The Design of A Serial Communication Program Written In Java

Tianyang Liu [a, *]

Institute of Computer Science and Technology, North China Electricity Power University,

Baoding 071000, China;

[a]1838623164@qq.com

Abstract

This article is based on java language-based serial communication program design and introduces the line of thinking in the process of programming, data reception and processing in detail. With CommPortIdentifier as the core class for serial port access, Implements management of serial ports and processing of received data, after the jar package RXTXcomm.jar was introduced.

Keywords

Java Serial port Data.

1. Introduction

This Serial communication program was written in Java. Java was originally designed for embedded consumer electronics applications. After many years of design optimization and multi-party support, It has evolved into a general-purpose, concurrent, class-based, object-oriented programming language. Java is widely used because of its universality, simplicity, object-oriented, distributed, robustness, security, platform independence and portability, multi-threading, and dynamics. The core function of serial communication is the reception and processing of data, so the programming focus also tends to it. For Java, there is a corresponding serial communication jar package RXTXcomm.jar, which facilitates the handling of the serial port and eases the programming difficulty.

2. Organization of the Text

2.1 The overall design thoughts and processes

The process of writing the program should include: available serial port search, specify the serial port is closed and open, set the serial port parameters, the management of the IO stream pipeline and the processing of the received data and send the processed results through the serial port.

2.2 Some important code and program analysis

2.2.1 Scan the serial port of the machine and open the serial port, set the serial port parameters

```
Public void scanPorts()
// Get all COM ports in this machine
// defines the core class object portId that manages and sets the serial port and controls access
 // Display the scanned serial port in the drop-down list
Public void openSerialPort()
//Open the desired port
// The core class object portId calls the open() function to set the serial port name and timeout
//wait time
// SerialPort serial object serialPort call setSerialPortParams() function to serial port
//Settings of rate, data bit, parity bit, stop bit
```

2.2.2 The setting of the event listener

The listener of the event is divided into two parts, one is to respond to the operation of each component, and the other is to monitor the port using the public void serialEvent (SerialPortEvent event) function.

2.2.3 Receiving, sending and processing of port data

The data type in the SCM is an unsigned character type, which is received in a byte array in a serial port written in Java. Because there is a delay in the transmission process, the serial port is scanned at intervals so that the data can be lost. After obtaining data from the serial port, the data is cached in byte[], and converted to a int value and converted to a String value. The name pwd is used to match the password entry in the MySQL database. When the match is successful, the data is returned and the confirmation frame is 1. When the match is unsuccessful, an error frame is returned. The specific function are as follows:

```
Private void haddle()
/ / Function of data processing
{
        InputStream inputStream = null;
        Try{
        inputStream = serialPort.getInputStream();
        Byte[] cache = new byte[1024];
/ / Define a byte array for buffering received data
        Int availableBytes = 0;
// availableBytes is used to record the number of bytes that have arrived at the serial port but have
//not been read
        While(true){
                availableBytes = inputStream.available();
                While(availableBytes > 0){
/ / If the number of available bytes is greater than zero
                        inputStream.read(cache);
// Store the data in the input stream into the cache data
                        String pwd="";
// Convert the data in the cache array to a string to match the data in the database
                        for(int j = 0;j < cache.length && j < availableBytes; j++){
                                Pwd+=(cache[j]&0xff);
// Convert byte to int and convert it to String. Java always treats a byte as a signed number and
 // composes it with 0xFF to get its unsigned value and convert it to an int value
                        }
                        Byte[] sendBuff = new byte[1];
// Sends to the array in the output stream, sends 1 when the password matches correctly, and
//sends 0 when the password is correct
sendBuff[0] = mydb.lianjie(pwd);
/ / Call the prepared lianjie () function to match the password
sendDataToSeriaPort(sendBuff);
 availableBytes = inputStream.available();
//Update the number of available bytes
```

```
                }
                Thread.sleep (3500);
// Sleep for 3500 milliseconds, allowing data to have enough time to reach and process
        }
        }catch(InterruptedException e){
                e.printStackTrace();
        }catch (IOException e) {
                e.printStackTrace();
    }
}
```

3. Problems and solutions

3.1 Java will be transmitted over byte data as signed data

Java uses complements to store int type which values of 4 bytes. Byte values of only 1 byte. And byte as a signed number in java. So when byte is converted to int type, it usually needs to be shifted or the number of bytes and 0xff Perform the AND operation.

3.2 Received packets when receiving data

When receiving data, it may happen that the received data is not received once, but received in several groups, such as 2 groups, 3 groups or 4 groups. However, we cannot predict the number of groups for a frame of data. One method may be to cache data, that is, a buffer is used to store the data received by the serial port. After the serial port receives data, determine whether the first two bytes of the buffer are header content. If it meets the requirements, the data of this frame is received according to the length of the data, and then the CRC check is performed. If it can be satisfied, the frame is correct. Second, if the data is small and there is no problem of transmission of a large number of characters, the sleep time can be increased so that the received data is processed as a frame of data. This method is less accurate but is more convenient to implement.

4. Conclusion

This article introduces a method of using java to realize serial communication, interprets and analyzes some important codes, and provides solutions to possible problems. The platform-independent nature of Java makes the program run less demanding on the environment, making the program more flexible. Serial communication is widely used in power carrier technologies and has a good application prospect.

References

[1] Zhang Jinbo, Zhang Xuewu, Zheng Xuefeng. Limiting Factors and Solutions of Low Voltage Power Line Communication[J]. Electric Applications, 2004(4):15-17.
[2] Power Line Communication (PLC) Technology and Application [M]. China Electric Power Press, edited by Qi Shuqing, 2005
[3] Ge Yanfeng, Lin Subin, Yu Xiren. Intelligent home control system design based on power line carrier communication technology [J]. Modern Building and Electric, 2011(3):15-18.