NFC Development

with Qt on Symbian and MeeGo

Andreas Jakl [@mopius]
Senior Technical Consultant
Nokia

NFC Development on the Windows 8 Platform
bit.ly/Win8NFC

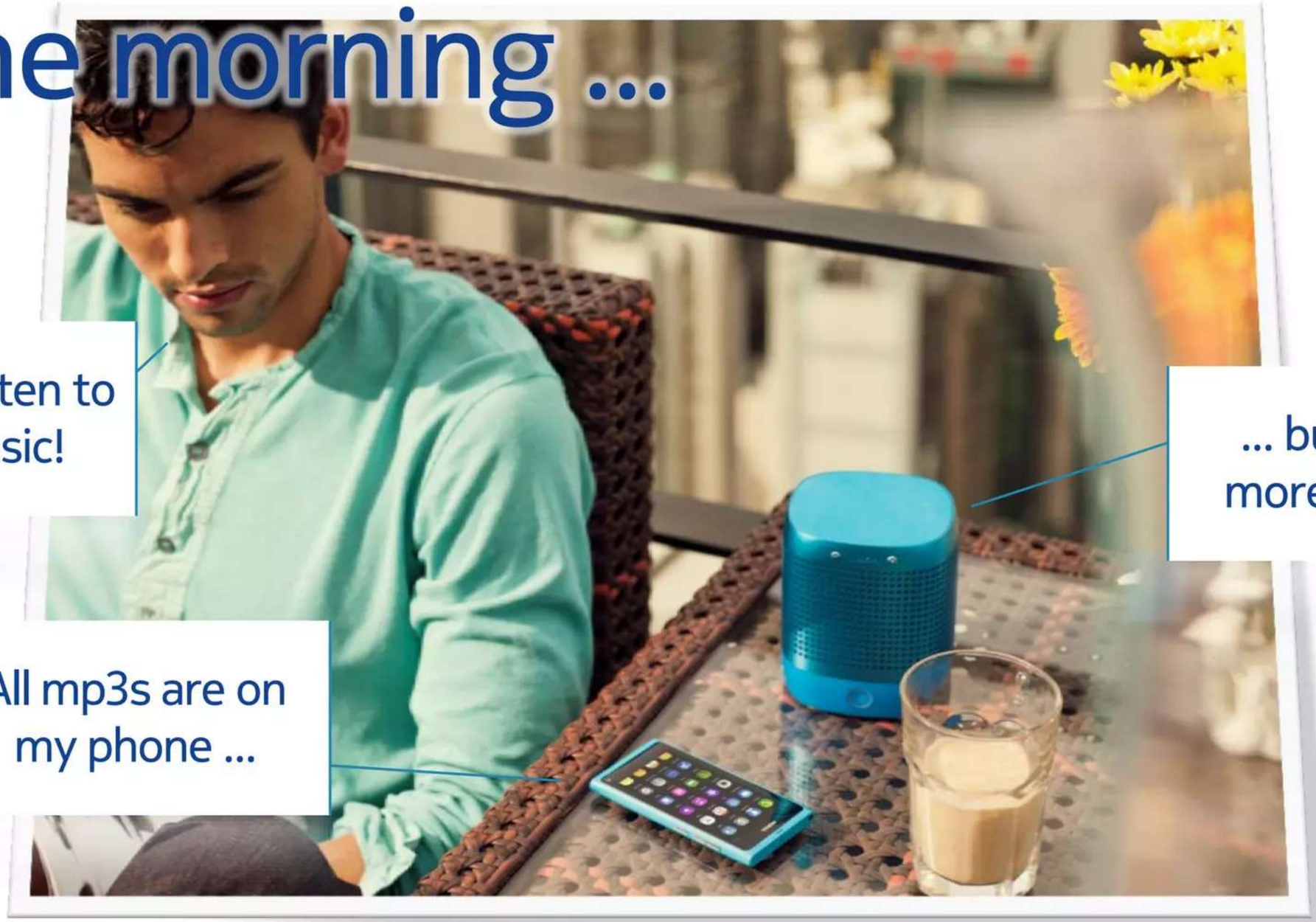
NOKIA
Connecting People

http://www.nokia.com/nfc

A day in your life **with NFC**

NOKIA
Connecting People

In the morning ...



NOKIA

Play it loud

nfc – tap to pair & play!

Nokia Play 360

Wireless Bluetooth connection

 Nokia NFC Development v2.1.2 May 8, 2012 Andreas Jakl

NOKIA

Near Field Communication?

