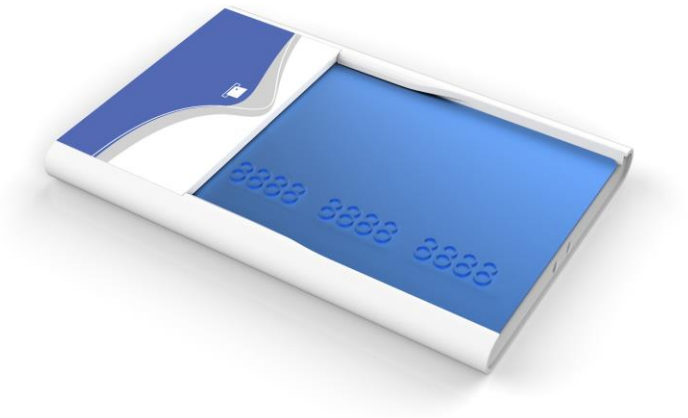
FEITIAN

PART 1 BR500 ANDROID DEVELOPER GUIDE



Revision History:

Date	Revision	Description
September. 2015	V1.0	Release of the first version
December. 2015	V1.1	Add reader control API

Software Developer's Agreement

All Products of Feitian Technologies Co., Ltd. (Feitian) including, but not limited to, evaluation copies, diskettes, CD-ROMs, hardware and documentation, and all future orders, are subject to the terms of this Agreement. If you do not agree with the terms herein, please return the evaluation package to us, postage and insurance prepaid, within seven days of their receipt, and we will reimburse you the cost of the Product, less freight and reasonable handling charges.

1. Allowable Use – You may merge and link the Software with other programs for the sole purpose of protecting those programs in accordance with the usage described in the Developer's Guide. You may make archival copies of the Software.

2. Prohibited Use – The Software or hardware or any other part of the Product may not be copied, reengineered, disassembled, decompiled, revised, enhanced or otherwise modified, except as specifically allowed in item 1. You may not reverse engineer the Software or any part of the product or attempt to discover the Software's source code. You may not use the magnetic or optical media included with the Product for the purposes of transferring or storing data that was not either an original part of the Product, or a Feitian provided enhancement or upgrade to the Product.

3. Warranty – Feitian warrants that the hardware and Software storage media are substantially free from significant defects of workmanship or materials for a time period of twelve (12) months from the date of delivery of the Product to you.

4. Breach of Warranty – In the event of breach of this warranty, Feitian's sole obligation is to replace or repair, at the discretion of Feitian, any Product free of charge. Any replaced Product becomes the property of Feitian.

Warranty claims must be made in writing to Feitian during the warranty period and within fourteen (14) days after the observation of the defect. All warranty claims must be accompanied by evidence of the defect that is deemed satisfactory by Feitian. Any Products that you return to Feitian, or a Feitian authorized distributor, must be sent with freight and insurance prepaid.

EXCEPT AS STATED ABOVE, THERE IS NO OTHER WARRANTY OR REPRESENTATION OF THE PRODUCT, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

5. Limitation of Feitian's Liability – Feitian's entire liability to you or any other party for any cause whatsoever, whether in contract or in tort, including negligence, shall not exceed the price you paid for the unit of the Product that caused the damages or are the subject of, or indirectly related to the cause of action. In no event shall Feitian be liable for any damages caused by your failure to meet your obligations, nor for any loss of data, profit or savings, or any other consequential and incidental damages, even if Feitian has been advised of the possibility of damages, or for any claim by you based on any third-party claim.

6. Termination – This Agreement shall terminate if you fail to comply with the terms herein. Items 2, 3, 4 and 5 shall survive any termination of this Agreement.

Contents

Chapter 1. Overview

This chapter describes how to develop bR500 reader applications, including the development interfaces supported by the product (bR500) and how to develop applications based on these interfaces.

FEITIAN bR500 is specially engineered to accommodate a range of smart card applications. Developers use it as a platform to generate and deploy related products and services. Moreover, FEITIAN bR500 is a terminal unit which is seamlessly integrated to all major systems of operation. Additional features such as the built-in inclusive support for different smart card interfaces has facilitated the wide scale and cross industry adoption of bR500.

bR500 suits customers where security concerns are the most salient and satisfies the demand for a flexible solution for ID authentication, e-commerce, e-payment, information security and access control.

BR500 and the rest of FEITIAN’s line of smart card readers offer each customer a complete solution for all manner of utilizations.

Chapter 2. Definitions

2.1 Error codes

The following is a list of commonly used errors. Since different cards produce different errors they must map over to these error messages.

