USER MANUAL

BT Mag

Android SDK

V.1.63

80125502-001-B

08/16/2013

Revision History

Revision	Date	Description of Changes	By
A	09/20/2012	Initial Release for SDK v1.61	Jenny Wang
B	08/16/2013	Released for SDK v1.63	Candy Han

Platform

Android OS 2.3.3 and above

Library

Add the JAR file (for example, idtech-bluetooth-v1.63.jar) to the Android project.

Right click on the Android project →Build Path→Configure Build Path→Add External JARs…→ (select the appropriate file) →Open→OK

API Summary

IDTechOpenHelper Class

connect(BluetoothDevice)
connect(String)
close()
getCurrentType()
isAvailable()
setOnReceiveListener(OnReceiveListener)
write(byte[])
xor(byte[])
postamble(String)
preamble(String)
review()
reviewSDKVersion()
reviewFirmwareVersion()
reviewKSN()
reviewSecurityLevel()
reviewSerialNumber()
setDecodingMethod(int)
setDefault()
setEncryption(int)
setKeyManagementType(int)
setMagneticTrack(int)
setMagneticTrackSeparator(char)
setMSRReading(int)
setSentinelOnTrack2(int)
terminator(int)
trackPrefix(int, String)
trackSuffix(int, String)
setDefault(byte[])
setEncryptionReview(byte[], boolean)

setKeyManagementTypeReview(byte[], boolean)
setMagneticTrackReview(byte[], boolean)
setMagneticTrackSeparatorReview(byte[], boolean)
setTerminatorReview(byte[], boolean)
setTrackPrefixReview(byte[])
setTrackSuffixReview(byte[])
setDefaultType()
void onCreatedDeviceOption();
void inputMethod(int)
void setTrackPrefixReview();
void setTrackSuffixReview();
void setDecodingMethodReview(byte[], boolean);

OnReceiveListener Class

void onConnected();
void onConnecting();
void onConnectedError(int, String);
void onDisconnected();
void onReceivedData(int, byte[]);
void onReceivedFailed(int);
void onReceivedSuccess(int);

IDTechUtils Class

xor(byte[]);
hasRestrictedCharacter(String)
byteToHexString(byte[]);

API Description

IDTechOpenHelper Class

void connect(BluetoothDevice device)	
Description:	Connect to the Bluetooth device
Parameter:	**device:** the Bluetooth device to be connected
Return:	None
Example:	String addr = "00:11:22:33:AA:BB"; BleutoothDevice device = mBluetoothAdapter.getRemoteDevice(addr); connect(device);

void connect(String address)	
Description:	Connect to the Bluetooth device
Parameter:	**address:** the address of the Bluetooth device **ex:** extended parameter, any value
Return:	None
Example:	connect("00:11:22:33:AA:BB");

void close()	
Description:	Close all connection and stop receiving data
Parameter:	None
Return:	None
Example:	close();

int getCurrentType()	
Description:	Get the type of the last calling function which operates the Bluetooth device. It is used to determine the type of value return from the BT Mag. Please refer the example of onReceivedData(byte[] bs, int type)
Parameter:	None
Return:	Return type can be one of the followings

type	function
TYPE_OF_SUFFIX	trackSuffix(**int**, String)
TYPE_OF_PREFIX	trackPrefix(**int**, String)
TYPE_OF_TRACK_SELECTION	setMagneticTrack(**int**)
TYPE_OF_TRACK_SEPARATOR	setMagneticTrackSeparator(**char**)
TYPE_OF_MSR_READING_DISABLE	setMSRReading(**int**)
TYPE_OF_MSR_READING_ENABLE	setMSRReading(**int**)
TYPE_OF_DECODING	setDecodingMethod(**int**)
TYPE_OF_TERMINATOR	terminator(**int**)
TYPE_OF_KEY_MANAGEMENT	setKeyManagementType(**int**)
TYPE_OF_TRACK2	setSentinelOnTrack2(**int**)
TYPE_OF_ENCRYPTION	setEncryption(**int**)