Tap a device or a tag: easy and intuitive gesture

Integrate proximity into local and social interactions

NFC = Wireless connectivity technology

Short distance: 0 - 4 cm

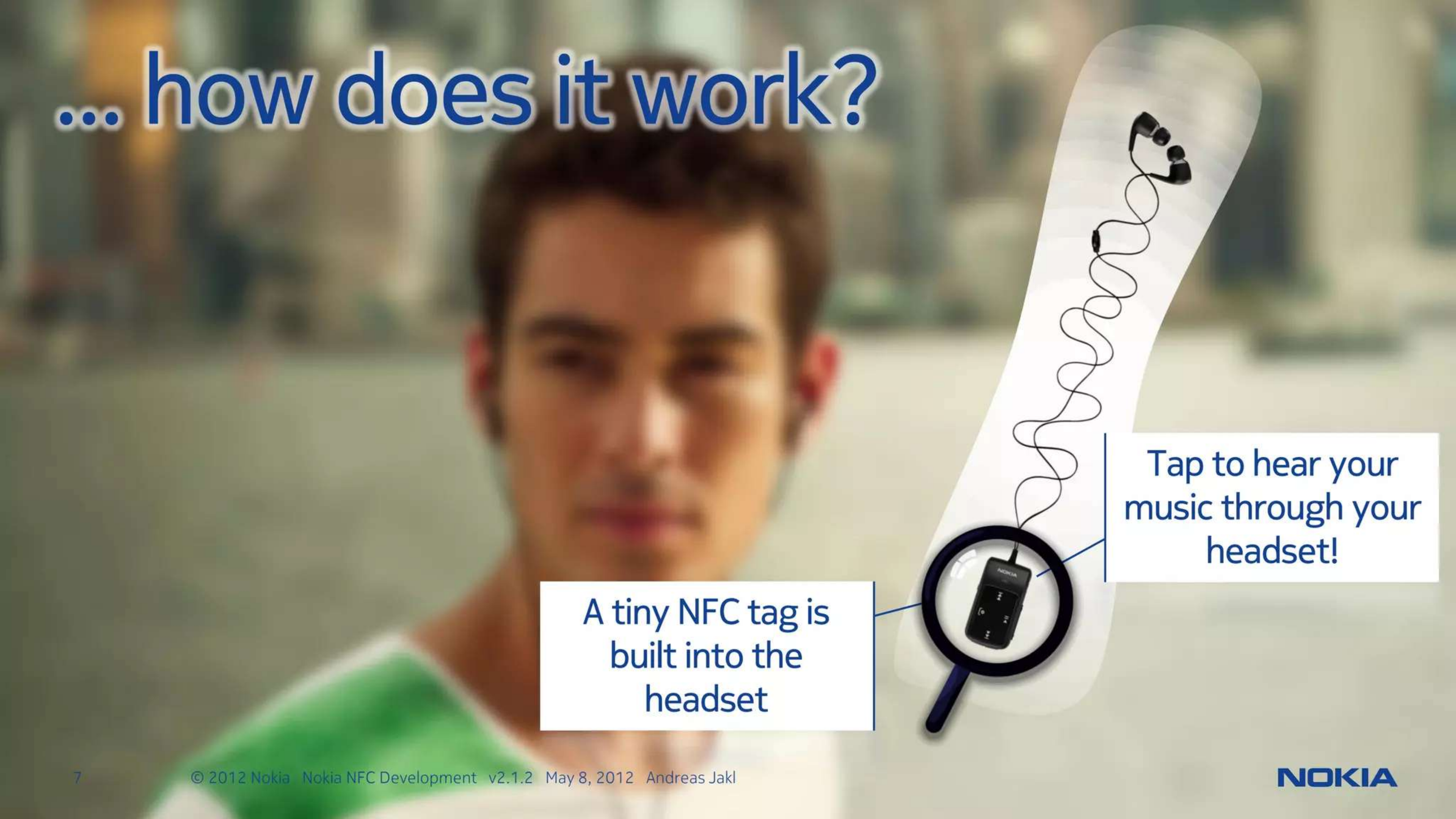
13.56 MHz, up to 424Kbits/sec (slow!)

Take the music with you ...

Tap to hear your music through your headset!

NOKIA

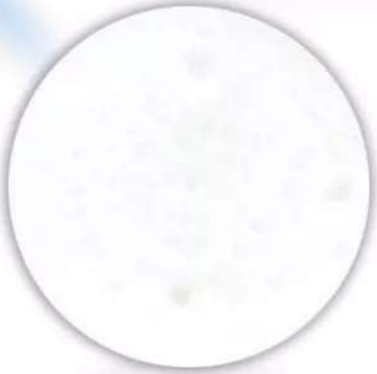
... how does it work?



NOKIA

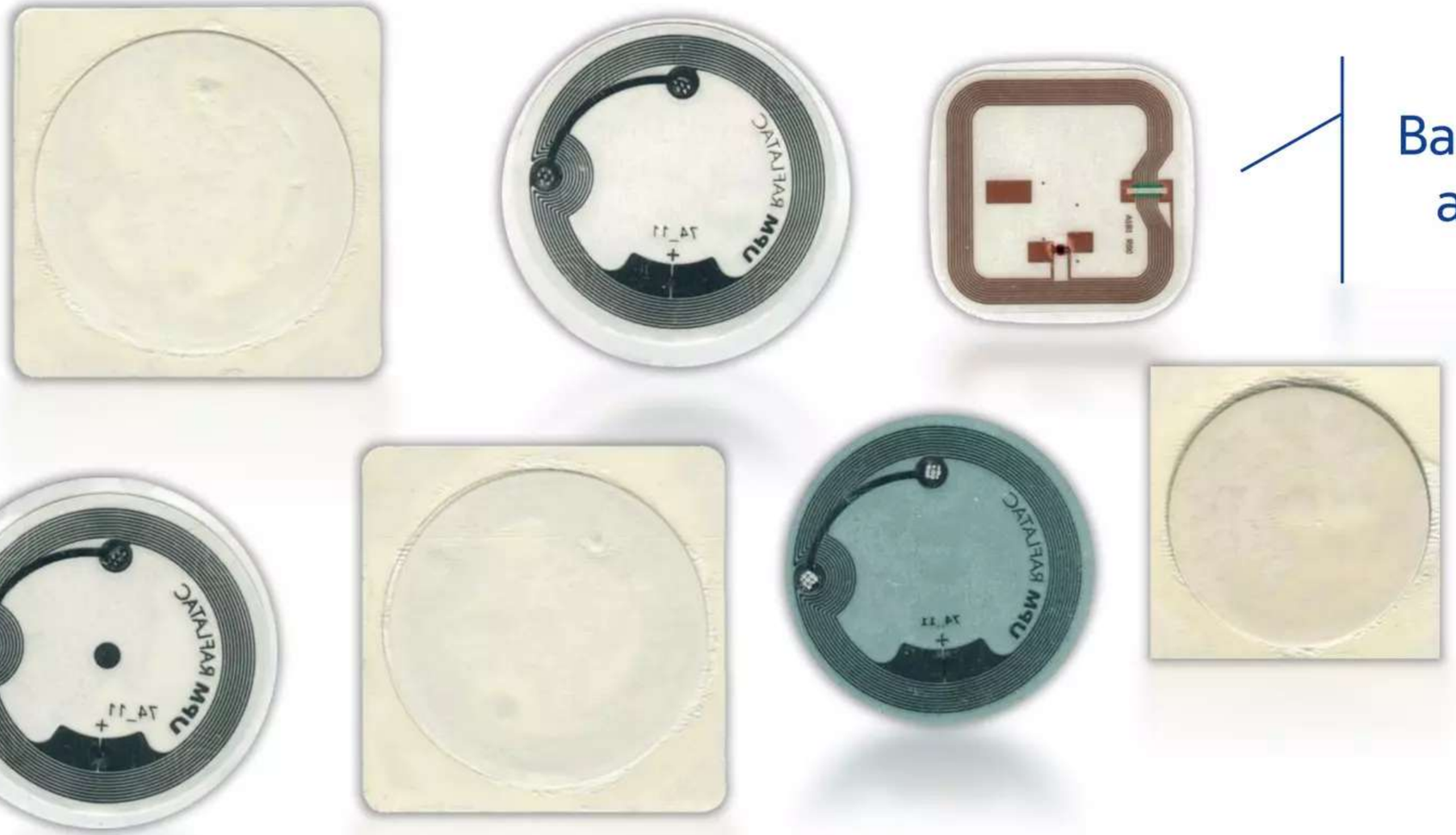
What do NFC tags look like?

