# 《Java ProgramingⅡ》实验报告

**班级：1622107022**

**学号：162210702213**

**姓名：韩玉皇**

**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**

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. **实验过程和结果分析**

static方法内部不能调用非静态方法，反过来是可以的。static关键字的基本作用：方便在没有创建对象的情况下来进行调用（方法/变量）。

this关键字主要有三个应用：this调用本类中的属性，也就是类中的成员变量；

public class ReferenceVariable {

private int a;

public ReferenceVariable(int a){

this.a = a;

}

public int getA(){

return a;

}

public void setA(int a){

this.a = a;

}

}

this 代表自身类的对象；

public class ReferenceObject {

ReferenceObject instance;

public ReferenceObject(){

instance = this;

}

public void test(){

System.out.println(this);

}

this引用构造方法

public class ReferenceConstructor {

int a;

public ReferenceConstructor(){

this(0);

}

public ReferenceConstructor(int a){

this.a = a;

}

}

String不可变类

 String s = "ABCabc";

System.out.println("s = " + s);

s = "123456";

System.out.println("s = " + s); 一个对象，在它创建完成之后，不能再改变它的状态，那么这个对象就是不可变的。不能改变状态的意思是，不能改变对象内的成员变量，包括基本数据类型的值不能改变，引用类型的变量不能指向其他的对象，引用类型指向的对象的状态也不能改变。

stringBuilder可变类的理解和使用

public StringBuilder(string value,int cap)

StringBuilder MyStringBuilder = new StringBuilder("Hello World");

线程安全StringBuffer，非线程安全StringBuilder

1. **总结**
2. **实验内容总结**
3. Static静态函数只能调用静态变量
4. Static静态函数不能调用this指针
5. Static往往表示这一个共享区域，为这个类所有成员所拥有
6. This指针代表着这个类自身
7. 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。
8. 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。
9. **心得体会**

这个实验内容很简单，都是一些java入门的知识要点。虽然简单但还是很重要，我也只是理解了这其中一些基本的东西，想要深入理解还是要下一番功夫。

**实验二 JDBC数据库编程**

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**

驱动程序装载

数据库连接

数据库数据的存取

数据库数据的处理

1. **实验过程和结果分析**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class TestDB {

public static final String url ="jdbc:mysql://localhost:3306/mydatabase?useSSL=false";

public static final String JDBC\_DRIVER="com.mysql.jdbc.Driver";

public static final String user="root";

public static final String password="980709";

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

try {

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

System.out.println("连接数据库。。。");

System.out.println(" 学号 "+"\t"+"姓名");

Connection conn=DriverManager.getConnection(url,user,password);

String sql="select \*from student";

Statement stmt=conn.createStatement();

ResultSet rs=stmt.executeQuery(sql);

while(rs.next()){

String number=rs.getString("num");

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

System.out.println(number+"\t"+name);

}

rs.close();

stmt.close();

conn.close();

} catch (ClassNotFoundException e) {

// TODO 自动生成的 catch 块

e.printStackTrace();

}catch(SQLException e){

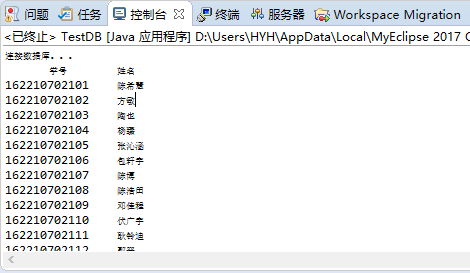
e.printStackTrace();

}catch(Exception e){

e.printStackTrace();

}

}

}****

1. **总结**
2. **实验内容总结**
3. 数据库加载需要相应的包和对应服务器地址
4. 服务器连接时需要对应和匹配的密码
5. Class.forName()函数用来加载相应的数据库驱动程序
6. 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。
7. Statement类是用于处理SQL语句的
8. ResultSet用于接收查询语句结果
9. PreparementStatement 用于处理SQL语句中某些待定值。
10. **心得体会**

连接数据库首先要上网下载对应数据库的驱动包，然后有同学分享了他整理好的数据库的表，拿过来导入到自己的数据库中，最后连接数据库，输入sql语句就好了。我就是因为sql语句中有一个单词拼写错误，检查半天才发现，所以以后还是要注意单词的拼写。

**实验三、四 Java web application**

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
  + JDBC

1. **实验过程和结果分析**

**Servlet代码：**

package demo;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

\* Servlet implementation class MyServlet

\*/

@WebServlet("/MyServlet")

public class MyServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

/\*\*

\* @see HttpServlet#HttpServlet()

\*/

public MyServlet() {

super();

// TODO Auto-generated constructor stub

}

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

String str=request.getParameter("num");

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

String name=getName(str);

PrintWriter out=response.getWriter();

out.write(name);

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

doGet(request, response);

}

//根据学号获取用户名

public String getName(String num) {

String str="";

//连接数据库

Connection connection;

ResultSet set;

try {

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

connection=DriverManager.getConnection("jdbc:mysql://localhost:3306/mydatabase?useSSL=false","root","980709");

Statement statement=connection.createStatement();

String sql="select name from student where num='"

+ num

+ "'";

set=statement.executeQuery(sql);

while(set.next()) {

str=set.getString("name");

}

set.close();

connection.close();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return str;

}

}

**HTML代码：**

<!DOCTYPE html>

<html lang=*"en"*>

<head>

<meta charset=*"UTF-8"*>

<title>Student\_info</title>

</head>

<body>

<form action=*"MyServlet"* method=*"get"*>

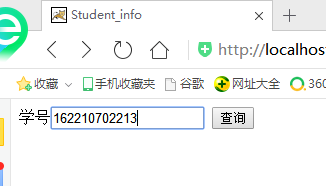
学号<input type=*"text"* name=*"num"*>

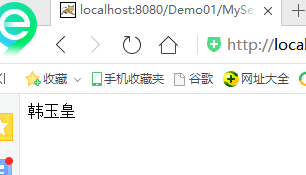
<input type=*"submit"* value=*"查询"*>

</form>

</body>

</html>





1. **总结**
2. **实验内容总结**
3. 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。
4. 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。
5. 对于数据的查找可使用PreparementStatement类。
6. Request.getParameter用于获得页面中表单元素的信息。
7. **心得体会**

这个实验做起来还是有一定难度的，通过很多途径：查阅书籍，上网查找，向同学请教，费了很多时间和精力，中间出现了各种大小问题，绞尽脑汁，最后终于能运行成功。当看到运行结果时，还是有一定的成就感和快感，之前的努力没有白费。