TapLinx Android SDK Release Notes

Ultralight, C and EV1
MIFARE Ultralight
ICODE L, S, X, X2 and DNA
ICODE
Classic and EV1
MIFARE Classic
TapLinX
The missing link for mobile NFC applications
NXP RFID PRODUCTS
MIFARE PRODUCTS
Increasing security
Increasing security
NTAG 210, 213, 215, 216 & 213F, 216F, I2C, I2C Plus
ntag
Plus, SE and EV1
MIFARE Plus
DESFire EV1 and EV2
MIFARE DESFire

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 V3.1.0

SL.No.	Device	Card	Issue
1	Google Pixel 3a Samsung Galaxy S20 FE Google Pixel 2 One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus EV2 SL0, Plus EV2 SL1, Plus EV2 SL3	Cards are not being detected by the devices
2	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus S 1K SL1, Plus S 2K SL1, Plus S 4K SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
3	Google Pixel 3a Google Pixel 2	Plus X 2K SL1, Plus X 4K SL1	Plus X 2K SL1 and Plus X 4K SL1 are being detected as MIFare Classic
4	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus X 2K SL1, Plus X 4K SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
5	Samsung Galaxy S20 FE	Plus EV1 SL0	Card is not being detected by the device
6	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus EV1 SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
7	Samsung Galaxy S20 FE	Plus EV1 SL3	Card is not being detected by the device
8	Samsung Galaxy S20 FE One Plus 5T LG Nexus 5x	Plus SE SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
9	Samsung Galaxy S20 FE Google Pixel 4a (All Android 13 devices)	DESFire EV3 DESFire EV3C	TransactionTimer API Failure
10	Samsung Galaxy S20 FE Samsung A51 (All Samsung Devices)	DESFire EV3	ReadData amd WriteData API failure for 16k card

Version V3.0.0

Same as previous version.

Version V2.0

SL.No.	Device	Card	Issue
1	Google Pixel 3a Samsung Galaxy S20 FE Google Pixel 2 One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus EV2 SL0, Plus EV2 SL1, Plus EV2 SL3	Cards are not being detected by the devices
2	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus S 1K SL1, Plus S 2K SL1, Plus S 4K SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
3	Google Pixel 3a Google Pixel 2	Plus X 2K SL1, Plus X 4K SL1	Plus X 2K SL1 and Plus X 4K SL1 are being detected as MIFare Classic
4	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus X 2K SL1, Plus X 4K SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
5	Samsung Galaxy S20 FE	Plus EV1 SL0	Card is not being detected by the device
6	Samsung Galaxy S20 FE One Plus 5T Nokia 7 Plus LG Nexus 5x	Plus EV1 SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices
7	Samsung Galaxy S20 FE	Plus EV1 SL3	Card is not being detected by the device
8	Samsung Galaxy S20 FE One Plus 5T LG Nexus 5x	Plus SE SL1	App is Crashing when cards are tapped to the devices and Cards are not being detected by the devices

Version V1.9

Same as previous version.

Version V1.8

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
5	All Android device that support NFC	PLUS EV2 SL1	PLUS EV2 SL1 is detected as PLUS EV1 SL1
6	Samsung S8, Samsung S9, Samsung J7	PLUS S SL0, PLUS S SL3, PLUS X SL0, PLUS S SL3, Plus SE SL0, Plus SE SL3	Cards are not being detected by mentioned device
7	Samsung S8, Samsung S9, Samsung J7, Samsung C7	PLUS S SL1, PLUS X SL1, Plus SE SL1	Cards are detected as MIFare Classic 1k
8	Samsung Note 10 Lite, Samsung A8 Star	Plus SE SL1	Plus SE SL1 is detected as Classic 1k

1) PLUS EV2 SL1 is not completely not supported, however PLUS Ev2 SL0 and SL3 are functional.

2) Configuration help API's are not available for NTAG Boost, Switch and Link.