Standardized:
NFC Forum Type 1 – 4
Also common: Mifare



Tag embedded in a credit-card style event badge

What do NFC tags look like?



Backside reveals the antenna and chip

Upper Austria University of Applied Sciences
NFC Research Lab Hagenberg

www.nfc-research.at · lab@nfc-research.at

Social Network Check In

With an NFC tag placed in the bar, it's a matter of seconds, even indoors!

What do you need to store on the tag to make this work?

NOKIA

Store Data on a Tag

Your app not yet installed?
Download it!

Standardized URL record

http://store.nokia.com/content/184295

Custom application record

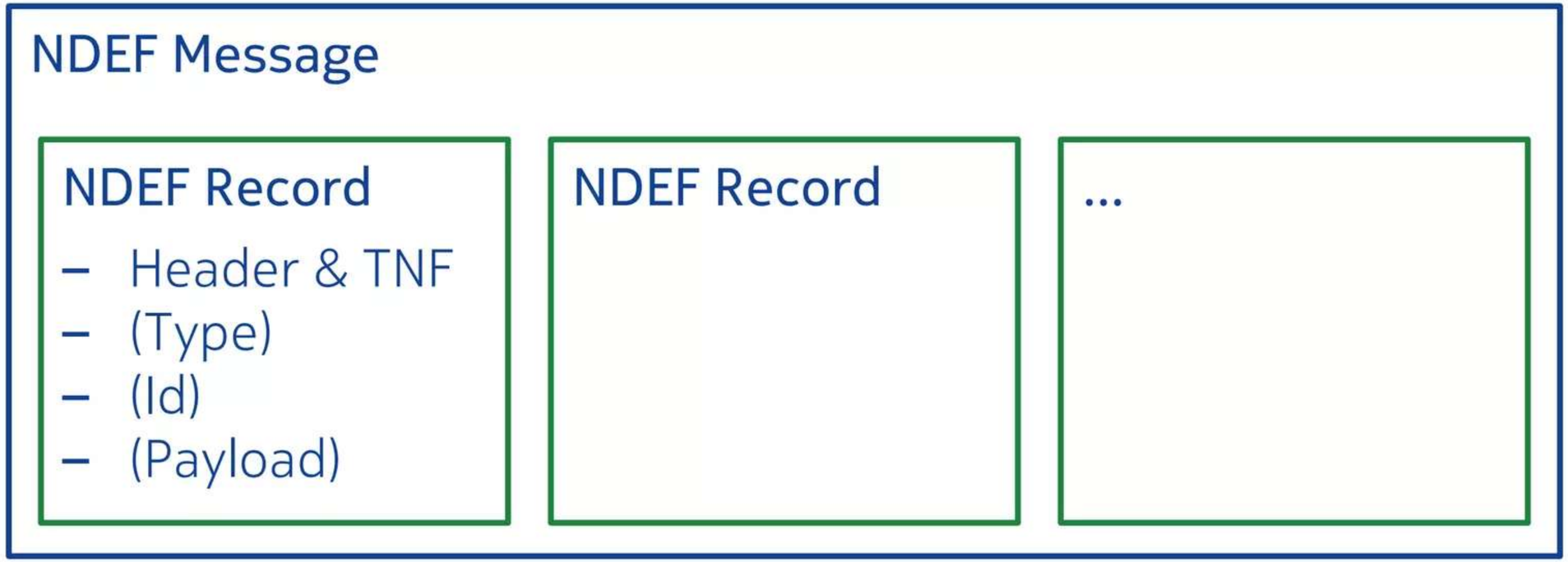
placeName=Schlabo's Bar;id=72XLPM3

Data	**Encapsulated in**	**Encoded through**	**Stored on**
NDEF Records	NDEF Message	NFC Forum Tag Type Platform	NFC Forum Tag

App already installed?
Read data and check in!

