NFC

NEAR FIELD COMMUNICATION

on Android*

Daniel Holmlund, Intel
Twitter: @agnathan

Agenda

1. NFC Basics and Fun Ideas
2. Manifest and Permissions
3. Integrating with Intents system
4. Reading NFC tags and NDEF messages
5. Creating NDEF messages and writing NFC tags
6. P2P communication with Android Beam™
7. Libraries and other resources

NFC Basics

NFC

NEAR FIELD COMMUNICATION

What is NFC?

Near field communication, or NFC for short, is an offshoot of radio-frequency identification (RFID) with the exception that NFC is designed for use by devices within close proximity to each other.

- NFC is a set of short-range wireless technologies, typically requiring a distance of 10 cm or less.
- NFC operates at 13.56 MHz on ISO/IEC 18000-3 air interface and at rates ranging from 106 kbit/s to 424 kbit/s.

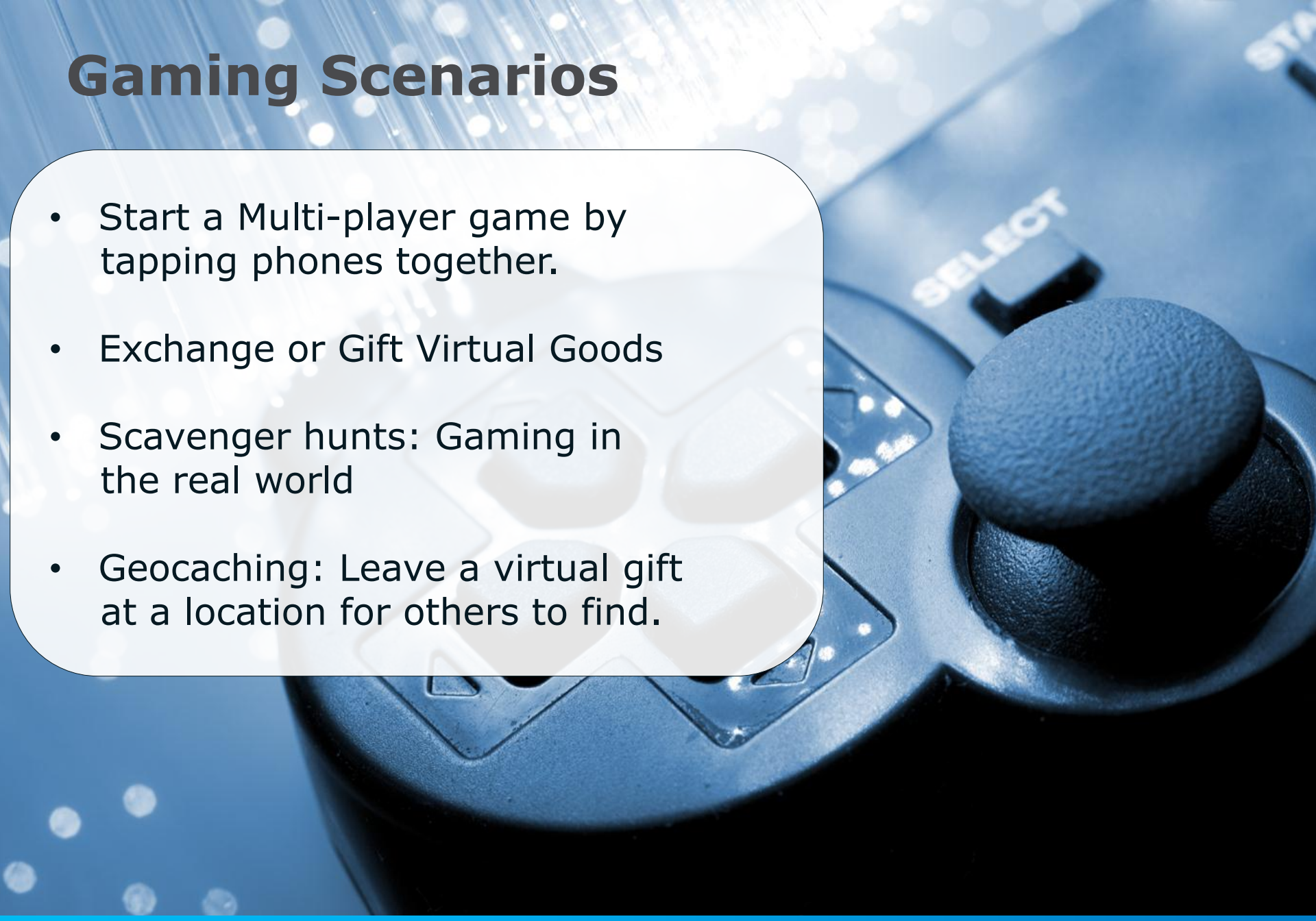
NFC Your World!

NFC

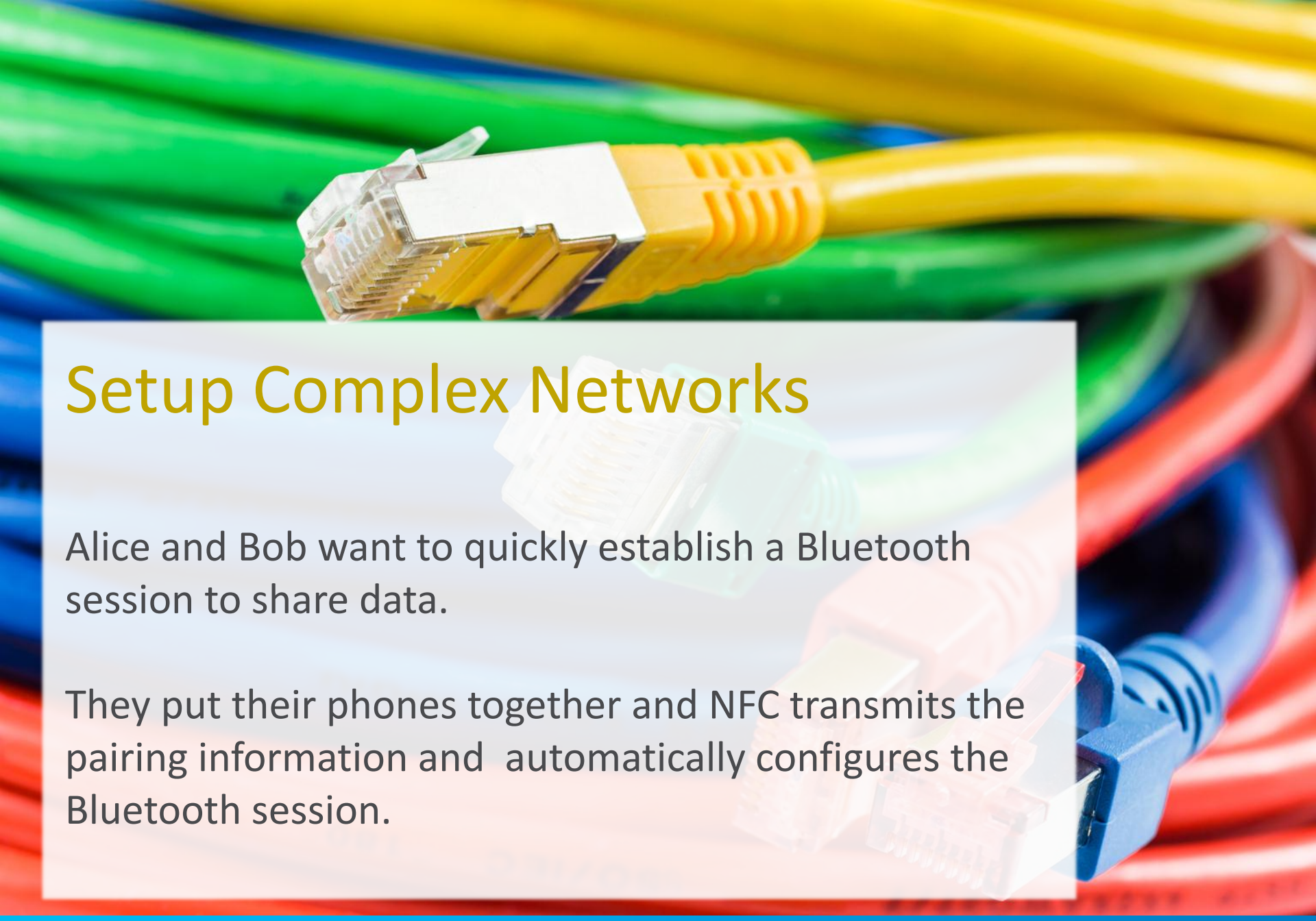
NEAR FIELD COMMUNICATION

Gaming Scenarios

- Start a Multi-player game by tapping phones together.
- Exchange or Gift Virtual Goods
- Scavenger hunts: Gaming in the real world
- Geocaching: Leave a virtual gift at a location for others to find.




GAME.
SOLVE THE CRIME! DOWNLOAD



Setup Complex Networks

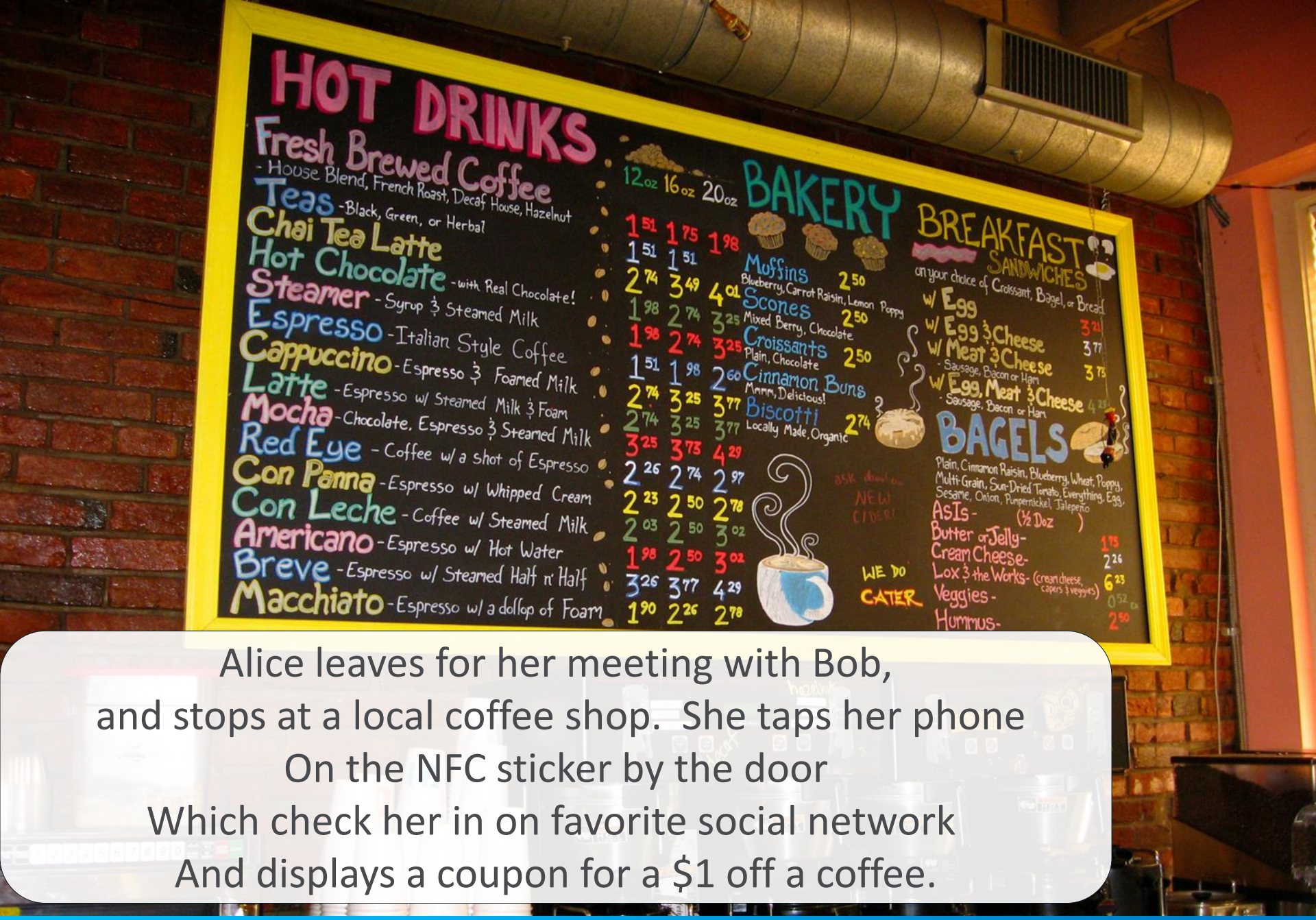
Alice and Bob want to quickly establish a Bluetooth session to share data.

