Java实验报告

实验一

1. 实验目的

掌握java 语言的一些基本技能。

1. 实验内容

static域和方法的理解和使用 ,

this理解和使用

string不可变类的理解和使用

stringBuilder可变类的理解和使用

1. 实验过程与结果分析

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. static域和方法的理解和使用  |  | | --- | | **package** static\_Attributes;  **import** java.util.Scanner;  **public** **class** ls {  **private** Scanner scan;  //静态变量  **public** **static** **int** *a* = 5;  **public** **int** m = 5;    **public** **static** **void** showNumber1(){  System.***out***.println("1 is a number");  //静态类中无法调用this指针  }    **public** **void** getTheInputValue(){  scan = **new** Scanner(System.***in***);  **int** a = scan.nextInt();  System.***out***.print(a);    }    } | | **package** static\_Attributes;  **public** **class** experiment {  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub    System.***out***.println(ls.*a*);  //System.out.println(ls.m); m不是静态变量所以无法打印出来  ls l = **new** ls();  System.***out***.println(l.m);    ls.*showNumber1*();  l.getTheInputValue();  //ls.getTheInputValue(); getTheInputValue()不是静态函数所以无法在不实例化的情况下进行调用  }  } |  1. this的理解和使用  |  | | --- | | **package** theUnderstandingOfThis;  **public** **class** getValue {  **int** a;  getValue(){  a = 0;  }  **public** getValue selfAdd(){  a++;  **return** **this**;  }  **public** **void** print(){  System.***out***.println(a);  }  } | | **package** theUnderstandingOfThis;  **public** **class** Experiment1 {    **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub  getValue m = **new** getValue();  m.selfAdd().selfAdd().selfAdd().print();  }  } |  1. String不可变类的理解和使用  |  | | --- | | **package** non\_variable\_class\_String;  **public** **class** NvcString {    **public** **static** String upcase(String s){  **return** s.toUpperCase();  }  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub  String a = "asdfg";  System.***out***.println(a);  String b = *upcase*(a);  System.***out***.println(b);  System.***out***.println(a);  //由此可见在调用upcase的过程中并没有对原先的字符串做出改变，而是新建了一个字符串  }  } |  1. stringBuilder可变类的理解和使用  |  | | --- | | **package** variable\_class\_StringBulider;  **public** **class** Vc\_StringBuilder {  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub  String str = "";  StringBuilder stb = **new** StringBuilder();  **long** startTime = System.*currentTimeMillis*();  **for**(**int** i = 0; i < 1000; i++){  str = str + i;  }  **long** endTime = System.*currentTimeMillis*();  **long** Time = startTime - endTime;  System.***out***.println("String消耗时间"+Time);  startTime = System.*currentTimeMillis*();  **for**(**int** i = 0; i < 1000; i++){  stb.append(i);  }  endTime = System.*currentTimeMillis*();  Time = startTime - endTime;  System.***out***.println("StringBuilder消耗时间"+Time);  //通过比对两者运算时间，可以很容易的发现StringBuilder所消耗的事件比String少    /\*\*\*\*\*  \* 创建 成功 的 字符串 对象， 其 长度 是 固定 的， 内容 不能 被 改变 和 编译。 虽然 使用“+” 可以达到 附加 新 字符 或 字符串 的 目的， 但“+” 会 产生 一个 新的 String 实例， 会在 内存 中创建 新的 字符串 对象。 如果 重 复地 对 字符串 进行 修改， 将 极大 地 增加 系统 开销。 而 J2SE 5. 0 新增 了 可变 的 字符 序列 String- Builder 类， 大大 提高 了 频繁增加 字符串 的 效率。\* \*/    }  } | |

1. 总结
2. Static静态函数只能调用静态变量
3. Static静态函数不能调用this指针
4. Static往往表示这一个共享区域，为这个类所有成员所拥有
5. This指针代表着这个类自身
6. String在附加另一个字符串时往往不会改变这个变量自身，而是重新创建一个变量将附加的字符串和原来的拼接起来重新存入一个新的变量。
7. StringBuilider类所使用的Append（）函数却不用再次经历这一个过程。因此在效率上StringBuilder会比string类更加有效。

实验二

1. 实验目的

学会使用JDBC存取典型关系数据库，比如MySql数据，对数据进行适当的处理。

1. 实验内容

1.驱动程序装载

2.数据库连接

3.数据库数据的存取

4.数据库数据的处理

1. 实验过程与结果分析

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验代码   |  |  | | --- | --- | | **package** pkg;  **import** java.sql.\*;  **public** **class** Main {  **public** **static** **void** main(String [] args) {  String driverName="com.microsoft.sqlserver.jdbc.SQLServerDriver";  String dbURL="jdbc:sqlserver://localhost:1433;DatabaseName=test";  String userName="sa";  String userPwd="shixing19970805";  **try**{  Class.*forName*(driverName);  System.***out***.println("加载驱动成功！");  }**catch**(Exception e){  e.printStackTrace();  System.***out***.println("加载驱动失败！");  }  **try**{  Class.*forName*(driverName);  Connection dbConn=DriverManager.*getConnection*(dbURL,userName,userPwd);  System.***out***.println("连接数据库成功！");  String sql = "SELECT \* FROM stu";  Statement stmt = dbConn.createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  System.***out***.print(rs.getInt("ID")+"\t");  System.***out***.print(rs.getString("name")+"\t");  System.***out***.println(rs.getString("score"));  }  String ins = "INSERT INTO stu VALUES (?,?,?)";  PreparedStatement ps = dbConn.prepareStatement(ins);  ps.setString(1, "17");  ps.setString(2, "老虎");  ps.setString(3, "71");  ps.execute();  sql = "SELECT \* FROM stu";  rs = stmt.executeQuery(sql);  **while**(rs.next()) {  System.***out***.print(rs.getInt("ID")+"\t");  System.***out***.print(rs.getString("name")+"\t");  System.***out***.println(rs.getString("score"));  }    rs.close();  stmt.close();  dbConn.close();  }**catch**(Exception e){  e.printStackTrace();  System.***out***.print("SQL Server连接失败！");  }  }  } |  | | 加载驱动成功！  连接数据库成功！  1 wanga 47  2 laotian 77  3 xiaozi 98  16 dvd 14  76 huangmao 76  1 wanga 47  2 laotian 77  3 xiaozi 98  16 dvd 14  17 老虎 71  76 huangmao 76 |  | |

四、总结

1. 数据库加载需要相应的包和对应服务器地址
2. 服务器连接时需要对应和匹配的密码
3. Class.forName()函数用来加载相应的数据库驱动程序
4. DriverManager.getConnection()是一个用于连接数据库的函数，会返回一个Connection类型的变量，同时三个参数类型分别是服务器地址，服务器用户名，服务器密码。
5. Statement类是用于处理SQL语句的
6. ResultSet用于接收查询语句结果
7. PreparementStatement 用于处理SQL语句中某些待定值。