RETURN_SUCCESS = 0;
RETURN_ERROR = 0xFF01;
ERROR_RECEIVE_LRC = 0xf0;
PROTO_NOT_SUPPORT = -1;
IFD_COMMUNICATION_ERROR = 612;
IFD_NOT_SUPPORTED = 614;

TRANS_RETURN_ERROR = 0xF001;
BUFFER_NOT_ENOUGH = 0xF002;
NOMROEDATA = 0xF003;
TIMEOUT = 0xF004;
CARD_TIMEOUT = 0xF005;

CARD_STATUS = 0xE001;
CARD_PRESENT = 1;
CARD_UNKNOWN = 2;
CARD_ABSENT = 3;
READER_NOT_SUPPORT = 0xF003;

Chapter 3. API Reference

3.1 ft_reader

Synopsis:
public ft_reader (String address, Context c)
Parameters:
N/A
Description:
Constructor Detail.

Example:
Please follow sample code.

Returns:

3.2 isPowerOn

Synopsis:
public boolean isPowerOn()
Parameters:
N/A
Description:
This function use to provide power to card ,before PowerOn() ,please make sure the getCardStatus() retrun CARD_PRESENT(that means bR500 find a card).

Example:
Please follow sample code.

Returns:
RETURN_SUCCESS = 0;
PROTO_NOT_SUPPORT = 1;
IFD_COMMUNICATION_ERROR = 612;
TRANS_RETURN_ERROR = 61441;

3.3 PowerOff

Synopsis:
public int PowerOff()

Parameters:
N/A
Description:
This function use to power off card.

Example:
Please follow sample code.

Returns:
RETURN_SUCCESS = 0;
PROTO_NOT_SUPPORT = 1;
IFD_COMMUNICATION_ERROR = 612;
TRANS_RETURN_ERROR = 61441;

3.4 getCardStatus

Synopsis:
public int getCardStatus()

Parameters:
N/A

Description:
Get card status:
CARD_PRESENT = 0;
CARD_UNKNOW = 1;
CARD_ABSENT = 2;

Example:
Please follow sample code.

Returns:
CARD_PRESENT = 0;
CARD_UNKNOW = 1;
CARD_ABSENT = 2;

3.5 transApdu

Synopsis:

public int transApdu(int tx_length,
byte[] tx_buffer,
int[] rx_length,
byte[] rx_buffer)

Parameters:

tx_length	IN	input data's length
tx_buffer	IN	input data
rx_length	OUT	return data's length
rx_buffer	OUT	return data from card

Description:

This function sends an APDU to the smart card contained in the reader. The card responds from the APDU and stores this response in rx_buffer and it's length in rx_length.

Example:

Please follow sample code.

Returns:

RETURN_SUCCESS = 0;
RETURN_ERROR = 0xFF01;
ERROR_RECEIVE_LRC = 0xf0;
PROTO_NOT_SUPPORT = -1;
IFD_COMMUNICATION_ERROR = 612;
IFD_NOT_SUPPORTED = 614;
TRANS_RETURN_ERROR = 61441;
BUFFER_NOT_ENOUGH = 61442;

Chapter 4. Card reader control API

4.1 getVersion

Synopsis:

public int getVersion (byte[] recvBuf,
int[] recvBufLen)

Parameters:

recvBuf	OUT	return data from reader
recvBufLen	OUT	return data's length

Description:

Get reader Firmware version::

recvBuf[0] The major version

recvBuf[1] The deputy version

Example:

Please follow sample code.

Returns:

4.2 getHardID

Synopsis:

public int getHardID(byte[] recvBuf,
int[] recvBufLen)

Parameters:

recvBuf	OUT	return data from reader
recvBufLen	OUT	return data's length

Description:

Get the serial No from card reader

Example:

Please follow sample code.

Returns:

4.3 getUserID

Synopsis:

public int getUserID(byte[] recvBuf,
int[] recvBufLen)

Parameters:

recvBuf	OUT	return data from reader
recvBufLen	OUT	return data's length

Description:

Get the user's ID from card reader

Example:

Please follow sample code.

Returns:

4.4 genUserID

Synopsis:

public int genUserID(byte[] sendBuf,
int sendLeng)

Parameters:

sendBuf	IN	input data
sendLeng	IN	input data's length

Description:

Let card reader to generate user's ID.

Example:

Please follow sample code.

Returns:

4.5 eraseUserID

Synopsis:

public int eraseUserID(byte[] sendBuf,
int sendLeng)

Parameters:

sendBuf	IN	input data
sendLeng	IN	input data's length

Description:

Let card reader to erase user's ID.

Example:

Please follow sample code.

Returns:

4.6 cmdReadFlash

Synopsis:

public int cmdReadFlash (byte[] recvBuf,
int offset,
int length)

Parameters:

offset	IN	The flash address offset
length	IN	The length to read
recvBuf	OUT	return data from reader

Description:

Read the card reader's flash.
Offset + length should be less than 255.

Example:

Please follow sample code.

Returns:

4.7 cmdWriteFlash

Synopsis:

public int cmdWriteFlash(byte[] `writebuf`,
int offset,
int length)

Parameters:

offset	IN	The flash address offset
length	IN	The length to write
recvBuf	IN	Input data

Description:

Write the card reader's flash.

Offset + length should be less than 255.

Example:

Please follow sample code.

Returns: