TapLinx Android SDK Release Notes

Table of Contents

1 General Points

TapLinx Android SDK consists of Android library that can be used to generate APDU to communicate with NXP Smartcards and uses Android SDK's transceive functionality. The NFC transceive operations in Android is dependent on the middleware provided by the OEMs. As we do not have control on the middleware; some APIs may not work in some devices. Below are the some of the know issues due to middleware found in the testing with different OEMs.

1.1 Known issues with certain Android Devices

Version 1.7

Same as previous version.

Version 1.6:

Same as previous version.

Version 1.5:

Same as previous version.

Version 1.3:

Sl.No.	Device	Card	Issue
1	Model-LG Nexus 5x	Plus EV1	causes re-detection continuously
2	Model-One Plus 1	Plus	NFC not responding properly
3	Model-Samsung A5	Plus EV1 SL3	Plus EV1 SL3 is detected as PLUS X SL3
4	Model- Samsung S7	Plus EV1 SL3	Plus EV1 SL3 & SL0 is detected as PLUS X SL3 & SL0 respectively

Version 1.2:

Same as previous version.

Version 1.1:

Sl.No.	Device	Card	Issue
1	Model-OnePlus	ICODE	Not getting detected.
2	Model-LG Nexus 5x	Plus SL1	causes re-detection continuously

3	Model-Moto X Play	DESFire EV1/EV2	ISO 7816 command mode does not work
4	Model-Nexus 5X (Android 7.1.1) , Model-OnePlus A3003 (Android 6.0.1)	Plus S , Plus X	OriginalityCheck does not work
5	Model-Samsung Galaxy S7	DESFireEV2	Not get detected if random id is set
6	Model-Samsung Galaxy S7(SM-G930FD & Android 6.0.1) , Model-Samsung Galaxy S5(SM-G900H & Android 6.0.1)	MIFARE Identity	As this card is DESFire EV2 based and by default RandomID enabled and hence this also not getting detected
7	Some devices	Ultralight , NTAG	Detection takes long time
8	All devices	Plus	Switching PLUS SL1 to SL3 is not possible because of android middleware issues
9	All devices	UltralightNano	Total NDEF maximum size returns 46 instead of 40

2 Release Contents

Version History

Date	Version
17-January-2017	1.1
08-February-2017	1.2
22-May-2017	1.3
16-November-2017	1.4
12-July -2018	1.5
21-Mar-2019	1.6
9-Aug-2019	1.7

