活此代理服务器配置
* protocol : 代理服务器的协议，例如http和https
* host : 代理服务器域名或者ip地址
* port : 代理服务器使用的端口
* username : 代理服务器使用的用户名
* password : 代理服务器使用的用户密码
* nonProxyHosts : 使用|分割不需要使用代理的网络地址或IP

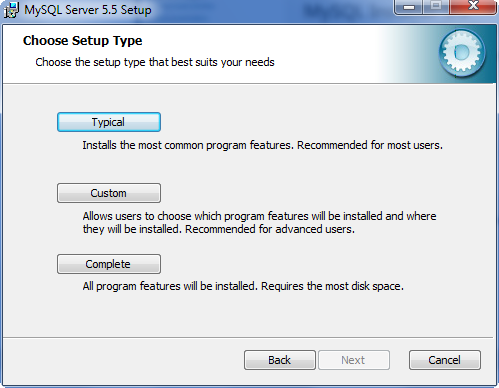
后面三个元素在上图中没有使用，如果用户的代理服务器需要用户名和密码登录，而且有些网络资源不需要代理服务器，则设定后三个元素

## 下载及安装MySQL

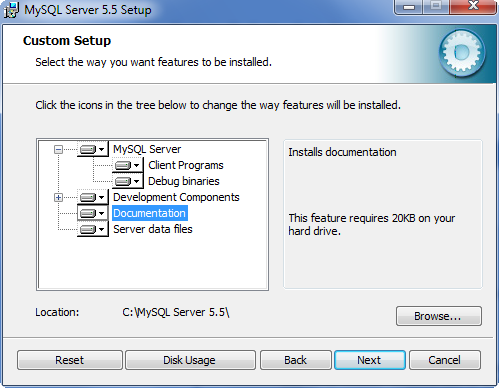
MySQL可以说是最为流行的开源关系型数据库，但目前MySQL被Oracle收购后，原来的分支独立出来成为社区版，而且与Oracle资源整合后退出的企业版在获得技术支持和高级特性的时候是需要支付费用的。项目使用社区版中比较流行的5.5作为开发数据库（<http://dev.mysql.com/downloads/mysql/5.5.html#downloads>）。



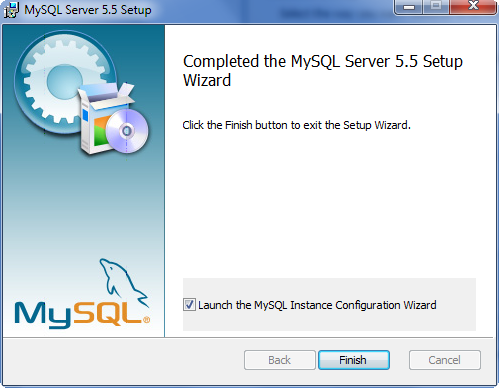
下载**Windows (x86, 64bit), MSI Installer**，下载完成后，以管理员身份运行安装，在选择安装类型的时选择**Custom…**



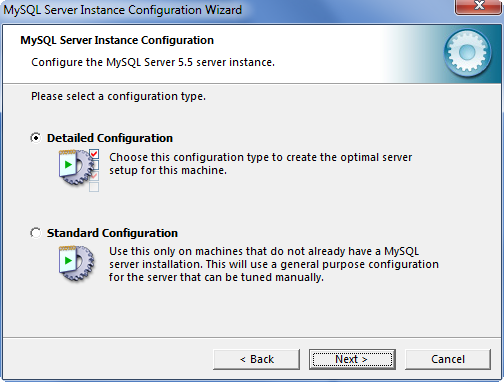
选择完全安装Documentation以便后期查看，点击**Browse…**将安装路径修改为任一分区的根目录，例如 **C:\MySQL 5.5**



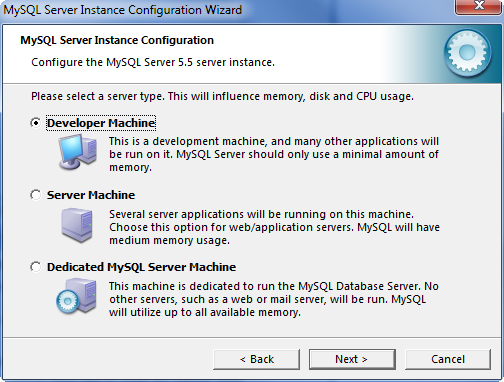
点击**Next**按钮进行安装，安装过程中有企业版的广告弹出，一路**Next**，只到安装界面如下，确保**Launch the MySQL Instance Configuration Wizard**前面的选择框是选中状态，点击**Finish**