实验三

1. 实验目的

了解和认识Java web application的涉及的核心技术、组成和开发过程

1. 实验内容

实现一个简单的学生信息管理：学生信息（学号、姓名、性别和年龄等）输入；学生信息的查询；

客户端设计与实现

 HTML

 CSS

 Javascript

服务器端设计与实现

 Servlet

 Java

1.  JDBC实验过程与结果分析

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学生数据查找前端网页代码   |  | | --- | | <!DOCTYPE html>  <html>  <head>  <title>MyHtml.html</title>    <meta name=*"keywords"* content=*"keyword1,keyword2,keyword3"*>  <meta name=*"description"* content=*"this is my page"*>  <meta name=*"content-type"*>  <meta http-equiv=*"content-type"* content=*"text/html;charset=UTF-8"*>    <!--<link rel="stylesheet" type="text/css" href="./styles.css">-->    </head>    <body>  <form action=*"Test"* method=*"POST"*>  姓名：<input type=*"text"* name=*"Nmae"*><br>  <input type=*"submit"* value=*"查询"*>  </form>  <form action=*"Insert"*>    </form>  </body>  </html> |   查找Severlet代码   |  | | --- | | **package** sev\_DB;  **import** java.io.IOException;  **import** java.io.PrintWriter;  **import** java.sql.\*;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebInitParam;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  @WebServlet(  urlPatterns = { "/Test" },  initParams = {  @WebInitParam(name = "Test", value = "/TomcatTest/Test")    })  **public** **class** Test **extends** HttpServlet {    **private** **static** **final** **long** ***serialVersionUID*** = 1L;  //数据库服务器端  **static** **final** String ***ODBC\_DRIVER*** = "com.microsoft.sqlserver.jdbc.SQLServerDriver";  **static** **final** String ***DB\_URL*** = "jdbc:sqlserver://localhost:1433;DatabaseName=test";    //用户登录名和登录密码  **static** **final** String ***USER*** = "sa";  **static** **final** String ***PASS*** = "shixing19970805";    **static** PreparedStatement *ps*;  /\*\*  \* Constructor of the object.  \*/  **public** Test() {  **super**();  }    **public** **void** destroy() {  **super**.destroy(); // Just puts "destroy" string in log  // Put your code here  }    **public** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  Connection conn = **null**;  Statement stmt = **null**;    response.setContentType("text/html;charset=UTF-8");  request.setCharacterEncoding("UTF-8");//防止乱码  PrintWriter out = response.getWriter();  String title = "查询学生信息";  String docType = "<!DOCTYPE html>\n";  out.println(docType +  "<html>\n" +  "<head><title>" + title + "</title></head>\n" +  "<body bgcolor=\"#f0f0f0\">\n" +  "<h1 align=\"center\">" + title + "</h1>\n");  **try**{  //加载数据库驱动  Class.*forName*("com.microsoft.sqlserver.jdbc.SQLServerDriver");    //连接数据库  conn = DriverManager.*getConnection*(***DB\_URL***,***USER***,***PASS***);  String na = request.getParameter("Nmae");  stmt = conn.createStatement();  String sql = "SELECT \* FROM stu WHERE name = '"+ na + "'";    //执行数据库的语句  ResultSet rs = stmt.executeQuery(sql);  //显示查询结果  **while**(rs.next()){    **int** id = rs.getInt("id");  String name = rs.getString("name");  String score = rs.getString("score");        out.println("ID: " + id);  out.println(" 姓名: " + name);  out.println(" 得分: " + score);  out.println("<br />");  }  out.println("</body></html>");      rs.close();  stmt.close();  conn.close();      } **catch**(SQLException se) {    se.printStackTrace();  } **catch**(Exception e) {    e.printStackTrace();  }**finally**{    **try**{  **if**(stmt!=**null**)  stmt.close();  }**catch**(SQLException se2){  }  **try**{  **if**(conn!=**null**)  conn.close();  }**catch**(SQLException se){  se.printStackTrace();  }  }  }    **public** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  doGet(request,response);  }  /\*\*  \* Initialization of the servlet. <br>  \*  \* **@throws** ServletException if an error occurs  \*/  **public** **void** init() **throws** ServletException {  // Put your code here  }  } |   插入的前端代码设计   |  | | --- | | <!DOCTYPE html>  <html>  <head>  <title>Insertl.html</title>    <meta name=*"keywords"* content=*"keyword1,keyword2,keyword3"*>  <meta name=*"description"* content=*"this is my page"*>  <meta name=*"content-type"* content=*"text/html; charset=UTF-8"*>  <meta http-equiv=*"content-type"* content=*"text/html;charset=UTF-8"*>  <!