

学习目的与要求

本章首先简要介绍关系数据库,描述 MySQL 的具体使用,重点介绍 JDBC 的基本原理以及应用编程接口。通过本章的学习,熟悉 MySQL 数据库系统的使用,掌握 JDBC 的基本原理以及编程接口的使用,能够开发数据库应用程序。

本章主要内容

- MySQL 数据库的使用。
- JDBC 概述及基本原理。
- JDBC 高级操作。

5.1 MySQL 数据库的使用

1. 连接 MySQL

连接 MySQL 的命令格式为:

```
mysql -h 主机地址 -u 用户名 -p 用户密码
```

例 5-1 连接到本机上的 MySQL。

首先打开 DOS 窗口,然后进入目录 mysql\bin,再输入命令 mysql -uroot -p,回车后提示输入密码,如果刚安装好 MySQL,超级用户 root 是没有密码的,故直接回车即可进入 MySQL。MySQL 的提示符是: mysql>。

例 5-2 连接到远程主机上的 MySQL。假设远程主机的 IP 地址为 110.110.110.110,用户名为 root,密码为 abc123,则输入以下命令:

```
mysql -h110.110.110.110 -uroot -pabc123
```

说明：u 与 root 可以不用加空格，其他也一样。

退出 MySQL 命令是 exit。

2. 修改密码

修改密码的命令格式为：

```
mysqladmin -u 用户名 -p 旧密码 password 新密码
```

例 5-3 给 root 加个密码 ab12。首先在 DOS 下进入目录 mysqlbin，然后输入以下命令(password 中不要加命令符)：

```
mysqladmin -uroot password ab12
```

说明：因为开始时 root 没有密码，所以-p 旧密码一项就可以省略了。

例 5-4 再将 root 的密码改为 djg345，输入以下命令：

```
mysqladmin -uroot -pab12 password djg345
```

3. 增加新用户

注意：和上面不同，下面的命令因为是 MySQL 环境中的命令，所以后面都带一个分号作为命令结束符。

增加新用户的命令格式为：

```
grant select on 数据库.* to 用户名@登录主机 identified by "密码"
```

例 5-5 增加一个用户 test1，密码为 abc，让他可以在任何主机上登录，并对所有数据库有查询、插入、修改、删除的权限。

首先以 root 用户连入 MySQL，然后输入以下命令：

```
grant select,insert,update,delete on *.* to test1@"%" identified by "abc";
```

但增加的用户是十分危险的，如果某个人知道 test1 的密码，那么就可以在互联网上的任何一台计算机上登录你的 MySQL 数据库并对你的数据为所欲为了，解决办法见例 5-6。

例 5-6 增加一个用户 test2，密码为 abc，让其只可以在 localhost 上登录，并可以对数据库 mydb 进行查询、插入、修改、删除的操作。

例 5-6 中的 localhost 指本地主机，即 MySQL 数据库所在的那台主机，这样用户即使知道 test2 的密码，也无法从 internet 上直接访问数据库，只能通过 MySQL 主机上的 Web 页来访问了。

```
grant select,insert,update,delete on mydb.* to test2@localhost identified by "abc";
```

如果不想让 test2 有密码，可以再输入一个命令将密码删掉：

```
grant select, insert, update, delete on mydb.* to test2@localhost identified by "";
```

上面介绍了登录、增加用户、密码更改等问题。下面来看看 MySQL 中有关数据库方面的操作。

注意：你必须首先登录到 MySQL 中，以下操作都是在 MySQL 的提示符下进行的，而且每个命令以分号结束。

4. 常用命令介绍

(1) 显示数据库列表：

```
show databases;
```

刚开始时只有两个数据库 mysql 和 test。数据库 mysql 很重要，它里面有 MySQL 的系统信息。修改密码和新增用户，实际上就是对这个库进行操作。

(2) 显示库中的数据表：

```
use mysql;
show tables;
```

(3) 显示数据表的结构：

```
describe 表名;
```

(4) 建库：

```
create database 库名;
```

(5) 建表：

```
use 库名;
create table 表名 (字段设定列表);
```

(6) 删库和删表：

```
drop database 库名;
drop table 表名;
```

(7) 将表中记录清空：

```
delete from 表名;
```

(8) 显示表中的记录：

```
select * from 表名;
```

如果输入命令时，回车后发现忘记加分号，无须重新输入一遍命令，只要输入一个分号回车即可。也就是说，可以把一个完整的命令分成几行来输入，完后用分号作结束标志。可以使用上、下键调出以前的命令。

