# 《Java ProgramingⅡ》实验报告

**班级：1622107022**

**学号：162210702226**

**姓名：汪远材**

**实验一、 java 语言基本技能**

1. **实验目的**

掌握java 语言的一些基本技能。

1. **实验内容**

1、static域和方法的理解和使用 ,

2、this理解和使用

3、string不可变类的理解和使用

4、stringBuilder可变类的理解和使用

1. **实验过程和结果分析**

**1、static**

静态域也称为类域或静态变量，每个类中只有一个静态域。当一个方法或者变量需要初始化加载，或者是经常被调用的时候可以加上static。用static修饰的方法可以用类名直接调用。

特点：

1. 随着类的加载而加载，也随着类的消失而消失.
2. 被所有对象共享.
3. 优先于对象存在.
4. 可以直接以类名调用.

注：

1. 静态方法只能访问静态成员
2. 静态方法中不可定义this，super关键字。
3. 代码示例：

package shiyan\_1;

public class javabase {

//设置静态变量和非静态变量

public static String s1="StaticValue1";

public static String s2="StaticValue2";

public String s3="NotStaticValue1";

public String s4="NotStaticValue2";

//静态方法只能调用静态变量

public static void str1(){

System.out.println(s1);

System.out.println(s2);

}

//非静态方法对静态和非静态变量的调用无限制

public void str2(){

System.out.println(s1);

System.out.println(s2);

System.out.println(s3);

System.out.println(s4);

}

public static void main(String[] args) {

javabase obj=new javabase();

//非静态方法可以调用静态变量和非静态变量

obj.str2();

//可直接通过类调用静态方法

javabase.str1();

}

}

**2、this**

this指自己所在的对象。在一个方法中，调用其他对象的变量或方法时，可以使用那个对象的对象名，比如a.abc();而调用自己所在对象的方法或变量时，不知道对象名，所以就用this.abc()。

用法：

1、当在一个类中要明确指出使用对象自己的变量或函数时就应该加上this引用。

2、this调用构造方法。

代码示例：

package shiyan\_1;

public class javabase {

public String h1;

public String h2;

//this可调用本类中的成员变量、方法

javabase(String x, String y){

this(x);

this.h2=y;

this.strbuilder();

}

public void strbuilder(){

StringBuilder sb=new StringBuilder();

sb.append(h1);

sb.append(h2);

System.out.println(sb);

}

public static void main(String[] args) {

javabase obj=new javabase("Hello ", "World!");

}

}

**3、String理解和使用**

String类是final类，也就意味着String类不能被继承，并且它的成员方法都默认为final方法。在Java中，被final修饰的类是不允许被继承的，并且该类中的成员方法都默认为final方法。

**package** shiyan\_1

**public** **class** javabase {

**public** **static** **void** main(String args[]){

String s1="String1";

String s2=s1;

//对象

String s3=**new** String("String1");

String s4=**new** String("String1");

System.***out***.println(s3==s4);

String s5=s1+s2+"String5";

System.***out***.println(s5);

}

}

**3、StringBuilder理解和使用**

String 对象是不可改变的。每次使用String 类中的方法之一时，都要在内存中创建一个新的字符串对象，这就需要为该新对象分配新的空间。在需要对字符串执行重复修改的情况下，与创建新的 String 对象相关的系统开销可能会非常昂贵。如果要修改字符串而不创建新的对象，则可以使用StringBuilder 类。

主要方法及使用：

**1、append(String str)/append(Char c)：字符串连接**

**2、toString()：返回一个与构建起或缓冲器内容相同的字符串**

**3、setCharAt(int i, char c)：将第 i 个代码单元设置为 c（可以理解为替换）**

**4、insert(int offset, String str)/insert(int offset, Char c)：在指定位置之前插入字符(串)**

**5、delete(int startIndex,int endIndex)：删除起始位置（含）到结尾位置（不含）之间的字符串**

代码示例：

**package** shiyan\_1

**import** java.lang.StringBuilder;

**public** **class** javabase {

**public** **static** **void** main(String[] args) {

StringBuilder sb1=**new** StringBuilder("Hello ");

System.***out***.println(sb1);

sb1.append("World");

sb1.append('!');

System.***out***.println(sb1);

sb1.setCharAt(5, '~');

System.***out***.println(sb1);

sb1.insert(5, " better");

System.***out***.println(sb1);

sb1.delete(6, 15);

System.***out***.println(sb1);

}

}

**四、总结**

1. **实验内容总结**

**从static到String，StringBuilder的百度了解理论和实际代码运行验证，我更加深入的了解到了static的特点注意点等等；同时也重温了this运用，使我更加的熟悉this；从内存角度理解了String的存储和其特点；也回顾了StringBuilder可变字符类的方法。**

1. **心得体会**

通过这次实验，我更加熟悉和灵活的运用Java基础知识来解决许多编程问题；很多知识点也更加的深入的理解了。重新找回原来的许多知识点并不是很难，而且还是有一些高兴的。有的原来不是很理解的点，从头来过并且更深入的了解后，印象更加深刻了。

**实验二 JDBC数据库编程**

**一、实验目的**

学会使用JDBC存取典型关系数据库，例如MySql数据，对数据进行操作和处理。

1. **实验内容**

1.驱动程序装载

驱动是实现了**java.sql.driver**接口的一个类，（此类可以在mysql提供的JDBC.jar包中找到），此类实现了**java.sql.Driver**接口。由于驱动的本质是一个类，所以加载驱动的过程其实就是加载类的过程：

**Class.forName("com.mysql.cj.jdbc.Driver");**

2.数据库连接

对于驱动加载后，获取指定的驱动程序，**DriverManager**的静态方法**getConnection(String url,String username,String password)**可以通过传递给的URL和相应的用户名及密码，链接上相应的数据库。

3.数据库数据的存取：

使用**Connection**[对象](http://courseware.ecnudec.com/zsb/zjs/zjs13/zjs13c/API_docs/api/java.sql.Connection.html)的[**CreateStatement()**方法](http://courseware.ecnudec.com/zsb/zjs/zjs13/API_docs/api/java.sql.Connection.html#createStatement())建立一个[**Statement**对象](http://courseware.ecnudec.com/zsb/zjs/zjs13/API_docs/api/java.sql.Statement.html)，再利用该对象的**executeQuery()**方法让数据库执行相应的SQL语句。

4.数据库数据的处理

用**executeQuery()**方法让数据库执行相应的SQL语句。实现的数据的处理。最后关闭相应的资源。

**总体代码：**

package shiyan\_2;

import java.sql.\*;

public class jdbc\_test {

public static void main(String[] args) throws SQLException {

// TODO Auto-generated method stub

String driver = "com.mysql.jdbc.Driver";

String url = "jdbc:mysql://localhost:3306/STU";

String userName = "root";

String passWord = "123";

Connection con=null;

PreparedStatement stmt=null;

ResultSet rs=null;

try

{

Class.forName(driver);

System.out.println("load driver class successfully");

con=DriverManager.getConnection(url, userName,passWord);

System.out.println("connect BC succseefully");

String sql="select \* from student";

stmt=con.prepareStatement(sql);

rs=stmt.executeQuery();

while(rs.next())

{

System.out.println(rs.getString(1)+"--"+rs.getString(2));

}

}

catch(Exception e)

{

e.printStackTrace();

System.exit(0);

}

finally {

if(con!=null){

con.close();

}

if(stmt!=null){

stmt.close();

}

if(rs!=null){

rs.close();

}

}

}

}

**调用工具类及数据库的查询操作：**

**package** just.hq.experment2;

**import** java.sql.Connection;

**import** java.sql.ResultSet;

**import** java.sql.Statement;

**public** **class** ToolTest {

**public** **static** **void** testResultSet(){

Connection conn=**null**;

Statement statement=**null**;

ResultSet rs=**null**;

**try** {

conn=JDBCTool.*getConnection*();

statement =conn.createStatement();

String sql="SELECT id,name " +"FROM student";

//name 后面要加一个空格

rs = statement.executeQuery(sql);

**while**(rs.next()){

**int** id=rs.getInt(1);

String name=rs.getString("name");

System.***out***.print(id+"-");

System.***out***.println(name);

}

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}**finally**{

JDBCTool.*releaseDB*(rs, statement, conn);

}

}

**public** **static** **void** main(String[] args) **throws** Exception {

String sql;

sql="INSERT INTO student(id)VALUES(21)";

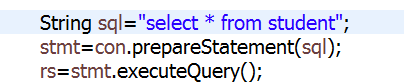
JDBCTool.*update*(sql);

}

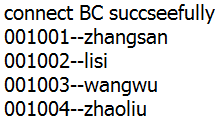
}

1. **实验结果分析及总结**

查询数据库STU中表student的内容：



结果：



总结：Java可通过驱动程序连接到其他数据库，并对数据库进行一系列的操作，了解jdbc编程后，可直接通过java程序来操作后台的数据库，为我们涉及到数据库的编程带来诸多方便。

**实验三、四 Java web application**

**一、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

**二．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；

1、客户端设计与实现

HTML

CSS

Javascript

2、务器端设计与实现

Servlet

Java

JDBC

1. **实验过程及结果分析**

1.首先建立Search类:

public class Search extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

request.setCharacterEncoding("UTF-8");

