RFID Reader Development Guide For Java

Written by: Tong Wan

V0.6.0.0

Contents

1. Preface

1.1. Overview

We provide the function library that work on Java language for the convenience of add-on development. This library is written by Java and zipped up to be a standard Jar package.

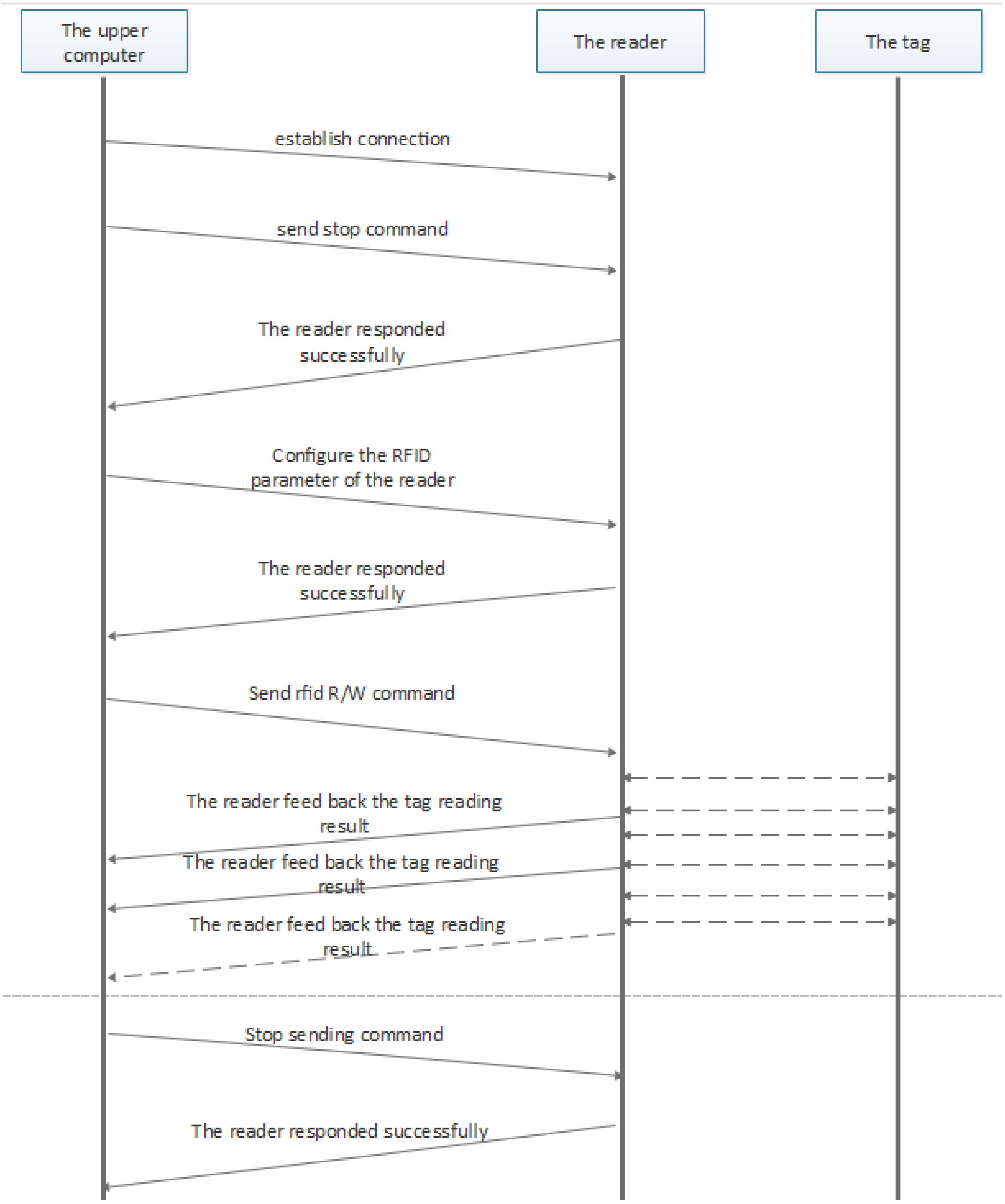
1.2. Applicable Devices

Function Mold	Applicable Devices Type
Reader Configuration and Management	**RF-M6006**
RFID Configuration and Operation	

1.3. Copyright Statements

All contents of this document, including text and pictures, are original and we reserve the right to pursue the legal liability for any unauthorized commercial use. The user should not add, modify or delete any content of this document and transmit the content via internet or CDs, etc. Anyone who violate will take the consequence at his or her own expensive.

1.4. Basic Flow of Read and Write


The upper computer
The reader
The tag
establish connection
send stop command
The reader responded successfully
Configure the RFID parameter of the reader
The reader responded successfully
Send rfid R/W command
The reader feed back the tag reading result
The reader feed back the tag reading result
The reader feed back the tag reading result
Stop sending command
The reader responded successfully