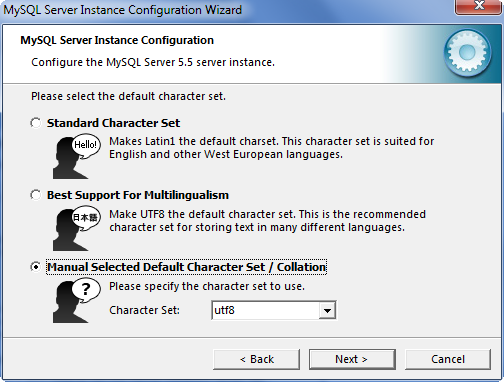
弹出的配置对话框中选择**Detailed Configuration**



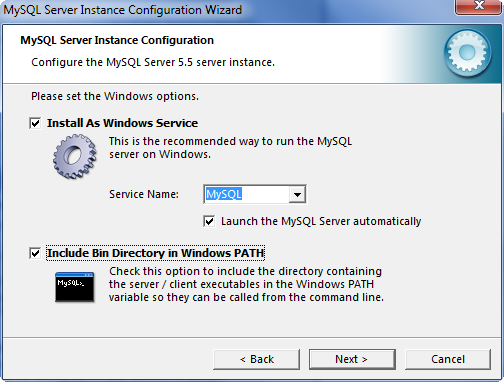
选择**Developer Machine**，以开发环境来配置MySQL



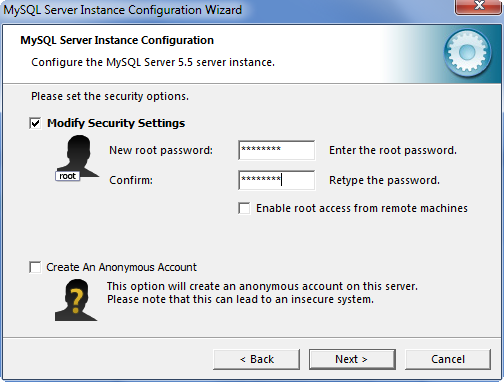
**Next**到选择字符集的界面，选择**Manual Selected Default Charset / Collation**选项，在下拉框中选择UTF-8



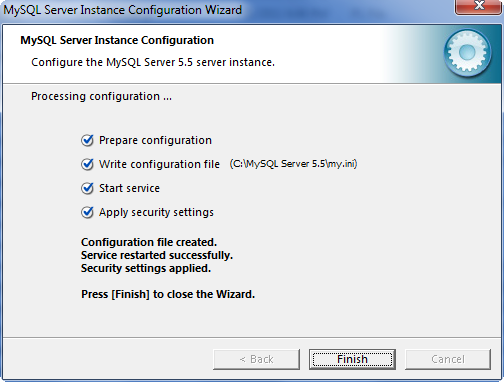
**Next**下一个界面中将**Include Bin Directory in Windows Path**选择勾选中，保证将MySQL命令行工具添加到Windows环境变量，以便在命令行窗口下直接使用



