



icaste JCommSerial 4.0

For 32 Bit Windows 7, Vista, XP, Server 2003, Server 2008 and Windows 2K

JCommSerial 4.0

Tutorial

Book 1 of 1

Document Release: Final 1.0

Date: July. 2010

Copyright 2008 icaste LLC. All rights reserved.

Produced in United States

The information in this document is subject to change without notice. The statements, configuration, technical data, and recommendations in this document are believed to be accurate and reliable, but are presented without express or implied warranty. Users must take full responsibility for their applications of any products specified in this document. The information in this document is proprietary to icaste LLC.

Icaste, icaste logo, JCommUSB and JCommSerial are trademarks of icaste LLC.

Microsoft, Windows, Windows Driver Framework, Visual C++, and Visual Studio are registered trademarks of the Microsoft Corporation.

Java is a registered trademark of Sun Microsystems Inc.

Revision History

July 2010

Final 1.0. New document for JCommSerial 4.0.

Contents

Revision History	2
July 2010	2
Contents	2
About this Document	3
Subject	3
Applicable Solutions	3
Intended Audience.....	3
Notations	3
Sample Code	3
Sample Code Comments	3
Additional Code Comments	3
Overview	4
Prerequisites.....	4
Developer	4
Tools	4
Dependencies	4
API Tutorial	5
Overview	5
API Model.....	5
Testing the API	6
Using the API.....	7
Creating the SerialPort Object	7
Opening the COM Port	7
Configuring the COM Port	8
Registering an Event Listener (optional)	8
Using the COM Port	8
Cleanup	8
Support.....	8

About this Document

Subject

This document provides a tutorial for the JCommSerial 4.0 API.

Applicable Solutions

JCommSerial 4.0

Intended Audience

This document is intended for software developers well versed in Java application development.

Notations

Sample Code

All code samples will be in Courier 9pt font black.

These are samples of the actual compiled source code

`Example Source Code`

Sample Code Comments

All code sample comments will be in Courier 9pt font green.

These are samples of the actual source code comments.

`Example Sample Code Comments`

Additional Code Comments

All additional comments to sample source code will be Arial 9pt blue italic.

These are additional comments for clarity and are not, and shall not be included in the source code.

Example Additional Comments to Source Code

Overview

Prerequisites

Developer

Understanding of Java and the Java Streams model is required.

Tools

A working install of the Java run time environment is required to run the API.

A working install of the Java software development kit is required to develop applications that leverage the API.

Dependencies

The following 32 bit Windows platforms are supported:

- Windows 7
- Windows Vista
- Windows Server 2008
- Windows XP (SP3)
- Windows Server 2003
- Windows 2000