2. Quick Start

```
import com.gg.reader.api.dal.GClient;
import com.gg.reader.api.dal.HandlerTagEpcLog;
import com.gg.reader.api.dal.HandlerTagEpcOver;
import com.gg.reader.api.protocol.gx.*;
```

```
public class App {
    public static void main(String[] args) {
        GClient client = new GClient();
        Scanner sc = new Scanner(System.in);
       // tcp connect, default port 8160
//     if (client.openTcp("192.1.1.168:8160", 2000)) {

        if (client.openSerial("COM11:115200", 2000)) {
            // subscribe Tag Reports Events
            client.onTagEpcLog = new HandlerTagEpcLog() {
                @Override
                public void log(String readName, LogBaseEpcInfo logBaseEpcInfo) {
                    // Any blocking inside the callback will affect the normal use of the API
                    // The number of tag callbacks is large. Please cache the tag data before doing business
                    if (null != logBaseEpcInfo && 0 == logBaseEpcInfo.getResult() ) {
                        System.out.println(logBaseEpcInfo);
                    }
                }
            };
            client.onTagEpcOver = new HandlerTagEpcOver() {
                @Override
                public void log(String s, LogBaseEpcOver logBaseEpcOver) {
                    if (null != logBaseEpcOver) {
                        System.out.println("Epc log over.");
                    }
                }
            };
            // Stop Command, idle state
            MsgBaseStop msgBaseStop = new MsgBaseStop();
            client.sendSynMsg(msgBaseStop);
            if (0 == msgBaseStop.getRtCode()) {
                System.out.println("Stop successful.");
            } else {
                System.out.println("Stop error.");
            }
            // Power configuration, set the power of the 4 antennas to be 30dBm.
            MsgBaseSetPower msgBaseSetPower = new MsgBaseSetPower();
            Hashtable<Integer, Integer> hashtable = new Hashtable<>();
            hashtable.put(1,30);
            hashtable.put(2,30);
            hashtable.put(3,30);
            hashtable.put(4,30);
            msgBaseSetPower.setDicPower(hashtable);

            client.sendSynMsg(msgBaseSetPower);
            if (0 == msgBaseSetPower.getRtCode()) {
                System.out.println("Power configuration successful.");
            } else {
                System.out.println("Power configuration error.");
            }
            //Press any key to start reading
            System.out.println("Enter any character to start reading the tag.");
            sc.nextLine();
            // 4 antennas read. The EPC data area and TID area will be read.
            MsgBaseInventoryEpc msgBaseInventoryEpc = new MsgBaseInventoryEpc();

            msgBaseInventoryEpc.setAntennaEnable(EnumG.AntennaNo_1|EnumG.AntennaNo_2|EnumG.AntennaNo_3|EnumG.AntennaNo_4);

            msgBaseInventoryEpc.setInventoryMode(EnumG.InventoryMode_Inventory);

            ParamEpcReadTid tid = new ParamEpcReadTid();
            tid.setMode(EnumG.ParamTidMode_Auto);
            tid.setLen(6);
            msgBaseInventoryEpc.setReadTid(tid);

              client.sendSynMsg(msgBaseInventoryEpc);
            if (0 == msgBaseInventoryEpc.getRtCode()) {
                System.out.println("Inventory epc successful.");
            } else {
                System.out.println("Inventory epc error.");
            }

            //press any key to stop reading
            sc.nextLine();
            // stop reading, idle state
            client.sendSynMsg(msgBaseStop);
            if (0 == msgBaseStop.getRtCode()) {
                System.out.println("Stop successful.");
            } else {
                System.out.println("Stop error.");
            }
        } else {
            System.out.println("Connect failure.");
        }
}
```

3. Connection Description

3.1. RS232 Connection

Package	com.gg.reader.api.dal
Object	GClient
Method	**public** boolean OpenSerial**(**String readerName**,** int timeout**)**
Description	readerName: connection string, such as"COM1:115200” timeout: connection time is confirmed exceeded limits (ms), like "1000”

3.2. RS485 Connection

3.3. TCP Client Connection

Package	com.gg.reader.api.dal
Object	GClient
Method	**public** boolean openTcp**(**String readerName**,** int timeout**)**
Description	readerName: connection string, such as"192.1.1.168:8160”, default port 8160 timeout: connection time is confirmed exceeded limits (ms), like "1000”
Android Description	**Add permissions according to the Appendix 2 Android environment building tips, and this method shall be used in sub threads.**
Heartbeat Detection for Disconnection	**public** void **setSendHeartBeat**(boolean isSendHeartbeat) Please set up(true) with this method after **Connected** to start Heartbeat Detection for Disconnection. And in the meanwhile, the disconnection messages of TCP will be subscribed.

3.4. TCP Server Monitoring

Package	com.gg.reader.api.dal
Object	GServer
Method	public boolean open(int param)
Description	param: Local Interface monitored by the upper computer UHF device shall be configured to be Client Mode if monitor with this method. Client Mode configuring method is detailed in Operation Manual of RFID Demonstration Software

3.5. Android RS232 Connection

Package	com.gg.reader.api.dal
Object	GClient
Method	public boolean openAndroidSerial(String readerName, int timeout)
Description	readerName: connection string, such as"/dev/ttyS1:115200” timeout: connection time is confirmed exceeded limits (ms), such as"1000”

3.6. Android RS485 Connection

Package	com.gg.reader.api.dal
Object	GClient
Method	public boolean open Android Rs485(String readerName, int timeout)
Description	readerName: connection string, such as"/dev/ttyS3:115200:1” timeout: connection time is confirmed exceeded limits (ms), such as"1000”

4. Events Description

4.1. ISO18000-6CTag Reports Events

Package	com.gg.reader.api.dal
Object	GClient
Events	public HandlerTagEpcLog OnTagEpcLog;

Description

```
client.onTagEpcLog = new HandlerTagEpcLog() {
                @Override
                public void log(String s, LogBaseEpcInfo logBaseEpcInfo) {}};
```

6C tag report events forwardly :when the reader is reading, the tag will report via this event.

Examples are detailed in Quick Start.

LogBaseEpcInfo: detailed in Reporting Object

Reporting Object

Package	com.gg.reader.api.protocol.gx
Object	LogBaseEpcInfo
Attribute	Epc: Hexadecimal EPC character string BEpc: EPC byte array Pc: PC value AntId: Antenna No. Rssi: Signal strength Result: Tag reading result, 0 means success and non-zero value means failure Tid: Hexadecimal TID character string BTid: TID byte array Userdata: Hexadecimal Userdata character string BUser: User data byte array Reserved: Hexadecimal reserved area character string BRes: reserved area byte array
Description	6C tag report parameter forwardly .