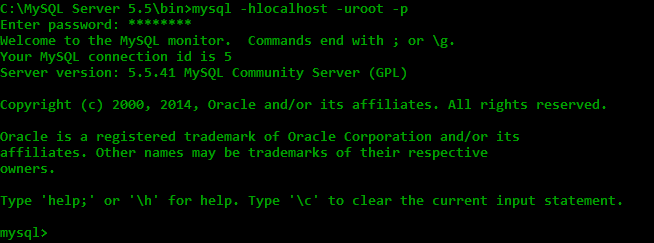
在下一个界面设定root用户密码



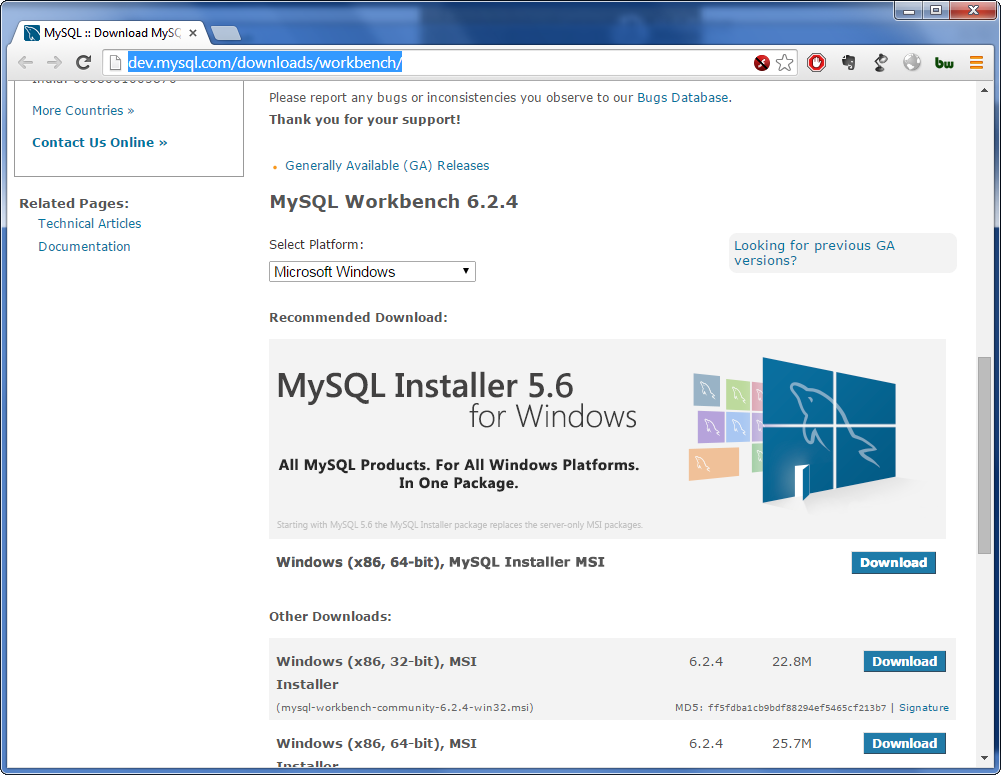
最后一步点击**Execute**按钮执行配置



打开命令行窗口，输入**mysql –hlocalhost –uroot –p，**输入设定的root用户密码，进入登录到MySQL数据库实例



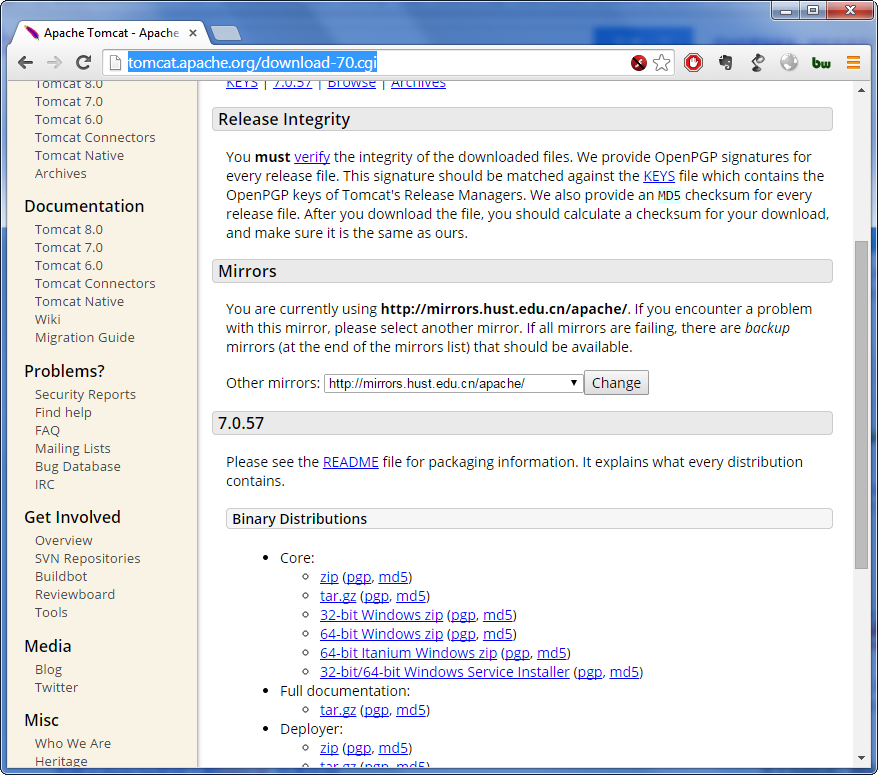
为了提高和简化开发过程，建议安装MySQL Workbench图形界面工具，下载页面为（<http://dev.mysql.com/downloads/workbench/>），选择64bit版本下载，已对应安装的MySQL 64bit 实例



注：MySQL Workbench的使用不在本文的描述范围内

## 下载及安装Tomcat

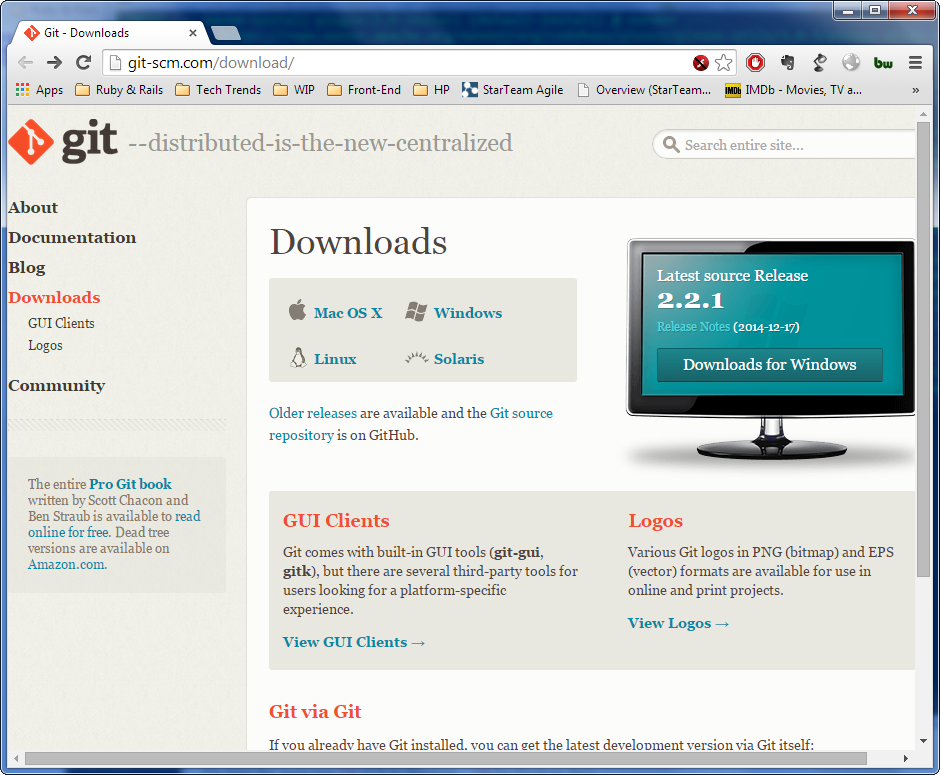
Tomcat是一个开源的WEB应用服务器，本项目采用Tomcat 7作为开发应用服务器，下载链接为<http://tomcat.apache.org/download-70.cgi>，点击Core下面的zip链接，下载zip格式的安装包



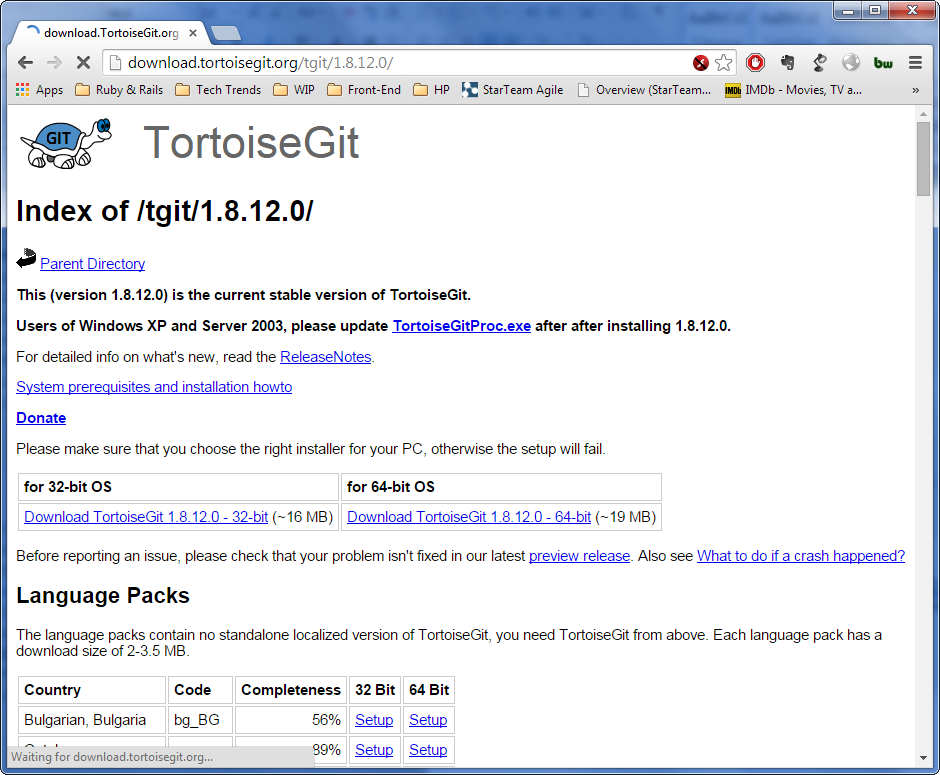
将下载的zip包解压到某一分区的根目录，例如**C:\Tomcat**，按照[2.1](#_下载及安装JDK)小节配置JDK环境变量的方法新增**CATALINA\_HOME**变量，值为**Tomcat**的安装目录，并将**;%CATALINA\_HOME%\bin**追加到**Path**变量中。

## 下载及安装Git

Git最初是Linux之父Linus为了对他的Linux内核代码做版本管理而开发的一个轻量级的版本控制工具，在开源之后迅速成为开源社区的主流版本管理工具之一。目前比较流行的Git公共仓库有GitHub，国内也有CSDN Code。Git不同于其他版本控制工具，例如CVS，SVN，之处是Git可以同时在本地和联网状态下进行代码版本管理。也就是说每个安装了Git的客户端都是一个版本仓库，无须在线仍然可以进行版本提交。Git的官方网站可以下载最新版的Git 1.8.9 （<http://git-scm.com/download/>），选择Windows版本，已管理员身份运行安装程序即可。



Git自带一个非常简陋的GUI工具，并且对中文支持不够好，这里推荐下载Tortoise Git工具，官方网站（<http://download.tortoisegit.org/tgit/1.8.12.0/>），这里对Git的使用和Git版本管理的概念不做详细描述。



## 导入项目

## 编译项目

# 项目目录结构说明

# 权限管理模块

## 用户登录

用户使用系统合法的用户名和密码登录到系统。对于不正确的用户名和密码信息进行提示。详细需求说明见UC\_005-用户登录

### 时序图



### 代码构成

实现用户登录系统的功能，需要下列文件支持，描述如下表

|  |  |  |
| --- | --- | --- |
| 文件名 | 文件路径 | 文件说明 |
| BaseVO.java | \java\com\innovaee\eorder\module \vo | 所有VO（值对象）类的父类源文件 |
| RoleLinkVo.java | \java\com\innovaee\eorder\module \vo | 页面链接值对象，用户显示用户可以看到的菜单。 |
| BaseEntity.java | \java\com\innovaee\eorder\module\entity | 所有实体类的基类 |
| User.java | \java\com\innovaee\eorder\module\entity | 用户实体类，该类与数据库的t\_user对应，通过注解进行映射。 |
| BaseService.java | \java\com\innovaee\eorder\module  \service | 所有Service类的基类 |
| AuthenticationService  .java | \java\com\innovaee\eorder\module  \service\security | 提供验证服务，验证用户输入的用户名和密码是否匹配。 |
| AuthorizationService  .java | \java\com\innovaee\eorder\module  \service\security | 提供授权服务，根据用户角色获取相关权限信息。 |
| SecurityMetadataSource Service.java | \java\com\innovaee\eorder\module  \service\security | 安全元数据源服务类，根据用户角色获取相关权限信息，生成用户权限列表。 |
| BaseAction.java | \java\com\innovaee\eorder\web\action | 所有Action类的基类 |
| LoginAction.java | \java\com\innovaee\eorder\web\action \login | 登录模块的Action，响应登录页面提交的登录请求或其他页面的退出系统请求。 |
| struts.xml | com\innovaee\eorder\web\action \login | 登录模块的Action配置文件 |
| login.jsp | eorder\src\main\webapp\pages\login | 登录页面 |

