```
*********************************** Library usage guide ********************************************

Example 1 : demonstrates how to open, close, read and write to a serial port.
Example 2 : demonstrates how to find serial ports available on system.
Example 3 : demonstrates how to register and use data listener for receiving data from serial port.
Example 4 : demonstrates how to register and use event listener for receiving data from serial port.
Example 5 : demonstrates how to register and use both data and event listeners for serial port.
Example 6 : demonstrates how to turn on/off debugging messages from java layer.


~~~ Example 1 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to open, close, read and write to a serial port.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;
import com.embeddedunveiled.serial.SerialComManager.BAUDRATE;
import com.embeddedunveiled.serial.SerialComManager.DATABITS;
import com.embeddedunveiled.serial.SerialComManager.FLOWCONTROL;
import com.embeddedunveiled.serial.SerialComManager.PARITY;
import com.embeddedunveiled.serial.SerialComManager.STOPBITS;

public class Test1 {
    public static void main(String[] args) {

        long handle = 0;

        // get serial communication manager instance
        SerialComManager sc = new SerialComManager();

        try {
            // try opening serial port for read and write without exclusive ownership
            handle = sc.openComPort("/dev/ttyUSB1", true, true, false);

            // configure data communication related parameters
            sc.configureComPortData(handle, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);

            // configure line control related parameters
            sc.configureComPortControl(handle, FLOWCONTROL.NONE, 'x', 'x', false, false);

            // try to send data out of serial port
            if(sc.writeString(handle, "test string \n", 0) == true) {
                System.out.println("write success \n");
            }

            // try to read data from serial port
            String data = sc.readString(handle);
```

```
            System.out.println("data read is :" + data);

            // close serial port
            sc.closeComPort(handle);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}


~~~ Example 2 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to find serial ports available on system.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;

public class Test2 {
    public static void main(String[] args) {

        SerialComManager sc = new SerialComManager();
        String[] ports = sc.listAvailableComPorts();
        for(String port: ports){
            System.out.println(port + "\n");
        }

    }
}

~~~ Example 3 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to register and use data listener for receiving data from serial port. For
demonstration we have used two serial port one for writing data and another for reading data via
listener.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;
import com.embeddedunveiled.serial.SerialComManager.BAUDRATE;
import com.embeddedunveiled.serial.SerialComManager.DATABITS;
import com.embeddedunveiled.serial.SerialComManager.FLOWCONTROL;
import com.embeddedunveiled.serial.SerialComManager.PARITY;
import com.embeddedunveiled.serial.SerialComManager.STOPBITS;
import com.embeddedunveiled.serial.ISerialComDataListener;
import com.embeddedunveiled.serial.SerialComDataEvent;
```

```java
class Data implements ISerialComDataListener{
    @Override
    public void onNewSerialDataAvailable(SerialComDataEvent data) {
        System.out.println("Read from serial port : " + new String(data.getDataBytes()) + "\n");
    }
}

public class Test3 {
    public static void main(String[] args) {

        long handle = 0;
        SerialComManager sc = new SerialComManager();

        // instantiate class which is will implement ISerialComDataListener interface
        Data dataListener = new Data();

        try {
            // open and configure port which will listen data
            handle = sc.openComPort("/dev/ttyUSB1", true, true, false);
            sc.configureComPortData(handle, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);
            sc.configureComPortControl(handle, FLOWCONTROL.NONE, 'x', 'x', false, false);

            // register data listener for this port
            sc.registerDataListener(handle, dataListener);

            // open and configure port which will listen data
            long handle1 = sc.openComPort("/dev/ttyUSB0", true, true, false);
            sc.configureComPortData(handle1, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);
            sc.configureComPortControl(handle1, FLOWCONTROL.NONE, 'x', 'x', false, false);
            sc.writeString(handle1, "test string", 0);

            // wait for data
            Thread.sleep(200);

            // unregister data listener
            sc.unregisterDataListener(dataListener);

            // close the port releasing handle
            sc.closeComPort(handle);
            sc.closeComPort(handle1);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
~~~ Example 4 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to register and use event listener for receiving event on serial port.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;
import com.embeddedunveiled.serial.SerialComManager.BAUDRATE;
import com.embeddedunveiled.serial.SerialComManager.DATABITS;
import com.embeddedunveiled.serial.SerialComManager.FLOWCONTROL;
import com.embeddedunveiled.serial.SerialComManager.PARITY;
import com.embeddedunveiled.serial.SerialComManager.STOPBITS;
import com.embeddedunveiled.serial.ISerialComEventListener;
import com.embeddedunveiled.serial.SerialComLineEvent;

class Event implements ISerialComEventListener{
    @Override
    public void onNewSerialEvent(SerialComLineEvent lineEvent) {
        System.out.println("Read from serial port : " + lineEvent.getCTS());
    }
}

public class Test1 {
    public static void main(String[] args) {

        long handle = 0;
        SerialComManager sc = new SerialComManager();

        // instantiate class which is will implement ISerialComEventListener interface
        Event eventListener = new Event();

        try {
            handle = sc.openComPort("/dev/ttyUSB0", true, true, false);
            sc.configureComPortData(handle, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);
            sc.configureComPortControl(handle, FLOWCONTROL.HARDWARE, 'x', 'x', false, false);

            // register data listener
            sc.registerLineEventListener(handle, eventListener);

            long handle1 = sc.openComPort("/dev/ttyUSB1", true, true, false);
            sc.configureComPortData(handle1, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);
            sc.configureComPortControl(handle1, FLOWCONTROL.HARDWARE, 'x', 'x', false, false);
            sc.setRTS(handle1, true);

            // wait for data
            Thread.sleep(200);

            // unregister data listener
```

```
            sc.unregisterLineEventListener(eventListener);

            // close the port releasing handle
            sc.closeComPort(handle);
            sc.closeComPort(handle1);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

~~~ Example 5 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to read the status of control lines (flowcontrol) of serial port.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;
import com.embeddedunveiled.serial.SerialComManager.BAUDRATE;
import com.embeddedunveiled.serial.SerialComManager.DATABITS;
import com.embeddedunveiled.serial.SerialComManager.FLOWCONTROL;
import com.embeddedunveiled.serial.SerialComManager.PARITY;
import com.embeddedunveiled.serial.SerialComManager.STOPBITS;

public class Test5 {
    public static void main(String[] args) {

        long handle = 0;
        SerialComManager sc = new SerialComManager();
        try {
            handle = sc.openComPort("/dev/ttyUSB0", true, true, false);
            sc.configureComPortData(handle, DATABITS.DB_8, STOPBITS.SB_1, PARITY.P_NONE, BAUDRATE.B115200, 0);
            sc.configureComPortControl(handle, FLOWCONTROL.HARDWARE, 'x', 'x', false, false);

            // read the status
            int[] state = sc.getLinesStatus(handle);
            System.out.println("CTS state = " + state[0]);
            System.out.println("DSR state = " + state[1]);
            System.out.println("CD state = " +  state[2]);
            System.out.println("RI state = " +  state[3]);

            sc.closeComPort(handle);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
~~~ Example 6 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This demonstrates how to turn on/off debugging messages from java layer.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

package example;

import com.embeddedunveiled.serial.SerialComManager;

public class Test6 {
    public static void main(String[] args) {
        SerialComManager sc = new SerialComManager();
        sc.enableDebugging(true);;
    }
}


############################ Document ends ########################################################################
```