4.2. ISO18000-6CTag Reports End Events

Package	com.gg.reader.api.dal
Object	GClient
Attribute	public HandlerTagEpcOver OnTagEpcOver;

Description

```
client.onTagEpcOver = new HandlerTagEpcOver() {
                          @Override
                          public void log(String s, LogBaseEpcOver logBaseEpcOver) {}};
```

6C tag reports the end parameter forwardly to ensure the asynchronous messages are synchronized.

4.3. ISO18000-6BTag Reports Events

Package	com.gg.reader.api.dal
Object	GClient
Events	public HandlerTag6bLog OnTag6bLog;

Description

```
client.onTagEpcLog = new HandlerTag6bLog() {
                          @Override
                          public void log(String s, LogBase6bInfo logBase6bInfo) {}};
```

6B tag reports the events forwardly. :when the reader is reading, the tag will report via this event.

Examples are detailed in Quick Start.

LogBase6bInfo:detailed in Reporting Object

Reporting Object

Package	com.gg.reader.api.protocol.gx
Object	LogBase6bInfo

Attribute	AntId: Antenna No. Rssi: Signal strength Result: Tag reading result, 0 means success and non-zero value means failure Tid: Hexadecimal TID character string BTid: TID byte array Userdata: Hexadecimal Userdata character string BUser: User data byte array
Description	6B tag reports the end parameter forwardly.

4.4. ISO18000-6BTag Reports End Events

Package	com.gg.reader.api.dal
Object	GClient
Attribute	`public HandlerTag6bOver OnTag6bOver;`

Description	`client.onTagEpcOver = new HandlerTag6bOver() {`
	`@Override`
	`public void log(String s, LogBase6bOver logBase6bOver) {}};`
	6b tag reports the end parameter forwardly to ensure the asynchronous messages are synchronized.

4.5. National standard Tag Reports Events

Package	com.gg.reader.api.dal
Category	GClient
Events	`public HandlerTagGbLog onTagGbLog;`

Description

```
client.onTagGbLog = new HandlerTagGbLog() {
            @Override
          public void log(String readerName, LogBaseGbInfo info) {}};
```

Description	GB tag reports the events forwardly. :when the reader is reading, the tag will report via this event. Examples are detailed in Quick Start. LogBaseGbInfo: detailed in Reporting Object

Reporting Object

Namespace	com.gg.reader.api.protocol.gx
Object	LogBaseGbInfo
Attribute	Epc: Hexadecimal EPC character string(tag coding area) BEpc: EPC byte array(tag coding area) Pc: PC value AntId: Antenna No. Rssi: Signal strength Result: Tag reading result, 0 means success and non-zero value means failure Tid: Hexadecimal TID character string(tag information area) BTid: TID byte array(tag information area) Userdata: Hexadecimal Userdata character string BUser: User data byte array
Description	GB Tag reports the parameter forwardly.

4.6. National Standard Tag Reports End Events

Namespace	com.gg.reader.api.dal
Category	GClient
Attribute	`public HandlerTagGbOver OnTagGbOver;`

Description

```
   client.onTagGbOver = new HandlerTagGbOver() {
                    @Override
                    public void log(String readerName, LogBaseGbOver info) {}
                };
```

GB tag reports the end parameters forwardly to ensure the asynchronous messages are synchronized.

4.7. GPI Triggers Report Events

Package	com.gg.reader.api.protocol.gx
Object	LogAppGpiStart
Attribute	`public HandlerGpiStart onGpiStart;`

Description

```
client.onGpiStart=new HandlerGpiStart() {
                    @Override
                    public void log(String readerName, LogAppGpiStart info) {}}
```

The reader will send a message forwardly when the starting condition is met to notify the upper computer that the triggering operation starts.

Reporting Object

Package	com.gg.reader.api.protocol.gx
Object	LogAppGpiStart
Attribute	gpiPort: GPI port No. triggering gpiPortLevel: electrical level of GPI port systemTime: current system time
Description	GPI trigger reporting parameter

4.8. GPI Triggers Report End Events

Package	com.gg.reader.api.protocol.gx
Object	LogAppGpiOver
Attribute	**public** HandlerGpiOver onGpiOver;

Description	`client.onGpiOver=new HandlerGpiOver() {`
	`@Override`
	`public void log(String readerName, LogAppGpiOver info) {}};`
	The reader will send a message forwardly when the stopping condition is met to notify the upper computer that the triggering operation starts.

Reporting Object

Package	com.gg.reader.api.protocol.gx
Object	LogAppGpiOver
Attribute	gpiPort: GPI port No. triggering gpiPortLevel: electrical level of GPI port systemTime: current system time
Description	GPI trigger stopping parameter report

4.9. TCP Connection Disconnect Events

Package	com.gg.reader.api.dal
Category	GClient
Events	**public** HandlerTcpDisconnected onDisconnected**;**

Description

```
Client.onDisconnected = new HandlerTcpDisconnected() {
                    @Override
                    public void log(String readName) {
                    }};
```

Description:

- ➢ The connection is under TCP. When the remote connection is actively disconnected or the physical layer is abnormal, the event will be reported.
- ➢ After the reporting of the event, the upper computer (the caller) need to release connection object, or the event reporting will loops.
- ➢ It is decided by the upper computer (the caller) itself that whether the remote device should be re-connected or not in order to meet different requests.

readerName: the name of the connection object.

4.10. TCP Connect Events

Package	com.gg.reader.api.dal
Category	GServer
Events	**public** HandlerGClientConnected onGClientConnected;

Reporting Object

Package	com.gg.reader.api.dal
Category	GClient
Attribute	N/A
Description	Description: this connection object is the identical to the other actively connection objects and share the same usage.
Heart beat Detection for Disconnection	**public** void **setSendHeartBeat**(boolean _isSendHeartbeat) Please set up(true) with this method after **Monitoring Reported** to start Heartbeat Detection for Disconnection. And in the meanwhile, the messages of disconnection events of TCP will be subscribed.

5. Messages Configuration and Query Description

5.1. Send Synchronous Messages

Namespace	com.gg.reader.api.dal
Object	GClient
Method	public void SendSynMsg(Message msg)
Method1	public void SendSynMsg(Message msg, int timeout)
Method2	public void SendSynMsgRetry(Message msg, int timeout, int retry)
Return value	msg.getRtCode():message return code, o means success, and non-zero value means failure. msg.getRtMsg(): the reason of the failed operation.
Description	**Send Synchronous Messages** is detailed Code Example . Tips: ”Reader Configuration and Management”, “RFID Configuration and Operation”, and other messages are sent with this method.

Code Example 1

```
// Stop Command,idle state
MsgBaseStop msgBaseStop = new MsgBaseStop();
clientConn.SendSynMsg(msgBaseStop);
if (0 == msgBaseStop.getRtCode())
{
    System.out.println("Stop successful.");
}
else { System.out.println("Stop error."); }
```

Code Example 2

```
//   Power configuration, set the power of the 4 antennas to be 30dBm.
MsgBaseSetPower msgBaseSetPower = new MsgBaseSetPower();
Hashtable<Integer, Integer> hashtable = new Hashtable<>();
hashtable.put(1, 30);
hashtable.put(2, 30);
```

```
 hashtable.put(3, 30);
hashtable.put(4, 30);
msgBaseSetPower.setDicPower(hashtable);
clientConn.SendSynMsg(msgBaseSetPower);
if (0 == msgBaseSetPower.getRtCode())
{
    System.out.println("Power configuration successful.");
}
else { System.out.println("Power configuration error."); }
```

Code Example 3

```
if (null != this.clientConn)
{
    // Query the power of the antennas
    MsgBaseGetPower msg = new MsgBaseGetPower();
    clientConn.SendSynMsg(msg);
    if (0 == msg.getRtCode())
    {
       System.out.println(msg);
    }
 }
```

6. Message Description

6.1. Reader Configuration and Management

6.1.1. Restart Reader

Package	com.gg.reader.api.protocol.gx
Object	MsgAppReset
Attribute	N/A
Description	Normally the restart message of the device will be executed after the modification of the configuration that need to come in to effect after restart.

6.1.2. Configure and Query COM Parameter

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgAppSetSerialParam
Query Object	MsgAppGetSerialParam
Attribute	serialBaudrate: Baud rate index (0,9600 bps; 1,19200 bps; 2,115200 bps; 3,230400 bps; 4,460800bps)
Description	(Persistent configuration, information saved when power off)configure the COM parameter of the device.

6.1.3. Configure GPO state Parameter

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgAppSetGpo
Attribute	gpo1: 0 (low, relay disconnected) 1 (high, relay closed) gpo2: 0 (low, relay disconnected) 1 (high, relay closed) gpo3: 0 (low, relay disconnected) 1 (high, relay closed) gpo4: 0 (low, relay disconnected) 1 (high, relay closed) ……
Description	(Persistent configuration, information saved when power off) configure the device GPO parameter. Notes: For GPO that do not require state control, no assignment is needed.

6.1.4. Query for GPI State Parameter

Package	com.gg.reader.api.protocol.gx
Query Object	MsgAppGetGpiState

Attribute	hpGpiState: it is corresponding to the GPI electrical level state(HashMap<Integer, Integer> , key: GPI index, value: electrical level state(0 low,1 high))
Description	Query the GPI status of the device. Notes: The index number starts from 1.

6.1.5. Configure and Query GPI Triggering Parameter

Namespace	com.gg.reader.api.protocol.gx
Configuration Object	MsgAppSetGpiTrigger
Query Object	MsgAppGetGpiTrigger
Attribute	gpiPort: GPI port No. , the index starts from 1 triggerStart: trigger start (0 trigger close ,1 low electrical trigger ,2 high electrical trigger ,3 rising edge trigger ,4 falling edge trigger ,5 random edge trigger) hexTriggerCommand: trigger binding command(Hex, can be null) triggerCommand: trigger binding command(Byte[],can be null) triggerOver: trigger stop(0 non-stop,1 low electrical trigger ,2 high electrical trigger ,3 rising edge trigger ,4 falling edge trigger ,5 random edge trigger ,6 delayed stop) overDelayTime: delayed stop time(take effect only if the stop condition is "delayed stop") levelUploadSwitch: uploading switch of the IO electrical level changes when triggering non-stop(0 do not upload, 1 upload)
Description	(Persistent configuration, information saved when power off) configure the device GPI trigger parameter. Notes: This configuration needs to be modified when the device is idle (that is, the configuration can be changed under loop reading).

6.1.6. Query Version for Software Baseband

Package	com.gg.reader.api.protocol.gx
Query Object	MsgAppGetBaseVersion

Attribute	baseVersions: version of the baseband software
Description	it is for acquiring the version No. of the baseband software

6.1.7. Confirmation for Connection State

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgAppHeartbeat
Attribute	param: connect and confirm the sequential Number of the message
Description	It is used for the confirming the connection state between reader and the upper computer . Both the reader and the upper computer can send messages for Connection State Confirmation. And the one receive the message must reply right away. The connection will be taken as failed if the sender didn't receive the confirmation message.