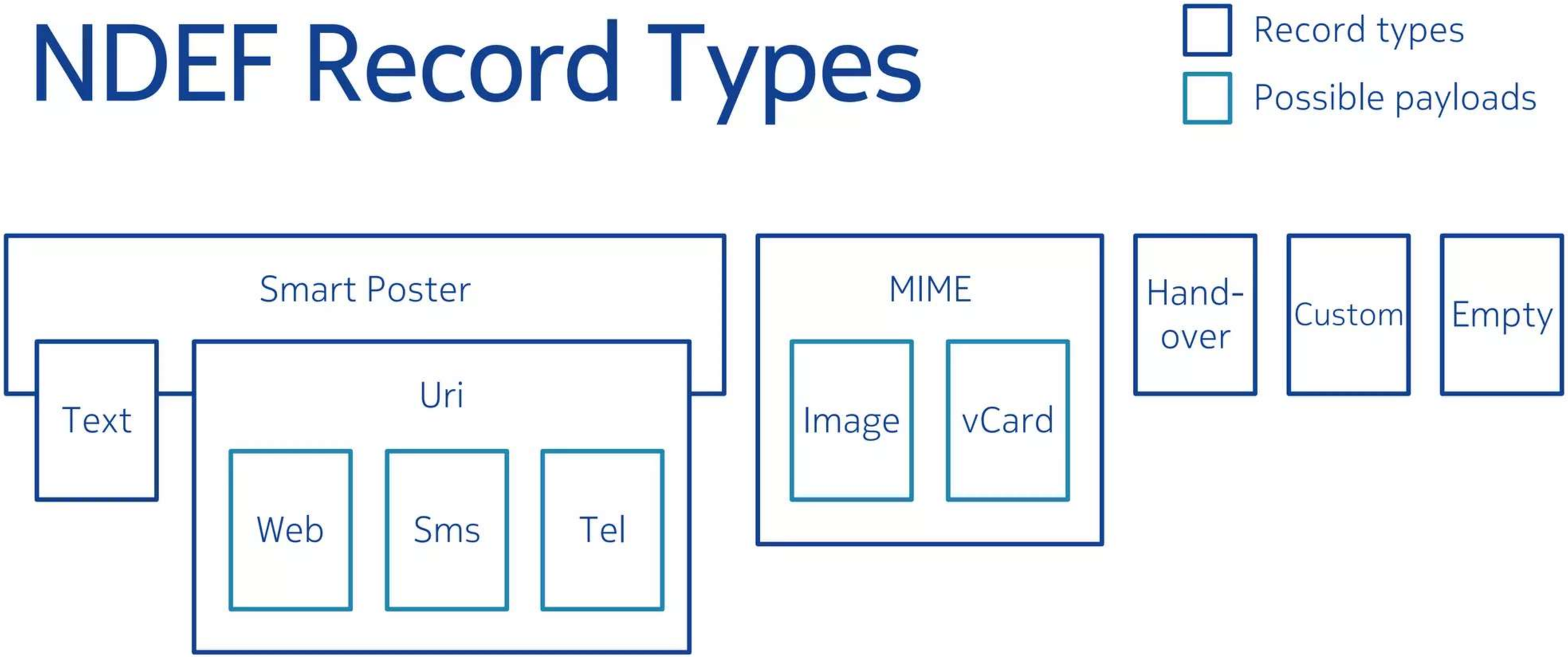
NOKIA

NDEF Messages & Records



NDEF = NFC Data Exchange Format, TNF = Type Name Format

NDEF Record Types



Public Transport

NOKIA

Public Transport

… or buy the ticket!

Either read through endless schedules and price lists…

… or just tap and see the next relevant connections

Similar possibilities w/ 2D Barcodes?

Combine?
Print 2D barcode on NFC tag!

- Works by touching, instant
- Design can be merged with product
- Larger data storage possible
- Multiple use cases (app launch, BT pairing)
- Re-writable (if desired)

Opposite

- Requires NFC HW
- Educate users

While Waiting: Unlock Game Content

Angry Birds Magic

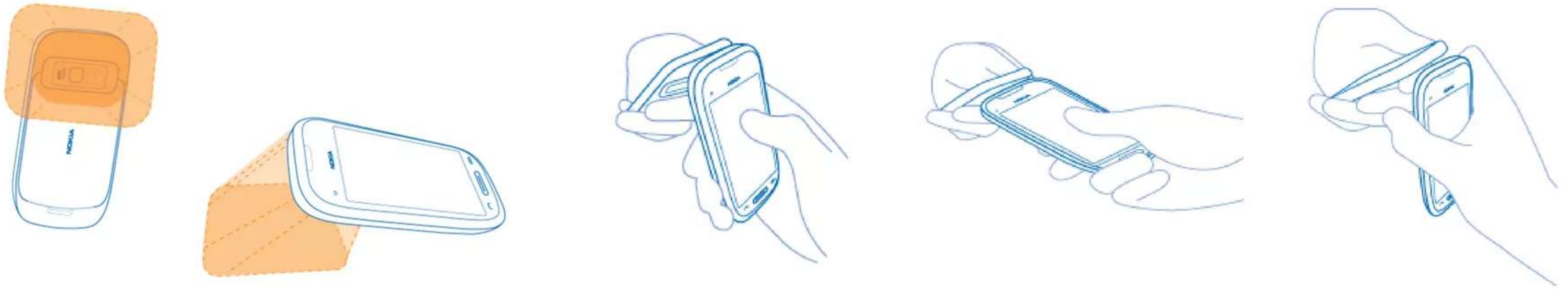
- First 5 levels available

Unlock more levels

- Touch the phone of your friend!

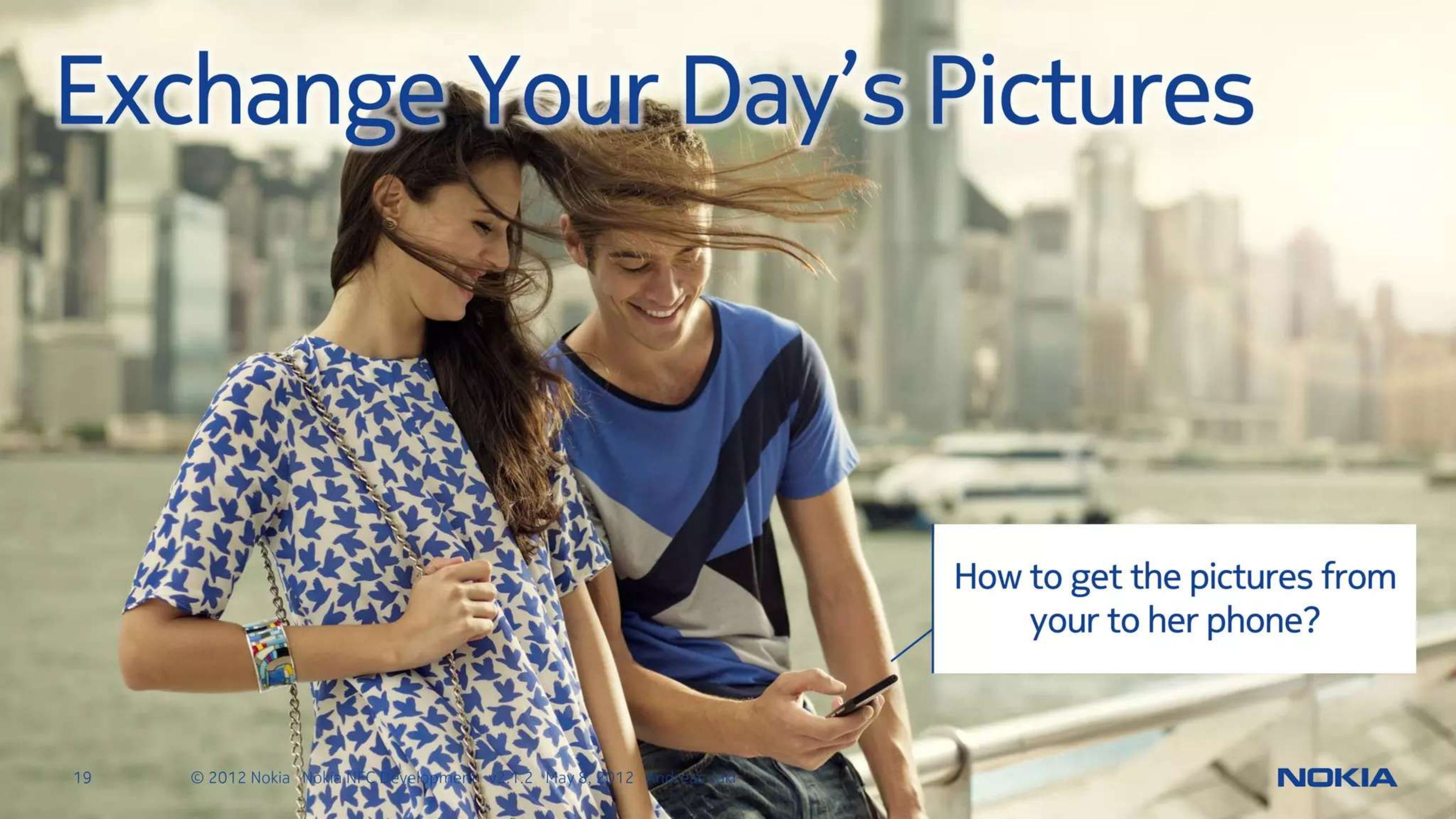
How to Touch?