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
27-Jan-2020	1.8
13-Jan-2022	1.9
05-Aug-2022	2.0
20-Jan-2023	3.0.0
28-Jun-2023	3.1.0

Application Note

Refer the URL - https://www.mifare.net/en/products/tools/taplinx-application-note/

3 New Features

Version 3.1.0:

- Added support to identify the manufacturing site of MIFARE DESFire EV3 and MIFARE DESFire EV3C with Get_Fab_Identifier API.
- Added support for MIFARE DESFire EV3 [16K] variant.

Version 3.0.0:

1. Offline License Key Length for Offline License Verification is changed from 256 bytes to 2048 bytes for the security reasons.

Note:

Existing customers (Using Taplinx Android SDK) will only be affected if they upgrade to Taplinx V3.0.0 and continue to use their earlier downloaded Offline License Key of 256 bytes length.

If the existing customers (Using Taplinx Android SDK only) upgrade to Taplinx 3.0.0 they must also regenerate the new Offline License Key of 2048 bytes from mifare.net for Offline License Verification feature to work.

Version 2.0:

1. Added support for MIFARE DESFire EV2 card and MIFARE DESFire EV2J/Pay Cards.
2. Added support for Android 12.
3. Bug Fixes

Version 1.9:

1. Added support for MIFARE DESFire EV3C card.
2. Added support for NTAG22x DNA cards, NTAG22x NDA SD cards.
3. Added support for MIFARE Ultralight AES cards

Version 1.8:

1. Added support for MIFARE DESFire EV3.

2. Added support for NTAG 5 Boost, NTAG 5 Switch and NTAG 5 Link.
3. Added support for MIFARE Plus EV2 card.

Version 1.7:

1. Added support for DESFire Ev2 proximity check.
2. Added support for DESFire Light NDEF format.

Version 1.6:

1. Introduced Offline Local License Verification.
2. Added support for DESFire EV2 XL.

Version1.5:

NA

Version 1.4:

1. Implemented a method for getting the current version of TapLinx.
2. 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 3.1.0:

NA.

Version 3.0.0:

NA.

Version 2.0:

NA.

Version 1.9:

Enhancements made for PLUS EV2 and DESFire EV2 to receive the MAC response.

Version 1.8:

NA.

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 3.1.0:

1. For DESFireEV2 card-> Failures in DESFireEV2ChangeKeyEV2Factory API on DESFire EV2 2k, 8k and 32k cards is fixed.
2. For MIFARE Classic 1k card -> Failure in UnitTestCases1 is fixed.

Version 3.0.0:

1. NTAG223DNASD --> Issue with setCTTCurrentTrimValue() and getCTTCurrentTrimValue() API's is Fixed

Version 2.0:

1. For NTAG223DNASD and NATG224DNASD**->**ShowStoredTTStatus and ShowActualTTStatus API's are updating on a wrong byte
2. For NTAG223DNA and NATG223DNASD**->**BlockLocking of Incorrect Pages id Fixed
3. NTAG223DNA and NATG223DNASD**->**Locking of Incorrect Pages id Fixed
4. For DESFire EV2J card->Issue with ApplicationKeySettings API.
5. For DESFire EV2J card->Issue in LinearRecordFile_PICC_AES_TestCase API is fixed
6. For DESFireEV2J Card ->Issue in NativeChainingReadWriteDataTestFactory_ISO API is fixed
7. Issue with Taplinx continuing to send Additional Frames even if there is an error response returned during first frame is fixed
8. For DESFire EV2/EV3 card -> LengthError in read data, record, getValue, getFreeMemory commands is fixed
9. For DESFire EV3 and DESFire EV3C cards -> issue in getFileCounter API is fixed
10. For DESFire EV3 and DESFire EV3C-> SDMCtrRetAccessByte missing issue fixed
11. For NTAG424 DNA card->Issue with ReadWriteNDEFTestCases Failure is fixed
12. ForPlusEV2 Card->Failure in SL1GetVersion and Read Signature commands is fixed.

