

How to Unleash NFC Connected Tags Potential with iOS 13

Discover new opportunities for NFC with Apple's latest generation of mobile devices

PRESENTED BY MOBILE KNOWLEDGE

MOBILE KNOWLEDGE IS AN NXP CONNECT GOLD PARTNER

PRESENTER: JORDI JOFRE, jordi.jofre@themobileknowledge.com

4/10/19



PUBLIC



SECURE CONNECTIONS
FOR A SMARTER WORLD



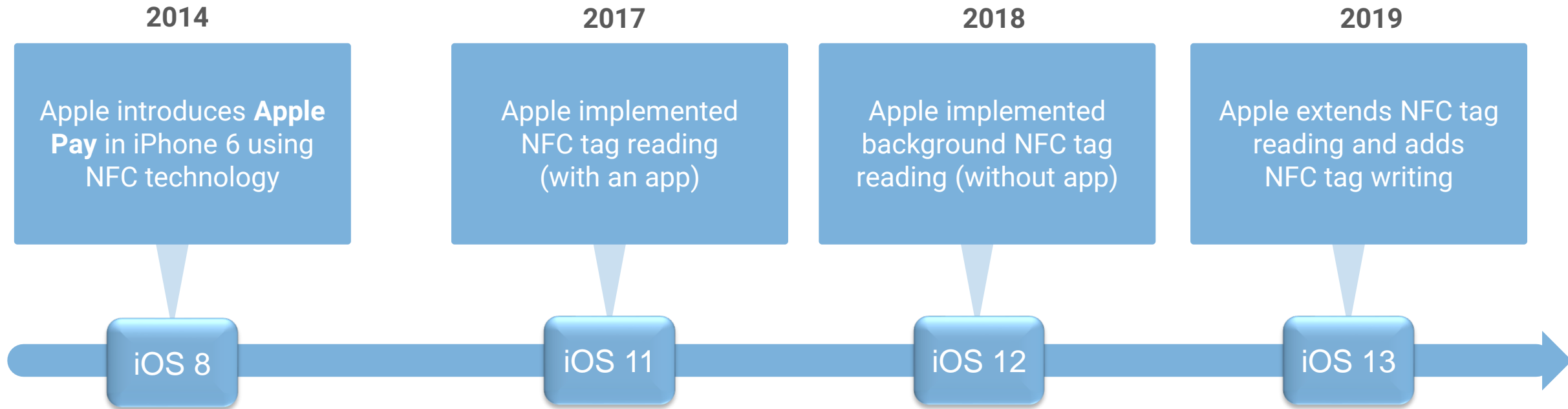
Agenda

- NFC capabilities in iOS 13
- Use cases with NFC-Connected Tags and iOS 13
- First steps with NFC on iOS 13
- NTAG I²C *plus* and NTAG 5 family
- Support tools for iOS 13 and Android.

NFC capabilities in iOS 13



Apple expands NFC in iOS 13



iOS 13 includes NFC tag writing in Core NFC framework

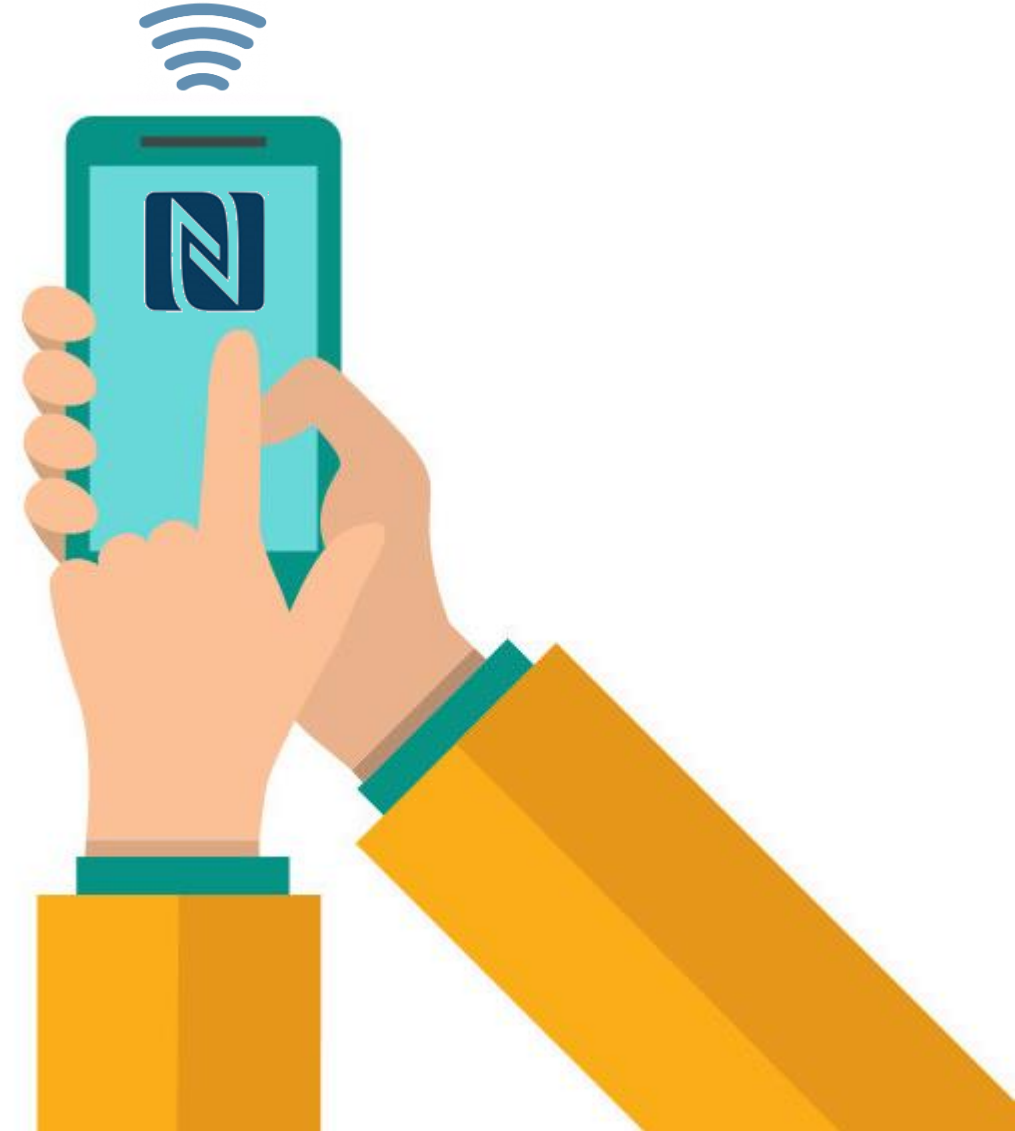
	iOS 8	iOS 11	iOS 12	iOS 13
NFC payment	Apple Pay	Apple Pay	Apple Pay	Apple Pay
Read NDEF	No	With an app	Yes, without app ¹	Yes, without app ¹
Write NDEF	No	No	No	Yes ²
Native tag reading	No	No	No	Yes ²
MIFARE support	No	No	No	Yes ²
ISO15693 support	No	No	No	Yes ²
ISO7816 support	No	No	No	Yes ²
FeliCa support	No	No	No	Yes ²

¹ Background reading supported in: iPhone XR, iPhone XS Max, iPhone XS, iPhone 11

² Tag writing supported in: iPhone 7, iPhone 7 Plus, iPhone 8, iPhone 8 Plus, iPhone XR, iPhone XS Max, iPhone XS, iPhone 11

NFC gets a lot more powerful in iOS 13

- Apple's announcement brings more NFC-based **convenience** to millions of iPhone users **worldwide**.
- Apple's Core NFC framework now supports tag reading and writing across the full range of NFC protocols for NFC tags deployed today.
- Developers can create new apps and solutions that can:
 - **Write** directly on **blank** tags, as well as communicate with tags through **native protocols**.
 - Interact with a range of contactless smartcards and tags, including NFC-enabled **passports** and other **government IDs**.



The major smartphone platforms offer extensive support for NFC



Secure Element
Host Card Emulation

Card emulation

Apple Pay
PassKit NFC Certificate

MIFARE
ISO/IEC 14443
ISO/IEC 15693
ISO/IEC 7816
FeliCa
NDEF

Tag reader / writer

MIFARE
ISO/IEC 14443
ISO/IEC 15693
ISO/IEC 7816
FeliCa
NDEF

iOS 13

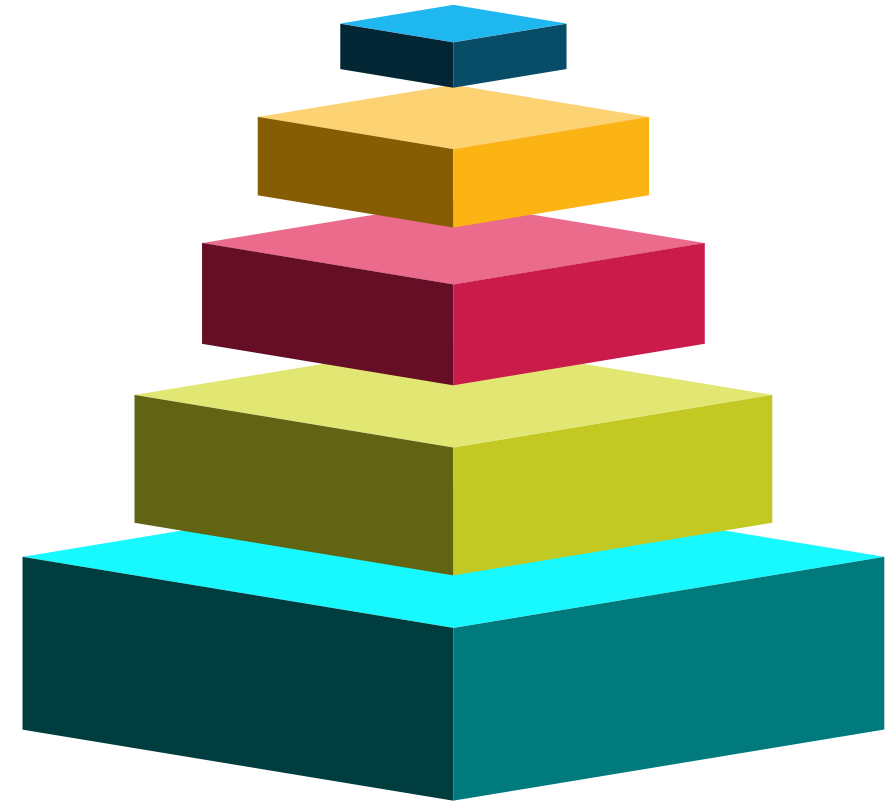
With extensive support for NFC from Android and iOS, there is no need to worry about limiting compatibility between devices

Use cases with NFC Connected Tags and iOS 13



NFC penetration continues to increase

- 2 billion NFC-enabled devices are already **in use** globally (one for every four people worldwide)¹
- NFC target market is continually expanding with the large number of **iPhones** in the market (900 million active iPhones²).
- 20 billion IoT devices will need commissioning, connection and control by 2020³.
- Automotive, IoT, public transportation, retail and payment are main growth areas.



¹ Source: NFC Forum 2018

² Source: The Verge 2019

³ Source: Gartner 2017

A world of use cases with NFC-Connected Tags and iOS 13



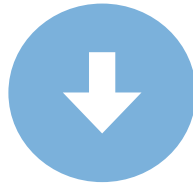
Parameterization

Use your iPhone as an external touchscreen to parameterize, configure or program update in unpowered state



Diagnosis and maintenance

Use your iPhone to read out dynamic device data, sensor readings, device information, error logs, or usage statistics



Firmware update

Use your iPhone to transfer software images to your devices without unmounting the device, cables, disks, or other means

Using NFC Connected Tags with iOS 13... in buildings



Bluetooth Motion Detector

Easy configuration of key parameters via an NFC phone, including wake-up intervals, button behavior, radio channels used, device address, etc.



Bluetooth light switch

NFC enables product set-up and contactless configuration via an NFC smartphone with one tap.



Room thermostat

Tap the thermostat with your iPhone and easily make more complex settings using the manufacturer app.



Alarms and security systems

Tap your iPhone to program the alarm or to provide access to guests or service personnel.

Using NFC-Connected Tags with iOS 13... in industrial

Signal conditioner



Parameterize the signal type, sample rate and product diagnosis even with the device unpowered using an app in your iPhone.

Industrial light curtains



Read out and copy the configuration to another safety gate or read out current status of each light beam with your iPhone.

Power supplies



Parameterize the DC output characteristic, signaling thresholds and product diagnosis using an app in your iPhone.

Power tool identification



Identify power tools even when their serial number are unreadable and show the tools repair history, purchase date, etc. with your iPhone device.