5. 实例

下面是建库和建表以及插入数据的实例。

```
drop database if exists school;    //如果存在 school,则删除
create database school;           //建立库 school
use school;                       //打开库 school
create table teacher              //建立表 teacher
```

```
(
id int(3) auto_increment not null primary key,
name char(10) not null,
address varchar(50) default '呼和浩特',
year date
);                                     //建表结束
//以下为插入字段
insert into teacher values('','glchengang','内蒙古大学','1976-10-10');
insert into teacher values('','jack','内蒙古师范大学','1975-12-23');
```

说明：

(1) 在建表中,将 ID 设为长度为 3 的数字字段: int(3);让它的每个记录自动加一: auto_increment;并不能为空: not null;而且让它成为主字段: primary key。

(2) 将 name 设为长度为 10 的字符字段。

(3) 将 address 设为长度 50 的字符字段,而且默认值为呼和浩特。

(4) 将 year 设为日期字段。

如果在 mysql 提示符输入上面的命令也可以,但不方便调试,可以将以上命令原样写入一个文本文件中且假设为 school.sql,然后复制到 C:\下,并在 DOS 状态进入目录 \mysql\bin,然后输入以下命令:

```
mysql -uroot -p 密码<c:\school.sql
```

如果成功,空出一行无任何显示;如有错误,会有提示。

6. 备份数据库

备份数据库命令在 DOS 的 \mysql\bin 目录下执行。

```
mysqldump --opt school>school.txt
```

说明：将数据库 school 备份到 school.txt 文件,school.txt 是一个文本文件。

在熟悉这些命令之后,可以使用 SQLYog、MySQLAdmin 等带有图形用户界面的客户端工具,从而让我们的操作更加简单方便。

5.2 JDBC 概述及基本原理

在开始介绍 JDBC 之前,先来澄清一个问题: JDBC 是不是过时了? 例如,现在 Java 开发中大量采用对象/关系映射(O/R Mapping)方案,典型的有 Hibernate、MyBatis、JDO 等技术,它们是不是完全取代了 JDBC 呢? 我们认为不能这样理解。首先这些技术的基础恰恰正是 JDBC。它们是封装 JDBC 后的数据持久化层。如果不了解 JDBC,就无法真正掌握 O/R Mapping。其次,JDBC 依然有其独特的适用范围,在要求速度和性能的情况下,比如海量数据处理等,JDBC 依然是不可取代的。

1. JDBC 是什么

在 Java Web 应用开发中,数据库管理系统(RDBMS)的使用是不可缺少的。JDBC

(Java Database Connectivity) 是一种用于执行 SQL 语句的 Java API。它由一组用 Java 编程语言编写的类和接口组成。JDBC 为工具/数据库开发人员提供了一个标准的 API, 使他们能够用纯 Java API 来编写数据库应用程序。

2. JDBC 如何使用

简单地讲,JDBC 的使用包括:

- (1) 与数据库建立连接。
- (2) 发送 SQL 语句。
- (3) 返回处理结果。

5.2.1 JDBC 驱动

首先介绍 JDBC 驱动(Driver), 因为它是 JDBC 开发的基础。有了驱动程序, 才能建立与数据库的连接, 然后传递并执行 SQL 语句。

目前使用的 JDBC 驱动程序可分为以下 4 种。

(1) JDBC-ODBC 桥加 ODBC 驱动程序(JDBC-ODBC bridge plus ODBC driver): 它利用 ODBC 驱动程序提供 JDBC 访问。注意, 必须将 ODBC 二进制代码(许多情况下还包括数据库客户机代码)加载到使用该驱动程序的每个客户机上。因此, 这种类型的驱动程序或者是用 Java 编写的 3 层结构的应用程序服务器代码最适用于企业网(这种网络上客户机的安装不是主要问题)。

(2) 本地 API 部分用 Java 来编写的驱动程序(Native-API partly-Java driver): 这种类型的驱动程序把客户机 API 上的 JDBC 调用转换为 Oracle、Sybase、Informix、DB2 或其他 DBMS 的调用。注意, 像桥驱动程序一样, 这种类型的驱动程序要求将某些二进制代码加载到每台客户机。