	TYPE_OF_DEFAULT	setDefault()
	TYPE_OF_POSTAMBLE	postamble(String)
	TYPE_OF_PREAMBLE	preamble(String)
	TYPE_OF_WAITING_SWIPE	When the swipe data is returned
Example:	int type = getCurrentType();	

boolean isAvailable()	
Description:	Check to see if the Bluetooth device is connected
Parameter:	None
Return:	True: Bluetooth device is connected, false: otherwise.
Example:	boolean isConnected = isAvailable();

void setOnReceiveListener(OnReceiveListener receiveListener)	
Description:	Register the listener to receive message
Parameter:	receiveListener: Object of OnReceiveListener class
Return:	None
Example:	setOnReceiveListener(receiveListener);

boolean write(byte[] bs)	
Description:	Send control commands
Parameter:	bs: commands
Return:	True: sent successful, false: otherwise
Example:	byte[] bs = new byte[] {0x02,0x53,0x1A,0x01,0x31,0x03,0x78}; boolean isSent = write(bs);

byte xor(byte[] bs)	
Description:	Take XOR operation of the byte array
Parameter:	bs: byte array for XOR operation
Return:	If bs is null, result would be all zeros.
Example:	byte[] bs = new byte[] {0x02,0x53,0x1A,0x01,0x31,0x03}; byte b = xor(bs);//result would be 0x78

void postamble(String postamble)	
Description:	To set postamble. The postamble serves the same purpose as the preamble, except it is added to the end of the data string, after any terminator characters
Parameter:	postamble: postamble to be added to the data
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener.

	OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType will be "TYPE_OF_POSTAMBLE"
Example:	postamble("IDTECH");

void preamble(String preamble)	
Description:	To set preamble. Characters can be added to the beginning of a string of data These can be special characters for identifying a specific reading station, to format a message header expected by the receiving host, or any other character string. Up to fifteen ASCII characters can be defined)
Parameter:	preamble: preamble to be added to the data
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType type will be TYPE_OF_PREAMBLE
Example:	preamble("IDTECH");

void review()	
Description:	To review current settings command
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the current setting. The funcType TYPE_OF_REVIEW will be returned
Example:	review ();

String reviewSDKVersion()	
Description:	To get the SDK'version
Parameter:	None
Return:	The string for SDK's version
Example:	str = reviewSDKVersion();

void reviewFirmwareVersion()	
Description:	This command is to get device firmware version
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the firmware version. The funcType TYPE_OF_REVIEW_FIRMWARE_VERSION will be returned. The returned

	firmware version format is <06> <02> <Firmware version string><03> <LRC>, where the LRC is an exclusive or of the data from <02> to <03>
Example:	reviewFirmwareVersion();

void reviewKSN()	
Description:	This command is to get DUKPT key serial number and counter
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the KSN. The funType TYPE_OF_REVIEW_KSN will be returned. The returned KSN is with format <06><02><51><KSN Legnth+1><KSN length><KSN><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>.
Example:	reviewKSN();

void reviewSecurityLevel()	
Description:	This command is to get the current security level
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the security level. The funcType TYPE_OF_REVIEW_SECURITY_LEVEL will be returned. The returned security level is with format <06><02><7E>< 01> <XX> <03> <LRC> Where, XX is the security level as below: 0x31 security level 1; No key injection 0x32 security level 2; Key injected but not activated; 0x33 security level 3; Key injected and activated; LRC is an exclusive or of the data from <02> to <03>
Example:	reviewSecurityLevel();

void reviewSerialNumber()	
Description:	This command is to get device serial number
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the serial number. Returned funcType is TYPE_OF_REVIEW_SERIAL_NUMBER Returned serial number format is <06><02><4E><SN length+1><SN length><SN ><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>

Example:	reviewSerialNumber();

void setDecodingMethod(int type)	
Description:	The BT Mag can support four kinds of decoding directions
Parameter:	**type**: **DECODING_BOTH_DIRECTIONS** --Decoding in Both Directions. If the encryption feature is enabled, the key management method used is DUKPT. **DECODING_HEAD_AGAINST_DIRECTIO**N --Moving stripe along head against direction of encoding. If the encryption feature is enabled, the key management method used is DUKPT. **DECODING_HEAD_IN_DIRECTION** --Moving stripe along head in direction of encoding. If the encryption feature is enabled, the key management method used is DUKPT. **DECODING_RAW_DATA_IN_IDTECH_MODE** --Raw Data Decoding in Both Directions, send out in ID TECH mode. **DECODING_RAW_DATA_IN_OTHER_MODE** --Raw Data Decoding in Both Directions, send out in other mode. If the encryption feature is enabled, the key management method used is fixed key.
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be TYPE_OF_DECODING
Example:	setDecodingMethod(IDTechReader.DECODING_BOTH_DIRECTIONS);

void setDefault()	
Description:	To set BTMag to default setting.
Parameter:	None
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be TYPE_OF_DEFAULT
Example:	setDefault();

void setEncryption(int type)	
Description:	Set Encryption Method.
Parameter:	**type**: **ENCRYPTION_AES** --Enable AES Encryption (Not for Raw Data Decoding in