6.1.8. Query RFID ability of the Reader

Package	com.gg.reader.api.protocol.gx
Query Object	MsgBaseGetCapabilities

Attribute	MaxPower: the maximum power supported MinPower: the minimum power supported AntennaCount: the quantity of the antennas FrequencyArray: the list of supported frequency band , 0， National standard 920~925MHz 1， National standard 840~845MHz 2， National standard 840~845MHz 和 920~925MHz 3， FCC,902~928MHz 4,ETSI,866~868MHz ProtocolArray: List of protocols supported 0， ISO18000-6C/EPC C1G2 1， ISO18000-6B 2， National standard GB/T 29768-2013 3， National military standard GJB 7383.1-2011
Description	N/A

6.1.9. Query Information of the Reader

Package	com.gg.reader.api.protocol.gx
Query Object	MsgAppGetReaderInfo
Attribute	readerSerialNumber: sequential number of the reader powerOnTime: power on time baseCompileTime: compiling time of the baseband appVersions: application software version(such as:”0.1.0.0”) appCompileTime: application compiling time systemVersions: version of the operation system
Description	N/A

6.2. RFID Configuration and Operation

6.2.1. Stop Command

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseStop
Attribute	N/A
Description	Stop all RFID operations of the reader and make the reader idle. Tips: When the reader is reading, all configuration messages will not be able to send, a stop command must be sent.

6.2.2. Configure and Query Power of the Reader

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgBaseSetPower
Query Object	MsgBaseGetPower
Attribute	dicPower : the corresponding antenna power of the reader(Hashtable<Integer, Integer> , key: antenna index No. , value: antenna power value)
Description	(Persistent configuration, information saved when power off)) configure the current power reader Configure according to the given antenna constants EnumG.AntennaNo_1, EnumG.AntennaNo_2.....

6.2.3. Configure and Query Working frequency band of the Reader

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgBaseSetFreqRange
Query Object	MsgBaseGetFreqRange

Attribute	FreqRangeIndex: frequency band index, the specific corresponding relationship is detailed in appendix1.
Description	(Persistent configuration, information saved when power off) for configuration of the current working frequency band of the reader.

6.2.4. Configure and Query Parameter of the EPC Baseband

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgBaseSetBaseband
Query Object	MsgBaseGetBaseband
Attribute	BaseSpeed: EPC baseband speed(Optional). QValue: Default Q value(Optional)(0~15). Session: (Optional)(0,Session0; 1,Session1; 2,Session2; 3,Session3). InventoryFlag: parameter of inventory taking mark(Optional)(0, use only Flag A for inventory taking ;1,use only Flag B for inventory taking;2,use Flag A and Flag B alternately).
Description	(Persistent configuration, information saved when power off) for configuration of the baseband parameter of the reader.

6.2.5. Configure and Query Tag for Uploading Parameter

Package	com.gg.reader.api.protocol.gx
Configuration Object	MsgBaseSetTagLog
Query Object	MsgBaseGetTagLog
Attribute	RepeatedTime: repeated tag filtering time (optional) (means during a reading instruction execution period, the same tag content can only be uploaded once in repeated tag filtering time, 0~65535,the time unit is 10ms). RssiTV: RSSI threshold value (optional) (give up uploading and discard when the RSSI value of the tag is lower than the threshold value).

Description	(Persistent configuration, information saved when power off) configure uploading parameter of the reader

6.2.6. Read EPC Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseInventoryEpc
Attribute	AntennaEnable: antenna port(using antenna enumeration is detailed in Quick Start) InventoryMode: continuous/single Read (0: single read, the reader read one time on each enabling antenna then stop reading and automatically enter idle state; 1: continuous read, the reader keeps reading it stop reading after receiving a stop command. Filter: select reading parameter(Optional)(detailed in Parameter Description) ReadTid: TID read parameter(Optional)(detailed in Parameter Description) ReadUserdata: user data area read parameter(Optional)(detailed in Parameter Description) ReadReserved: reserved area read parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)
Description	Used for configuring the reader parameter for tag reading and start reading. Any reading operation for the tag data require the EPC code of the tag, so a EPC code can be acquired through any reading operation.

6.2.7. Write EPC Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseWriteEpc

Attribute	AntennaEnable: antenna port Area: tag data area to be written (0,reserved area;1,EPC area;2,TID area;3, user data area) Start: word initial address of the tag data area to be written HexWriteData: data content to be written(Optional)(hex) BwriteData: data content to be written Filter: select reading parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)
Description	Reader writes EPC Tags, the writing that this command refined should be single operation.
Writing result	0， wrote successfully 1， antenna port parameter error 2， parameter selecting error 3， parameter writing error 4， CRC verification error 5， Insufficient power 6， Data area overflow 7， Data area locked 8， access password error 9， Other tag errors 10,tag missing 11,reader command sending error

6.2.8. Lock EPC Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseLockEpc
Attribute	AntennaEnable: antenna port Area: tag data area to be locked(0,inactivated password area; 1,access password area; 2,EPC area; 3,TID area ; 4,user data area) Mode: locking operation type(0,unlock; 1,lock; 2,unlock permanently; 3,lock permanently) Filter: select reading parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)
Description	To lock or unlock the tag. The operation this command defined is single.

6.2.9. Inactivate EPC Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseDestoryEpc
Attribute	AntennaEnable: antenna port HexPassword: destroy password Filter: select reading parameter(Optional)(detailed in Parameter Description)
Description	Inactivate the tag. The tag inactivated will be in permanent failure and this operation is irreversible. The operation this command defined is single.

6.2.10. Read 6B Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseInventory6b
Attribute	AntennaEnable: antenna port InventoryMode: Continuous/Single Read (0: single read, the reader read one time on each enabling antenna then stop reading and automatically enter idle state; 1: continuous read, the reader keeps reading it stop reading after receiving a stop command. Area: Reading content (0,read only 6B TID;1,read 6B TID + user data;2,read only user data) ReadUserdata: user data area read parameter(Optional)(detailed in Parameter Description) HexMatchTid: TID code of the 6B tag to be matched(Optional)(hex) BMatchTid: TID code of the 6B tag to be matched(Optional)
Description	Used for the data reading for ISO18000-6B tag.