### 类说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.vo | | |
| 类名 | BaseVo | | |
| 父类 | n/a | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| n/a |  |  | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| copy | void | Object orig | 将传入对象的属性值复制到本对象 |
| equals | boolean | Object obj | 指示其他某个对象是否与此对象“相等”。 |
| hashCode | int | n/a | 返回该对象的哈希码值。 |
| toString | String | n/a | 返回该对象的字符串表示。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.vo | | |
| 类名 | RoleLinkVo | | |
| 父类 | com.innovaee.eorder.module.vo. BaseVo | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| order | String | 菜单项排序号 | |
| id | Integer | 菜单ID | |
| name | String | 菜单名 | |
| functionName | String | 功能名称 | |
| link | String | 链接地址 | |
| flag | String | 菜单等级，1为父节点，2为子节点 | |
| list | List<RoleLinkVo> | 子节点菜单列表 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| n/a |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.entity | | |
| 类名 | BaseEntity | | |
| 父类 | n/a | | |
| 实现接口 | java.io.Serializable | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| n/a |  |  | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getPK | Serializable | n/a | 返回主键的值 |
| copy | void | Object orig | 将传入对象的属性值复制到本对象 |
| equals | boolean | Object obj | 指示其他某个对象是否与此对象“相等”。 |
| hashCode | int | n/a | 返回该对象的哈希码值。 |
| toString | String | n/a | 返回该对象的字符串表示。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.entity | | |
| 类名 | UserEntity | | |
| 父类 | com.innovaee.eorder.module.entity.BaseEntity | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userId | Integer | 用户ID | |
| username | String | 用户名称 | |
| password | String | 密码 | |
| cellphone | String | 手机号码 | |
| levelId | Integer | 用户等级ID | |
| userStatus | Boolean | 用户状态 | |
| createAt | Timestamp | 创建时间 | |
| updateAt | Timestamp | 更新时间 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| n/a |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service | | |
| 类名 | BaseService | | |
| 父类 | n/a | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| n/a |  |  |  |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| n/a |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | AuthenticationService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.core.userdetails.UserDetailsService | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getUserInfo(String) | User | String username | 通过用户名查找用户对象。 |
| checkUserPassword(String, String) | boolean | String username, String password | 校验用户名密码是否匹配，如果匹配成功，返回true，否则返回false。 |
| loadUserByUsername(String) | UserDetails | String username | 通过用户名查找用户详细信息对象。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | AuthorizationService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.access.AccessDecisionManager | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| decide | void | Authentication authentication, Object object, Collection<ConfigAttribute> configAttributes | AccessDecisionManager 使用方法参数传递所有信息，在真实的安全方法期望调用的时候，传递安全Object启用那些参数。如果访问被拒绝，实现将抛  出一个AccessDeniedException 异常。 |
| supports | boolean | ConfigAttribute attribute | 在启动的时候被AbstractSecurityInterceptor调用， 来决定AccessDecisionManager 是否可以执行传递ConfigAttribute 。 |
| supports | boolean | Class<?> clazz | 此方法被安全拦截器实现调用， 包含安全拦截器将显示的AccessDecisionManager 支持安全对象的类型。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | SecurityMetadataSourceService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.web.access.intercept.FilterInvocationSecurityMetadataSource | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| functionDao | FunctionDao | 功能数据访问对象 | |
| roleDao | RoleDao | 角色数据访问对象 | |
| userRoleDao | UserRoleDao | 用户角色数据访问对象 | |
| roleFunctionDao | RoleFunctionDao | 角色功能数据访问对象 | |
| allConfigAttributes | Collection<ConfigAttribute> | 配置属性集合 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getAllFunctions | List<Function> | n/a | 返回所有功能列表 |
| getUserRoles | List<UserRole> | String username | 根据用户名称返回用户角色列表 |
| findRoleFunctionsByRoleId | List<RoleFunction> | Integer roleId | 根据用户角色ID查找角色功能列表 |
| getUserFunctions | List<UserFunctionVo> | String username | 根据用户名称查找用户功能值对象列表 |

### 配置文件说明



#### struts.xml

下面配置文件展现了如何将用户登录Action与URI路径以及对应的Java类配置起来

<struts>

<package name="com.innovaee.eorder.web.action.login" namespace="/login"

extends="struts-base">

<action name="login" class="com.innovaee.eorder.web.action.login.LoginAction"

method="login">

<result name="success">/pages/login/login.jsp</result>

</action>

<action name="doLogin" class="com.innovaee.eorder.web.action.login.LoginAction"

method="doLogin">

<result type="redirect">../user/doUser.action</result>

</action>

</package>

</struts>

#### applicationContext-total.xml

在applicationContext-total.xml 中添加如下配置，使得用户登录相关Java对象成为Spring容器托管对象

<bean id="baseDAO" class="com.innovaee.eorder.module.dao.BaseDao"

abstract="true">

<property name="sessionFactory" ref="sessionFactory" />

</bean>

<bean id="roleFunctionDao" class="com.innovaee.eorder.module.dao.RoleFunctionDao"

parent="baseDAO" />

<bean id="functionDao" class="com.innovaee.eorder.module.dao.FunctionDao"

parent="baseDAO" />

<bean id="roleDao" class="com.innovaee.eorder.module.dao.RoleDao" parent="baseDAO" />

<bean id="userDao" class="com.innovaee.eorder.module.dao.UserDao" parent="baseDAO" />

<bean id="userRoleDao" class="com.innovaee.eorder.module.dao.UserRoleDao"

parent="baseDAO" />

<bean id="userLevelDao" class="com.innovaee.eorder.module.dao.UserLevelDao"

parent="baseDAO" />

<bean id="baseService" class="com.innovaee.eorder.module.service.BaseService"

abstract="true" />

<bean id="authorizationService"

class="com.innovaee.eorder.module.service.security.AuthorizationService"

parent="baseService" />

<bean id="authenticationService"

class="com.innovaee.eorder.module.service.security.AuthenticationService"

parent="baseService" />

<bean id="securityMetadataSourceService"

class="com.innovaee.eorder.module.service.security.SecurityMetadataSourceService"

parent="baseService" />

<bean id="userService" class="com.innovaee.eorder.module.service.UserService"

parent="baseService" />

#### Hibernate 配置

项目使用Java 5之后的注解特性来进行Hibernate ORM配置，比之前XML文件配置更为简洁，用户实体与数据库表映射配置如下

@Entity

@Table(name = "t\_user")

public class User extends BaseEntity {

@Id

@Column(name = "USER\_ID", unique = true, nullable = false)