	Both Directions, send out in other mode.) **ENCRYPTION_DISABLE** –Encryption method not defined **ENCRYPTION_TDES** --Enable TDES Encryption
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be TYPE_OF_ENCRYPTION
Example:	setEncryption(IDTechReader.ENCRYPTION_AES);

void setKeyManagementType(int type)	
Description:	Select key management type
Parameter:	**type**: **KEY_MANAGEMENT_DUKPT** **KEY_MANAGEMENT_FIXED**
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be TYPE_OF_KEY_MANAGEMENT
Example:	setKeyManagementType(IDTechReader.KEY_MANAGEMENT_DUKPT);

void setMagneticTrack(int type)	
Description:	There are up to three tracks of encoded data on a magnetic stripe. This option selects the tracks that will be read and decoded.
Parameter:	**type**: **TRACK_ANY** **TRACK_1_ONLY** **TRACK_2_ONLY** **TRACK_1_AND_2** **TRACK_3_ONLY** **TRACK_1_AND_3** **TRACK_2_AND_3** **TRACK_ALL_3** **TRACK_ANY_1_AND_2** **TRACK_ANY_2_AND_3**
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType will be TYPE_OF_TRACK_SELECTION.
Example:	setMagneticTrack(IDTechReader.TRACK_ALL_3);

void setMagneticTrackSeparator(char c)	
Description:	This option allows the user to select the character to be used to separate data decoded by a multiple-track reader. Only one ASCII Character. The default value is CR(0Dh), 0h means no track separator.
Parameter:	**c**： **TRACK_SEPARATOR_CR** or **TRACK_SEPARATOR_NONE** or any ASCII character
Return:	None The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be `TYPE_OF_TRACK_SEPARATOR`.
Example:	setMegneticTrackSeparator(IDTechReader.TRACK_SEPARATOR_CR); or setMegneticTrackSeparator('$');//any ASCII character， for example '$'

void setMSRReading(int type)	
Description:	Enable or Disable the SecureHead. If the reader is disabled, no data will be sent out to the host.
Parameter:	**type**：**MSR_READING_DISABLE** or **MSR_READING_ENABLE**
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType will be `TYPE_OF_MSR_READING_DISABLE` or `TYPE_OF_MSR_READING_ENABLE`
Example:	setMSRReading(IDTechReader.MSR_READING_ENABLE);

void setSentinelOnTrack2(int type)	
Description:	The BTMag can be set to either send, or not send, the Start/End sentinel, and to send either the Track 2 account number only, or all the encoded data on Track 2. (The Track 2 account number setting doesn't affect the output of Track 1 and Track 3.)
Parameter:	**type**： **TRACK2_NO_SENTINEL_SEND_DATA** --Don't send start/end sentinel and send all data on Track 2 **TRACK2_HAS_SENTINEL_SEND_DATA** --Send start/end sentinel and send all data on Track 2 **TRACK2_NO_SENTINEL_SEND_ACCOUNT_23H** --Don't send start/end sentinel and send account # on Track 2 **TRACK2_HAS_SENTINEL_SEND_ACCOUNT_NUMBER**--Send start/end sentinel and send account number on Track 2
Return:	None

	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The funcType returned will be TYPE_OF_TRACK2.
Example:	setSentinelOnTrack2(IDTechReader.TRACK2_HAS_SENTINEL_SEND_DATA);

void terminator(int c)	
Description:	To set the terminator. Terminator characters are used to end a string of data in some applications
Parameter:	**C**: **TERMINATOR_CR** or **TERMINATOR_NONE** or any ASCII character
Return:	None
Note	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType will be TYPE_OF_TERMINATOR
Example:	terminator(IDTechReader.TERMINATOR_CR); or terminator('#');