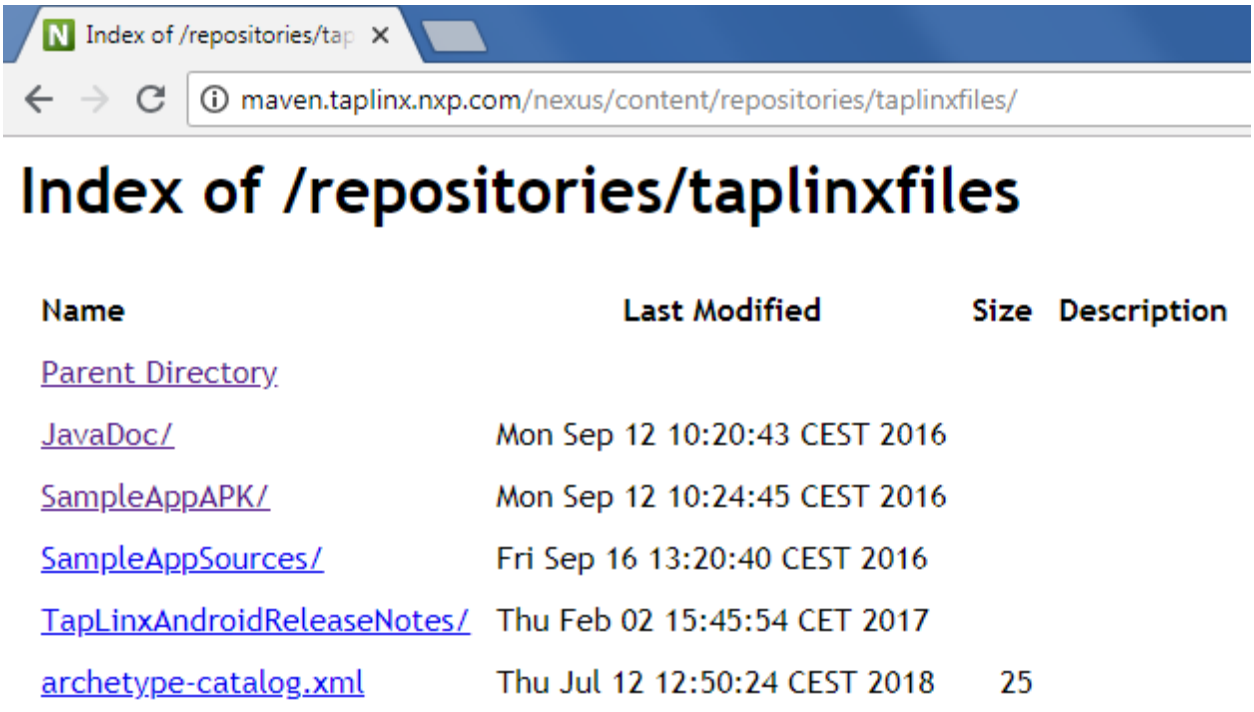
Application Note

Refer the URL - https://www.mifare.net/en/products/tools/taplinx-application-note/ for setting up your project build gradle file for pulling the TapLinx Android Library “taplinx-android:nxpnfcandroidlib:<ver>” from public maven repo.

Repo for Supporting Files

Refer the following URL for release contents of latest and previous release versions SDK.

http://maven.taplinx.nxp.com/nexus/content/repositories/taplinxfiles/



3 New Features

Version 1.7:

1) Added support for DESFire EV2 proximity check.
2) Added support for DESFire Light NDEF format.

Version 1.6:

3) Introduced Offline Local License Verification.
4) Added support for DESFire EV2 XL.

Version1.5:

NA

Version 1.4:

5) Implemented a method for getting the current version of TapLinx.
6) Implemented authentication using predictable challenge for DESFireEV1.

Version 1.3:

1)Added support for INTAG213TagTamper

Version 1.2:

NA

Version 1.1:

1) Added support for MIFARE Identity

2) Added support for NTAG413 DNA

4 Enhancements

Version 1.7:

NA.

Version 1.6:

1) Enhanced app registration by adding Offline Local License Verification, where user can use the TapLinx library without verifying the license Online, provided the offline verification key is genuine.

2) Replaced Google Analytics by Firebase Analytics.

Version 1.5:

1) Added ISO14443-L4 support to Identify for CID,NAD is supported in RATS response

2) Added Command exchange as per ISO14443-L4.

3) Added Implement Sample application using ISO14443-L4 implementation and using Plus EV1 card.

4) Taplinx- Value of Mirror Page and Mirror Byte is set to 0 if bytes required calculation is 0 in NTAG21X

Version 1.4:

1) Add support for predictable challenge for DESFire EV1

2) ISO File Id should be accepted Uniformly all the APIs and cards.

3) Refactoring of Sample Application source code.

Version 1.3:

1) Large Frame Size support in DESFire EV2 Family cards

2) Updated clone detection for newly identified clone cards

3) Split IDESFireEV2#getKeySetVersion in to two APIs IDESFireEV2#getAllKeySetVersion and IDESFireEV2#getKeyVersionFromKeySet

Version 1.2:

NA

Version 1.1:

1) Getter methods are added to the FileSettings & its derived classes.

2) Strict length error check is added on responses from all the cards in the detection logic.

3) Symmetric originality check in DESFireEV2 is tested.

4) New ISO Select file API is added which returns FCI.

5) NTAG213 & Ultralight detection speed improvement.

6) FileSettings base class made abstract & hence users cannot create it directly.

7) Added support for MIFARE Identity and NTag413 DNA in sample application.

8) Merge InvalidArgumentException & UsageException to single exception and updated the javadoc suitably.

9) As NTAG413 DNA is a derivative of DESFire in spite of being named as NTAG413, and hence this will be under com.nxp.nfclib.desfire package. So please import the desfire package to use the NTAG413 DNA.

10) *"public void enableReaderMode(final int presenceCheckDelay, NfcAdapter.ReaderCallback readerCallback, int flags) throws NxpNfcLibException"* API is added in the NxpNfcLib class to set the custom presence check delay on certain devices where presence check happens too often and cause issues in the session based communication with advanced cards like DESFire EV1 and EV2.