API Tutorial

Overview

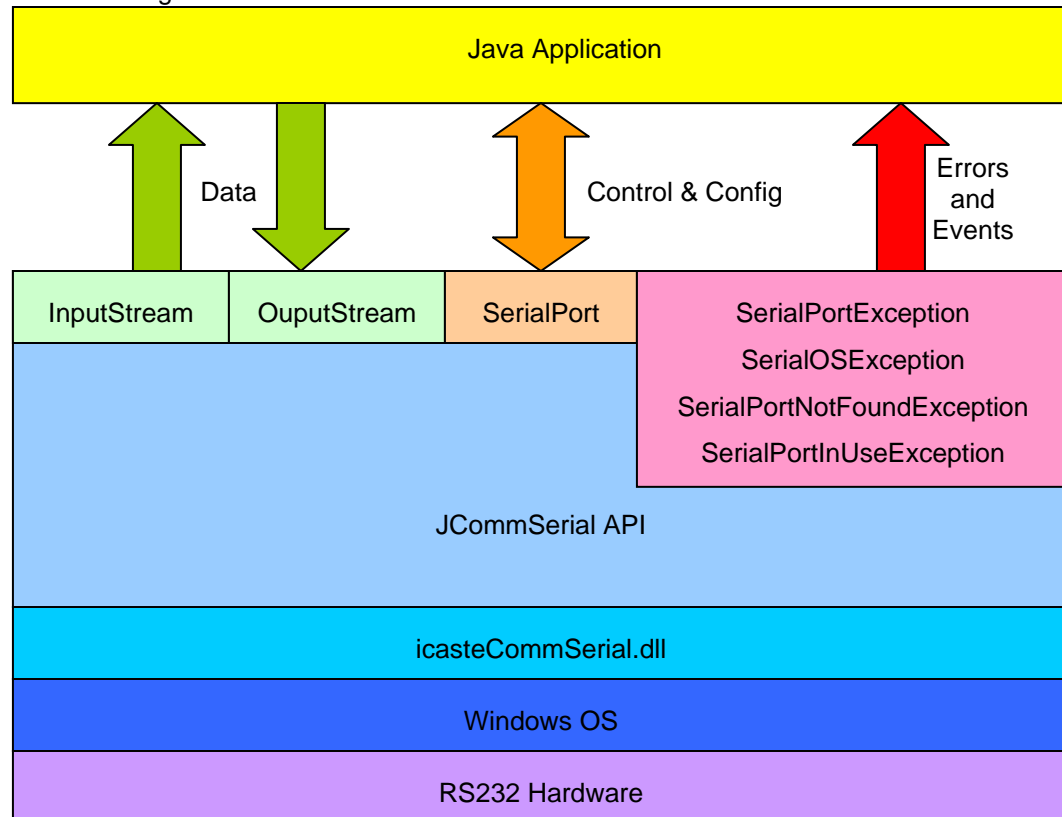
The JCommSerial API provides Java application access to RS232 serial ports on the Windows based OS. The API provides com port setup, configuration and access, including but not limited to:

- Baud Rate
- Data Bits
- Stop Bits
- Parity
- Flow Control (Hardware and Software)
- Read and Write Methods (based on the IO stream model)

The API supports access to physical com ports present on the system, and virtual com ports such as those available to Bluetooth devices.

API Model

The API integrates with the Windows OS as follows:



Testing the API

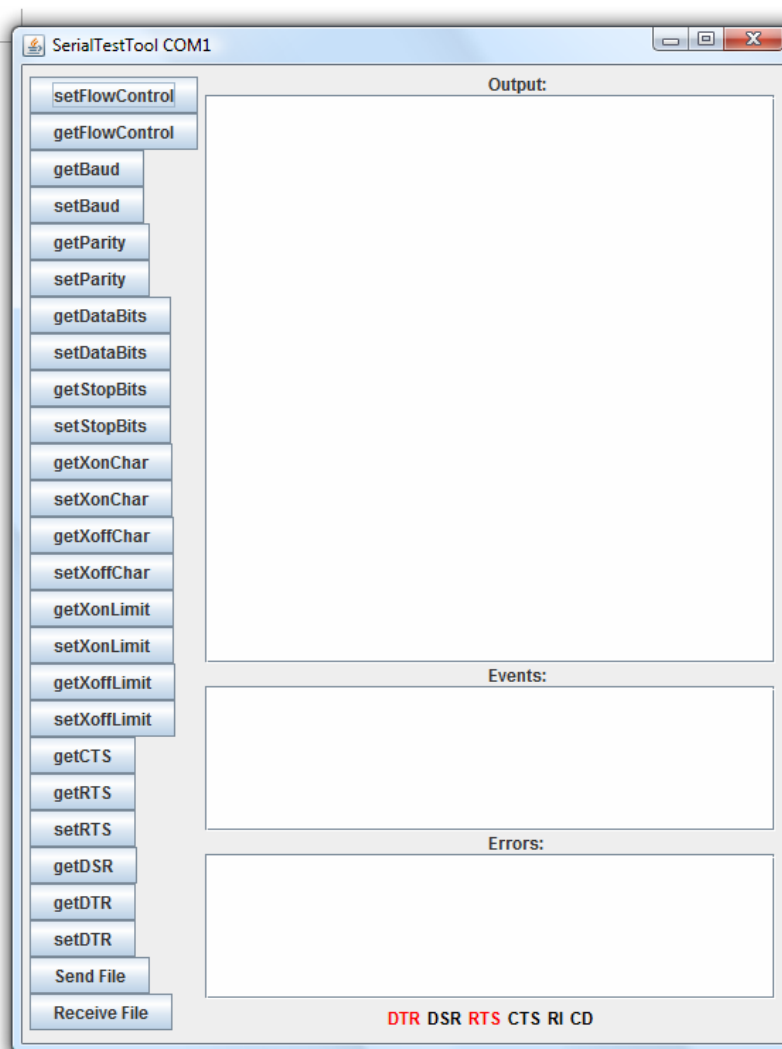
The SerialTestTool class is a GUI application included in the API and provides simple read/write and control operations to COM1.

SerialTestTool can be run directly from

“Programs->JCommSerial_4_0->SerialTestTool.bat”

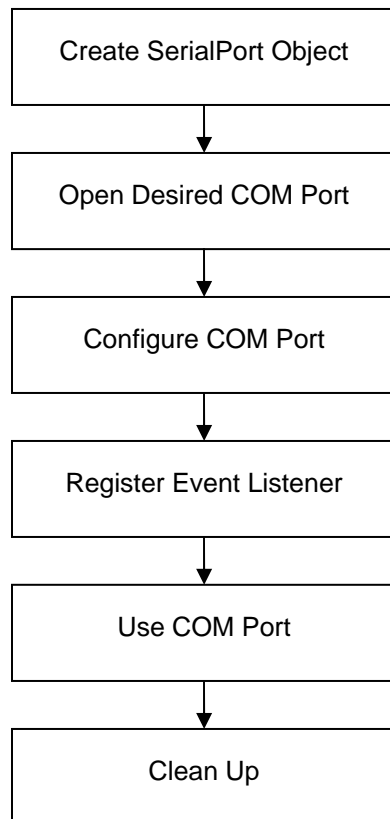
Upon initialization the SerialTestTool will attempt to open COM1 at 9600 baud, no parity, 8 data bits, 1 stop bit. All keyboard input directed to the GUI is written to COM1, all data received from COM1 is written to the “Output” window.

All JCommSerial get/set functions can be tested using the buttons on the left. The source code for the API test utility can serve as a basis for your application, and can be found in “Start->Programs->JCommSerial_4_0”.



Using the API

The API uses the SerialPort object as the direct interface to the desired COM port. The following flow chart outlines the life cycle of the API SerialPort object :



Creating the SerialPort Object

For every port you would like to access you will need instantiate a SerialPort object.

```
SerialPort myPort = new SerialPort();
```

Opening the COM Port

Now we have a SerialPort object we can use it to open the desired COM port. When defining the com port note we use the format "COMX:" where X is the com port number. Here we also define the input/output buffer size used by the serial port driver. Note that this is simply a buffer size request, the driver may choose a more appropriate size.

```
myPort.openPort("COM1:", 2000, 2000);
```

Configuring the COM Port

Now we can configure the port specifics, such as baud rate and flow control. Always check the return value from these requests.

```
myPort.setBaudRate(9600);  
myPort.setDataBits(8);  
myPort.setStopBits(SerialPort.ONESTOPBIT);  
myPort.setParity(SerialPort.NOPARITY);
```

Registering an Event Listener (optional)

If you would like your application to be notified of serial port events, such as a change in CTS signal you must register it as a `SerialPortEventListener` with the `SerialPort` object. The listener object must implement the `SerialPortEventListener` interface.

```
myPort.registerSerialPortEventListener(SerialPortEventListener X);
```

Where X is the name of the listener object.

After registering your listener you must enable the events on which to be notified.

```
myPort.notifyOnCTS(true);
```

Using the COM Port

At this point we are ready to retrieve the `OutputStream` and `InputStream` objects that can be used to write/read data to/from the COM Port.

```
OutputStream osOut = myPort.getOutputStream();  
InputStream isIn = myPort.getInputStream();
```

Now `osOut` and `isIn` can be used to send/receive data from the serial port just as you would use the default Java IO Stream object types.

Cleanup

When you have finished using the port be sure to call `close()` on the `SerialPort` object. This ensures resources related to that port are properly released.

```
myPort.close();
```

Support

If you encounter problems during the API installation process, contact support@icaste.com for best effort support..