void trackPrefix(int track, String prefix)	
Description:	Characters can be added to the beginning of a track data. These can be special characters to identify the specific track to the receiving host, or any other character string. Up to six ASCII characters can be defined.
Parameter:	**track**: **PREFIX_TRACK1** or **PREFIX_TRACK2** or **PREFIX_TRACK3** **prefix**: maximum 6 characters
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType will be TYPE_OF_PREFIX.
Example:	trackPrefix(IDTechReader.PREFIX_TRACK1, "$$$");

void trackSuffix(int track, String suffix)	
Description:	Characters can be added to the end of track data. These can be special characters to identify the specific track to the receiving host, or any other character string. Up to six ASCII characters can be defined.
Parameter:	**track: SUFFIX_TRACK1** or **SUFFIX_TRACK2 or SUFFIX_TRACK3** **suffix**: maximum 6 characters
Return:	None
Note	The handle result will be returned by interface OnReceiveListener.

	OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The returned funcType will be TYPE_OF_SUFFIX
Example:	trackSuffix(IDTechReader.SUFFIX_TRACK1, "@@@");

void setDefault(byte[] cmd)	
Description:	Set Bluetooth device to default setting. The commands are sent out in sequence, if one command fails, it will be resent.
Parameter:	**cmd**: IDTechReader.CHANGE_TO_DEFAULT_CMD1 IDTechReader.CHANGE_TO_DEFAULT_CMD2 IDTechReader.CHANGE_TO_DEFAULT_CMD3 IDTechReader.CHANGE_TO_DEFAULT_CMD4 IDTechReader.CHANGE_TO_DEFAULT_CMD5
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails. The results of funcType will be TYPE_OF_DEFAULT
Example:	setDefault(IDTechReader.CHANGE_TO_DEFAULT_CMD1);

void setEncryptionReview(byte[] cmd, boolean isControl)	
Description:	To get current encryption type.
Parameter:	**cmd**: IDTechReader.ENCRYPTION_REVIEW_CMD **isControl** true: get setting value and set control option (with a dialog box popped up)， false: get setting value only
Return:	None
Note:	OnReceiveListener OnReceivedData will return the current encryption method. FuncType TYPE_OF_ENCRYPTION will be returned. The encryption method will be returned with format<06><02><4C><01><XX><02><LRC> Encryption method XX is as below: 30: encryption method not defined 31: TDES 32: AES LRC is an exclusive or of the data from <02> to <03>
Example:	setEncryptionReview(IDTechReader.ENCRYPTION_REVIEW_CMD, true);

void setKeyManagementTypeReview(byte[] cmd, boolean isControl)	
Description:	Review current key management type.
Parameter:	**cmd**: IDTechReader.KEY_MANAGEMENT_REVIEW_CMD

	isControl true : get setting value and set control option (with a dialog box popped up)， false: get setting value only
Return:	None
Note	Interface OnReceiveListener will return the settings. OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the funcType *TYPE_OF_KEY_MANAGEMENT*. The return key management type will be with format: <06><02><52><58><01><XX><03><LRC> Where, XX is the key type as below: 30 Fix key management 31 DUKPT Key management LRC is an exclusive or of the data from <02> to <03>
Example:	setKeyManagementTypeReview(IDTechReader.KEY_MANAGEMENT_REVIEW_CMD, true);

void setMagnetTrackReview(byte[] cmd, boolean isControl)	
Description:	Review current track selection
Parameter:	**cmd**: IDTechReader.SELECT_MAGNETIC_TRACK_REVIEW_CMD **isControl** true: get setting value and set control option (with a dialog box popped up)， false: get setting value only
Return:	None

Note:	Interface OnReceiveListener will return the settings. OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return the track review. Fucntion type is TYPE_OF_TRACK_SELECTION. The returned track selection is with format <06><02><13>< 01><XX>< 03><LRC>
	Where, the XX is as below:
	`int TRACK_ANY = 0x30;`
	`int TRACK_1_ONLY = 0x31;`
	`int TRACK_2_ONLY = 0x32;`
	`int TRACK_1_AND_2 = 0x33;`
	`int TRACK_3_ONLY = 0x34;`
	`int TRACK_1_AND_3 = 0x35;`
	`int TRACK_2_AND_3 = 0x36;`
	`int TRACK_ALL_3 = 0x37;`
	`int TRACK_ANY_1_AND_2 = 0x38;`
	`int TRACK_ANY_2_AND_3 = 0x39;`
	LRC is an Exclusive or of the data from <02> to <03>