- Touching with a natural NFC antenna placement
  - Give instant feedback in your app!



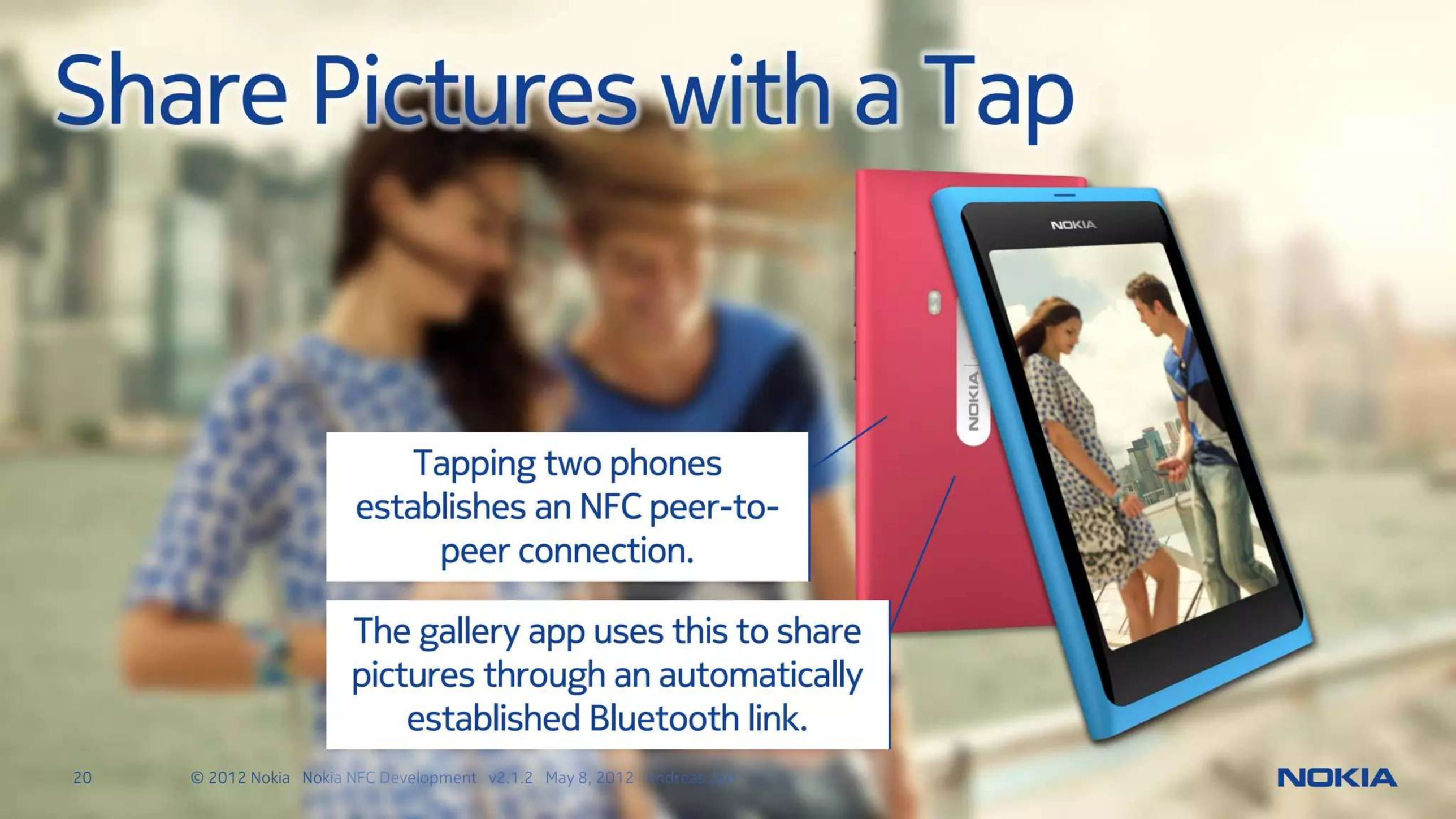
http://www.developer.nokia.com/Resources/Library/Design_and_UX/designing-for-nokia-devices/interaction-design/designing-nfc-applications.html

Exchange Your Day's Pictures



How to get the pictures from your to her phone?

Share Pictures with a Tap



Tapping two phones establishes an NFC peer-to-peer connection.

The gallery app uses this to share pictures through an automatically established Bluetooth link.

NOKIA

Peer-to-peer

- Touch two phones to …
- … establish instant peer-to-peer connection
  - LLCP (Logical Link Control Protocol)
  - Direct communication via sockets
- Content transfer
  - Slow (< 424 kBit/s, Bluetooth: < 3 MBit /s))
  - Only while touching

  → great to exchange business card or unlock game items

  → transfer more data (e.g., images)?