They put their phones together and NFC transmits the pairing information and automatically configures the Bluetooth session.

Setup More Complex Connections

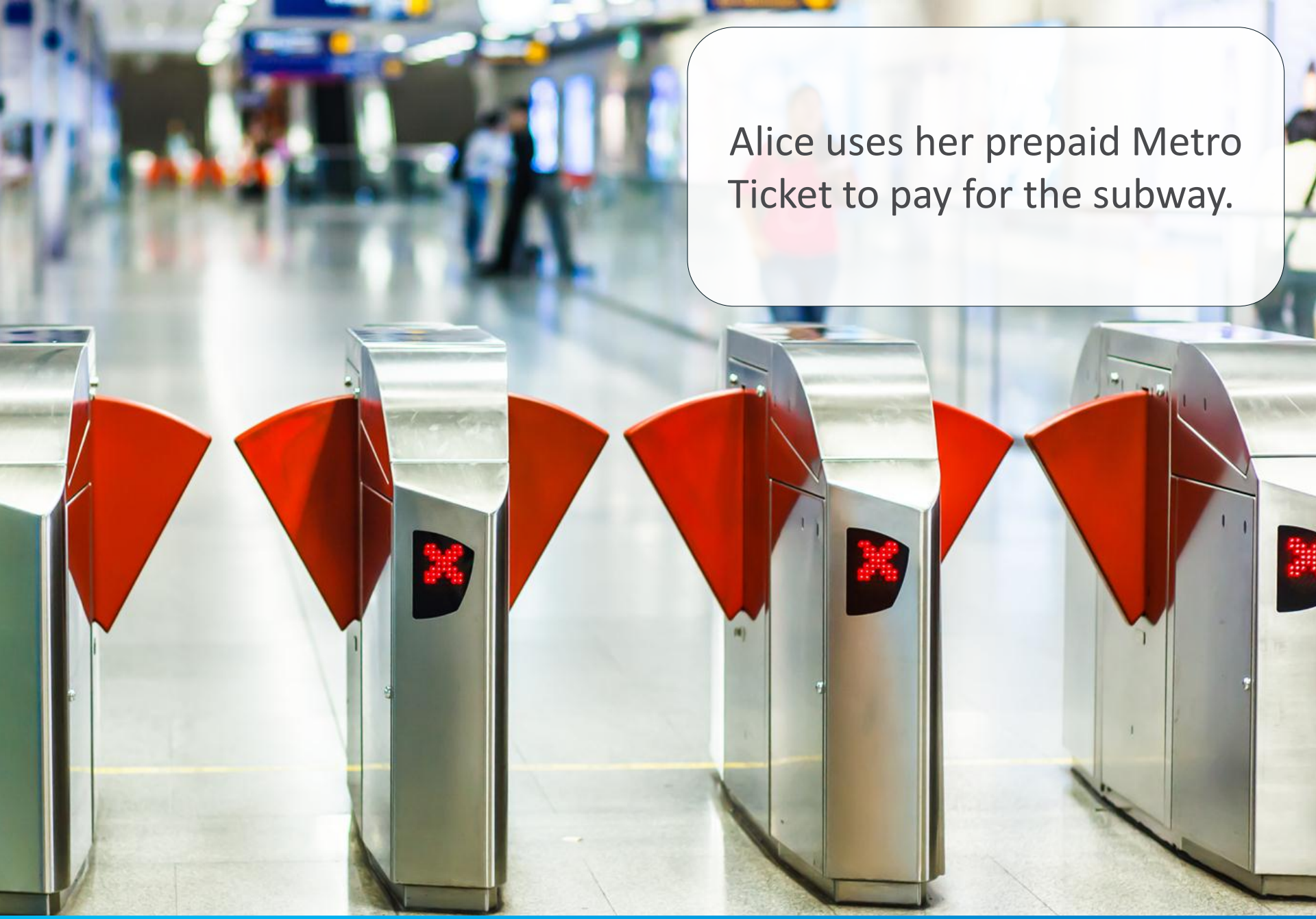
NFC offers a low-speed connection with extremely simple setup, and can be used to bootstrap more capable wireless connections.[16] For example, the Android Beam software uses NFC to complete the steps of enabling, pairing and establishing a Bluetooth connection when doing a file transfer.[17] Nokia, BlackBerry and Sony[18] have used NFC technology to pair Bluetooth headsets, media players, and speakers with one tap in its NFC-enabled devices.[*citation needed*] The same principle can be applied to the configuration of Wi-Fi networks.

- Wikipedia



Alice leaves for her meeting with Bob,
and stops at a local coffee shop. She taps her phone
On the NFC sticker by the door
Which check her in on favorite social network
And displays a coupon for a $1 off a coffee.

Alice then pays by making a contactless wireless payment


Alice uses her prepaid Metro Ticket to pay for the subway.

Social networking

- Exchanging business cards
- Contact information
- Twitter
- Facebook
- Google Plus
- Whatever

Hyundai and other car manufactures are reportedly working on NFC Keyless Car Entry Systems

Send Applications

Not Really. It just sends a link to the Google Play Store.

1. Open the application
2. Touch two NFC Phones
3. Beam the link to the application

Alice and Bob, as all Cryptographers know also share Public/Private or Symmetric Cipher keys.

NFC Contactless payment,
NFC Person to person payment,
NFC Prepaid ticketing,
NFC Proof of Presence,
NFC Proximity Marketing and
NFC access control.

NFC Key Ideas

NFC

NEAR FIELD COMMUNICATION

NFC always involves an initiator and a target

The initiator actively generates an RF field that can power a passive target.

Powers tags, stickers, key fobs, or cards that do not require batteries

NFC tags contain data and are typically read-only, but may be rewriteable.

Short Range, i.e. Proof of Presence

- Mail carriers such as the US Post Office need to verify a package at each point along its trip.
- Passing evident for a court trial must leave a record when it is moved from any location.
- Measure the effectiveness of marketing. An advertiser can count the number of times a people visited a particular store or posted advertisement
- NFC is more accurate then GPS for checking in to a social network.