Example:	setMagnetTrackReview(IDTechReader.SELECT_MAGNETIC_TRACK_REVIEW_CMD, true);

void setMagnetTrackSeparatorReview(byte[] cmd, boolean isControl)	
Description:	To review current track separator
Parameter:	**cmd:** IDTechReader.TRACK_SEPARATOR_REVIEW_CMD **isControl** true: get setting value and set control option (with a dialog box popped up)，false: get setting value only
Return:	None
Note:	OnReceiveListener will call OnReceivedData to return the result. The returned funcType will be TYPE_OF_TRACK_SEPARATOR. The returned data will be with format <06><02><17><01><TrackSeparator><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>.
Example:	setMagnetTrackSeparatorReview(IDTechReader.TRACK_SEPARATOR_REVIEW_CMD, true);

void setTerminatorReview(byte[] cmd, boolean isControl)	
Description:	Review current terminator.
Parameter:	**cmd:** IDTechReader.TERMINATOR_REVIEW_CMD **isControl** true: get setting value and set control option (with a dialog box popped up)，false: get setting value only
Return:	None
Note:	The returned fucnType will be TYPE_OF_TERMINATOR. The returned track separator will be with format <06><02><21><01><Terminator ><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>.
Example:	setTerminatorReview(IDTechReader. TERMINATOR_REVIEW_CMD, true);

void setDefaultType()	
Description:	Set default type to IDTechReader.TYPE_OF_WAITING_SWAPPING
Parameter:	None
Return:	None
Note:	The handle result will be returned by interface OnReceiveListener. OnReceiveListener will call onReceivedSuccess(int funcType) if setting succeeds and onReceivedFailed(int funcType) if setting fails.
Example:	setDefaultType();

void onCreatedDeviceOption()	
Description:	Create dialog box to select Bluetooth Device connection
Parameter:	None
Return:	None

Example:	onCreatedDeviceOption();

void inputMethod(int type)	
Description:	Display input dialog for the type selected
Parameter:	**type**： INPUT_METHOD_PREAMBLE INPUT_METHOD_POSTAMBLE INPUT_METHOD_PREFIX INPUT_METHOD_SUFFIX INPUT_METHOD_SEPARATOR INPUT_METHOD_TERMINATOR
Return:	None
Example:	inputMethod(IDTechOpenHelper.INPUT_METHOD_PREFIX);

void setTrackPrefixReview()	
Description:	Review current prefix
Parameter:	None
Return:	None
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return current prefix. The return data format is <02><52><34><Prefix><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>.
Example:	setTrackPrefixReview();

void setTrackSuffixReview()	
Description:	Review current track suffix review
Parameter:	None
Return:	setDecodingMethodReviewNone
Note:	OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return current prefix. The return data format is <02><52><34><Prefix><03><LRC>, where the LRC is an exclusive or of the data from <02> to <03>.
Example:	setTrackSuffixReview();

void setDecodingMethodReview(byte[] cmd, boolean isControl)	
Description:	Get decoding Method
Parameter:	**cmd**: IDTechReader.DECODING_METHOD_REVIEW_CMD **isControl**：true: get setting value and set control option (with a dialog box popped up)，false: get setting value only
Return:	None
Note:	Interface OnReceiveListener will return the settings. OnReceiveListener will call onReceivedData(int funcType, byte[] reviewDatas) to return current

	decoding method. The funcType returned will be TYPE_OF_DECODING. The returned data is with format <06><02><52><1D><01><XX><03><LRC> Where, XX is as below **31 Decoding in Both Directions. If the encryption feature is enabled, the key management method used is DUKPT.** 32 Moving stripe along head in direction of encoding. If the encryption feature is enabled, the key management method used is DUKPT. 33 Moving stripe along head against direction of encoding. If the encryption feature is enabled, the key management method used is DUKPT. 34 Raw Data Decoding in Both Directions, send out in other mode. If the encryption feature is enabled, the key management method used is fixed key. LRC is an exclusive or of the data from <02> to <03>
Example:	`setDecodingMethodReview(IDTechReader.DECODING_METHOD_REVIEW_CMD, true);`

OnReceiveListener class:

void onConnectErrorInfo(int errorType, String errorMsg**)**	
Description:	Bluetooth device connection error information
Parameter:	**errorMsg** **errorType** **ERROR_UNSUPPORTED** **ERROR_EXCEPTION** **ERROR_INVALID_ADDRESS** **ERROR_NO_BLUETOOTH_DEVICE**
Return:	None

Example:	`public void onConnectErrorInfo(String msg, int reasion) {`
	`  switch (reasion) {`
	`  case OnReceiveListener.ERROR_EXCEPTION :`
	`  break;`
	`  case OnReceiveListener.ERROR_INVALID_ADDRESS :`
	`  break;`
	`  case OnReceiveListener.ERROR_NO_BLUETOOTH_DEVICE :`
	`  break;`
	`  case OnReceiveListener.ERROR_UNSUPPORTED :`
	`  break;`
	`  default : break;`
	`  }`
	`  Toast.makeText(this, msg, 0).show();`
	`}`

void onReceivedData(int funcType, byte[] reviewDatas)

Description:	Receive data from the Bluetooth device
Parameter:	funcType： IDTechReader.TYPE_OF_DECODINGIDTechReader.TYPE_OF_ENCRYPTIONIDTechReader.TYPE_OF_KEY_MANAGEMENTIDTechReader.TYPE_OF_POSTAMBLEIDTechReader.TYPE_OF_PREAMBLEIDTechReader.TYPE_OF_PREFIX IDTechReader.TYPE_OF_REVIEWIDTechReader.TYPE_OF_REVIEW_FIRMWARE_VERSION IDTechReader.TYPE_OF_REVIEW_KSN IDTechReader.TYPE_OF_REVIEW_SECURITY_LEVEL IDTechReader.TYPE_OF_REVIEW_SERIAL_NUMBER IDTechReader.TYPE_OF_SUFFIXIDTechReader.TYPE_OF_TERMINATOR IDTechReader.TYPE_OF_TRACK_SELECTION IDTechReader.TYPE_OF_TRACK_SEPARATOR IDTechReader.TYPE_OF_WAITING_SWIPEreviewDatas
Return:	None

Example:	**public void** onReceivedData(int funcType, byte[] reviewDatas)(**byte**[] bs, **int** type) {
	switch (mIDTechHelper.getCurrentType()) {
	case IDTechReader.TYPE_OF_DECODING :
	case IDTechReader.TYPE_OF_ENCRYPTION :
	case IDTechReader.TYPE_OF_KEY_MANAGEMENT :
	case IDTechReader.TYPE_OF_POSTAMBLE :
	case IDTechReader.TYPE_OF_PREAMBLE :
	case IDTechReader.TYPE_OF_PREFIX :
	case IDTechReader.TYPE_OF_REVIEW :
	case IDTechReader.TYPE_OF_REVIEW_FIRMWARE_VERSION :
	case IDTechReader.TYPE_OF_REVIEW_KSN :
	case IDTechReader.TYPE_OF_REVIEW_SECURITY_LEVEL :
	case IDTechReader.TYPE_OF_REVIEW_SERIAL_NUMBER :
	case IDTechReader.TYPE_OF_SUFFIX :
	case IDTechReader.TYPE_OF_TERMINATOR :
	case IDTechReader.TYPE_OF_TRACK_SELECTION :
	case IDTechReader.TYPE_OF_TRACK_SEPARATOR :
	case IDTechReader.TYPE_OF_WAITING_SWIPE :
	default : **break**;
	}
	//TODO handling data
	}

void onConnected()

Description:	Listener to be calledwhen Bluetooth connection is successful
Parameter:	None
Return:	None

Example:	`public void onConnected() {`
	`  //TODO update connection status`
	`  Toast.makeText(this, "Connect Success", 0).show();`
	`}`

void onConnecting()	
Description:	Listener to be called while the BTMag is connecting
Parameter:	None
Return:	None

Example:	`public void onConnecting() {`
	`  //TODO update connection status`
	`  Toast.makeText(this, "Connecting.., please wait……",`
	`  0).show();`
	`}`

void onDisconnected()	
Description:	Listener to be called when the BTMag is disconnected
Parameter:	None
Return:	None