Version 1.9:

1. For DESFire EV2 → CommitTransaction API to return TMC and TMV values
2. For DESFire EV3 → Issue with ISO and AES Auth Types when using getFileSettings
3. For DESFire EV2 → Issue with Read Records API.
4. For DESFire EV2 → Reading Large data from Record File.
5. Plus EV1 2k cards getting detected as PlusSL0 Card
6. Classic Family cards are detected multiple times on Android 10

Version 1.8:

1. Bug fixes with Proximity check on DESFire Ev2.
2. DESFire EV2 Read Signature API fails in the authenticated state (Native, ISO or AES)
3. Integrity error in Change Key settings using DESFire EV2 secure messaging
4. For DESFire Ev2, Failed to append MAC while reading record file in Enciphered/MACed communication mode.
5. For DESFire Ev2, getFreeMemory() API fails after Secure Messaging & AuthenticateNonFirst is restricted by Key Length in DESFire EV2.
6. Bug fix for Read/Write NDEF and FormatT2T API in NTagI2CPlus and all T2T cards.
7. Bug fix on boundary values for Linear record API.
8. Fixed the Analytics performance issue on Authentication for DESFire EV2.

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.
12. 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 inauthentication 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:

1. Add Failure Exceptions in ICODE DNA
2. ChangeKey issue fixed for application level in DESFire EV2 card
3. getKeySetVersion Method does not return the correct version fixed
4. DESFire EV1/EV2 setConfiguration bug fixes
5. IDESFireEV1.getFreeMemory() return invalid values fixed
6. Plus SL1 do originality check should accept SL1CardAuthKey to perform originality check
7. Misbehavior on wrong password on an Ultralight EV1
8. Improvement for misusing TapLinx for attacks on an Ultralight EV1
9. 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 3.1.0:

NA

Version 3.0.0:

Class	Version 2.0	Version 3.0.0
NTAG223DNAStatusDetect	int getCTTCurrentTrimValue( )	Change in return type from CTTCurrentTrimValue to int.
NTAG223DNAStatusDetect	void setCTTCurrentTrimValue (int)	Change in signature from CTTCurrentTrimValue to int.
NTAG224DNAStatusDetect	int getCTTCurrentTrimValue( )	Change in return type from CTTCurrentTrimValue to int.
NTAG224DNAStatusDetect	void setCTTCurrentTrimValue (int)	Change in signature from CTTCurrentTrimValue to int.
UltralightAES	void setNegativePwdLimit(byte)	Method was inherited from com.nxp.nfclib.ultralight.UltralightEV1, but is now defined locally.

Version 2.0:

Class	Version 1.9	Version 2.0
NTAG224DNA	void authenticate (IKeyData, KeyType)	void authenticate(IKeyData, NTAGKeyType)
NTAG224DNA	void programKey (KeyType, byte[])	void programKey(NTAGKeyType, byte[])
UltralightAES	void authenticate (IKeyData, KeyType)	void authenticate(IKeyData, UltralightKeyType)
UltralightAES	void programKey (IKeyData, KeyType)	void programKey(UltralightKeyType, byte[])

Version 1.9:

Class	Version 1.8	Version 1.9
desfire.EV3PICCConfigurationSettings	void setVcTypeIdentifier(EV3VCTypeIdentifier)	void setVCTypeIdentifier(VirtualCardTypeIdentifier)

Version 1.8:

Class	Version 1.7	Version 1.8
IDESFireEV2	byte[] isoSelect(byte[], IKeyData)	Change in return type from void to byte[].
ValueFileConfigurationStructure	ValueFileConfigurationStructure getInstance(int, int, int, int, boolean, boolean)	Change from final to non-final.

Version 1.7:

NA

Version 1.6:

NA

Version 1.5:

Class	Version 1.3	Version 1.5
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(	public StdDataFileSettings(final CommunicationType comSettings, final byte readAccess,

	final CommunicationType comSettings, final byte readAccess, final byte writeAccess, final byte readWriteAccess, final byte changeAccess, final int fileSize, final byte nrAddARS, final byte[] addAccessRights)	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 3.1.0:

IPlusEV1	IPlusEV1SL3 getSL3SectorHelper()	Gets the Plus SL3 object which can be used to do SL3 operations on sectors that have been moved to SL3.
IPlusEV2	void proximityCheck(IKeyData, int)	1. Prepare Proximity Check. 2. Proximity Check. 3. Verify Proximity Check.

Version 3.0.0: NA

Version 2.0:

IDESFireEV3	void updateRecord(int, int, int, b yte[])	
IDESFireEV3	void updateRecord(int, int, int, b yte[])	

Version 1.9:

ntag.INTag213215216	void setMemProtectionAndPwdVerificationForReadWriteAccess(byte, boolean)	Now deprecated. Use enablePasswordProtection or enableAesProtection API instead.

Version 1.8:

DESFireFile.FileSettings	DESFireFile.FileSettings(FileType, CommunicationType, byte, byte, byte, byte)	Base class Constructor.
DESFireFile.FileType	byte mValue	holds the file mType enum value.
EV1ApplicationKeySettings. Builder	byte appKeyChangeAccessRight	
EV1ApplicationKeySettings. Builder	boolean appKeySettingsChangeable	
EV1ApplicationKeySettings. Builder	boolean appMasterKeyChangeable	
EV1ApplicationKeySettings. Builder	boolean authenticationRequiredForDirectoryConfigurationData	

EV1ApplicationKeySettings. Builder	boolean authenticationRequiredForFileManagement	
EV1ApplicationKeySettings. Builder	boolean isoFileIdentifierPresent	
EV1ApplicationKeySettings. Builder	KeyType keyTypeOfApplicationKeys	
EV1ApplicationKeySettings. Builder	int maxNumberOfApplicationKeys	

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 3.1.0:

Class	API Name	Description
IPlusEV1	void authenticateFirst(int, IKeyData, byte[])	Performs first authenticate.
	void authenticateNonFirst(int, IKeyData)	Performs non first authenticate.
DESFireUtil	void checkIfApplicationSelected(int)	Check if application selected, throw exception if no application is selected
	int updateKeyNumOrFileNumIfSAIEnabled(int, int, int)	Update Key Number or File Number if Secondary Application is active.
IDESFireEV3	String getFABIdentifier()	This API retrieves the FAB Identifier by sending getVersion and checking SW Minor version.
IPlusEV1SL3	void proximityCheck(IKeyData, int)	1. Prepare Proximity Check. 2. Proximity Check. 3. Verify Proximity Check.
IPlusEV1SL1	byte[] readSignature()	Retrieve the ECC originality check signature
	byte[] readTransactionMac(int)	Reads the TransactionMac data from TransactionMACBlock.
	void resetAuth()	Resets the authentication session Performs a Reset Auth command.
	int sectorNumberToBlockNumberForAESKeys(byte)	Helper to convert from sector number to block number which store corresponding AES 128 keys.
	boolean vcSupportLastISOL3(byte[], byte[])	Check if the Virtual Card Concept is supported, communicate PCD capabilities and retrieve the UID This command is supported only in Layer 3.

Version 3.0.0:

Class	API Name	Description
NTAG22xTagTamperStatus	byte[] getCounterValue2_ext()	
NTAG22xTagTamperStatus	byte getCttCfilt()	
NTAG22xTagTamperStatus	byte getCttCurrTrim()	

NTAG22xTagTamperStatus	byte[] getDifferenceMeasurement()	
NTAG22xTagTamperStatus	byte getMeasDblRange()	
UltralightAES	void setNegativeAuthLimit(int)	This API allows to set a negative authentication limit value {(0 < limit < 1022)}.

Version 2.0:

Class	ApiName	Description
IDESFireEV1	void isoExternalAuthenticate(int, KeyType)	Authenticate the PCD (second part of ISO7816-4 authentication) If the previous command was not ISOGetChallenge, isoExternalAuthenticate command will be rejected.
IDESFireEV1	void isoGetChallenge(KeyType, IKeyData)	Get a challenge (first part of ISO7816-4 authentication) Once the ISO/IEC7816-4 authentication has been initiated, it must be completed with ISOExternalAutheticate and ISOInternalAuthenticate
IDESFireEV1	void isoInternalAuthenticate(int, KeyType)	Authenticate the PICC (third part of ISO7816-4 authentication) If the previous command was not Cmd.ISOExternalAutheticate, isoInternalAuthenticate command will be rejected.
IDESFireEV2	CARDPLATFORM getCardPlatForm()	This api will return the PLATFORM used for DESFIRE cards
IDESFireEV2	void isoSelect(byte[])	Selects the given IID or DF name as per ISO 7816 terminology Use this API if the AuthVCMandatory is set to false which means the response of the data is not expected and no need to call IsoExternalAuthenticate api.
IDESFireEV2	void updateRecord(int, int, int, byte[])	The updateRecord command allows to update a record in a Cyclic or Linear Record File.
IDESFireEV2	void updateRecord(int, int, int, byte[])	The updateRecord command allows to update a record in a Cyclic or Linear Record File.
DESFireUtil	boolean isValidResponse(byte[], CommandSet)	Check if it is valid Response or not

Version 1.9:

Class	API Name	Description
IDESFireEV3C	void changeDESFireEV3CFileSettings(int, EV3CFileSettings)	This API changes the access parameters of an existing file.
IDESFireEV3C	void changeToMFCStateKilled()	This methods changes the MiFare Classic Support State Unlocked to Killed.
IDESFireEV3C	byte[] computeMFCLicenseMAC(byte[], byte[], IKeyData)	This methods Compute the MFC License MAC.
IDESFireEV3C	byte[] constructMFCLicense(EV3CMFCMappingBlockSettings[])	This methods constructs the MFC License.
IDESFireEV3C	void createFile(int, byte[], EV3CFileSettings)	Creates a file within a DESFire EV3C application.
IDESFireEV3C	void createFile(int, EV3CFileSettings)	Creates a file within a DESFire EV3C application.
IDESFireEV3C	void createMFCMapping(int, EV3CMFCMappingFileSettings, EV3CMFCMappingBlockSettings[], byte[], byte[])	This methods creates Mifare Classic Mapping in the file.
IDESFireEV3C	void enableMFCLicenseMACKey(byte[], byte[])	This methods enables MFC License MAC Key.
IDESFireEV3C	EV3CFileSettings getDESFireEV3CFileSettings(int)	This API gets the information on the properties of a specific DESFireEV3C file.
IDESFireEV3C	byte[] getFileCounters(int)	This API retrieves the current values associated with the SDMReadCtr related with a StandardData file after enabling Secure Dynamic Messaging.
IDESFireEV3C	EV3CMFCState getMifareClassicCardState()	This methods checks card for the MiFare Classic Support State Unlocked or Killed.
IDESFireEV3C	void restoreTransfer(int, int, boolean, boolean)	This methods copies the value from one FileType.Value file to another.
IDESFireEV3C	void restrictMFCUpdate(EV3CMFCMappingBlockSettings[], byte[], byte[])	This methods restricts Mifare Classic Update in the Classic Blocks.
IDESFireEV3C	void restrictMFCUpdate(EV3CMFCValuePairConfiguration[], byte[], byte[])	This methods restricts Mifare Classic Update in the Classic Blocks.
IPlusEV1SL3	byte[] commitReaderIDWithTMRIPrev(int, byte[])	Commits a ReaderID for the ongoing transaction.

IPlusSL3	byte[] decrementTransferForTmcTmvValue(boolean, int, int, int)	Decrement followed by a transfer on a value block
IPlusSL3	byte[] incrementTransferForTmcTmvValue(boolean, int, int, int)	Increment followed by transfer on a value block.
IPlusSL3	byte[] transferForTmcTmvValue(boolean, int)	Transfers the contents of the transfer buffer to the specified address.
IPlusSL3	byte[] writeForTmcTmv(WriteMode, int, byte[])	Writes the data to the given block number
Crypto	IAsymmetricCryptoGram getECDSASecp192CryptoGram()	
DESFireUtil	boolean isCommunicationTypePlainIfmIsAuthenticated(EV3CFileSettings)	Is Communication type plain authenticated
DESFireUtil	boolean isCommunicationTypePlainIfmIsNotAuthenticated(EV3CFileSettings)	Is Communication type plain not authenticated
NTagFactory	INTAG223DNA getNTAG223DNA(CustomModules)	@param supportModules CustomModules object holds all the user customized or library default modules for sharing across the card objects.
NTagFactory	INTAG223DNAStatusDetect getNTAG223DNAStatusDetect(CustomModules)	@param supportModules CustomModules object holds all the user customized or library default modules for sharing across the card objects.
NTagFactory	INTAG224DNA getNTAG224DNA(CustomModules)	@param supportModules CustomModules object holds all the user customized or library default modules for sharing across the card objects.
NTagFactory	INTAG224DNAStatusDetect getNTAG224DNAStatusDetect(CustomModules)	@param supportModules CustomModules object holds all the user customized or library default modules for sharing across the card objects.
NTagFactory	boolean isTagNTAG223DNA(CustomModules)	helper method to check if the tag under operation is NTAG223DNA or not.
NTagFactory	boolean isTagNTAG223DNAStatusDetect(CustomModules)	helper method to check if the tag under operation is NTAG223DNA StatusDetect or not.
NTagFactory	boolean isTagNTAG224DNA(CustomModules)	helper method to check if the tag under operation is NTAG224DNA or not.
NTagFactory	boolean isTagNTAG224DNAStatusDetect(CustomModules)	helper method to check if the tag under operation is NTAG224DNA StatusDetect or not.
UltralightFactory	UltralightAES getUltralightAES(CustomModules)	@param customModules CustomModules object holds all the user customized or library default customModules for sharing across the card objects.

UltralightFactory	boolean isTagUltralightAES(CustomModules)	helper method to check if the tag under operation is UltralightAES or not.
IDESFireEV3	void proximityCheckEV3(IKeyData, int)	The API performs the 3 operations: 1.Prepare Proximity Check: The PD is prepared to perform the Proximity Check by drawing a 7-byte random number RndR. 2.Proximity Check: The PD answers with a prepared random number at the published response time in Cmd.PreparePC This command may be repeated up to 8 times splitting the random number for different time measurements. 3.Verify Proximity Check: The random numbers are verified using cryptographic methods.
IDESFireEV3	void updateRecord(int, int, int, byte[])	The updateRecord command allows to update a record in a Cyclic or Linear Record File.
IDESFireEV3	void updateRecord(int, int, int, byte[], CommunicationType)	The updateRecord command allows to update a record in a Cyclic or Linear Record File.

Version 1.8:

Class	API Name	Description
DESFireFactory	IDESFireEV3 getDESFireEV3(CustomModules)	Returns a DESFire EV3 Object.
DESFireFactory	boolean isCardDESFireEV3(byte[])	Fetch DESFire EV3 Status
DESFireUtil	boolean checkCommand(DESFireCommand)	Check commands
DESFireUtil	boolean isCommunicationTypePlainIfmIsAuthenticated(FileSettings)	Is Communication type plain authenticated
DESFireUtil	boolean isCommunicationTypePlainIfmIsNotAuthenticated(FileSettings)	Is Communication type plain not authenticated

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 3.1.0:

NA

Version 3.0.0:

NA

Version 2.0:

NA

Version 1.9:

NA

Version 1.8:

NA

Version 1.7:

NA

Version 1.6:

1. INTAG5.java

Version 1.5:

NA

Version 1.4:

1. DESFireEV1PredictableChallenge.java
2. IDESFireEV1PredictableChallenge.java

Version 1.3:

NA

Version 1.2:

NA

Version 1.1:

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