5 Bug Fixes

Version 1.7:

1) Offline registration with TapLinx for Android 21 is failed – resolved

Version 1.6:

1) Key diversification for AES with diversification input as 15 bytes is not valid - resolved

2) User is unable to write NDEF message for MIFARE Ultralight card – resolved

3) DESFire - Predictable Challenge AuthCounter is not getting increased – resolved

4) Issue in isoSelect API (For Plus EV1) - resolved

Version 1.5:

1) In a TapLinx Desktop Java Sample App, connected reader name is not displayed - resolved

2) In a TapLinx Desktop Java Sample App, No Error message is shown when Reader is not connected ad execute button is clicked- reader - resolved

3) In a TapLinx Desktop Java Sample App, Application doesn't respond/hangs when card is not available on reader and execute button is clicked - resolved

4) In a TapLinx Desktop Java Sample application --> Response is shown as null when EV1 card is placed on reader and In Plus EV1 SL3 tab is selected. – resolved

5) In a TapLinx Desktop Java Sample App, Log communication doesn't change when user is navigated from one tab to another.

6) Andriod sample app after second tap null pointer exception is thrown – resolved.

7) Finalize keyset operation is failling after change key with integrity error – resolved

8) In DESFire EV2, Linear and Cyclic Records Write /Read fails with Boundary Error - resolved

9) Taplinx - WriteData on DESFireEV1 for file whose size is greater than 0xFF is failing - resolved

10) TapLinx - Can not communicate with Mifare Plus EV1 SL1 with any sector moved to Sl1SL3Mix mode

11) Taplinx - WriteData on DESFireEV1 for file whose size is greater than 0xFF is failing.

15) TapLinx - NTAG 413 DNA Interaction Counter not visible in Read flow

Version 1.4:

1) Issue related to changeKeySettings at PICC level for DESFire EV1/2 cards in which key settings bitmap wrongly generated - Resolved
2) Verified support for NTAG413 DNA and some bugs fixes related to file settings - Resolved
3) Key Diversification for AES128, 2k3DES and 3k3DES verified and issue related to padding - Resolved
4) For NTAG413 DNA getNFCCounter was failing in TapLinx - Resolved
5) For DESFire EV2, ChangeKey with diversified key giving integrity error - Resolved
6) For DESFire EV2, Credit/Debit operations are failing after EV2 non-first authentication - Resolved
7) For DESFire EV2, Read/Write data is giving length error in EV2 secure messaging - Resolved
8) For DESFire EV2 ISO-Chaining was not working for writeData/Record commands - Resolved
9) AuthStatus is not lost when new tag gets connected - Resolved
10) For DESFire EV2, getValue() always sent with communicationMode = MACed - Resolved

11) For DESFire, ISO select is failing with null pointer exception - Resolved
12) For DESFire EV2, creating Transaction MAC File in DAM application returns length error - Resolved
13) For DESFire EV2, not able to add Multiple file access rights while creating file - Resolved
14) For DESFire, If Card or application is configured for 2k3DES and if user pass 16 bytes in authentication sometime invalid key length exception is thrown - Resolved
15) Get Free Memory issue - Resolved
16) TapLinx-Static Key authentication for getting Card UID for Diversified keys - Resolved
17) For DESFire, file number issue - Resolved
18) For DESFire, change Key issue in 3k3DES - Resolved
19) For DESFire EV2, bug in transcieve command - Resolved
20) For NTAG I2C plus, in fastWrite(Data) API session register were not getting updated- Resolved
21) For DESFire, Illegal Command Code using Predictable Challenge - Resolved

Version 1.3:

22) Add Failure Exceptions in ICODE DNA
23) ChangeKey issue fixed for application level in DESFire EV2 card
24) getKeySetVersion Method does not return the correct version fixed
25) DESFire EV1/EV2 setConfiguration bug fixes
26) IDESFireEV1.getFreeMemory() return invalid values fixed
27) Plus SL1 do originality check should accept SL1CardAuthKey to perform originality check
28) Misbehavior on wrong password on an Ultralight EV1
29) Improvement for misusing TapLinx for attacks on an Ultralight EV1
30) Mifare DESFire EV1/2 auth/read/identify problems on Samsung Galaxy S7/S7 Edge

Version 1.2:

1) Issue related to DESFire EV1 standard file communication in encrypted mode resolved.