6.2.11. Write 6B Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseWrite6b

Attribute	AntennaEnable: antenna port HexMatchTid: TID code of the 6B tag to be matched(Optional)(hex) BMatchTid: TID code of the 6B tag to be matched Start: Byte initial address for the data area of the tag to be written HexWriteData: data content to be written(Optional)(hex) BwriteData: data content to be written
Description	Write 6B tag. The writing operation this command defined is single.

6.2.12. Lock 6B Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseLock6b
Attribute	AntennaEnable: antenna port HexMatchTid: TID code of the 6B tag to be matched(Optional)(hex) BMatchTid: TID code of the 6B tag to be matched LockIndex: byte address of the data to be locked
Description	Lock 6B tag. The operation is irrevocable and reversible. The locking operation this command defined is single.

6.2.13. Query for 6B Tag Locking

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseLock6bGet
Attribute	AntennaEnable: antenna port HexMatchTid: TID code of the 6B tag to be matched(Optional)(hex) BMatchTid: TID code of the 6B tag to be matched LockIndex: byte address of the data to be locked and queried.

Description	Query the locking state for the 6B tag. The query operation this command defined is single.

6.2.14. Read GB Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseInventoryGb
Attribute	AntennaEnable: antenna port(using antenna enumeration is detailed in Quick Start) InventoryMode: Continuous/Single Read (0: single read, the reader read one time on each enabling antenna then stop reading and automatically enter idle state; 1: continuous read, the reader keeps reading it stop reading after receiving a stop command. Filter: select reading parameter(Optional)(detailed in Parameter Description) ReadTid: TID read parameter(Optional)(detailed in Parameter Description) ReadUserdata: user data area read parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)
Description	Used for configuring the reader parameter for tag reading and start reading. Any reading operation for the tag data require the tag code , so a EPC code can be acquired with any reading operation.

6.2.15. Write GB Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseWriteGb
Attribute	AntennaEnable: antenna port(using antenna enumeration is detailed in Quick Start) Area: Tag data area to be written (0x10, tag coding area; 0x20, tag secure area; 0x30~0x3F,user subarea 0~15 area) Start: word initial address of the tag data area to be written HexWriteData: data content to be written(Optional)(hex) BwriteData: data content to be written Filter: select reading parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)

Description	➢ The reader writes GB tag. The writing operation this command defined is single. ➢ The GB protocol specifies that the minimum data unit for R/W operations is a word.

6.2.16. Lock GB Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseLockGb
Attribute	AntennaEnable: antenna port(using antenna enumeration is detailed in Quick Start) Area: tag data area to be locked(0x00tag information area, 0x10tag coding area, 0x20tag secure area , 0x30~0x3Fuser subarea 0~15) Mode: locking operation type 0x00,Readable and writable 0x01,Readable but not Writable 0x02,Unreadable but Writable 0x03,Unreadable and unwritable 0x11,Set no identification is required under secure mode ; the operation area must be tag secure area. 0x12,Set identification is required but no secure communications is required under secure mode. The operation area must be tag secure area. 0x13,Set identification and secure communications are required under secure mode. The operation area must be tag secure area. Filter: select reading parameter(Optional)(detailed in Parameter Description) HexPassword: access password (Optional)
Description	To lock or unlock the tag. The operation this command defined is single.

6.2.17. Inactivate GB Tag

Package	com.gg.reader.api.protocol.gx
Object	MsgBaseDestoryGb

Attribute	AntennaEnable: antenna port(using antenna enumeration is detailed in Quick Start) HexPassword: destroy password Filter: select reading parameter(Optional)(detailed in Parameter Description)
Description	Inactivate the tag. The tag inactivated will be in permanent failure and this operation is irreversible. The operation this command defined is single.

7. Parameter Description

7.1. 6C Tag Select Parameter

Package	com.gg.reader.api.protocol.gx
Object	ParamEpcFilter
Attribute	Area: data area to be matched (1,EPC area; 2,TID area; 3, user data area) BitStart: starting bit address of the matching data BitLength: the data bit length to be matched HexData: the data content to be matched (Optional)(hex) BData: the data content to be matched
Description	optional parameter (The first 32 bit of the EPC area is PC value. So usually the starting bit address is 32 when distinguishing EPC.)

7.2. 6C Tag Read TID Parameter

Package	com.gg.reader.api.protocol.gx
Object	ParamEpcReadTid
Attribute	Mode: TID reading mode configuration(0,the TID reading length is self-adaptable, but the maximum length should not be longer than the length defined by byte 1; read TID according to the length defined by byte 1) Len: the reader need to read the word (word, 16bits, similarly hereinafter) length of the TID data
Description	parameter optional

7.3. 6C Tag Read Parameter of User Area

Package	com.gg.reader.api.protocol.gx
Object	ParamEpcReadUserdata
Attribute	Start: starting word address Len: the word length of the user data that need to be read by the reader
Description	parameter optional

7.4. Parameter of User Area for 6B Tag Reading

Package	com.gg.reader.api.protocol.gx
Object	Param6bReadUserdata
Attribute	Start: the starting byte address of the user data Len: the byte length of the user data
Description	parameter optional

7.5. Parameter for GB Tag Reading User Data Area

Package	com.gg.reader.api.protocol.gx
Object	ParamGbReadUserdata
Attribute	ChildArea: user subarea Start: byte starting address for user data Len: the byte length of the user data
Description	parameter optional

8. Appendix 1

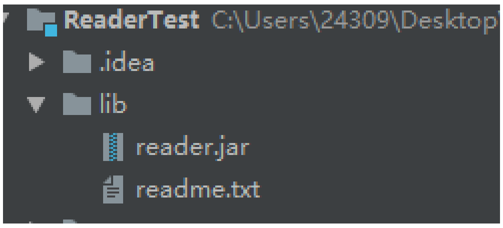
List of the frequency band the Readers Supported

