实验报告

第 页

专业\_软件工程\_\_\_\_ 班级\_软工一班\_\_\_ 学号\_116052017001\_ 姓名 周靖赟

实验日期： 2019年 9 月 2 日 报告退发 (订正 、 重做)

课程 ERP 实验名称 高级编程语言与数据库系统的可配置连接

一、实验目的

ERP系统是计算机和数据库知识在企业管理中的应用。因此一个成功的ERP系统必定是高级编程语言与数据库系统的结合。高级编程语言与数据库的连接是开发ERP系统的基础。

1. 掌握高级程序与数据库系统的连接。
2. 考虑系统的移植性，该连接应该是可配置的。

二、实验环境（描述实验的软件、硬件环境）

1. 高性能计算机1台
2. 数据库系统
3. 高级开发语言

三、实验内容、步骤和结果分析

将高级编程语言与数据库系统连接的方法中的数据库服务器名、登陆的用户名和密码等关键元素以合适的方式进行处理，最终实现通过配置方式适应服务器名的变化。系统运行后，读取配置文件中的内容，形成连接字符串，调用相应的方法实现高级编程语言与数据库系统连接。

1. 讨论

### Db.properties

|  |
| --- |
| db.host=localhost  db.port=3306  db.user=arsia  db.passwd=154785  db.dbname=arsiadb |

### ReadConfigure.java

|  |
| --- |
| package main;  import java.io.BufferedInputStream;  import java.io.File;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.InputStream;  import java.util.Properties;  public class ReadConfigure {      private File configFile;      public ReadConfigure(String configFile) {          this.configFile = new File(configFile);      }      /\* Getter and Setter \*/      public void setConfigFile(String configFileName) {          this.configFile = new File(configFileName);      }      public void setConfigFile(File configFile) {          this.configFile = configFile;      }      public File getConfigFile() {          return configFile;      }      /\* Achieving Goals \*/      public Properties getAllConfigure() throws IOException {          Properties config = new Properties();          FileInputStream configFileInputStream = new FileInputStream(configFile);          InputStream configStream = new BufferedInputStream(configFileInputStream);          config.load(configStream);          return config;      }  } |

### DataBaseSystem.java

|  |
| --- |
| package main;  import java.sql.Connection;  import java.sql.SQLException;  import java.sql.Statement;  /\*\*   \* @package: jdbc   \* @Description: 所有数据库类的父类   \* @program: order-management-system   \* @Author: Arsia Chou   \* @date: 2019-05-29 16:16   \*/  public abstract class DataBaseSystem {      public static final String LOCALHOST = "localhost";      private Connection connect;      private Statement statement;      protected abstract void flushUrl();      public abstract boolean connect();      public boolean disconnect() {          boolean success = true;          try {              if (connect != null) {                  if (!connect.isClosed()) {                      connect.close();                  }              }              if (statement != null) {                  if (!statement.isClosed()) {                      statement.close();                  }              }          } catch (SQLException e) {              success = false;              System.out.println("DatabaseSystem: 关闭连接失败 " + e.getMessage());          }          return success;      }      public Connection getConnect() {          return connect;      }      public void setConnect(Connection connect) {          this.connect = connect;          try {              statement = connect.createStatement();          } catch (SQLException e) {              System.out.println("DataBaseSystem: 创建statement失败");          }      }      public Statement getStatement() {          return statement;      }  } |

### MysqlConnect.java

|  |
| --- |
| package main;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.util.Properties;  public class MysqlConnect extends DataBaseSystem {      private String url = "jdbc:mysql://%HOST%:%PORT%/%DATABASE%?useUnicode=true&characterEncoding=utf-8&useSSL=false";      private Properties config;      public MysqlConnect(Properties config) {          this.config = config;          flushUrl();      }      /\*\*       \* load driver       \*/      static {          try {              Class.forName("com.mysql.jdbc.Driver");          } catch (ClassNotFoundException e) {              System.out.println("Mysql: Driver Not Found. " + e.getMessage());          }      }      protected void flushUrl() {          url = url.replace("%HOST%", config.getProperty("db.host")).replace("%PORT%", config.getProperty("db.port"))                  .replace("%DATABASE%", config.getProperty("db.dbname"));      }      public boolean connect() {          boolean result;          try {              super.setConnect(                      DriverManager.getConnection(url, config.getProperty("db.user"), config.getProperty("db.passwd")));              result = true;          } catch (SQLException e) {              result = false;              System.out.println("Mysql: Connection Failed. " + e.getMessage());          }          return result;      }      public String getUrl() {          return url;      }      public void setConfig(Properties config) {          this.config = config;          flushUrl();      }  } |

### Test.java

|  |
| --- |
| package main;  import java.io.IOException;  import main.MysqlConnect;  import main.ReadConfigure;  public class Test {      private static final String DATA\_DB\_PROPERTIES = "../data/db.properties";      public static void main(String[] args) {          ReadConfigure readConfig = new ReadConfigure(DATA\_DB\_PROPERTIES);          boolean isConnect = false;          try {              MysqlConnect mysql = new MysqlConnect(readConfig.getAllConfigure());              System.out.println("url: " + mysql.getUrl());              isConnect = mysql.connect();              System.out.println("connect: " + isConnect);          } catch (IOException e) {              e.printStackTrace();          }      }  } |