(3) JDBC 网络纯 Java 驱动程序(JDBC-net pure Java driver): 这种驱动程序将 JDBC 转换为与 DBMS 无关的网络协议, 之后这种协议又被某个服务器转换为一种 DBMS 协议。这种网络服务器中间件能够将它的纯 Java 客户机连接到多种不同的数据库上。所用的具体协议取决于提供者。通常, 这是最为灵活的 JDBC 驱动程序。所有这种解决方案的提供者都有可能提供适合于 Intranet 使用的产品。为了使这些产品也支持 Internet 访问, 它们必须处理 Web 所提出的安全性、通过防火墙的访问等方面的额外要求。

(4) 本地协议纯 Java 驱动程序(Native protocol pure Java driver): 这种类型的驱动程序将 JDBC 调用直接转换为 DBMS 所使用的网络协议。这将允许从客户机机器上直接调用 DBMS 服务器, 是 Intranet 访问的一个很实用的解决方法。

5.2.2 JDBC 开发应用编程接口介绍

1. 主要对象和接口

和程序员密切相关的是如何使用 JDBC 提供的 API, 它包括以下两部分。

- (1) java.sql: Primary features of JDBC in Java 2 platform, standard edition(J2SE)。
- (2) javax.sql: Extended functionality in Java 2 platform, enterprise edition(J2EE)。

JDBC 中的主要对象和接口, 如图 5-1 所示, 说明如下。

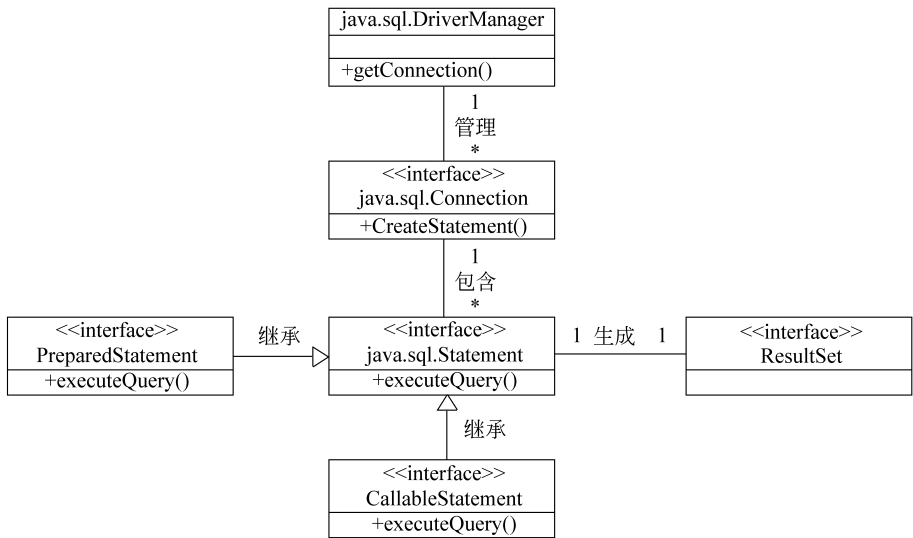


图 5-1 主要对象和接口

- (1) DriverManager：驱动管理类。
- (2) Connection：数据库连接对象封装接口。
- (3) Statement：陈述语句。
- (4) PreparedStatement：预处理语句。
- (5) CallableStatement：存储过程调用接口。
- (6) ResultSet：结果集接口。

2. 编写 JDBC 应用程序的基本流程

所有的 JDBC 应用程序都具有下面的基本流程。

- (1) 注册 JDBC 驱动程序。
- (2) 建立到数据库的连接。
- (3)通过 Statement 创建 SQL 语句。
- (4) 执行 SQL 语句。
- (5) 处理得到的结果。
- (6) 从数据库断开连接。