Example:	`public void onDisconnected() {`
	`  //TODO update connection status`
	`  Toast.makeText(this, "BlueTooth device is unconnected。`
	`  ", 0).show();`
	`}`

void onReceivedFailed(int funcType)	
Description:	Function failed
Parameter:	funcType: IDTechReader.TYPE_OF_DECODING IDTechReader.TYPE_OF_ENCRYPTION IDTechReader.TYPE_OF_KEY_MANAGEMENT IDTechReader.TYPE_OF_POSTAMBLE IDTechReader.TYPE_OF_PREAMBLE IDTechReader.TYPE_OF_PREFIX IDTechReader.TYPE_OF_SUFFIX IDTechReader.TYPE_OF_TERMINATOR IDTechReader.TYPE_OF_TRACK_SELECTION IDTechReader.TYPE_OF_TRACK_SEPARATOR IDTechReader.TYPE_OF_DEFAULT

	IDTechReader.TYPE_OF_MSR_READING_DISABLE IDTechReader.TYPE_OF_MSR_READING_ENABLE
Return:	None

Example:	**public void** onReceivedFailed(int funcType) {
	    //TODO update connection status, or others
	}

void onReceivedSuccess(int funcType)	
Description:	Function succeed
Parameter:	funcType: IDTechReader.TYPE_OF_DECODING IDTechReader.TYPE_OF_ENCRYPTION IDTechReader.TYPE_OF_KEY_MANAGEMENT IDTechReader.TYPE_OF_POSTAMBLE IDTechReader.TYPE_OF_PREAMBLE IDTechReader.TYPE_OF_PREFIX IDTechReader.TYPE_OF_SUFFIX IDTechReader.TYPE_OF_TERMINATOR IDTechReader.TYPE_OF_TRACK_SELECTION IDTechReader.TYPE_OF_TRACK_SEPARATOR IDTechReader.TYPE_OF_DEFAULT IDTechReader.TYPE_OF_MSR_READING_DISABLE IDTechReader.TYPE_OF_MSR_READING_ENABLE
Return:	None
Example:	**public void onReceivedSuccess(int funcType) {** //TODO update connection status, or others **}**

IDTechUtils Class

static void xor(byte[] bs)	
Description:	XOR operation
Parameter:	**bs** operant
Return:	XOR result, or 0
Example:	**IDTechUtils.xor(new byte[] {1, 2, 3, 4, 5});**

static void hasRestrictedCharacter(String text)	
Description:	If there is restricted character "% E ? ; +"
Parameter:	**text**
Return:	true; false
Example:	**IDTechUtils.hasRestrictedCharacter("this is a + test");**

static void byteToHexString(byte[] bArray)

Description:	Convert byte array to Hex String. For example, bArray={0x0f, 0x0e} will be converted to "0F 0E", lower case alphabets are converted to upper case alphabets
Parameter:	**bArray**
Return:	Hex String
Example:	**IDTechUtils.byteToHexString(new byte[] {1, 2, 3, 4, 5});**

Return values:

int TYPE_OF_SUFFIX = 0x00;
int TYPE_OF_PREFIX = 0x01;
int TYPE_OF_TRACK_SELECTION = 0x02;
int TYPE_OF_TRACK_SEPARATOR = 0x03;
int TYPE_OF_MSR_READING_DISABLE = 0x04;
int TYPE_OF_MSR_READING_ENABLE = 0x05;
int TYPE_OF_DECODING = 0x06;
int TYPE_OF_TERMINATOR = 0x07;
int TYPE_OF_KEY_MANAGEMENT = 0x08;
int TYPE_OF_TRACK2 = 0x09;
int TYPE_OF_ENCRYPTION = 0x0A;
int TYPE_OF_DEFAULT = 0x0B;
int TYPE_OF_POSTAMBLE = 0x0C;
int TYPE_OF_PREAMBLE = 0x0D;
int TYPE_OF_REVIEW = 0x0E;
int TYPE_OF_REVIEW_FIRMWARE_VERSION = 0x0F;
int TYPE_OF_REVIEW_KSN = 0x10;
int TYPE_OF_REVIEW_SECURITY_LEVEL = 0x11;
int TYPE_OF_REVIEW_SERIAL_NUMBER = 0x12;
int TYPE_OF_WAITING_SWIPE = 0x13;

int DECODING_RAW_DATA_IN_IDTECH_MODE = 0x30;
int DECODING_BOTH_DIRECTIONS = 0x31;
int DECODING_HEAD_IN_DIRECTION = 0x32;
int DECODING_HEAD_AGAINST_DIRECTION = 0x33;
int DECODING_RAW_DATA_IN_OTHER_MODE = 0x34;