Privacy and Security

- NFC is off when the Android phone screen is off
- NFC is by design low power, an attach needs to be near.
- There is no encryption of the NFC payloads by default

NFC vs. Bluetooth!

NFC

NEAR FIELD COMMUNICATION

Differences between NFC and Bluetooth

Bluetooth and Wi-Fi seem similar to near field communication on the surface. All three allow wireless communication and data exchange between digital devices like smartphones. Yet near field communication utilizes electromagnetic radio fields while technologies such as Bluetooth and Wi-Fi focus on radio transmissions instead.

- much lower power consumption than Bluetooth
- Even lower than (Bluetooth 4.0) Bluetooth LE
- NFC doesn’t require pairing. Easy setup

Wikipedia Comparison Chart

Aspect	NFC	Bluetooth	Bluetooth Low Energy
RFID compatible	ISO 18000-3	active	active
Standardisation body	ISO/IEC	Bluetooth SIG	Bluetooth SIG
Network Standard	ISO 13157 etc.	IEEE 802.15.1	IEEE 802.15.1
Network Type	Point-to-point	WPAN	WPAN
Cryptography	not with RFID	available	available
Range	< 0.2 m	~100 m (class 1)	~50 m
Frequency	13.56 MHz	2.4–2.5 GHz	2.4–2.5 GHz
Bit rate	424 kbit/s	2.1 Mbit/s	~1.0 Mbit/s
Set-up time	< 0.1 s	< 6 s	< 0.006 s
Power consumption	< 15mA (read)	varies with class	< 15 mA (read and transmit)

NFC vs. QRCodes

- Lower friction
- Reprogrammable after deployment
- NFC triggers an Android Intent that brings you to the correct application for processing the new data. QRCodes require that the user launch and use a bar code scanner.

NFC Standards and Implementations

NFC

NEAR FIELD COMMUNICATION

NFC Types

Passive NFC Forum Tags

- Tag Type 1: Topas™
- Tag Type 2: MiFare Ultralight™
- Tag Type 3: Song Felica™
- Tag Type 4 Mifare Desfire™

Proprietary NFC Tags

- Mifare Classic™

Peer to Peer Devices

- Android to Android

Mifare and NXP

- NXP is the co-inventor of near field communication (NFC) technology along with Sony and supplies NFC chip sets which enable mobile phones to be used to pay for goods, and store and exchange data securely.[9]
- NXP ranks number one in chips for eGovernment applications such as electronic passports; number one in transport and access management, with the chip set and contactless card for MIFARE used by many major public transit systems worldwide; and is number one in RFID tags and labels.[10]
- Source Wikipedia

Mifare Classic Specs

- 30mm diameter
- 124bytes of storage, 752 usable, split into 16 *sectors*
- 2cm distance
- Water proof
- Does not work on metal
- Available with 7 byte unique identifier (7B UID) or 4 byte non-unique identifier (4B NUID)
- 1 KB or 4 KB EEPROM
- Memory access conditions freely programmable

Introducing

Samsung TecTiles™

Get Started with TecTiles!