@GeneratedValue(strategy = GenerationType.AUTO)

private Integer userId;

@Column(name = "USERNAME")

private String username;

@Column(name = "PASSWORD")

private String password;

@Column(name = "CELLPHONE")

private String cellphone;

@Column(name = "LEVEL\_ID")

private Integer levelId;

@Column(name = "USER\_STATUS")

private Boolean userStatus;

### 数据库脚本说明

通过命令行或者GUI工具执行下面的SQL语句到目标数据库生成用户实体表

-- ----------------------------

-- Table structure for `t\_user`

-- ----------------------------

DROP TABLE IF EXISTS `t\_user`;

CREATE TABLE `t\_user` (

`user\_id` int(32) NOT NULL AUTO\_INCREMENT,

`username` varchar(20) COLLATE utf8\_bin DEFAULT NULL,

`password` varchar(256) COLLATE utf8\_bin DEFAULT NULL,

`cellphone` varchar(16) COLLATE utf8\_bin DEFAULT NULL,

`user\_score` int(32) DEFAULT NULL,

`level\_id` int(32) DEFAULT NULL,

`user\_status` tinyint(1) DEFAULT '1',

`create\_at` datetime DEFAULT NULL,

`update\_at` datetime DEFAULT NULL,

PRIMARY KEY (`user\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8 COLLATE=utf8\_bin COMMENT='user''s basic information';

# 模块设计



## 功能模板设计说明

## 用户管理

### 目标

管理员通过用户管理模块，对用户进行增删改查操作。

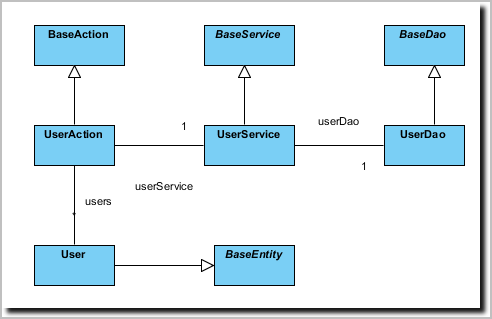
### 用户需求

|  |  |
| --- | --- |
| 用例编号 | UC\_006 |
| 用例描述 | 用户管理 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 |  |
| 用例图 |  |
| 触发条件 | 管理员得到用户信息更新需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | n/a |
| 备选流程 | n/a |
| 例外 | n/a |
| 包含用例 | UC\_007,UC\_008 |
| 非功能性需求 | n/a |
| 其他说明 | n/a |

|  |  |
| --- | --- |
| 用例编号 | UC\_007 |
| 用例描述 | 用户管理-创建用户 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 | n/a |
| 用例图 |  |
| 触发条件 | 管理员得到增加用户信息需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | 管理员点击eOrder订餐管理系统后台页面左侧用户管理菜单，进入用户管理页面  页面显示系统所有的用户信息列表  在增加新用户的页面内，输入用户信息  管理员点击增加创建新用户按钮提交新用户信息，系统显示Msg701  系统显示用户加入了用户列表 |
| 备选流程 | 管理员没输入所有的用户信息，系统提示Msg702  回到主流程4 |
| 例外 | 员工号重复，提示Msg703  手机号重复，提示Msg704 |
| 包含用例 | n/a |
| 非功能性需求 | 用户列表按照用户拼音字母a-z显示 |
| 其他说明 | 用户信息包括：用户名（用户名为姓名的拼音，忽略大小写，相同用户的加数字区分），员工号（为5位数字），手机号，密码（默认密码为手机号后6位），确认密码，用户状态（在职，离职） |

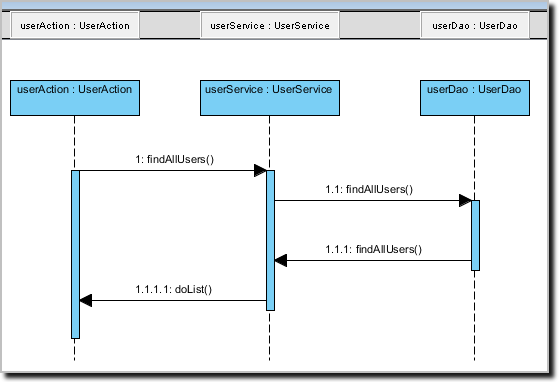
|  |  |
| --- | --- |
| 用例编号 | UC\_008 |
| 用例描述 | 用户管理-更新用户 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 | n/a |
| 用例图 |  |
| 触发条件 | 管理员得到用户信息更新需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | 管理员进入用户信息列表  管理员点击需要更新的用户名对应的编辑按钮  在用户信息的页面内，输入更新需求的用户信息  管理员点击更新用户信息页面的完成按钮提交用户更新信息，系统显示Msg801  系统显示更新的用户信息 |
| 备选流程 | 用户点击删除按钮，系统不显示该用户信息  回到主流程1  4.1管理员删除了用户的信息某一个或者多个字段的值，系统提示Msg802  回到主流程4  管理员提交更新不成功，系统提示Msg803  回到主流程4 |
| 例外 | n/a |
| 包含用例 | n/a |
| 非功能性需求 | 用户列表按照拼音显示 |
| 其他说明 | 用户信息包括：用户名（用户名为姓名的拼音），手机号，密码（默认密码为手机号后6位），用户角色 |

### 类图

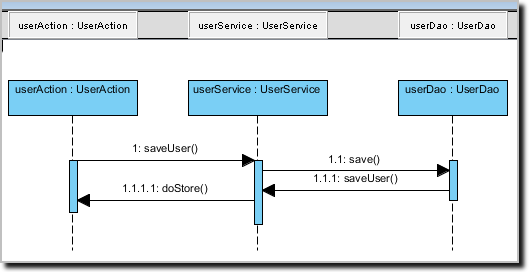


### 时序图

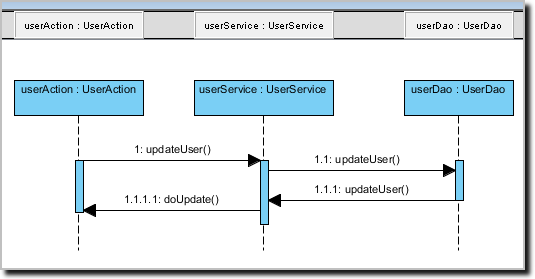
1、查找用户



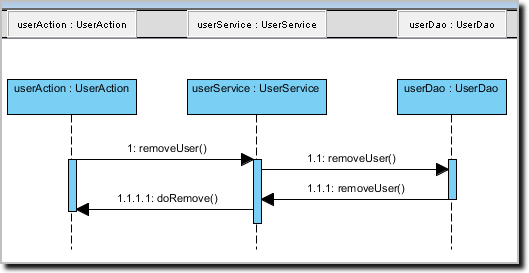
2、新增用户



3、修改用户



4、删除用户



### 组件

* Entity层实现对数据表及关联关系进行封装 ,命名就是数据库表的名字，遵循Java的编码规范；
* Dao接口层实现对数据访问层封装；
* Sevice接口层实现对业务逻辑封装；
* Action层实现对业务事件的封装。

### 类说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.entity | | |
| 类名 | User | | |
| 父类 | com.innovaee.eorder.module.entity.BaseEntity | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userId | Integer | 用户ID | |
| username | String | 用户名 | |
| password | String | 密码 | |
| cellphone | String | 手机号码 | |
| levelId | Integer | 用户等级 | |
| userStatus | Boolean | 用户状态 | |
| createAt | Timestamp | 创建时间 | |
| updateAt | Timestamp | 更新时间 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getPK | Serializable | n/a | 用户ID |
| User | n/a | n/a | 默认构造函数 |
| User | n/a | Integer userId | 构造一个新的 User，它具有指定的userId。 |
| User | n/a | String username | 构造一个新的 User，它具有指定的username。 |
| User | n/a | String username, String password, String cellphone, Integer levelId, Boolean userStatus, Timestamp createAt | 构造一个新的 User，它具有指定的username，password，cellphone，levelId，userStatus，createAt。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.dao | | |
| 类名 | UserDao | | |
| 父类 | com.innovaee.eorder.module.dao.BaseDao | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getEntityClass() | Class | n/a |  |
| findUsersByUserName() | User | String username | 通过用户名查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findAllUsers() | List<User> | n/a | 返回所有用户实体对象列表。 |
| getUserByPassword(String, String) |  | String username,  String password | 通过用户名和密码查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| saveUser(User) | User | User | 将传入的User对象写入数据库 |
| updateUser(User) | void | User | 更新数据库中传入的User对象对应的记录 |
| removeUser(User) | void | User | 删除数据库中传入的User对象对应的记录 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service | | |
| 类名 | UserService | | |
| 父类 | com.innovaee.eorder.module.service.BaseService | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象类，用于对用户数据操作 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| loadUser | User | Integer userId | 通过用户ID查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findAllUsers() | List<User> | n/a | 返回所有用户实体对象列表。 |
| getUserByPassword(String, String) | User | String username,  String password | 通过用户名和密码查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findUsersByUserName(String) | User | String username | 通过用户名查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| saveUser(User) | User | User | 将传入的User对象新增到数据库中 |
| updateUser(User) | void | User | 更新数据库中传入的User对象对应的记录 |
| removeUser(User) | void | User | 删除数据库中传入的User对象对应的记录 |
| removeUser(Integer) | void | Integer userId | 根据userId删除数据库中的对应的User记录 |
| removeUsers(String[]) | void | Integer[] userId | 根据userId数组，批量删除数据库中的对应的User记录 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.web.action.admin.user | | |
| 类名 | UserAction | | |
| 父类 | com.innovaee.eorder.web.action.BaseAction | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| list | List<RoleLinkVo> | 用户角色链接列表，用于显示用户可以看到的菜单 | |
| userId | String | 用户ID | |
| userIds | String[] | 用户ID数组 | |
| user | User | User对象，用户前台页面和后台数据交互 | |
| users | List<User> | User对象列表，用户前台页面和后台数据交互 | |
| userService | UserService | 用户服务类，用于对User对象的操作 | |
| userRoleService | UserRoleService | 用户角色服务类，用于对UserRole对象的操作 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| login() | String | n/a | 进入页面执行的方法。 |
| doLoad() | String | n/a | 根据前台传来的用户ID查到User对象，并保存到user对象中。 |
| doList() | List<User> | n/a | 返回所有用户实体对象列表。 |
| doStore() | User | User | 将传入的User对象新增到数据库中 |
| doUpdate() | void | User | 更新数据库中传入的User对象对应的记录 |
| doRemove() | void | User | 删除数据库中传入的User对象对应的记录 |
| doLeft() | String | n/a | 根据用户信息，生成对应的菜单 |

