

Servlet 数据库访问

本教程假定您已经了解了 JDBC 应用程序的工作方式。在您开始学习 Servlet 数据库访问之前，请访问 [Java MySQL 连接](#) 来设置相关驱动及配置。

注意：

你可以下载本站提供的 jar 包：[mysql-connector-java-5.1.39-bin.jar](#)

在 java 项目中，只需要在 Eclipse 中引入 mysql-connector-java-5.1.39-bin.jar 就可以运行 java 项目。

但是在 Eclipse web 项目中，当执行 `Class.forName("com.mysql.jdbc.Driver");` 时不会去查找驱动的。所以本实例中我们需要把 mysql-connector-java-5.1.39-bin.jar 拷贝到 tomcat 下 lib 目录。

从基本概念下手，让我们来创建一个简单的表，并在表中创建几条记录。

创建测试数据

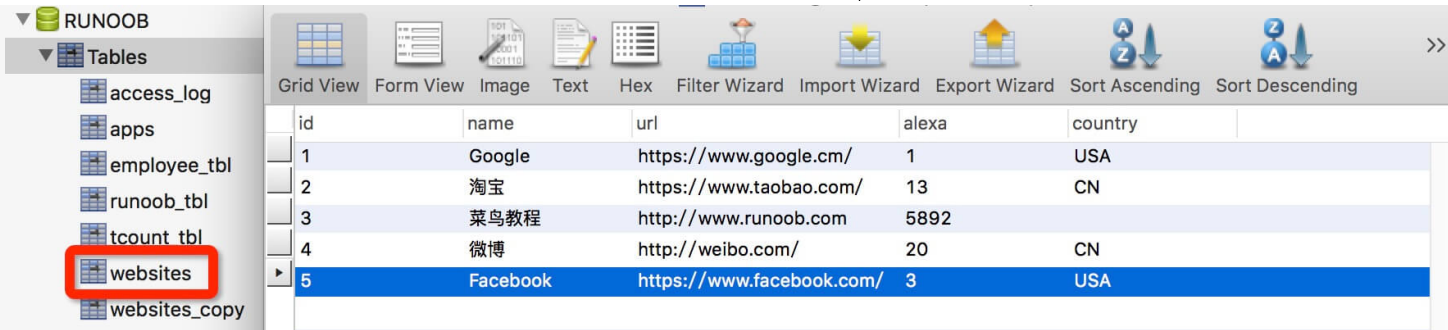
接下来我们在 MySQL 中创建 RUNOOB 数据库，并创建 websites 数据表，表结构如下：

```
CREATE TABLE `websites` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `name` char(20) NOT NULL DEFAULT '' COMMENT '站点名称',  
  `url` varchar(255) NOT NULL DEFAULT '',  
  `alexa` int(11) NOT NULL DEFAULT '0' COMMENT 'Alexa 排名',  
  `country` char(10) NOT NULL DEFAULT '' COMMENT '国家',  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8;
```

插入一些数据：

```
INSERT INTO `websites` VALUES ('1', 'Google', 'https://www.google.cm/', '1', 'USA'), ('2', '淘宝', 'http  
s://www.taobao.com/', '13', 'CN'), ('3', '菜鸟教程', 'http://www.runoob.com', '5892', ''), ('4', '微博',  
'http://weibo.com/', '20', 'CN'), ('5', 'Facebook', 'https://www.facebook.com/', '3', 'USA');
```

数据表显示如下：



| Grid View Form View Image Text Hex Filter Wizard Import Wizard Export Wizard Sort Ascending Sort Descending >> | | | | | |
|--|----------|---------------------------|-------|---------|--|
| id | name | url | alexa | country | |
| 1 | Google | https://www.google.cm/ | 1 | USA | |
| 2 | 淘宝 | https://www.taobao.com/ | 13 | CN | |
| 3 | 菜鸟教程 | http://www.runoob.com | 5892 | | |
| 4 | 微博 | http://weibo.com/ | 20 | CN | |
| 5 | Facebook | https://www.facebook.com/ | 3 | USA | |

访问数据库

下面的实例演示了如何使用 Servlet 访问 RUNOOB 数据库。

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.*;