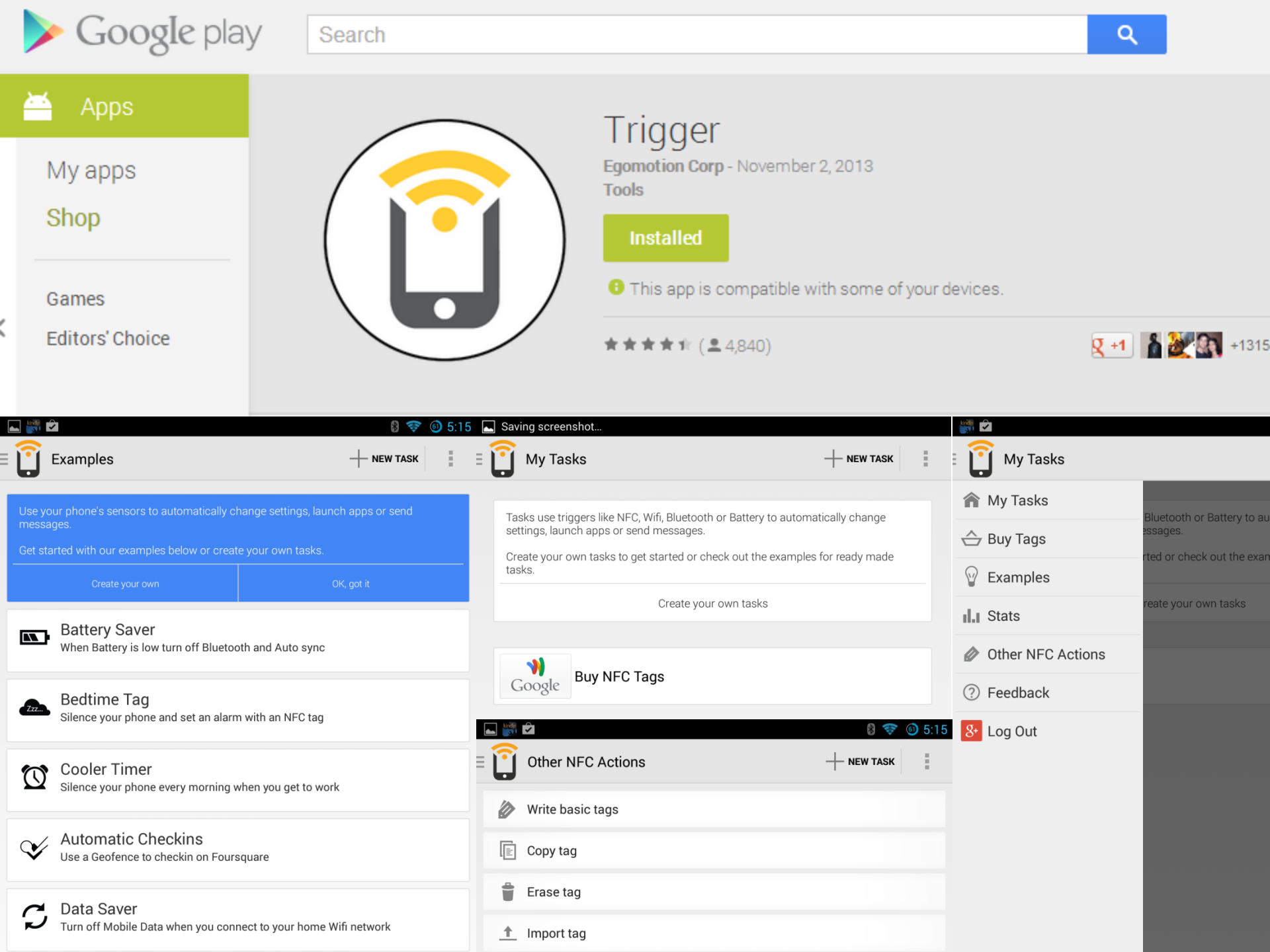
Buy Now ▸

Download the App ▸

NFC Applications on Android

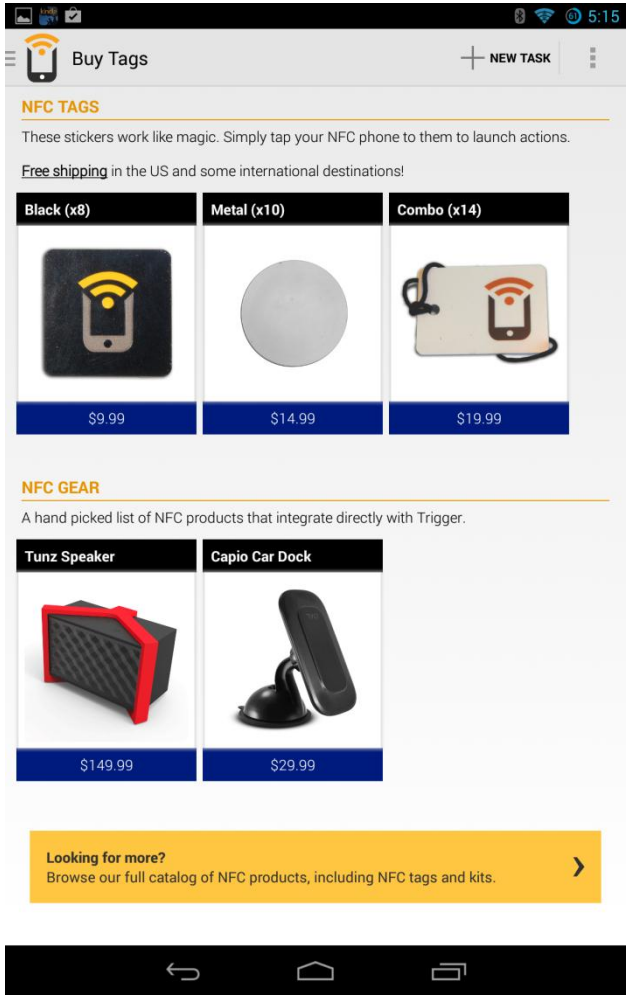
NFC

NEAR FIELD COMMUNICATION


Google play
Search
Apps
My apps
Shop
Games
Editors' Choice
Trigger
Egomotion Corp - November 2, 2013
Tools
Installed
This app is compatible with some of your devices.
(4,840)
+1
+1315
5:15
Saving screenshot...
Examples
NEW TASK
Use your phone's sensors to automatically change settings, launch apps or send messages.
Get started with our examples below or create your own tasks.
Create your own
OK, got it
Battery Saver
When Battery is low turn off Bluetooth and Auto sync
Bedtime Tag
Silence your phone and set an alarm with an NFC tag
Cooler Timer
Silence your phone every morning when you get to work
Automatic Checkins
Use a Geofence to checkin on Foursquare
Data Saver
Turn off Mobile Data when you connect to your home Wifi network
My Tasks
NEW TASK
Tasks use triggers like NFC, Wifi, Bluetooth or Battery to automatically change settings, launch apps or send messages.
Create your own tasks to get started or check out the examples for ready made tasks.
Create your own tasks
Google
Buy NFC Tags
5:15
Other NFC Actions
NEW TASK
Write basic tags
Copy tag
Erase tag
Import tag
My Tasks
My Tasks
Buy Tags
Examples
Stats
Other NFC Actions
Feedback
Log Out

Trigger will help determine the type of Tag you should Buy

- Setup you Actions that will be triggered when you scan an NFC tag.
- The Trigger Application will tell you how many bytes you need in your tag to store the commands
- Then it tells you which Tags will store than amount
- And it links you to sites where you can purchase the correct tags.



Android Beam

Android Beam

With Galaxy Nexus you can now easily share contacts, websites, apps, maps, directions and YouTube videos to other people close by. Simply hold two NFC enabled Android phones close to each other and touch to beam and share.

Android NFC Programming

NFC

NEAR FIELD COMMUNICATION

Android.nfc package

Classes available

- Tag: Represents a discovered NFC Tag
- NfcAdapter: Represents the NFC adapter of the mobile device
- NfcManager: Gets an instance of the NFC adapter
- NdefMessage: Represents an NDEF message
- NdefRecord: Represents an NDEF record
- NfcEvent: Information about an NFC Event

Android.nfc.tech package

The Android.nfc.technology package exposes classes to handle specific NFC technologies.

- iSODep
- NfcA
- NfcB
- NfcF
- NfcV
- Ndef
- Mifare Classic
- Mifare Ultralight

Receiving NDEF messages an detecting NFC tags

Ndef stands for NFC Data Exchange Format

When a tag is scanned there are three different options:

1. ACTION_NDEF_DISCOVERED - The tag contains NDEF payload that can be mapped to a MIME type or URI

2. ACTION_TECH_DISCOVERED - The tag contains NDEF payload that cannot be mapped to a MIME type or URI

3. ACTION_TAG_DISCOVERED - The tag does not contain NDEF payload but is of a known tag type

Android* NFC application manifest

- First ask permission for use of NFC:

```
<uses-permission android:name="android.permission.NFC" />
```

- NFC is well supported starting with API level 10:

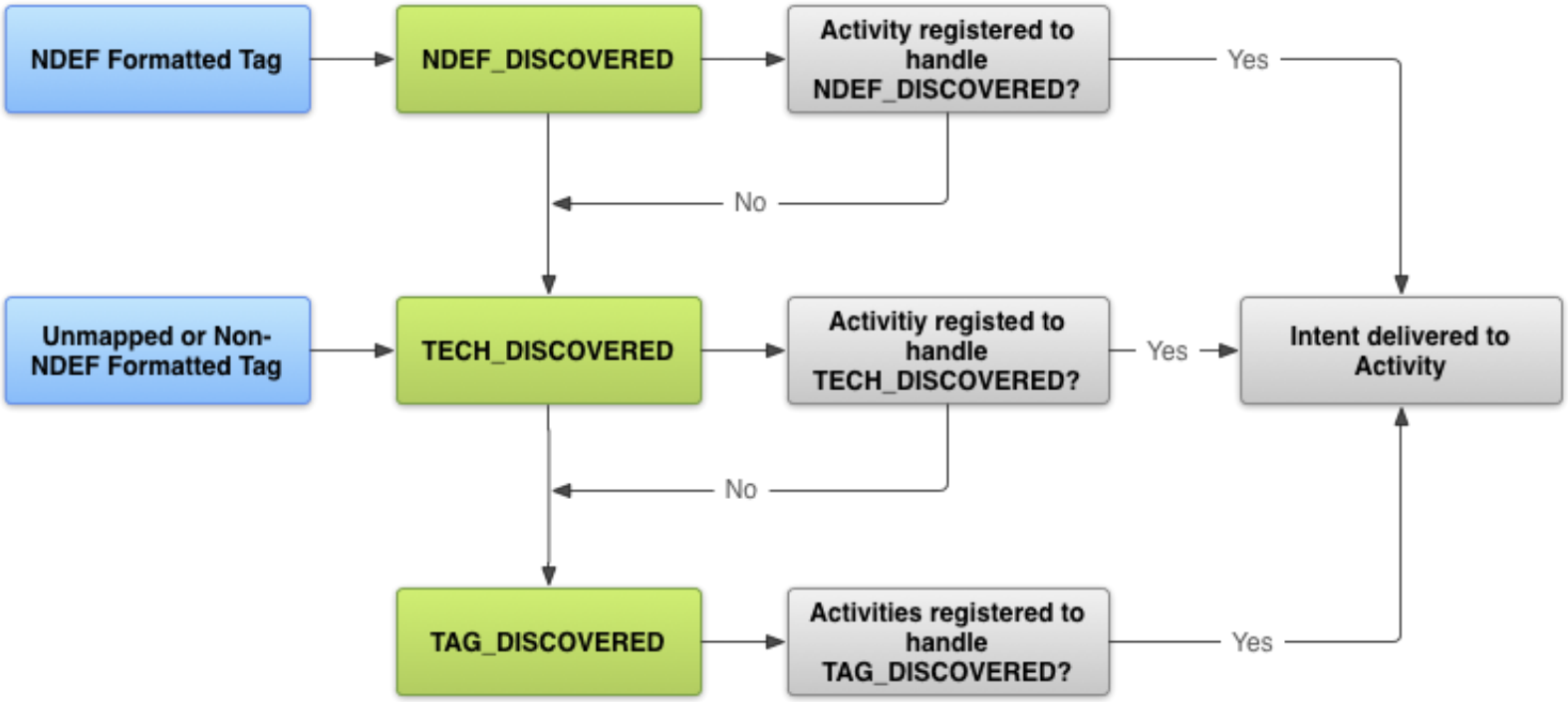
```
<uses-sdk android:minSdkVersion="10"/>
```

  Android* Beam -> level 14
  Android* Beam + BT/Wifi-Direct -> level 16

- For the application to be available only to NFC-enabled devices:

```
<uses-feature android:name="android.hardware.nfc"
android:required="true" />
```

NFC Intent propagation



There are two ways to register an activity to an intent:

- From the manifest
- From the on-screen activity, using foreground dispatch

When several activities are registered, priority is given to the one registered through foreground dispatch, then to the one with whose an AAR is associated to the NDEF message.

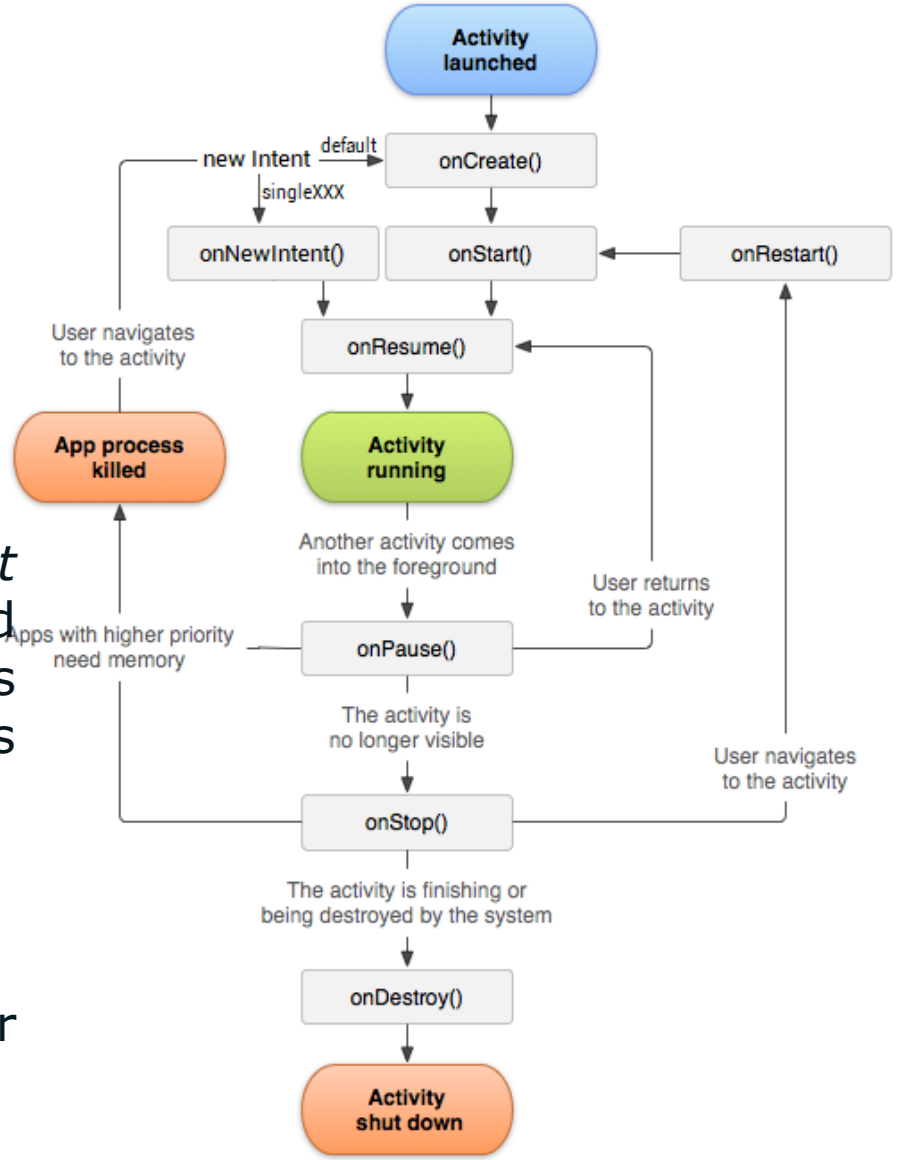
NFC Intent handling

```
<activity …
  android:launchMode=
     [ "default"
     | "singleTop"
     | "singleTask"
     | "singleInstance" ] …
```