图 5-2 为编写 JDBC 程序的一般过程。

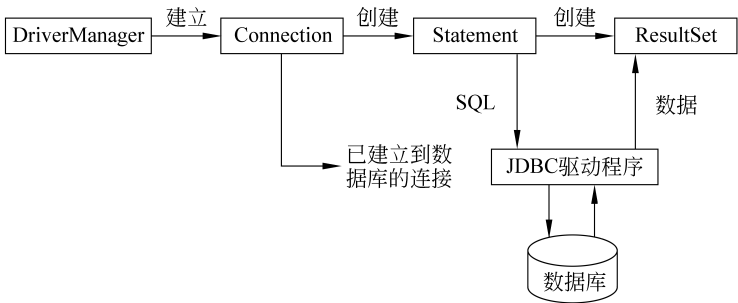


图 5-2 编写 JDBC 应用程序的一般过程

下面举一个开发 JDBC 的例子。

(1) 在 MySQL 的 test 数据库下建表,脚本如下。

```
USE 'test';

/* Table structure for table 'llxtable' */

DROP TABLE IF EXISTS 'llxtable';

CREATE TABLE 'llxtable' (
  'name' varchar(255) NOT NULL,
  'gender' varchar(255) DEFAULT NULL,
  'age' int DEFAULT NULL,
  PRIMARY KEY ('name')
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

(2) 在 MyEclipse 中选择 New→Java Project,建立一个 Java 项目,如图 5-3 所示。