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 DatabaseAccess
 */
@WebServlet("/DatabaseAccess")
public class DatabaseAccess extends HttpServlet {
    private static final long serialVersionUID = 1L;
    // JDBC 驱动名及数据库 URL
    static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/RUNOOB";

    // 数据库的用户名与密码，需要根据自己的设置
    static final String USER = "root";
    static final String PASS = "123456";
    /**
     * @see HttpServlet#HttpServlet()
     */
    public DatabaseAccess() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
```

```
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
*/
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    Connection conn = null;
    Statement stmt = null;
    // 设置响应内容类型
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    String title = "Servlet Mysql 测试 - 菜鸟教程";
    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{
        // 注册 JDBC 驱动器
        Class.forName("com.mysql.jdbc.Driver");

        // 打开一个连接
        conn = DriverManager.getConnection(DB_URL,USER,PASS);

        // 执行 SQL 查询
        stmt = conn.createStatement();
        String sql;
        sql = "SELECT id, name, url FROM websites";
        ResultSet rs = stmt.executeQuery(sql);

        // 展开结果集数据库
        while(rs.next()){
            // 通过字段检索
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String url = rs.getString("url");

            // 输出数据
            out.println("ID: " + id);
            out.println(", 站点名称: " + name);
            out.println(", 站点 URL: " + url);
            out.println("<br />");
        }
        out.println("</body></html>");

        // 完成后关闭
        rs.close();
        stmt.close();
        conn.close();
    } catch(SQLException se) {
```

```
        // 处理 JDBC 错误
        se.printStackTrace();
    } catch (Exception e) {
        // 处理 Class.forName 错误
        e.printStackTrace();
    } finally {
        // 最后是由于关闭资源的块
        try {
            if (stmt != null)
                stmt.close();
        } catch (SQLException se2) {}
        try {
            if (conn != null)
                conn.close();
        } catch (SQLException se) {
            se.printStackTrace();
        }
    }
}

/**
 * @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);
}
}
```

现在让我们来编译上面的 Servlet，并在 web.xml 文件中创建以下条目：

```
....
<servlet>
    <servlet-name>DatabaseAccess</servlet-name>
    <servlet-class>com.runoob.test.DatabaseAccess</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>DatabaseAccess</servlet-name>
    <url-pattern>/TomcatTest/DatabaseAccess</url-pattern>
</servlet-mapping>
....
```

现在调用这个 Servlet，输入链接：<http://localhost:8080/TomcatTest/DatabaseAccess>，将显示以下响应结果：

Servlet Mysql 测试 - 菜鸟教程

ID: 1, 站点名称: Google, 站点 URL: https://www.google.cm/
ID: 2, 站点名称: 淘宝, 站点 URL: https://www.taobao.com/
ID: 3, 站点名称: 菜鸟教程, 站点 URL: http://www.runoob.com
ID: 4, 站点名称: 微博, 站点 URL: http://weibo.com/
ID: 5, 站点名称: Facebook, 站点 URL: https://www.facebook.com/

[← Servlet Session 跟踪](#)[Servlet 文件上传 →](#)

1 篇笔记

[写笔记](#)

进行数据库插入操作的时候使用 PreparedStatement 更好，好处如下：

- 1. PreparedStatement 可以写动态参数化的查询；
- 2. PreparedStatement 比 Statement 更快；
- 3. PreparedStatement 可以防止 SQL 注入式攻击

实例：

```
//编写预处理 SQL 语句
String sql= "INSERT INTO websites1 VALUES(?,?,?,?,?)";

//实例化 PreparedStatement
ps = conn.prepareStatement(sql);

//传入参数，这里的参数来自于一个带有表单的jsp文件，很容易实现
ps.setString(1, request.getParameter("id"));
ps.setString(2, request.getParameter("name"));
ps.setString(3, request.getParameter("url"));
ps.setString(4, request.getParameter("alexa"));
ps.setString(5, request.getParameter("country"));

//执行数据库更新操作，不需要SQL语句
ps.executeUpdate();
sql = "SELECT id, name, url FROM websites1";
ps = conn.prepareStatement(sql);

//获取查询结果
ResultSet rs = ps.executeQuery();
```

Alan_scut 2年前 (2017-03-23)