Using another mode than *default* will avoid the activity to be launched again each time an intent is received and to fill up the activities stack.

Use precise intent filters

Don't send bulky data, XML or sockets over NFC



NFC Intent handling

```
@Override
public void onCreate(Bundle savedInstanceState) {
        …
        resolveIntent(getIntent());
}

@Override
public void onNewIntent(Intent intent) {
        super.onNewIntent(intent);
        setIntent(intent);
        resolveIntent(intent);
}

private void resolveIntent(Intent intent) {
        if(NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())){
                ... //handle intent
        }
}
```

Getting Available Tag Technologies

In order to connect to connect to a tag, you need to know the type of the tag. If you wish to get the available tag types that a tag owns, use the getTechList() method as shown in the following code:

```
if (NfcAdapter.ACTION_TECH_DISCOVERED.equals(getIntent(). getAction())) {
    Tag tag = getIntent().getParcelableExtra(NfcAdapter.EXTRA_TAG);
    myText.setText(" Technologies available in this tag =" +
                        Arrays.toString(tag.getTechList()) );
}
```

After getting the supported tag technologies of a tag, you may connect to tag one of these technologies. For example, if a scanned tag supports NfcV and NdefFormatable, then you can get the tag with either of the classes (android.nfc.tech.NfcV or android.nfc.tech.NdefFormatable) and perform the required I/ O operations.

Checking for the NFC Adapter

Checking for the NFC Adapter

```
@Override
public void onCreate( Bundle savedInstanceState) {
    super.onCreate( savedInstanceState);
    setContentView( R.layout.main);
    myText = (TextView) findViewById( R.id.myText);
    myNfcAdapter = NfcAdapter.getDefaultAdapter(this);
    if (myNfcAdapter = = null)
            myText.setText(" NFC is not available for the device!!!");
    else
            myText.setText(" NFC is available for the device");
}
```

Reading NFC tags and NDEF messages

Reading NFC tags and NDEF messages

Reading an NDEF message from a tag or sent from an active device: same implementation.

3 steps:

1. Subscribe to the NDEF message dispatch
2. Get the content of the NDEF message
3. Read the content of the NDEF message

1. Subscribing to the NDEF message dispatch

Subscription done for each activity, usually done in the manifest.

Different filters can be handled.

Example 1 – plain text :

```
<intent-filter>
    <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:mimeType="text/plain" />
</intent-filter>
```

2. Subscribing to the NDEF message dispatch

Example 2 – URI ("http://developer.android.com/index.html") :

```
<intent-filter>
    <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
    <category android:name="android.intent.category.DEFAULT"/>
   <data android:scheme="http"
            android:host="developer.android.com"
            android:pathPrefix="/index.html" />
</intent-filter>
```

Example 3 - TNF_EXTERNAL_TYPE ("com.example:externalType") :

```
<intent-filter>
    <action android:name="android.nfc.action.NDEF_DISCOVERED" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:scheme="vnd.android.nfc"
        android:host="ext"
        android:pathPrefix="/com.intel.ssg:myexttype"/>
</intent-filter>
```

2. Getting NDEF message content

```
@Override
public void OnCreate(Bundle savedInstanceState) {
        …
        resolveIntent(intent);
}

@Override
public void onNewIntent(Intent intent) {
        super.onNewIntent(intent);
        //setIntent(intent);
        resolveIntent(intent);
}

private void resolveIntent(Intent intent) {
  if(NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())){
        Parcelable[] rawMsgs =
intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
        if(rawMsgs != null && rawMsgs.length > 0) {
            NdefMessage message = (NdefMessage) rawMsgs[0];
            readNdefRecords(message.getRecords());
        }
  }
}

private void readNdefRecords(NdefRecord[] records ) {
         … //handle NDEF records
}
```

3. Reading NDEF message content

API Level >= 9:

```
short NdefRecord.getTnf() & byte[] NdefRecord.getType()
byte[] NdefRecord.getPayload();
```

API level >= 16:

```
Uri NdefRecord.toUri();
String NdefRecord.toMimeType();
```

We would have to wait for Jelly Bean to be able to get easily an URI from a tag ?

Tips: use Uri intent.getData()

3. Reading NDEF message content - example

```
…
Uri intentUri = intent.getData();
…
   NdefMessage message = (NdefMessage) rawMsgs[0];
   try {
     textView.setText(new
String(message.getRecords()[0].getPayload(), "UTF-8"));
   }
   catch (UnsupportedEncodingException e) {
     e.printStackTrace();
   }
}

…
```

Formatting NFC tags and writing NDEF messages

1. Subscribing to the ACTION_TECH_DISCOVERED intent

We generally use the foreground dispatch:

```
public void OnCreate(){
  NfcManager manager = (NfcManager) getSystemService(Context.NFC_SERVICE);
  mNfcAdapter = manager.getDefaultAdapter();

  mPendingIntent = PendingIntent.getActivity(getApplicationContext(), 0, new
        Intent(this, getClass()).addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP), 0);

  mIntentFiltersArray = new IntentFilter[]{ new
        IntentFilter(NfcAdapter.ACTION_TECH_DISCOVERED) };
  mTechListsArray = new String[][]{new String[]{NdefFormatable.class.getName()},
                                   new String[]{ Ndef.class.getName() }};
}
protected void OnResume(){
  super.OnResume();
  mNfcAdapter.enableForegroundDispatch(this, mPendingIntent, mIntentFiltersArray,
mTechListsArray);
}
protected void OnPause(){
  super.OnPause();
  mNfcAdapter.disableForegroundDispatch(this);
}
public void onNewIntent(Intent intent) {
  super.onNewIntent(intent);
  setIntent(intent);
  resolveIntent(intent);
}
```

Evolution of NDEF Content Classes

API Level >= 9:

- `new NdefMessage(byte[]);`
- `new NdefMessage(new NdefRecord[]{new NdefRecord(short tnf, byte[] type, byte[] id, byte[] payload)});`

API level >= 14:

- `new NdefMessage(new NdefRecord[]{ NdefRecord.createUri(…) });`
- `new NdefMessage(new NdefRecord[]{ NdefRecord.createUri(…), NdefRecord.createApplicationRecord(…) });`

API level >= 16:

- `new NdefMessage(NdefRecord.createMime(…));`
- `new NdefMessage(NdefRecord.createExternal(…));`

3. Writing the NDEF message

```
NdefMessage ndefMessage; // previously created ndef message.
final Tag tag =
intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);

Ndef ndefTag = Ndef.get(tag); // if tag is already formatted
if (ndefTag != null) {
    ndefTag.connect(); // throws IOException
    if (ndefTag.isWritable() &&
NdefMessage.toByteArray().length <= ndefTag.getMaxSize())
            ndef.writeNdefMessage(ndefMessage); // throws
TagLostException, FormatException, IOException
        ndefTag.close(); // throws IOException
    }
```

Libraries

http://code.google.com/p/ismb-snep-java/ et http://www.grundid.de/nfc/ - SNEP data exchange between a PC NFC device and Android*

http://mobisocial.github.com/EasyNFC/apidocs/reference/mobisocial/nfc/Nfc.html - Facilitate NFC use under Android*

http://code.google.com/p/ndef-tools-for-android/ - Manipulate NDEF messages more easily

Tools

ndefeditor.com: create NDEF messages online

https://code.google.com/p/nfc-eclipse-plugin/: same through an eclipse plugin

http://play.google.com/store/apps/details?id=com.antares.nfc – write the NDEF messages built with ndefeditor using an Android* phone

https://play.google.com/store/apps/details?id=com.nxp.nfc.tagwriter – manage/write NFC tags

https://play.google.com/store/apps/details?id=com.nxp.taginfolite – Analyze NFC tags

Resources

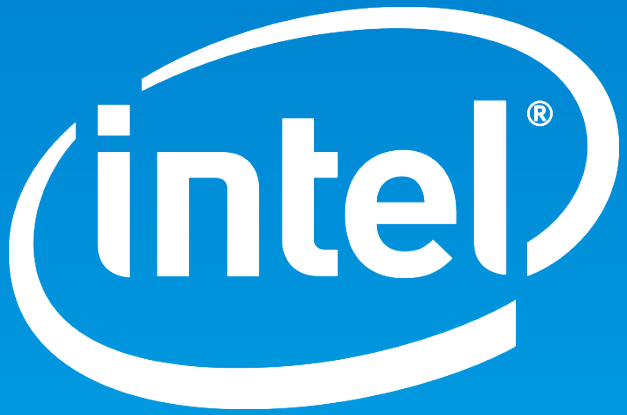
http://www.tappednfc.com/android-nfc-developer-guide/ - Android* NFC guide

http://blog.zenika.com/index.php?tag/nfc - NFC I-V (french)

Daniel Holmlund

Daniel.W.Holmlund@intel.com

Twitter: @agnathan



Thank You!

Backup

Subscribing to NDEF_DISCOVERED using foreground dispatch

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        NfcManager manager = (NfcManager) getSystemService(Context.NFC_SERVICE);
        mNfcAdapter = manager.getDefaultAdapter();
        mPendingIntent = PendingIntent.getActivity(getApplicationContext(), 0, new
Intent(this, getClass()).addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP), 0);
        IntentFilter intentFilter = new
IntentFilter(NfcAdapter.ACTION_NDEF_DISCOVERED);
        intentFilter.addDataScheme("http");
        intentFilter.addDataAuthority("software.intel.com", null);
        intentFilter.addDataPath(".*", PatternMatcher.PATTERN_SIMPLE_GLOB);
        mIntentFiltersArray = new IntentFilter[]{ intentFilter };
        resolveIntent(getIntent());
}
protected void onPause() {
        super.onPause();
        mNfcAdapter.disableForegroundDispatch(this);
}
protected void onResume() {
        super.onResume();
        mNfcAdapter.enableForegroundDispatch(this, mPendingIntent,
mIntentFiltersArray, null);
        };
```

Peer-to-peer communication

Peer-to-peer communication

Reminder of peer-to-peer communication standards:

- LLCP channel
- SNEP or NPP protocols
- NDEF messages exchange

Android* Beam

- Work on top of LLCP and SNEP (since ICS 4.0.3)
- Only send one NDEF message
- Requires user to press on the interface
- By default: Sends current application AAR

Sending an NDEF (setNdefPushMessage[Callback])

API level 14 (ICS)

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
    if (nfcAdapter == null) return;  // NFC not available
    nfcAdapter.setNdefPushMessage(ndefMessage, this);
// or nfcAdapter.setNdefPushMessageCallback(callback,
        this);
}
```

Sending files (setBeamPushUris[Callback])

API level 16 (JB)

Transfer initiated using NFC

Then a more adapted transport is used:

- Bluetooth®
- Wi-Fi direct™

Takes in arguments URIs with these schemes:

- file://
- content://

Hands-On !

Hands-On – Android* Beam

1. Create a new Android* app with NFC permission
2. Use setNdefPushMessage or setNdefPushMessageCallback to send an URI myscheme://myapp/id where id is entered by the user.
3. Add an AAR to the sent NDEF message

P2P Writing

Foreground Activities can register and NDEF payload for P2P push.

The receiving side can handles an NDEF payload as though it is reading a passive tag.

Data Range (1-4cm)

Low Data Rate (106-414 kbps)

Some NFC Devices and Tags are not like the Others

Mifare vs NTAG203

Hands-On – Writing an URI to a NFC Tag

1. Write a tag with that content: myscheme://myapp/id using ndefeditor or TagWriter
2. Create a new Android* application
3. Add the NFC permission to the manifest
4. Add an intent-filter for myscheme://myapp/id where id can vary
5. Read the end of the URI (id) to display it