索引	Description
0	National standard 920~925MHz
1	National standard 840~845MHz
2	National standard 840~845MHz 和 920~925MHz
3	FCC,902~928MHz
4	ETSI,866~868MHz

9. Appendix 2

9.1. IDEA(maven) Environment Building

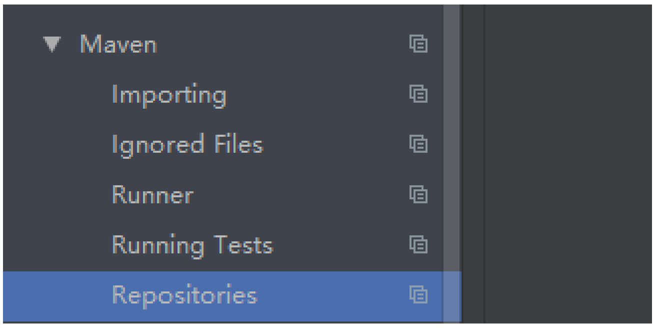
① Create maven project in IDEA, Create a new lib folder under the project directory and copy reader.jar to the directory.



② Open Terminal in IDEA. Execute the installing jar Package command to the local repository in turn.

```
1.  mvn install:install-file -Dfile=lib/reader.jar -DgroupId=com.gg.reader -DartifactId=greader-api -Dversion=1.0
    -Dpackaging=jar
```

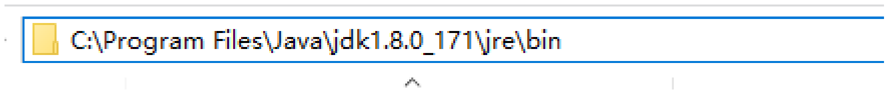
③ Open maven configuration after installation. File-->Settings-->search for maven. Check if the current repository is the local repository needed. Click Repositories under maven, and choose local repository. Then click the update button on the right.



④ Open pom document, and add the following dependencies.

```
<dependency>
   <groupId>com.gg.reader</groupId>
   <artifactId>greader-api</artifactId>
   <version>1.0</version>
</dependency>
```

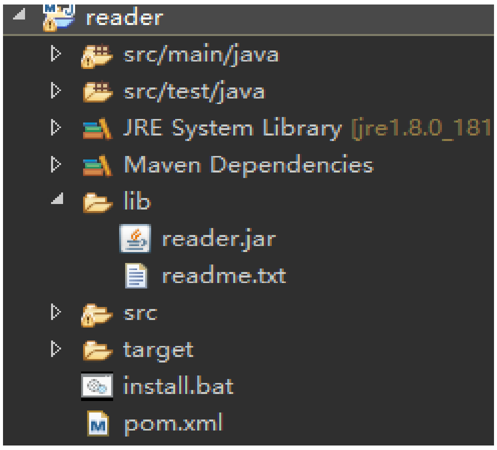
⑤ Add the static library files of the corresponding system to the JRE path where JDK located. JDK --> jre --> bin.



C:\Program Files\Java\jdk1.8.0_171\jre\bin

9.2. Eclipse(maven) Environment Building

① Create Maven project in Eclipse, Create a new lib folder under the project directory and copy reader.jar to the directory.



```
reader
  src/main/java
  src/test/java
  JRE System Library [jre1.8.0_181
  Maven Dependencies
  lib
    reader.jar
    readme.txt
  src
  target
  install.bat
  pom.xml
```

② Click File --> Import --> Maven --> Install or deploy.......(install locally)

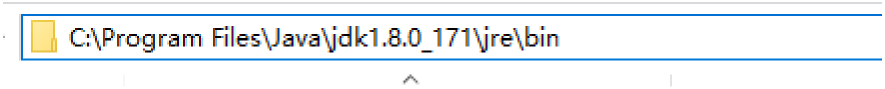
Reader:

1) Artifact file-->the path of jar's location
2) GroupId-->com.gg.reader
3) Artifact Id-->greader-api
4) Version-->1.0

③ Open pom document, and add the following dependencies.

```
<dependency>
   <groupId>com.gg.reader</groupId>
   <artifactId>greader-api</artifactId>
   <version>1.0</version>
</dependency>
```

④ Add the static library files of the corresponding system to the JRE path where JDK located, JDK --> jre --> bin.



C:\Program Files\Java\jdk1.8.0_171\jre\bin

9.3. Non-Maven Environment

① IDEA:

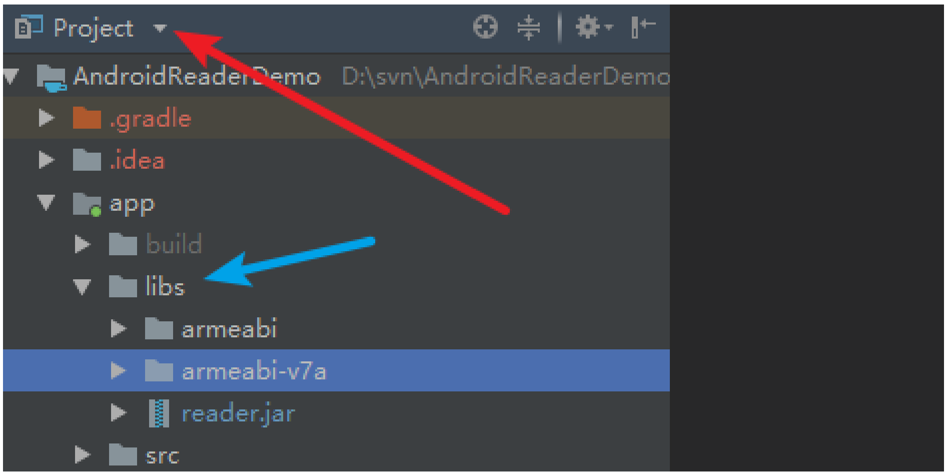
Select jar Package, right click Add as Library.. confirm

② Eclipse:

Select jar Package, right click Build Path-->Add to Build Path.