Using NFC-Connected Tags with iOS 13... in industrial (II)

Sensors and switches



Get sensor readings, configure start-up behavior or access user manual by touching the device with your iPhone.

Panel-mount programmable controllers



Control, manage and maintain coolers based on individual system requirements using your iPhone.

Relays, timers and dimmers



Configure industrial equipment that comes with limited user interfaces but with advanced settings and configurations with your iPhone.

Motor control



Use your iPhone to interact with motor controls to configure speed, stop condition or usage statistics.

Using NFC-Connected Tags with iOS 13... in fitness & healthcare



Sport bike computer

Use your NFC smart phone as an external user interface to collect and track your activities and easily adjust settings of your bike computer.



Fitness trackers

Read workout statistics from your fitness equipment with a tap of your phone.



Golf Watch

Access your game performance and upload your statistics to social networks with just a tap



Medical devices

Transfer data from blood pressure monitors, glucose meters, pedometers by a simple tap of any iPhone.

Using NFC-Connected Tags with iOS 13... in ESL and lighting



Electronic labeling for retail

Enable price updates at any time, simplified customer interactions, reassignments, firmware updates, and much more via a simple tap.



Electronic labeling for luggage

Transfer flight data from your iPhone to the label on the luggage. Ease your check-in procedure, just do it at home.



Led driver

Adjust the output current of power supply which regulates an LED or array of LEDs.



Event light system

Configuration of light devices to an electronic control unit via NFC.

...and many applications unleash by iOS 13 NFC capabilities

First steps with NFC on iOS 13



Building an NFC app in iOS 13



Configure the app to detect NFC tags

Create a project in Xcode for an app with permissions to read/write NFC tags.

Start a Reader Session

A reader session for detecting ISO7816, ISO15693, FeliCa, MIFARE tags or NDEF formatted tags.

Adopt the Reader Session Delegate Protocol

Implement the protocol delegate callbacks to receive the new tag objects.

Connect to the desired NFC tag

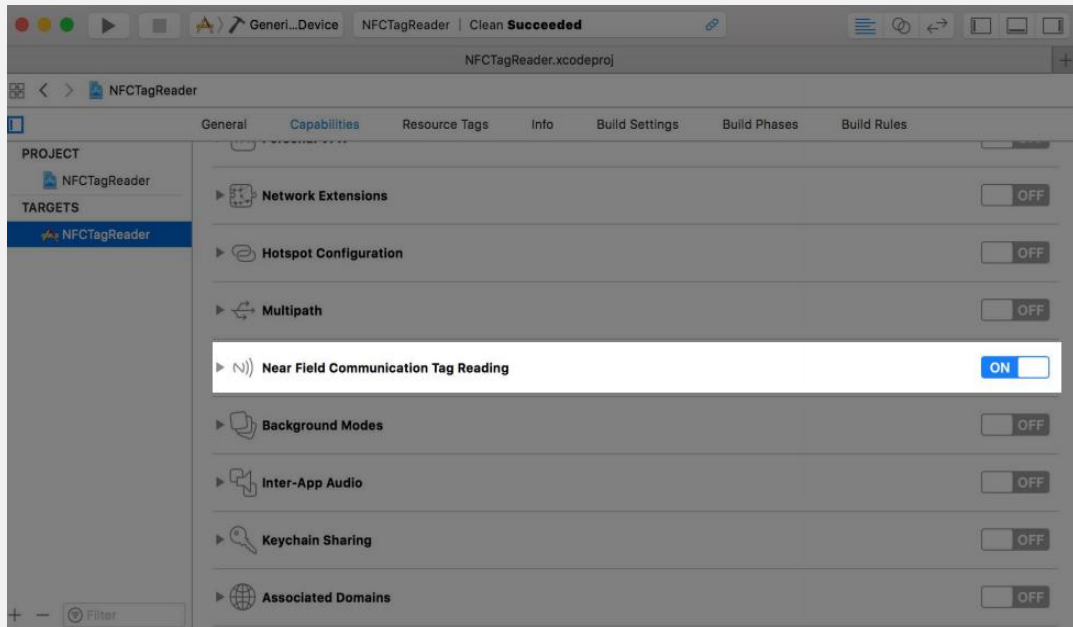
Connects the reader session to a tag and activates selected tag.

Perform operations with the NFC tag object

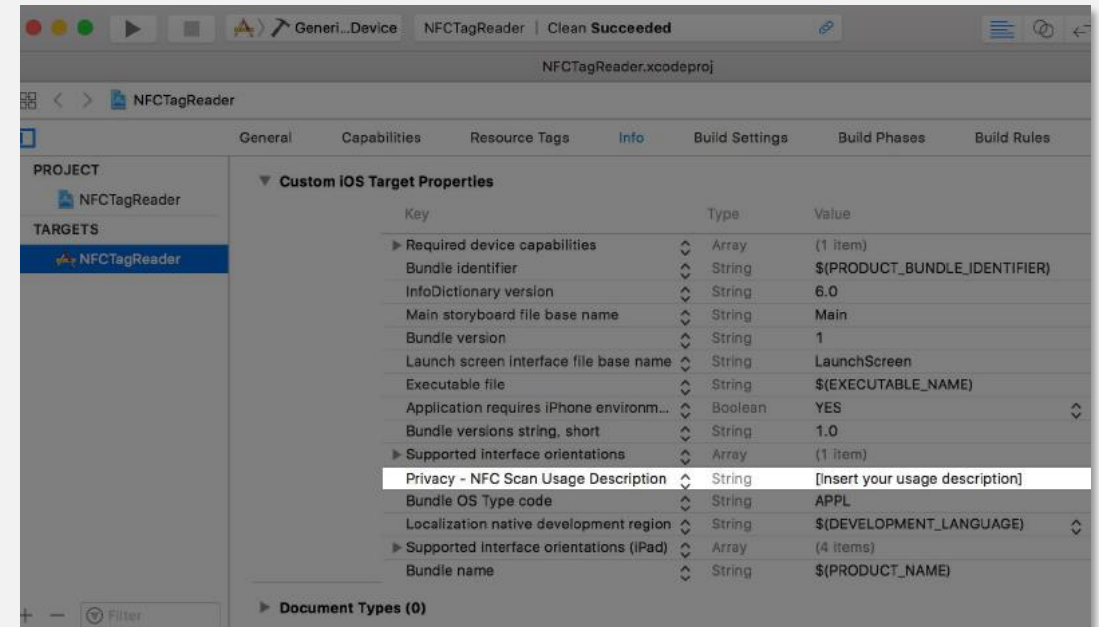
Perform all your interactions using the tag object.

Configure the app to detect NFC tags

1 Add NFC capabilities to your Xcode project



2 Add `NFCScanUsageDescription` in your app's Info.plist file



Start a Reader Session

A reader session is used to **scan** and **detect** NFC tags. The **two** types of reader sessions supported by iOS 13 are:

`NFCNDEFReaderSession`

A reader session for detecting NFC Data Exchange Format (NDEF) tags.

Supports read and write operations with NDEF tag objects



`NFCNDEFReaderSession` has been modified to support NDEF tag **writing**.

`NFCTagReaderSession`

A reader session for detecting ISO7816, ISO15693, FeliCa, and MIFARE tags

Discovery callback provides protocol specific tag objects



New in iOS 13



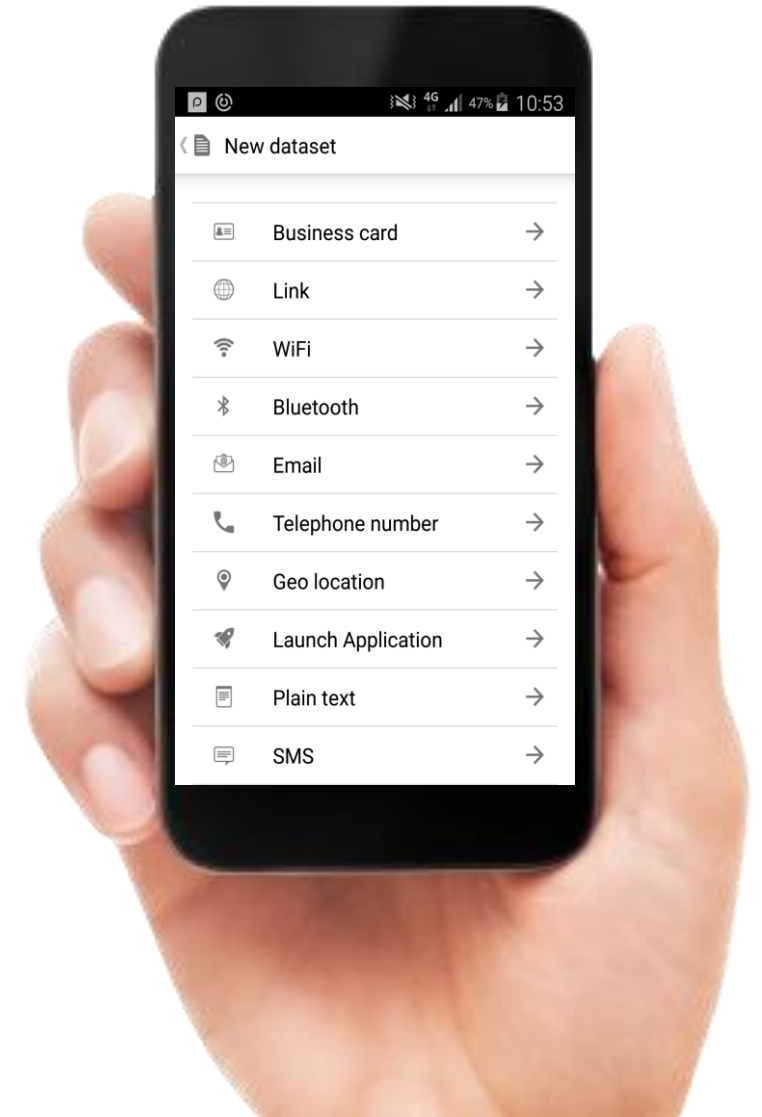
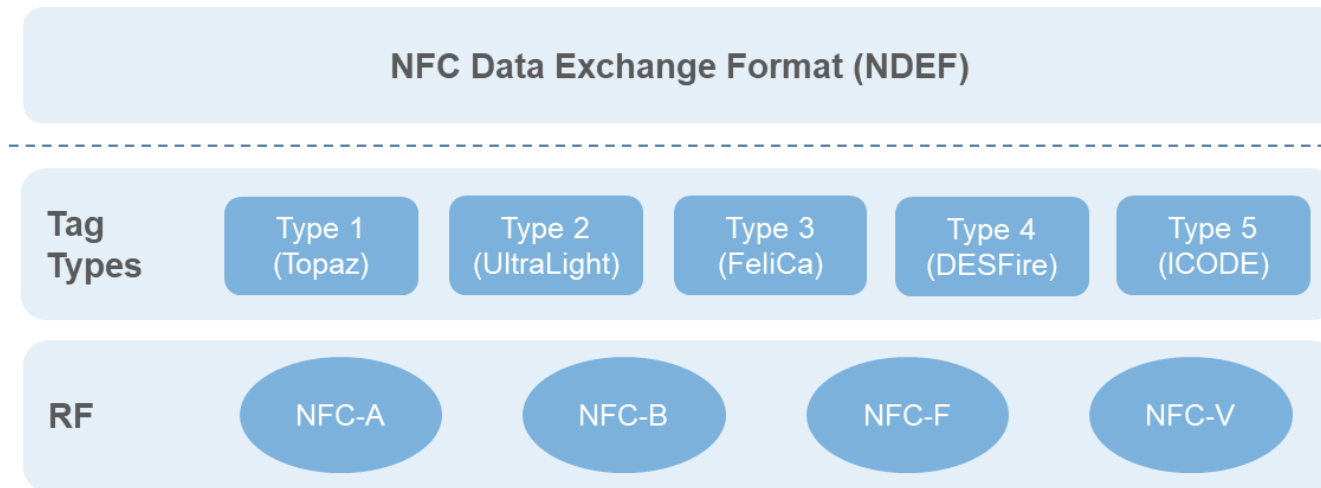
Use the appropriate reader session whether your application requires NDEF or native access.

How to use NFCNDEFReaderSession



NFC data exchange format (NDEF)

- Specifies a common data format for NFC Forum-compliant devices and NFC Forum-compliant tags.
- It is used to describe how a set of actions are to be encoded onto an NFC tag (e.g., open a URL, create an SMS, create an email, etc.).
- The benefit of using NDEF is that you do not need to have custom software running on the touching device.



Using the `NFCNDEFReaderSession`

- 1 Create session with a `NFCNDEFReaderSessionDelegate` object:

```
@IBAction func beginScanning(_ sender: Any) {  
    session = NFCNDEFReaderSession(delegate: self, queue: nil, invalidateAfterFirstRead: false)  
    session?.alertMessage = "Hold your iPhone near the item to learn more about it."  
    session?.begin()  
}
```

- 2 Implement the new `didDetect` tags callback method to receive the NDEF tag objects:

```
optional func readerSession(_ session: NFCNDEFReaderSession, didDetect tags: [NFCNDEFTag])
```

- 3 Use NDEF tag protocol:

```
var isAvailable: Bool { get }  
func queryNDEFStatus(completionHandler: @escaping (NFCNDEFStatus, Int, Error?) -> Void)  
func readNDEF(completionHandler: @escaping (NFCNDEFMessage?, Error?) -> Void)  
func writeNDEF(_ ndefMessage: NFCNDEFMessage, completionHandler: @escaping (Error?) -> Void)  
func writeLock(completionHandler: @escaping (Error?) -> Void)
```

Example implementation of `didDetect()` method

```
func readerSession(_ session: NFCNDEFReaderSession, didDetect tags: [NFCNDEFTag]) {  
    // Connect to the found tag and perform NDEF message writing  
    let tag = tags.first!  
    session.connect(to: tag) { (error: Error?) in  
        tag.queryNDEFStatus() { (ndefStatus: NFCNDEFStatus, capacity: Int, error: Error?) in  
            let myMessage = NFCNDEFMessage(data: Data())  
            tag.writeNDEF(myMessage) { (error: Error?) in  
                session.invalidate() } }  
        }  
    }
```

1

Connect to the NDEF tag

2

Query NDEF status

3

Write NDEF message

4





Invalidate session on completion

How to use NFCTagReaderSession



What is new in the `NFCReaderSession` API

The `NFCReaderSession` allows your applications to scan and connect to tags based on their underlying technologies:

- **Protocol** `NFCISO7816Tag`
An interface for interacting with an ISO 7816 tag.  Now, it is possible to send any ISO7816 command
Note: Supports ISO14443Type A & B.
- **Protocol** `NFCISO15693Tag`
An interface for interacting with an ISO 15693 tag.  New API to communicate with ISO/IEC15693 tags
- **Protocol** `NFCFeliCaTag`
An interface for interacting with a FeliCa™ tag.  New API to communicate with FeliCa tags
- **Protocol** `NFCMiFareTag`
An interface for interacting with a MIFARE® tag.  Additional MIFARE API to make connection easily

Protocol `NFCISO7816Tag` requirements:



AIDs should be listed explicitly in Info.plist file.
Payment related application identifiers are not supported.

Using the NFCISO7816Tag protocol

- 1 Create session with a `NFCTagReaderSessionDelegate` object:

```
@IBAction func beginScanning(_ sender: Any) {  
    session = NFCTagReaderSession(pollingOption: .iso14443, delegate: self)  
    session?.alertMessage = "Hold your iPhone near the ISO7816 tag to begin transaction."  
    session?.begin()}}
```

- 2 Implement the new `didDetect` and `didInvalidateWithError` tags callback method to receive the tag objects:

```
func tagReaderSession(_ session: NFCTagReaderSession, didDetect tags: [NFCTag])  
func tagReaderSession(_ session: NFCTagReaderSession, didInvalidateWithError: Error)
```

- 3 Use `NFCISO7816Tag` tag protocol:

```
func sendCommand(apdu: NFCISO7816APDU, completionHandler: @escaping (Data, UInt8,  
    UInt8, Error?) -> Void)
```


Example implementation of `didDetect()` method

```
func tagReaderSession(_ session: NFCTagReaderSession, didDetect tags: [NFCTag]) {  
    if case let NFCTag.iso7816(tag) = tags.first {  
        session.connect(to: tag) { (error: Error?) in  
            let myAPDU = NFCISO7816APDU(instructionClass:0 instructionCode:0xB0 p1Parameter:0  
p2Parameter:0 data: Data() expectedResponseLength:16)  
            tag.sendCommand(apdu: myAPDU) { (response: Data, sw1: UInt8, sw2: UInt8, error: Error?)  
in  
                guard error != nil && !(sw1 == 0x90 && sw2 == 0) else {  
                    session.invalidate(errorMessage: "Application failure")  
                    return }  
                }  
            }  
        }  
    }  
}
```

1

Connect to the ISO7816 tag

2

Send APDU and receive
response NFCISO7816APDU

3

Terminate session with error
(optional)

How to use MIFARE native tag reading



MIFARE native tag reading

Properties:

- Identifier (UID)
- HistoricalBytes
- MIFARE family: MIFARE Ultralight , MIFARE Plus, MIFARE DESFire



MIFARE Classic tags are not supported

Methods:

```
func sendMiFareCommand(commandPacket command: Data,  
completionHandler: @escaping (Data, Error?) -> Void)  
func sendMiFareISO7816Command(_ apdu: NFCISO7816APDU,  
completionHandler: @escaping (Data, UInt8, UInt8, Error?) -> Void)
```



We can send MIFARE native commands or APDU-wrapped commands (e.g., MIFARE DESFire, MIFARE Plus)

Using the MIFARE protocol

- 1 Create session with a `NFCTagReaderSessionDelegate` object:

```
@IBAction func beginScanning(_ sender: Any) {  
    session = NFCTagReaderSession(pollingOption: .iso14443, delegate: self)  
    session?.alertMessage = "Hold your iPhone near the MIFARE tag to begin transaction."  
    session?.begin() }
```

- 2 Use the `didDetect` and callback method to receive the tag objects:

```
func tagReaderSession(_ session: NFCTagReaderSession, didDetect tags: [NFCTag])
```

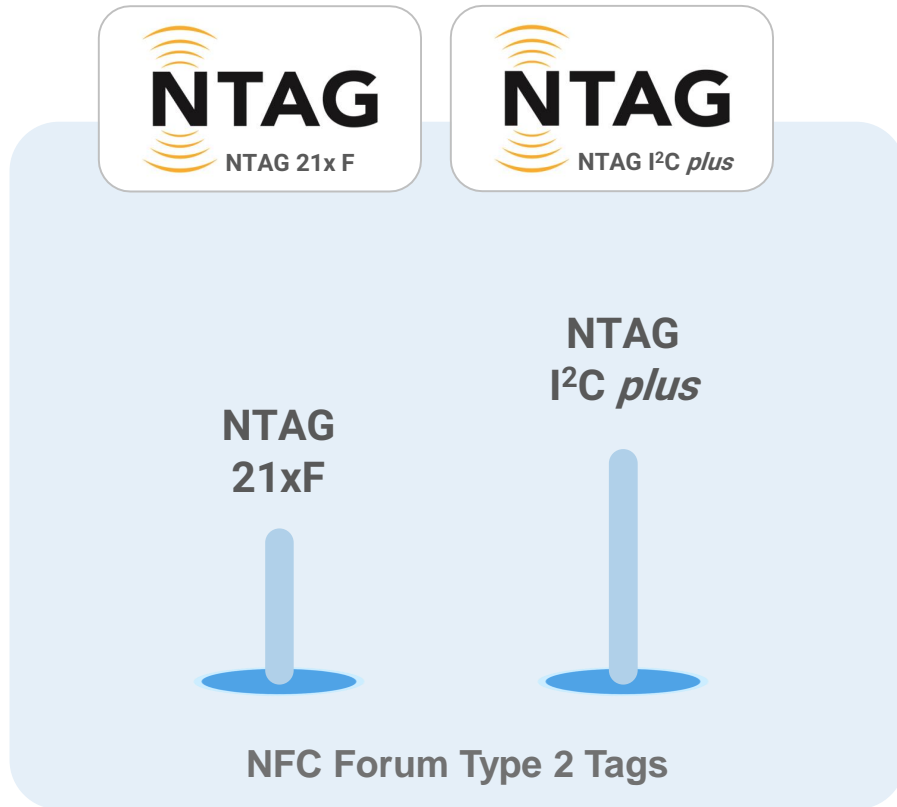
- 3 Connect and send commands:

```
func tagReaderSession(_ session: NFCTagReaderSession, didDetect tags: [NFCTag]) {  
    if case let NFCTag.miFare(tag) = tags.first {  
        session.connect(to: tag) { (error: Error?) in  
            ...  
            tag.sendMiFareCommand(commandPacket: command) { (response: Data, error: Error?) in  
                ... } } } }
```

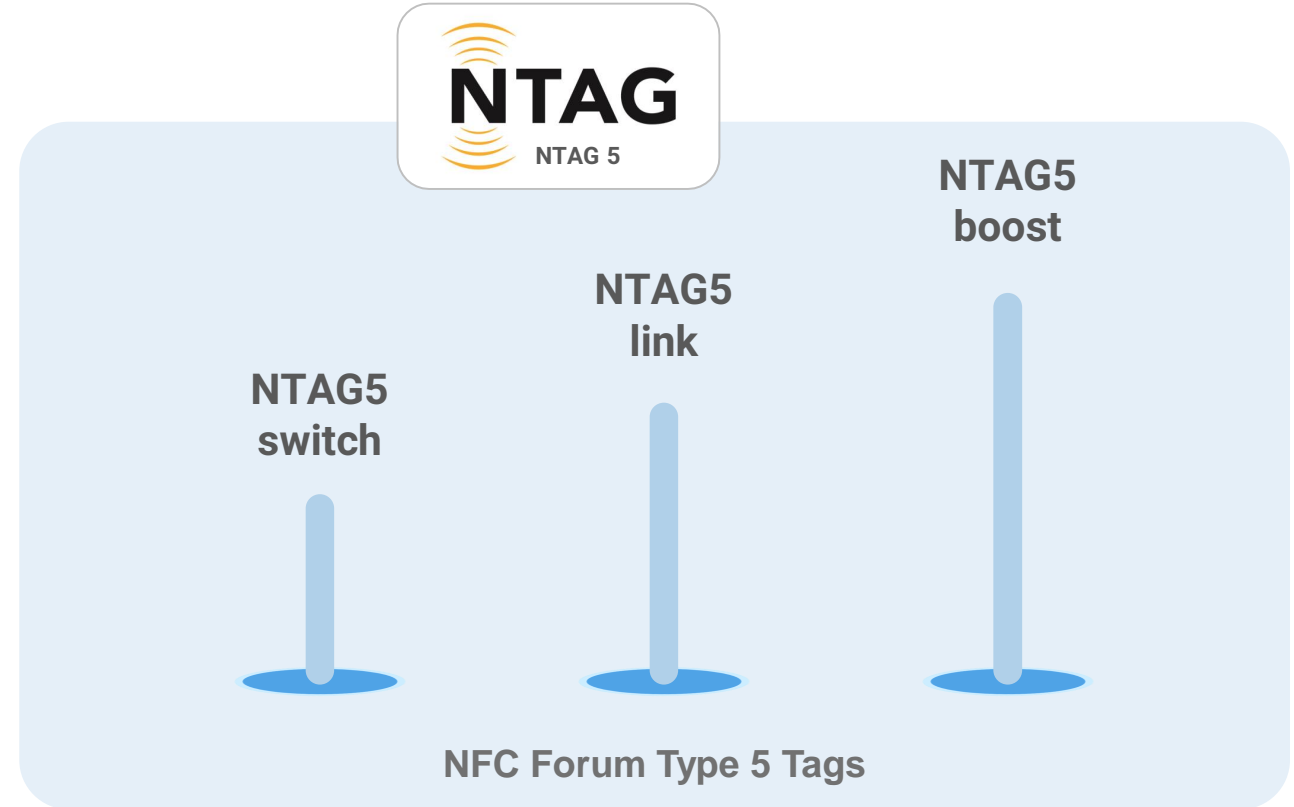
NTAG I²C *plus* and NTAG 5 family



NFC tags for electronics



NTAG 21xF - NFC passive tag with a configurable field detection pin to wake up connected electronic devices.
NTAG I²C plus – NFC passive tag with a contact I²C interface for bidirectional communication with the MCU

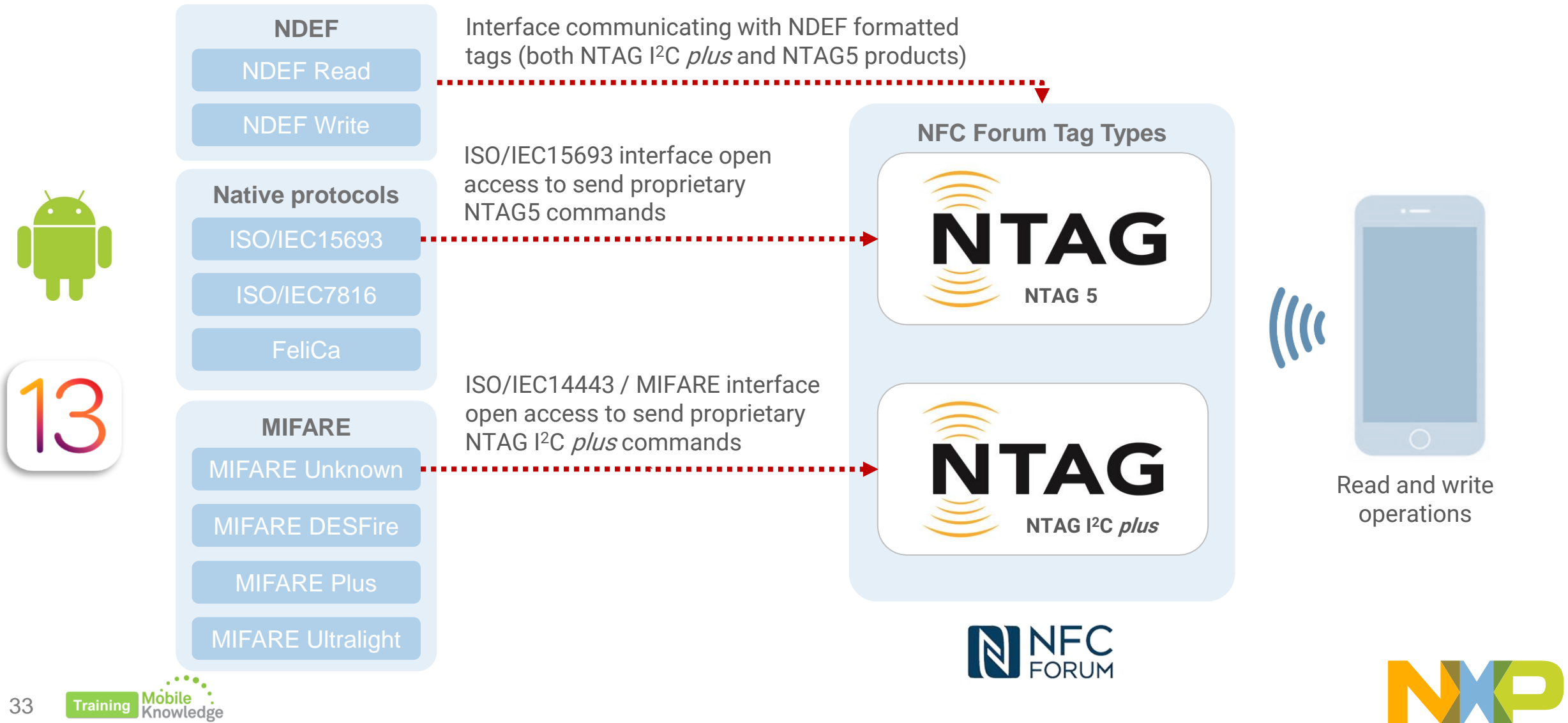


NTAG 5 boost - NFC Forum-compliant I²C bridge for tiny devices
NTAG 5 link - NFC Forum-compliant I²C bridge for IoT on demand
NTAG 5 switch - NFC Forum-compliant PWM and GPIO bridge for lighting and gaming

NFC tags for electronics – high level features comparison

Feature	NTAG I ² C <i>plus</i>	NTAG 5 switch	NTAG 5 link	NTAG5 boost
NFC interface	ISO/IEC14443	ISO/IEC15693	ISO/IEC15693	ISO/IEC15693
Energy harvesting	Yes up to 15 mW	Regulated up to 30 mW	Regulated up to 30 mW	Regulated up to 30 mW (in passive mode)
GPIO + PWM	-	✓	✓	✓
Memory areas	2	3	3	3
Memory protection	Password	Password	Password and AES authentication	Password and AES authentication
I ² C interface	Slave	-	Slave / Master	Slave / Master
Pass-through via SRAM	Proprietary	-	Proprietary and standardized	Proprietary and standardized
Active load modulation	-	-	-	✓

Full potential of NFC-connected tags is supported by iOS/Android



Support tools for iOS 13 and Android

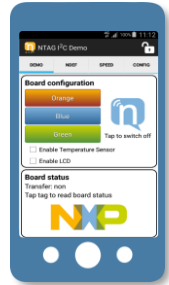


NTAG I²C *plus* and NTAG 5 family demo apps



NTAG I2C plus demo app

- Android: Demonstration of the tag energy harvesting, bi-directional communication, EEPROM storage, pass-through mode, and tag register configuration.
- iOS: Demonstration of the tag energy harvesting, bi-directional communication, EEPROM storage, pass-through mode, and tag register configuration (**Planned Oct**)



NTAG5 demo app

- Android: Demonstration of I²C master channel, GPIO, PWM, Pass-Through mode and Active Load Modulation (ALM) - (**Planned Nov**)
- iOS: Demonstration of I2C master channel, GPIO, PWM, Pass-Through mode and Active Load Modulation (ALM)



NFC software tools for tag interaction & development



TagInfo

- TagInfo Android: All low-level protocol information including UID, ATQs, SAK,ATS
 - TagInfo iOS: Know AID-based, limited low level protocol information (**Planned October**)
-



TagWriter

- TagWriter Android: Read / Writer NDEF, erase / formatting NFC tags
 - TagWriter iOS: Read / Writer NDEF, erase / formatting NFC tags (**Planned December**)
-



TapLinX

- TapLinX Java: SDK for laptops (Windows, MacOS and Linux)
- TapLinX Android: SDK for Android
- TapLinX iOS: SDK for iOS (**Planned 2020**)

Tap into new NFC potential with iOS 13 and NXP IC tag solutions

Unleash the full potential of NFC with iOS 13 and NXP®

Tap into new NFC potential with iOS 13 and NXP IC tag solutions - and let us become part of your NFC implementation.

SOLUTIONS

DEVELOPER RESOURCES

NFC – the technology that powers Apple Pay® and other smart features for iOS apps got a big upgrade with [iOS 13](#). Announced at the WWDC Conference in June, and unveiled with the new iPhone® 11 series on the 10th of September, it's now released on the 19th of September. Instead of just allowing iPhone to read NFC tags, apps will now be able to write directly to NFC tags and interact with tags through native protocols.

With this upgrade, Apple's Core NFC framework now supports NFC tag reading and writing across multiple protocols, including NFC NDEF Tags (for end-user focused content), ISO/IEC 7816 combined with ISO/IEC 14443 (e.g. for passports and ID cards), ISO/IEC 15693 (for vicinity tags, e.g. in the supply chain), MIFARE®, and FeliCa™. That means iPhone will work in more places with more types of tags than before. Enhanced iOS 13 NFC capabilities, running on iPhone 7 and up, can be fully leveraged with NXP's extensive NFC IC tag portfolio.

NXP, the leading provider of NFC IoT technology solutions, welcomes Apple® in creating a world of innovative services and rich experiences, powered by NFC. Hundreds of millions of iPhone users worldwide can benefit from the convenient 'tap to interact' functionality, and businesses can benefit from new marketing opportunities, increased operational efficiencies and rising revenues.

Anticipated NFC Scale Up

We at NXP expect this will further accelerate the momentum behind NFC. Adding iOS 13 users into the total smartphone universe means that 2 billion people worldwide who own a smartphone now also have an NFC reader in their hands to interact with the NFC tags and smart cards around them, as part of smart home, smart business and smart city applications.

Discover our products:

- [NTAG® NFC Tags](#)
- [NTAG® Connected Tags](#)
- [NTAG® SmartSensor Tags](#)
- [ICODE® Vicinity Tags](#)
- [MIFARE® Contactless ICs](#)

Find out more about new iOS 13 NFC capabilities and how these can be best leveraged by NXP's advanced NFC product solutions

https://www.nxp.com/products/rfid-nfc/nfc-hf/nfc-with-ios-13-:NFC_IOS_13#solutions

Last words

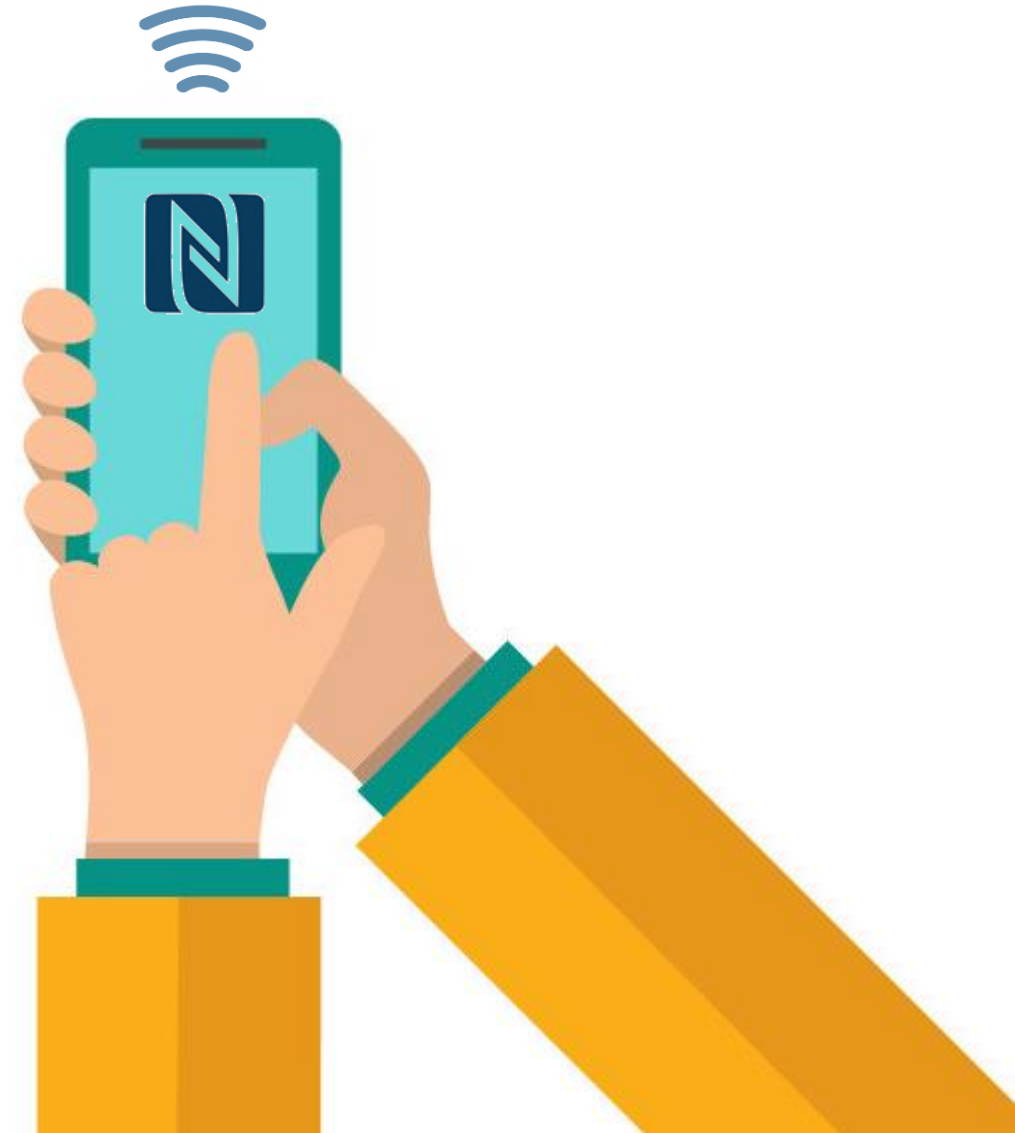
NFC potential unlocked with iOS 13

Major NFC upgrade in iOS 13 (iPhone 7 and newer models)

- Support for interacting with NDEF formatted tags (read and **write** capabilities)
- Support for native protocols. Developers can create new apps and solutions that can **write** directly on **blank** tags using ISO/IEC7816, ISO/IEC14443, ISO/IEC15693, MIFARE, FeliCa protocols



New opportunities for NFC-connected tag products with multiple applications such as parameterization, configuration, diagnosis, firmware update and many others.





Time for
Q & A





MobileKnowledge

MobileKnowledge is a team of HW, SW and system engineers, experts in **smart, connected and secure** technologies for the IoT world. We are your ideal **engineering consultant** for any specific support in connection with your **IoT** and **NFC** developments. We design and develop secure HW systems, embedded FW, mobile phone and secure cloud applications.

Our services include:

- **Secure hardware design**
- **Embedded software development**
- **NFC antenna design and evaluation**
- **NFC Wearable**
- **EMV L1 pre-certification support**
- **Mobile and cloud application development**
- **Secure e2e system design**

www.themobileknowledge.com

mk@themobileknowledge.com



We help companies leverage
the secure IoT revolution





SECURE CONNECTIONS
FOR A SMARTER WORLD