int TERMINATOR_NONE = 0x00;
int TERMINATOR_CR = 0x0D;
int TERMINATOR_OTHER = 0x01;

int PREFIX_TRACK1 = 0x34;
int PREFIX_TRACK2 = 0x35;
int PREFIX_TRACK3 = 0x36;

```
int SUFFIX_TRACK1 = 0x37;
int SUFFIX_TRACK2 = 0x38;
int SUFFIX_TRACK3 = 0x39;

int TRACK_ANY = 0x30;
int TRACK_1_ONLY = 0x31;
int TRACK_2_ONLY = 0x32;
int TRACK_1_AND_2 = 0x33;
int TRACK_3_ONLY = 0x34;
int TRACK_1_AND_3 = 0x35;
int TRACK_2_AND_3 = 0x36;
int TRACK_ALL_3 = 0x37;
int TRACK_ANY_1_AND_2 = 0x38;
int TRACK_ANY_2_AND_3 = 0x39;

char TRACK_SEPARATOR_NONE = 0x00;
char TRACK_SEPARATOR_CR = 0x0D;
char TRACK_SEPARATOR_OTHER = 0x01;

int KEY_MANAGEMENT_FIXED = 0x30;
int KEY_MANAGEMENT_DUKPT = 0x31;

int TRACK2_NO_SENTINEL_SEND_DATA = 0x30;
int TRACK2_HAS_SENTINEL_SEND_DATA = 0x31;
int TRACK2_NO_SENTINEL_SEND_ACCOUNT_23H = 0x32;
int TRACK2_HAS_SENTINEL_SEND_ACCOUNT_NUMBER = 0x33;
int TRACK2_NO_SENTINEL_SEND_ERROR = 0x34;
int TRACK2_HAS_SENTINEL_SEND_ERROR = 0x35;
int TRACK2_NO_SENTINEL_SEND_ACCOUNT_ERROR = 0x36;
int TRACK2_HAS_SENTINEL_SEND_ACCOUNT_NUMBER_ERROR = 0x37;
int TRACK2_NO_SENTINEL_SEND_DATA_ALT = 0x38;
int TRACK2_HAS_SENTINEL_SEND_DATA_ALT = 0x39;
int TRACK2_NO_SENTINEL_SEND_ACCOUNT_ALT = 0x3a;
int TRACK2_HAS_SENTINEL_SEND_ACCOUNT_NUMBER_ALT = 0x3b;
int TRACK2_NO_SENTINEL_SEND_ERROR_ALT = 0x3c;
int TRACK2_HAS_SENTINEL_SEND_ERROR_ALT = 0x3d;
int TRACK2_NO_SENTINEL_SEND_ACCOUNT_ERROR_ALT = 0x3e;
int TRACK2_HAS_SENTINEL_SEND_ACCOUNT_NUMBER_ERROR_ALT = 0x3f;

int ENCRYPTION_DISABLE = 0x30;
int ENCRYPTION_TDES = 0x31;
int ENCRYPTION_AES = 0x32;
```

int MSR_READING_DISABLE = 0x30;
int MSR_READING_ENABLE = 0x31;

Example:

```
public class IDTechDemoActivity extends Activity
              implements OnReceiveListener {

    private IDTechOpenHelper mIDTechHelper = null;

    protected void onCreated(Bundle savedInstanceState ) {
        super.onCreate(savedInstanceState);
        //set listener
        mIDTechHelper = new IDTechOpenHelper();
        mIDTechHelper.setOnReceiveListener(this);

        //Initialization
    }
    protected void onDestroy() {
        super.onDestroy();
        mIDTechHelper.close();// disconnect Bluetooth connection
    }
    /*
     * to be implemented
     */
    public void onConnectErrorInfo(int errorType, String errorMsg) {
        // Bluetooth connection failed
    }
    public void onConnected() {
        // Bluetooth connected
    }
    public void onConnecting() {
        //Bluetooth is connecting
    }
    public void onDisconnected() {
        // Disconnect Bluetooth
    }
    public void onReceivedData(int funcType, byte[] reviewDatas) {
        // Receive data
    }
    public void onReceivedFailed(int funcType) {
        // on received failed
```

```
    }
    public void onReceivedSuccess(int funcType) {
        //  on received success
    }
}
//Bluetooth device premission
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```