--<link rel="stylesheet" type="text/css" href="./styles.css">-->  </head>    <body>  This is my HTML page. <br>  <form action=*"Insert"* method=*"post"*>  学号：<input type=*"text"* name=*"ID"*><br>  姓名：<input type=*"text"* name=*"name"*><br>  分数：<input type=*"text"* name=*"score"*><br>  <input type=*"submit"* value=*"插入"*>  </form>  </body>  </html> |   插入的Severlet设计代码   |  | | --- | | **package** sev\_DB;  **import** java.io.IOException;  **import** java.io.PrintWriter;  **import** java.sql.\*;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebInitParam;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  @WebServlet(  urlPatterns = { "/Insert" },  initParams = {  @WebInitParam(name = "Insert", value = "/TomcatTest/Insert")    })  **public** **class** Insert **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  //数据库服务器端  **static** **final** String ***ODBC\_DRIVER*** = "com.microsoft.sqlserver.jdbc.SQLServerDriver";  **static** **final** String ***DB\_URL*** = "jdbc:sqlserver://localhost:1433;DatabaseName=test";    //用户登录名和登录密码  **static** **final** String ***USER*** = "sa";  **static** **final** String ***PASS*** = "shixing19970805";    **static** PreparedStatement *ps*;  /\*\*  \* Constructor of the object.  \*/  **public** Insert() {  **super**();  }  /\*\*  \* Destruction of the servlet. <br>  \*/  **public** **void** destroy() {  **super**.destroy(); // Just puts "destroy" string in log  // Put your code here  }    **public** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  Connection conn = **null**;  Statement stmt = **null**;    response.setContentType("text/html;charset=UTF-8");  request.setCharacterEncoding("UTF-8");//防止乱码  PrintWriter out = response.getWriter();  String title = "插入学生信息";  String docType = "<!DOCTYPE html>\n";  out.println(docType +  "<html>\n" +  "<head><title>" + title + "</title></head>\n" +  "<body bgcolor=\"#f0f0f0\">\n" +  "<h1 align=\"center\">" + title + "</h1>\n");  **try**{  //加载数据库驱动  Class.*forName*("com.microsoft.sqlserver.jdbc.SQLServerDriver");    //连接数据库  conn = DriverManager.*getConnection*(***DB\_URL***,***USER***,***PASS***);  String na = request.getParameter("Nmae");  stmt = conn.createStatement();  String ins = "INSERT INTO stu VALUES (?,?,?)";  *ps* = conn.prepareStatement(ins);  *ps*.setString(1, request.getParameter("ID"));  *ps*.setString(2,request.getParameter("name"));  *ps*.setString(3, request.getParameter("score"));  *ps*.execute();  String sql = "SELECT \* FROM stu";  ResultSet rs = stmt.executeQuery(sql);  //显示查询结果  **while**(rs.next()){    **int** id = rs.getInt("id");  String name = rs.getString("name");  String score = rs.getString("score");        out.println("ID: " + id);  out.println(" 姓名: " + name);  out.println(" 得分: " + score);  out.println("<br />");  }  out.println("</body></html>");      rs.close();  stmt.close();  conn.close();      } **catch**(SQLException se) {    se.printStackTrace();  } **catch**(Exception e) {    e.printStackTrace();  }**finally**{    **try**{  **if**(stmt!=**null**)  stmt.close();  }**catch**(SQLException se2){  }  **try**{  **if**(conn!=**null**)  conn.close();  }**catch**(SQLException se){  se.printStackTrace();  }  }    }    **public** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  **this**.doGet(request, response);    }    **public** **void** init() **throws** ServletException {  // Put your code here  }  } |   WebXML设计代码   |  | | --- | | <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN" "http://java.sun.com/dtd/web-app\_2\_3.dtd">  <web-app>  <servlet>  <servlet-name>Test</servlet-name>  <display-name>This is the display name of my J2EE component</display-name>  <description>This is the description of my J2EE component</description>  <servlet-class>sev\_DB.Test</servlet-class>  </servlet>  <servlet>  <servlet-name>Insert</servlet-name>  <display-name>This is the display name of my J2EE component</display-name>  <description>This is the description of my J2EE component</description>  <servlet-class>sev\_DB.Insert</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>Test</servlet-name>  <url-pattern>/TomcatTest/Test</url-pattern>  </servlet-mapping>  <servlet-mapping>  <servlet-name>Insert</servlet-name>  <url-pattern>/TomcatTest/Insert</url-pattern>  </servlet-mapping>  </web-app> | |

1. 总结
2. 环境配置时要将驱动包放在lib文件内，否则会出现找不到驱动的异常情况，其次直接将jar文件直接拖入lib文件中，通过importPath操作方法无法使对应的数据库驱动文件放置在lib文件夹内。
3. HTML文件中<form>标签内action属性的值要对应相关的Severlet名称。Name属性是放置在<form>标签的子元素内，用于使Severlet获得相应表单的元素。
4. 对于数据的查找可使用PreparementStatement类。
5. Request.getParameter用于获得页面中表单元素的信息。