Version 1.1:

1) TapLinx sending an annoying popup if NFC is off is removed.
2) Desfire getting detected as Plus on certain devices is fixed.
3) Bug found in MISMART Application in EV2 secure messaging is fixed.
4) Fixed create cyclic record file creation API in DESFire.
5) Fixed ChangeKeySettings API for DESFire EV2 at application level.
6) In DESFireEV2, isoSelectFile does not returns FCI template.

7) In DESFireEV1, readNDEF is failing with exception format exception in ISO CommandSet mode.

8) NtagFactory#isNTAG203x API is fixed.

6 API Signature Changes

Version 1.7:

NA

Version 1.6:

NA

Version 1.5:

Class	Version 1.3	Version 1.2
INTag413DNA	boolean verifySecureDynamicMessagingMac(byte[], byte[], byte[], byte[], byte[])	Change in signature from (byte[], byte[], byte[], IKeyData, byte[]) to (byte[], byte[], byte[], byte[], byte[]).

Version 1.4:

NA

Version 1.3:

Class	Version 1.3	Version 1.2
IplusEV1SL1	byte[] activateLayer4()	Void activateLayer4()
EV1ApplicationKeySettings	EV1ApplicationKeySettings(byte, byte)	EV1ApplicationKeySettings(byte[])
IDESFireEV1	byte getKeyVersion(int)	Byte[] getKeyVersion(int)

Version 1.2:

NA

Version 1.1:

Class	Version 1.0	Version 1.1
DESFireFile.RecordFileSettings	public RecordFileSettings (final FileType type, final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int recordSize, final int maxNrOfRecords, final int currNoOfRecords)	public RecordFileSettings (final FileType type, final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int recordSize, final int maxNumberOfRecords, final int currentNumberOfRecords, final byte numberOfAdditionalAccessConditions, final byte[] numberOfAdditionalAccessRights)

DESFireFile.StdDataFileSettings	public StdDataFileSettings(final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int fileSize, final byte nrAddARS, final byte[] addAccessRights)	public StdDataFileSettings(final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int fileSize, final byte numberOfAdditionalAccessConditions, final byte[] numberOfAdditionalAccessRights)
DESFireFile.BackupDataFileSettings	public BackupDataFileSettings(final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int fileSize, final byte nrAddARS, final byte[] addAccessRights)	public BackupDataFileSettings(final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int fileSize, final byte numberOfAdditionalAccessConditions, final byte[] numberOfAdditionalAccessRights)
IPlusEV1SL1	void sectorSwitchToSL3(final IKeyData key, final int sectorCount, final Map<Integer, IKeyData> keyBBlockNumberMap);	void sectorSwitchToSL3(final IKeyData sectorSwitchKey, final Map<Integer, IKeyData> keyBBlockNumberMap);
	void sectorSwitchToSL1SL3Mix(final IKeyData key, final int sectorCount, final Map<Integer, IKeyData> blockNumberToKeyMap);	void sectorSwitchToSL1SL3Mix(final IKeyData sectorSwitchKey, final Map<Integer, IKeyData> blockNumberToKeyMap);
EV1ApplicationKeySettings	public Builder setAuthenticationRequiredForFileManagement(final boolean authenticationRequiredForFileManagement)	public Builder setAuthenticationRequiredForApplicationManagement(final boolean authenticationRequiredForApplicationManagement)
EV1PICCConfigurationSettings	public void setPICCConfigurations(final boolean useRandomID, final boolean disableFormatCommand)	public void setRandomIDAndFormatConfiguration(final boolean useRandomID, final boolean disableFormatCommand)
EV2PICCConfigurationSettings	public void setPICCConfigurations(final boolean authVCMandatory, final boolean pcMandatory)	public void setVCAndPICConfigurations(final boolean authVCMandatory, final boolean pcMandatory)
DESFireEV1	byte[] getCardUID();	byte[] getManufacturerUID()

7 Removed/Obsolete APIs

Version 1.7:

NA

Version 1.6:

Class	ApiName	Alternate API
INTAG5	getNTAG5(CustomModules)	No Alternative

Version 1.5:

NA

Version 1.4:

Class	ApiName	Alternate API
IDESFireEV2	void authenticateEV2NonFirst (int, IKeyData, byte[])	void authenticateEV2NonFirst(int, IKeyData)

Version 1.3:

Class	ApiName	Alternate API
IMIFAREIdentity	byte[] commitTranscation()	byte[] commitTransaction()
IDESFireEV2	byte[] getKeySetVersion()	byte[] getKeyVersionFromKeySet() byte[] getAllKeySetVersion()

Version 1.2:

NA

Version 1.1:

Class	ApiName	Alternate API
IICodeDNA	readBuffer()	No Alternative
	void challenge()	No Alternative
IUltralightEV1	byte readVCSL()	No Alternative
LibraryManager	void setLogger(final ILogger logger)	No Alternative

8 Added Class/APIs

Version 1.6:

Class	API Name	Description
IPlusEV1SL0	void authenticateNonFirst(int, IKeyData)	Performs non first authenticate.
IPlusSL0	void authenticateNonFirst(int, IKeyData)	Performs non first authenticate.
IPlusSL3	void authenticateNonFirst(int, IKeyData)	Performs non first authenticate.
IDESFireEV1PredictableChallenge	byte[] getAuthenticationCounter()	Return current authentication counter value.
IPlusEV1	void isoSelect(byte[], IKeyData)	Selects the given IID or DF name as per ISO 7816 terminology.
LibraryManager	boolean isRegistered()	Returns true if registered.
LibraryManager	void registerJavaApp(String)	Registers java app using local path for the license file generated during application registration.

Version 1.5:

Class	API Name	Description
INTag210	boolean isPwdAuthenticated()	Returns true if the password is authenticated.
INTag213215216	void setMemProtectionAndPwdVerificationForReadWriteAccess(byte, boolean)	This method is used to set AUTH0 byte (page address from which password verification is required) and PROT bit (defines the memory protection, read or read and write)
INTag213215216	void setPwdProtectionForReadWriteAccess(boolean)	This method is used to set PROT bit (defines the memory protection, read or read and write)
INTag213215216	void setStartPageAddressForPwdAuth(byte)	This method is used to set AUTH0 byte (page address from which password verification is required)
INTag213TagTamper	boolean isPwdAuthenticated()	return true if the password is authenticated.
INTag413DNA	boolean doAsymmetricOriginalityCheck(byte[])	Performs the Asymmetric originality check.
INTag413DNA	byte[] readData(int, int, int, CommunicationMode, int)	The readData commands allows to read data from a Standard Data File.
INTag413DNA	void writeData(int, int, byte[], CommunicationMode)	The writeData command allows to write data to a Standard Data File or a Standard Backup File.

IUtility	String byteToHexString(byte[], String)	Converts the byte array to Hex String.

Version 1.4:

Class	API Name	Description
DESFireEV1PredictableChallenge	authenticateHelper (int cardkeyNo, AuthType auth, KeyType type, IKeyData keyInfo)	Helper method to authenticate in predictable challenge flow.
DESFireEV1PredictableChallenge	sendCommandAndGenerateRandomNumberB (byte [] cmd, AuthType authtype, KeyType keyType, IKeyData keyInfo)	Method to generate Random B based on Derived data for Predictable Challenge
DESFireEV1PredictableChallenge	generateCMACForNative (final byte[] data)	Method to generate CMA for Native auth type
DESFireEV1PredictableChallenge	generateCMACForISO (final byte [] data)	Method to generate CMA for ISO auth type
DESFireEV1PredictableChallenge	generateCMACForAES (final byte [] data)	Method to generate CMA for AES auth type
NTag413DNA	getKeyVersion (int iKeyNo)	Depending on the currently selected AID and given key number parameter, return key version of the key targeted.
LibraryManager	getTaplinxVersion ()	Method to retrieve the current version of TapLinx library.