### 其他说明

# 数据库设计

## 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 用户表 | | t\_user | |
| 数据项描述 | 数据项名 | 类型(长度) | 默认值 | 限制 |
| 用户ID | User\_id | Int(32) | n/a | 主键，自动增加 |
| 用户名 | username | Varchar(128)r | n/a | 非空，英文字母和数字组成不少于6位 |
| 密码 | password | Varchar(128) | n/a | 非空，英文字母和数字组成不少于6位 |
| 手机号 | cellphone | Varchar(16) | n/a |  |
| 会员等级ID | level\_id | Int(32) | n/a |  |
| 用户状态 | User\_status | tinyint | 1 |  |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 角色表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 角色表 | | t\_role | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 角色ID | role\_id | Int(32) | n/a | 主键 |
| 角色名称 | role\_name | Varchar(128) | n/a |  |
| 角色描述 | role\_desc | Varchar(128) | n/a |  |
| 角色状态 | role\_status | tinyint | 1 |  |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 功能表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 功能表 | | t\_function | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 功能ID | function\_id | Int(32) | n/a | 主键 |
| 功能名称 | function\_name | Varchar(128) | n/a | 非空 |
| 功能描述 | function\_desc | Varchar(128) | n/a | 非空 |
| 功能路径 | function\_path | Varchar(128) | n/a | 非空 |
| 父功能ID | function\_parent | Int(32) | n/a | 可为空 |
| 功能排序 | function\_order | Varchar(128) | n/a | 非空 |
| 功能状态 | function\_status | tinyint | 1 | 非空 |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 用户角色关联表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 用户角色关联表 | | t\_user\_role | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 用户角色ID | User\_role\_id | int(32) | n/a | 主键 |
| 用户ID | user\_id | Int(32 | n/a | 外键 |
| 角色ID | role\_id | Int(32 | n/a | 外键 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 角色功能关联表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 角色功能关联表 | | t\_role\_function | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 角色功能ID | Role\_function\_id | int(32) | n/a | 主键 |
| 功能ID | function\_id | Int(32 | n/a | 外键 |
| 角色ID | role\_id | Int(32 | n/a | 外键 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 菜品分类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 菜品分类表 | | t\_dish\_category | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 菜品类别ID | category\_id | Int(32) | n/a | 主键，自动增加 |
| 类别名称 | category\_name | Varchar(r128) | n/a | 非空 |
| 类别图片 | category\_picture | Varchar(r128) | n/a | 非空 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 菜品表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 菜品表 | | t\_dish | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 菜品ID | dish\_id | Int(32) | n/a | 主键，自动增加 |
| 菜品分类ID | category\_id | Int(32) | 0 | 外键 |
| 菜品名称 | dish\_name | Varchar(128) | n/a | 非空 |
| 菜品图片 | dish\_picture | Varchar(128) | n/a | 非空 |
| 价格 | price | double(64) | 0.0 | 精确到小数点后一位 |
| 是否在售 | on\_sell | tinyint | True | 非空 |
| 其他信息 | misc | Varchar(128) | n/a | n/a |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 订单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 订单表 | | t\_order | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 订单ID | order\_id | Int(32) | n/a | 主键，自动增加 |
| 订单流水号 | order\_seq | Varchar(128) | n/a | 格式为<yyyymmddxxxxx>  yyyy四位数年份  mm两位数月份  dd两位日期  xxxxx当天订单序号 |
| 订单状态 | order\_status | Varchar(128) | ‘new’ | 非空 |
| 台号 | table\_number | Int(32) | n/a |  |
| 人数 | attendee\_number | Int(32) | 1 |  |
| 实收金额 | total\_price | double | 0.0 | 精确小数点后一位 |
| 服务员ID | servent\_id | Int(32) | n/a | 非空 |
| 会员ID | member\_id | Int(32) | n/a | n/a |
| 收银员ID | casher\_id | Int(32) | n/a | n/a |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 订单条目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 订单条目表 | | t\_order\_item | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 订单条目ID | item\_id | Int(32) | n/a | 主键，自动增加 |
| 订单ID | order\_id | Int(32) | 0 |  |
| 菜品ID | dish\_id | Int(32) | 0 | 非空 |
| 份数 | dish\_account | Int(32) | 1 | 最大99 |
| 创建时间 | create\_at | datetime | 当前日期 | 当前日期 |
| 修改时间 | update\_at | datetime | 当前日期 | 当前日期 |

## 会员等级表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 会员等级表 | | t\_user\_level | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 会员等级ID | level\_id | Int(32) | n/a | 主键，自动增加 |
| 等级名称 | level\_name | Int(32) | n/a |  |
| 等级分数 | level\_score | Int(32) | n/a |  |
| 等级状态 | level\_status | tinyint | n/a | 非空 |
| 创建时间 | create\_at | datetime | 当前日期 | 非空 |
| 修改时间 | update\_at | datetime | 当前日期 | 非空 |

# 接口设计



## 内部接口

## 外部接口

### 接口说明

### 调用方式

# 系统认证安全设计

## 访问控制设计

## 访问控制流程图

## 访问控制流程说明

## 系统性能设计

## 概念

## EnCache配置

# 系统出错处理

## 出错信息输出

## 出错处理对策

# 附录