Java、数据库MySQL、socket编程

2021年7月8日　　23:45

- 使用JDBC连接数据库需要4步：
  1、加载驱动程序；
  2、连接数据库；
  3、访问数据库；
  4、执行查询；

参考学习链接：	https://www.it610.com/article/1296527839755051008.htm

- JVM：
  Java Virtual Machine（Java虚拟机）的缩写，JVM是一种用于计算设备的规范，它是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。
- 加载驱动：
  Class.forName(String className)
  使用装载当前类的类装载器来装载制定的类
  在JDBC中 Class.forName("com.mysql.jdbc.Driver")
  作用是要求JVM查找并加载指定的类，
  也就是说JVM会执行该类的静态代码段
- e.printStackTrace()：
  SQLException e是Throwable的实例异常对象，用在catch语句中相当于一个形参，一旦try捕获到了异常，那么就将这个异常信息交给e，由e处理，printStackTrace()是异常类的一个方法，用于查看完整错误信息。
- JDBC里面连接数据库：
  语法一：
  Connection conn=DriverManager.getConnection(String url);
  Connection是接口，DriverManager是一个普通的类，
  getConnection()方法是静态方法，url是访问数据库的 URL 路径。
  MySQL URL路径："jdbc:mysql://localhost:3306/?user=root&password=Root"
  语法二：
  getConnection(String url,Properties info);
  url是访问数据库的 URL 路径；info是一个持久的属性集对象，包括 user 和 password 属性。
- 数据库 Connection.createStatement() 方法
  用于创建一个 Statement 对象，封装 SQL 语句发送给数据库，通常用来执行不带参数的 SQL 语句。

比较prepareStatement和Statement：

在对数据库只执行一次性存取的时侯，用 Statement 对象进行处理。PreparedStatement 对象的开销比Statement大，对于一次性操作并不会带来额外的好处。

而statement每次执行sql语句，相关数据库都要执行sql语句的编译，preparedstatement是预编译得, preparedstatement支持批处理。

例子：

```
String sqlQuery="SELECT * FROM log";
Statement statement=connection.createStatement();
String sql = "select * from dictionary where receive = ?";
PreparedStatement ps = c.prepareStatement(sql);
```

- 执行语句：ResultSet rs=statement.executeQuery(sqlQuery)：

  executeQuery()方法会把数据库响应的查询结果存放在ResultSet类对象中供我们使用。

  例子：

  ```
  //这是直接将查询结果打印出来了
   while(rs.next())
  {
              String IDrs=rs.getString("ID");
              String passrs=rs.getString("password");
              String statusrs=rs.getString("status");
              System.out.println(IDrs+"    "+passrs+"    "+statusrs);
  }
  ```

  //这是将查询结果存进dictionary

  ```
  while(rs.next()){
      Dictionary d = new Dictionary();
      int id = rs.getInt(1);
      String receive = rs.getString("receive");
      String response = rs.getString("response");
      d.id = id;
      d.receive = receive;
      d.response = response;
      ds.add(d);
  }
  ```

-