  Use NFC to exchange Bluetooth / Wifi handover information

NFC App Ideas

Put tag into your merchandise to unlock bonus content in your game

Museum or tourist guide: touch tags to get more info*

Treasure hunt game with tags spread around the city

Augment an interactive multimedia installation with touch

Social networking app: touch friend's phone to connect

Concert app: provide video and music samples of the band when touching a poster

Virtual message board: people post short texts at specific, real-world places

Farm game: touch friend's phone to exchange virtual sheep or money

Multiplayer board game: touch friend's phone to switch turns

Call a taxi to the office by touching a tag **

* http://conversations.nokia.com/2011/09/26/byte-at-the-museum/

** http://conversations.nokia.com/2011/10/11/nokia-and-nfc-create-world%e2%80%99s-first-taxi-booking/

NFC Standards

- NFC Forum standardizes technology
  - Members: Nokia, Microsoft, Google, Research in Motion, Samsung, MasterCard, NXP, Skidata and many more*
- **Compatible phones, no matter which brand, can:**
  - Interact with NFC Forum Type 1 – 4 tags
  - Read / write NDEF messages and common record types
  - Communicate between devices using LLCP
- **More Information & technical specifications**
  - http://www.nfc-forum.org/

* see: http://www.nfc-forum.org/member_companies/

Implementation Overview

NFC phones have different operating modes:

- **Reader/writer:** phone is polling for a passive NFC tag
- **Peer to Peer:** communicate with another NFC device
- **Card Emulation:** NFC modem polls for another NFC device and establishes a data connection *

NFC is deactivated after certain device idle time. Wake up polling, e.g., by tapping the display.

* supported in Nokia 603 and 808

Summary – Nfc Experiences

Service Initiation

Start a service
or an application

Reader/writer mode

Sharing

Transfer content
between any
two NFC devices

Peer to peer mode

Pairing

Connect easily
with other NFC
devices

Reader mode

Secure NFC

Mobile device
is a credit card
and travel card

Card emulation mode

Extensive experience in NFC globally

- First to bring commercial NFC phones to the market
- More than 80 trials and pilots globally (AT&T, O2, Citibank, Maxis, Visa, MasterCard)
- First to deploy commercial services (Austria, Malaysia, India)

2005	2007	2008	2011	2012
Nokia 5140i, 3220	Nokia 6131 NFC	Nokia 6212 classic	Nokia C7 / Astound / Oro, 603, 700, 701, 801T (Symbian) Nokia N9 (MeeGo Harmattan)	Nokia 808 PureView, Lumia 610 NFC

NFC Development

NFC Development Alternatives

Java ME

Works on Series 40 and Symbian devices.

Limited use of smartphone features.

Qt Mobility

Cross-platform API.

Best flexibility and ease of app development.

Symbian Native

Low level control over devices' NFC support.

More complex, platform knowledge required.

Windows Phone

Nokia-specific APIs – restricted publishing

http://bit.ly/WpNfc

NFC Developer Offering

Tools

Qt Development

Qt SDK 1.2
Qt Mobility 1.2

Symbian Development

Symbian Belle SDK

Java ME development

Nokia 6212 Classic NFC SDK or
Symbian Belle SDK

Developer HW

NFC Device KIT *

- C7 with Symbian Anna
- Sample NFC Tags

Channel

Nokia Store
NFC Collection

* available to Nokia Developer Launchpad / PRO company members

www.developer.nokia.com/NFC

Java ME NFC Development

NetBeans

Both free IDEs come with extensive, generic Java ME support on board.

Extend with Nokia 6212 Classic NFC SDK (or Symbian Belle SDK)*

* http://www.developer.nokia.com/Community/Wiki/Developing_NFC_Applications#Java_ME_based_development

Contactless Communication API

- JSR 257
  - Tag discovery and data exchange
  - NDEF messages
- **Example project: Nfc Creator**
  - https://projects.developer.nokia.com/nfccreator

Symbian Platform NFC API

- Exposes NFC middleware at the platform level
  - Aimed at Symbian platform developers
  - Platform specific coding/skills needed
  - Low abstraction
- Development
  - Carbide.c++ IDE
  - Symbian Belle SDK
  - Optional: Qt for Symbian

Qt Mobility NFC

What is Qt?

Using Qt SDK for
- Dev. environment
- Testing
- Translation
- Help

Publish to
- Nokia Store (Symbian, MeeGo)
- Windows
- Linux
- Mac
- Embedded

Developer

App Source Code

Using Qt framework for
- User Interface
- Networking
- Location (GPS)
- Web integration
- ...

Qt NFC Development

More information:
bit.ly/StartNfc

- Requirements
  - Qt SDK 1.2
    - Contains: Qt 4.7.4, Qt Mobility 1.2
  - Windows: Nokia Suite
- Symbian
  - Nokia C7 / 801T with Symbian Anna / Belle or 603 / 700 / 701 / 808 with Nokia Belle+
  - Install Qt SDK's Symbian Anna or Belle target *
- MeeGo
  - Nokia N9
  - Install Qt SDK's MeeGo Harmattan target

* Apps compiled with the Symbian Anna target are compatible to Symbian^3, Anna & Belle.

Qt Mobility

Mobility 1.0

Bearer Management API
Contacts
Location
Messaging
Multimedia
Publish and Subscribe
Service Framework
Sensors
System Information
Versit

Mobility 1.1

Camera
Document Gallery
Feedback
Landmarks
Maps/Navigation
Organizer
Service Framework – Out of process