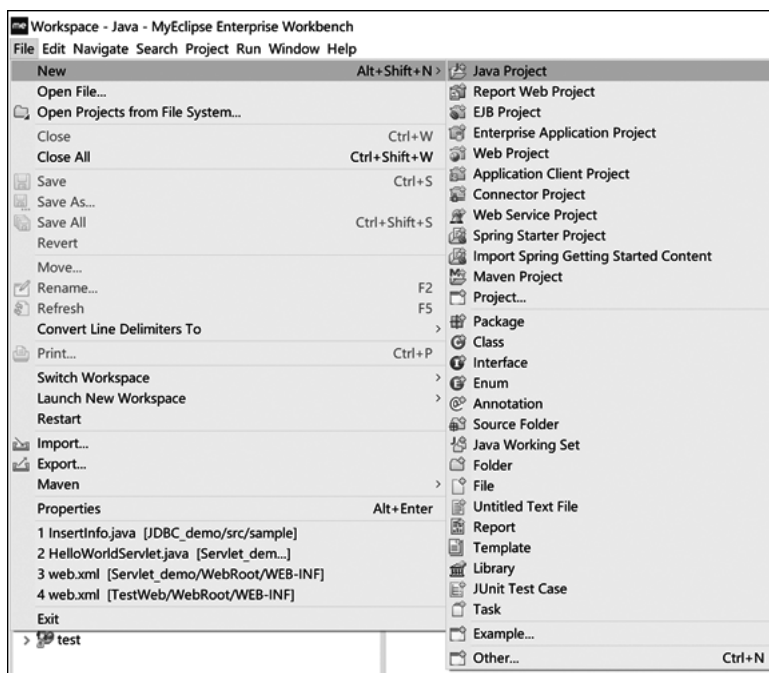


图 5-3 新建项目

(3) 命名 Project name 为 JDBC_demo,如图 5-4 所示。

(4) 单击 Next 按钮,选中 Libraries,单击 Add External JARs 选项,如图 5-5 所示。

(5) 浏览文件目录,找到 MySQL Connector Jar 包,这里使用的是 mysql-connector-java-5.1.46,如图 5-6 所示。

注意: 因为这个实例介绍的是 JDBC 基本特性,所以选择 5.1.46 版本的库就好,如果要使用 JDBC 高级特性,请相应更新到更高版本的 Jar 包。

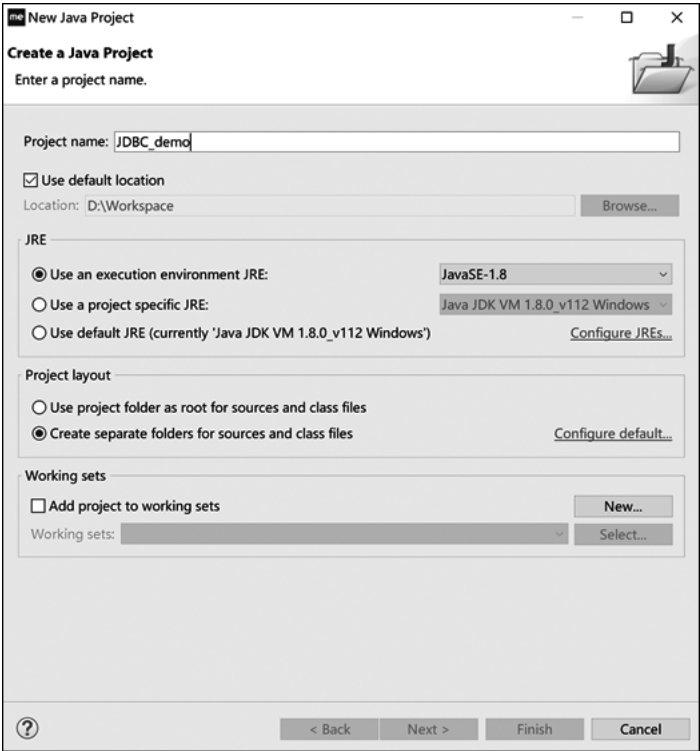


图 5-4 命名项目

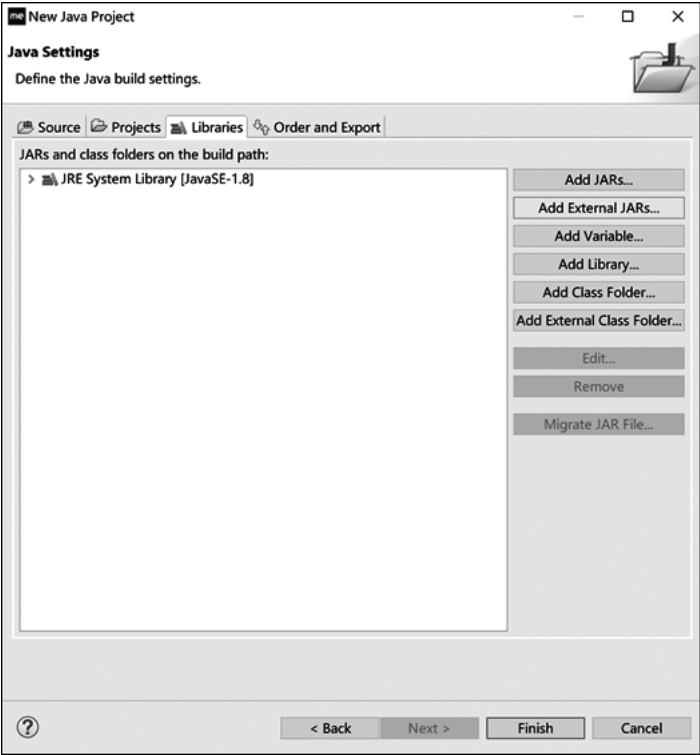


图 5-5 单击 Add External JARs 选项

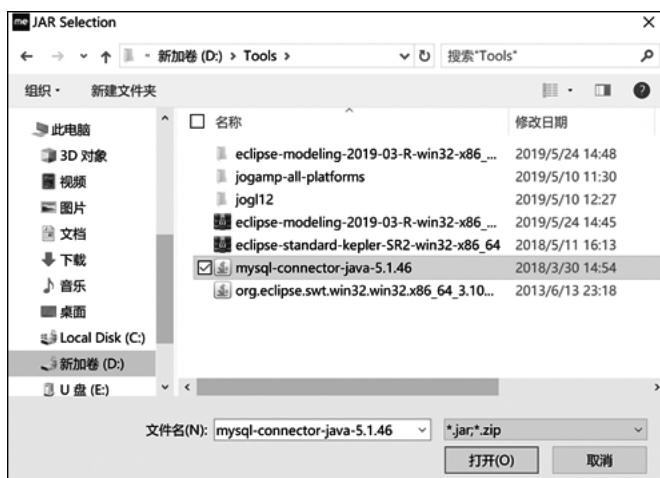