Version 1.3:

Class	API Name	Description
KeyData	Key getKey()	To Retrieve Key
SecureKeyGenerator	byte[] getCMACDiversifiedKeyBytesFromKeyBytes(byte[], byte[], KeyType)	Provides the CMAC based diversified key
SecureKeyGenerator	SecureKeyGenerator getInstance(CustomModules, Provider)	To Retrieve singleton instance
SecureKeyGenerator	IKeyData getKeyFromKeyBytes(byte[], KeyType)	Gives the Secure key object from the given key bytes
IPlusSL1	boolean doOriginalityCheck(IKeyData)	For Plus SL1 we need to use SL1 card authentication key to verify chip originality
IUltraLightEV1	void setNegativePwdLimit(byte)	Sets the Limitation of negative password verification attempts
UltraLightEV1	void setNegativePwdLimit(byte)	Sets the Limitation of negative password verification attempts

IMIFAREIdentity	byte[] commitTransaction()	Commits the transaction and reads back the TMV and TMC
IMIFAREIdentity	IReader getReader()	Returns reader associated with Mifare Identity
IMIFAREIdentity	List<byte[]> readOfflineTransaction()	Read offline transaction records
IMIFAREIdentity	byte[] selectVirtualCard(byte[], IKeyData, IKeyData)	This method allows to select Virtual card base on DF name
IMIFAREIdentity	void writeOfflineTransaction(IssueTransactionParams)	Issues the offline transaction
IdentityFCIInfoForC1Type	Complete Class	Utility class for decoding FCI information
MIFAREIdentityUtility	Complete Class	Utility class for decoding FCI information
PICCFrameSize	Complete Class	Different PICC Frame size of DESFire Family cards
ICODEUtility	Complete Class	Utility class for ICODE operation
INTAG213TagTamper	Complete Class	New Tag supported
INTAG213TagTamper.MirrorType	Complete Class	Different types of mirroring supported by NTAG213 Tag Tamper
IDESFireEV2	byte[] getKeyVersionFromKeySet(byte keyNumber , byte keySetNumber)	Get the version of Key from specific keySet of selected application Note: If application 0 has been selected, only key number 0 is valid.
IDESFireEV2	byte[] getAllKeySetVersion ()	Retrieves the all key set versions of selected application.

Version 1.2:

NA

Version 1.1:

Class	ApiName
IDESFireEV2	byte[] isoSelectFile(final boolean isPICCMasterFile, final byte selectionControl, final boolean isReturnFCI, final byte[] data);
	SubType getSubType();
	IKeyData getDerivedSymmetricOriginalityKey(final IKeyData masterKey, final byte originalityKeyNo, final byte[] uid);
	byte[] commitReaderId(final byte[] tMRI)
	byte[] commitAndGetTransactionMac()
	void changeMISMARTKey(final int cardkeyNumber, final KeyType keyType,

	final byte[] oldKey, final byte[] newKey, final byte newKeyVersion);
	void changeVCKey(final int cardkeyNumber, final byte[] oldKey, final byte[] newKey, final byte newKeyVersion);
DESFireFile.FileSettings	FileType getType()
	CommunicationType getComSettings()
	getReadAccess()
	getWriteAccess() getReadWriteAccess()
	getChangeAccess()
DESFireFile.StdDataFileSettings	getNumberOfAdditionalAccessConditions
	getNumberOfAdditionalAccessRights
DESFireFile. ValueFileSettings	isGetFreeValueEnabled()
	getNumberOfAdditionalAccessConditions()
	getNumberOfAdditionalAccessRights()
	isLimitedCreditValueEnabled()
	isGetFreeValueEnabled()
	getUpperLimit()
	getLowerLimit()
	getInitialValue()
DESFireFile .LinearRecordFileSettings & DESFireFile. CyclicRecordFileSettings	getCurrentNumberOfRecords()
	getMaxNumberOfRecords()
	getRecordSize()
	getNumberOfAdditionalAccessConditions()
	getNumberOfAdditionalAccessRights()
EV2ApplicationKeySettings	setAppMasterKeyChangeable
EV2PICCConfigurationSettings	public void setISODFNameForMiSmartApplication(final IKeyData keyData, final KeyType keyType, final byte[] oldIsoDFName, final byte[] newIsoDFName)
DESFireFactory	public INTag413DNA getNTag413DNA(final CustomModules customModules)
	public boolean isCardNTag413DNA(final CustomModules customModulesObj)

9 Removed/Added Classes

Version 1.7:

NA

Version 1.6:

1) INTAG5.java

Version 1.5:

NA

Version 1.4:

2) DESFireEV1PredictableChallenge.java
3) IDESFireEV1PredictableChallenge.java

Version 1.3:

NA

Version 1.2:

NA

Version 1.1:

1) InvalidArgumentException class is merged with UsageException to avoid ambiguity.