Mobility 1.2

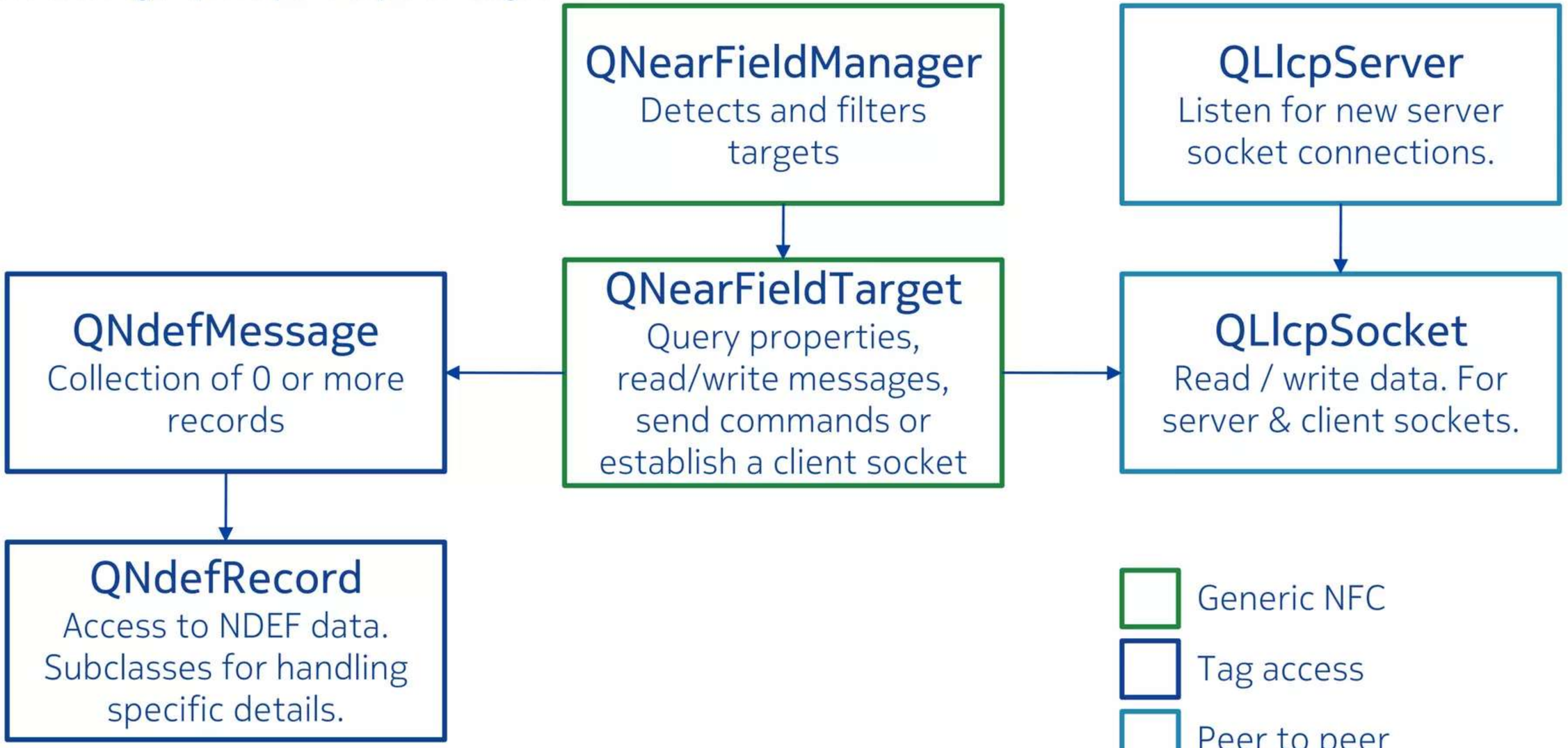
Connectivity (Bluetooth, NFC)

Qt Mobility 1.2 NFC

- Interact with
  - NFC Forum **tags**
    - Target detection
    - NDEF message handlers
    - Reading and writing NDEF messages
    - Send tag specific commands
    - Register for app autostart on tag touch
  - NFC Forum **devices**
    - LLCP peer-to-peer sockets

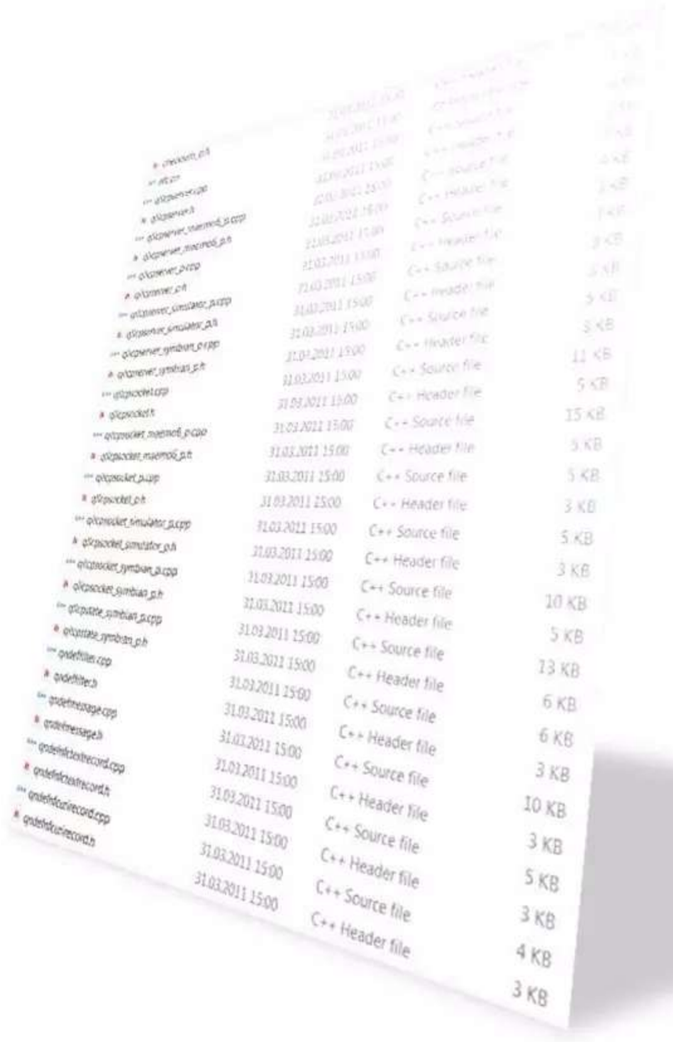
API Overview*

* Not a class diagram, but explains sequential usage order



Qt NFC Source Code

- Qt Mobility is Open Source
  - How does a method work? How to make best use of it? Take a look at the source code!
- **Download complete source package**
  - Latest source: http://qt.gitorious.org/qt-mobility/
  - Find implementation in: C:\QtMobility\src\connectivity\nfc



Refresh your Qt Knowledge

- The following tutorials require basic C++ & Qt skills
- Need more?