图 5-6 添加 MySQL Connector Java 包

(6) 单击“打开”按钮,完成项目环境搭建。

(7) 建立 package,右击项目的 src 文件夹,选择 New→Package,如图 5-7 所示。

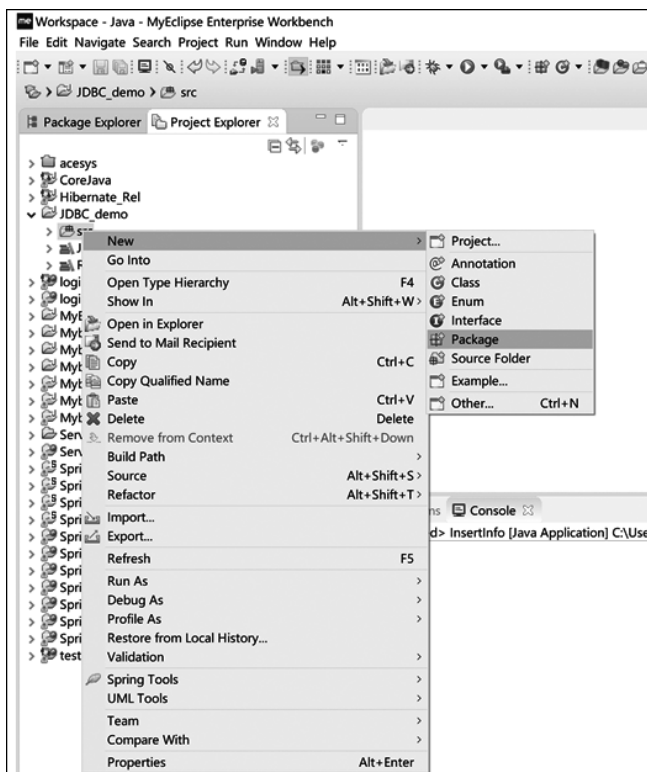


图 5-7 新建 package

(8) 在 Name 文本框中输入 sample,建立 sample 包,如图 5-8 所示。

(9) 建立类,右击 sample 文件夹,选择 New→Class,如图 5-9 所示。

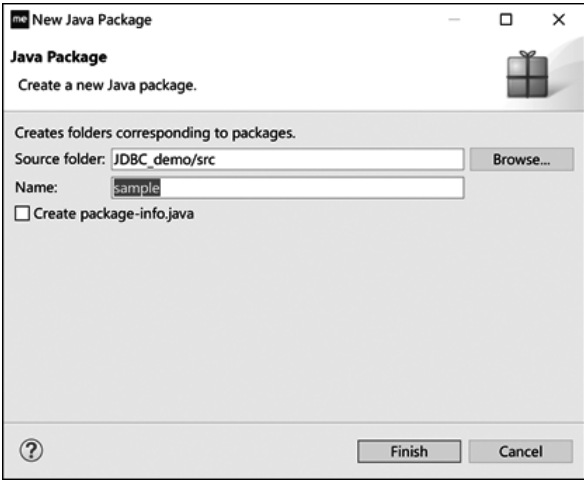


图 5-8 命名 package

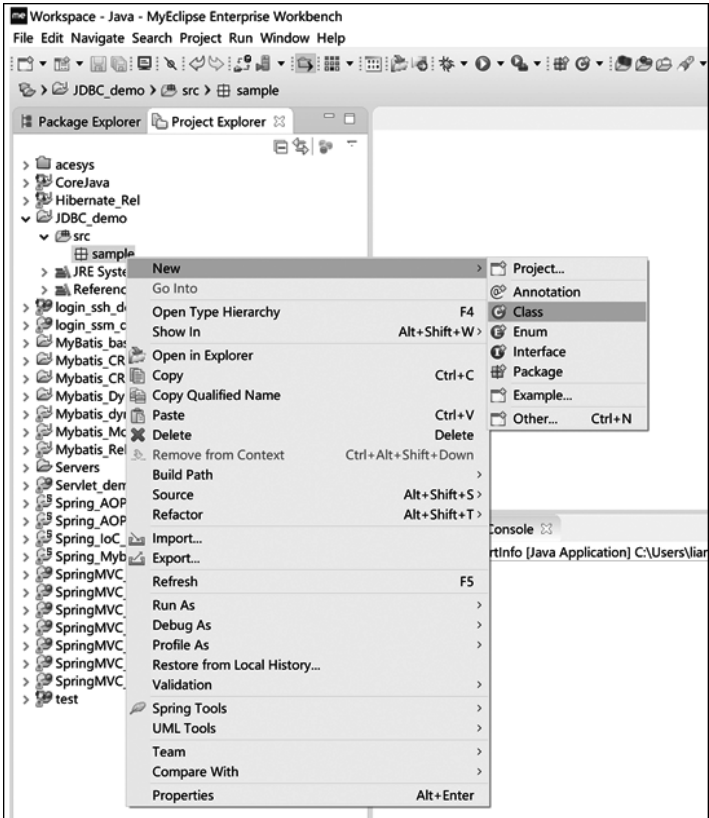


图 5-9 新建 class

- (10) 在 Name 文本框中输入 InsertInfo 给 class 命名,单击 Finish 按钮,如图 5-10 所示。
- (11) 编写 InsertInfo.java 代码如下。