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

Connection con;

try {

System.out.print("Loading Database…");

System.out.println();

Class.forName("com.mysql.jdbc.Driver");

System.out.println("Loading Successfully!");

con = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/test2?useSSL=false","root","123");

System.out.println("Connecting Successfully!");

String name=request.getParameter("usernamesearch");

System.out.print("姓名："+name);

System.out.println();

String sql = "SELECT \* FROM table3 where name="+"'"+name+"'";

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while(rs.next()){

out.print(rs.getString("id")+"<br>");

out.print(rs.getString("name")+"<br>");

out.print(rs.getString("sex")+"<br>");

out.print(rs.getString("age")+"<br>");

out.print(rs.getString("score")+"<br>");}

con.close();

out.close();

} catch (ClassNotFoundException | SQLException e) {

e.printStackTrace();

}

}

public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

this.doGet(request, response);

}

}

2.创建Insert插入类。

public class Insert extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

request.setCharacterEncoding("UTF-8");

response.setContentType ("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

Connection con;

try {

Class.forName("com.mysql.jdbc.Driver");

System.out.println("Loading Successfully!");

con = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/test2?useSSL=false","root","123");

System.out.println("Connecting Successfully!");

System.out.println("输入学生信息:");

String id=request.getParameter("id");

String name=request.getParameter("name");

String sex=request.getParameter("gender");

String age=request.getParameter("age");

String score=request.getParameter("score");

String sql2="insert into table3(id,name,sex,age,score)value(?,?,?,?,?)";

PreparedStatement pstmt = con.prepareStatement(sql2);

pstmt.setString(1, id);

pstmt.setString(2, name);

pstmt.setString(3, gender);

pstmt.setString(4, age);

pstmt.setString(5, score);

pstmt.execute();

out.print(id+"<br>");

out.print(name+"<br>");

out.print(sex+"<br>");

out.print(age+"<br>");

out.print(score+"<br>信息插入…");

con.close();

System.out.print("插入成功!");

} catch (ClassNotFoundException | SQLException e) {

e.printStackTrace();

}

}

public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

this.doGet(request, response);

}

}

3.然后是Servlet接口

public class stu extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");

out.println("<HTML>");

out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");

out.println(" <BODY>");

out.print(" This is ");

out.print(this.getClass());

out.println(", using the GET method");

out.println(" </BODY>");

out.println("</HTML>");

out.flush();

out.close();

}

/\*\*

\* The doPost method of the servlet. <br>

\* This method is called when a form has its tag value method equals to post.

\* @param request the request send by the client to the server

\* @param response the response send by the server to the client

\* @throws ServletException if an error occurred

\* @throws IOException if an error occurred

\*/

public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");

out.println("<HTML>");

out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");

out.println(" <BODY>");

out.print(" This is ");

out.print(this.getClass());

out.println(", using the POST method");

out.println(" </BODY>");

out.println("</HTML>");

out.flush();

out.close();

}

}

4.最后是Javascript网页代码，构建系统主页，包含输入系统的学号姓名等输入框及提交按钮，查询系统的姓名输入框及查询按钮。

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<html>

<head>

<title>java web applicatioin</title>

</head>

<body>

<form action="insert" method="post">

学生信息管理系统<br/>

学号：<input type="text" name="id"><br/>

姓名：<input type="text" name="name"><br/>

性别：

<input type="radio" name="sex" value="male"/>Male

<input type="radio" name="sex" value="female"/>Female<br>

年龄：<input type="text" name="age"><br/>

分数：<input type="text" name="score"><br/>

<input type="submit" value="提交">

</form>

<br>

<form action="search" method="post">

学生信息查询系统<br>

输入姓名：<input type="text" name="usernamesearch"><br>

<input type="submit" value="查询">

</form>

</body>

</html>

**五、实验总结**

1、实验内容总结

本实验的目的是建立Java web应用程序，开发一个简单的学生信息管理系统。实验步骤如下：首先打开Myeclipse，新建一个Web Project，在Web文件夹新建一个index.jsp文件作为客户端。编写一个Java类实现servlet接口，收集来自网页表单的用户输入，显示来自数据库的数据记录。然后再创建insert和search类，用于插入学生信息和查询学生信息，编写代码使用JDBC操作 MySql的数据，再完成之后一系列的操作。

2、心得体会

通过本次实验，我对JDBC的理解又有了极大的进步。但是本实验大大的超出了我的能力范围，而且之前从未接触过这次实验的内容，在同学的帮助和指导下，又上网查找了很多相关的内容，涉及到很多应用程序开发所需要的东西。至于系统客户端的设计方面没有什么问题，主要是HTML、CSS、JS技术。