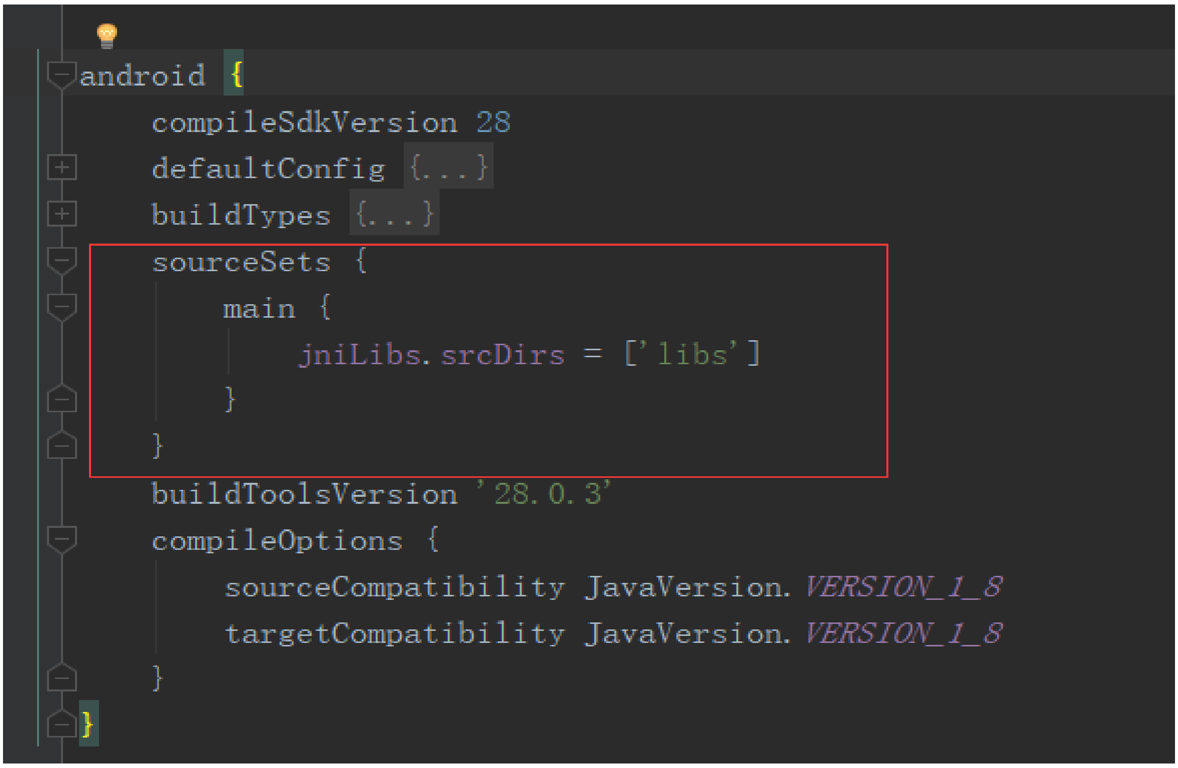
9.4. Android Studio Environment Building

① As the picture shows, copy the three files in lib to the project LIBS folder and click the compile button to load the jar files automatically in the project state.



② Then add the following content to the Android node under the build.gradle of the app directory. And then click compile.

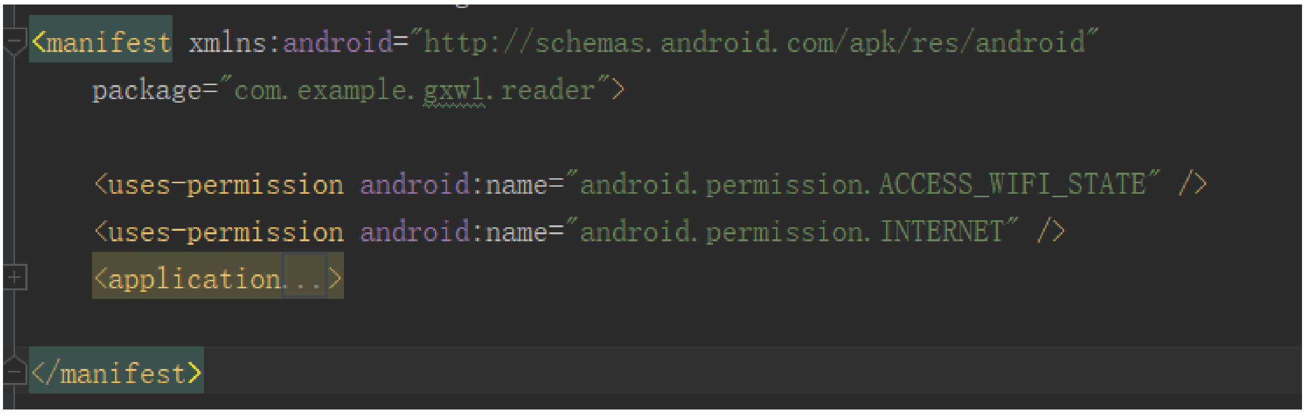
```
sourceSets {
    main {
        jniLibs.srcDirs = ['libs']
    }
}
```



```
android {
    compileSdkVersion 28
    defaultConfig {...}
    buildTypes {...}
    sourceSets {
        main {
            jniLibs.srcDirs = ['libs']
        }
    }
    buildToolsVersion '28.0.3'
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}
```

③ Finally, add the following permissions to the AndroidManifest. XML file (TCP connections require this permission).

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
```



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.gxwl.reader">

    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application...>

</manifest>
```