Qt Introduction
http://slidesha.re/QtAppDev

Qt Resources
http://www.developer.nokia.com/Develop/Qt/

Learning Qt
http://qt.nokia.com/learning

Nfc Corkboard

https://projects.developer.nokia.com/nfccorkboards

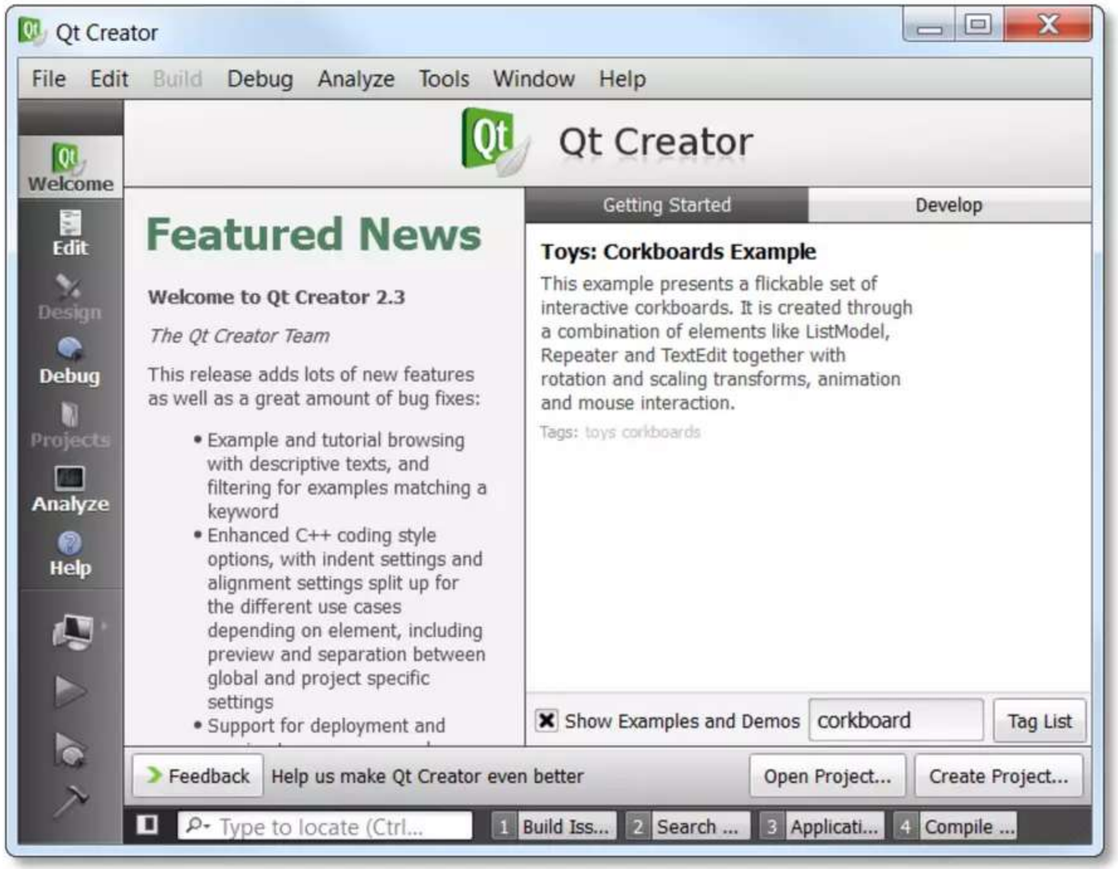
Tasks

- Extend Qt SDK example with NFC: Touch tag to create note
- Qt Quick based UI
- C++ based NFC
- Integrate C++ with QML

Importing the Example

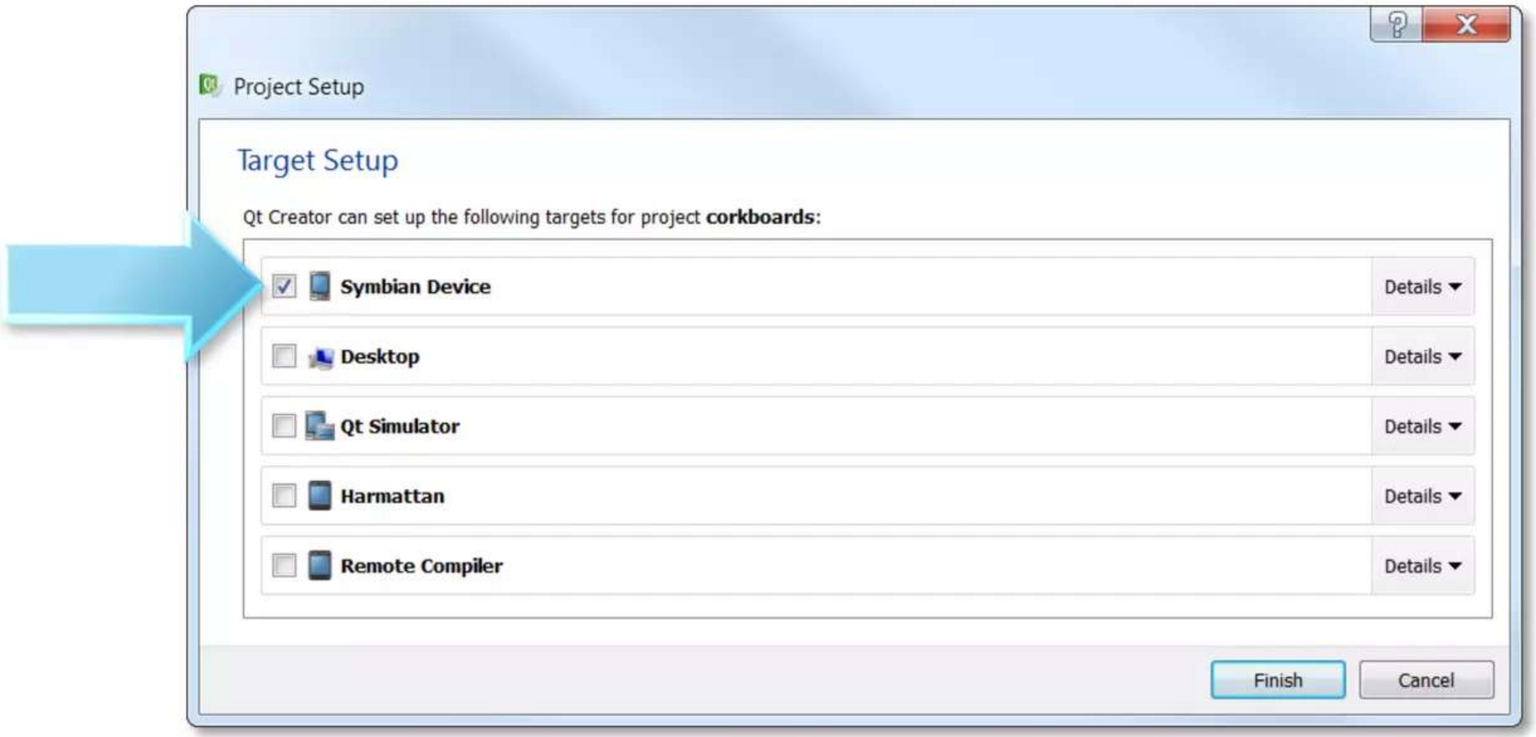
- From Qt Creator's Getting Started screen
  - Search for Corkboards example
  - Or open:

    C:\QtSDK\Examples\4.7\declarative\toys\corkboards\corkboards.pro



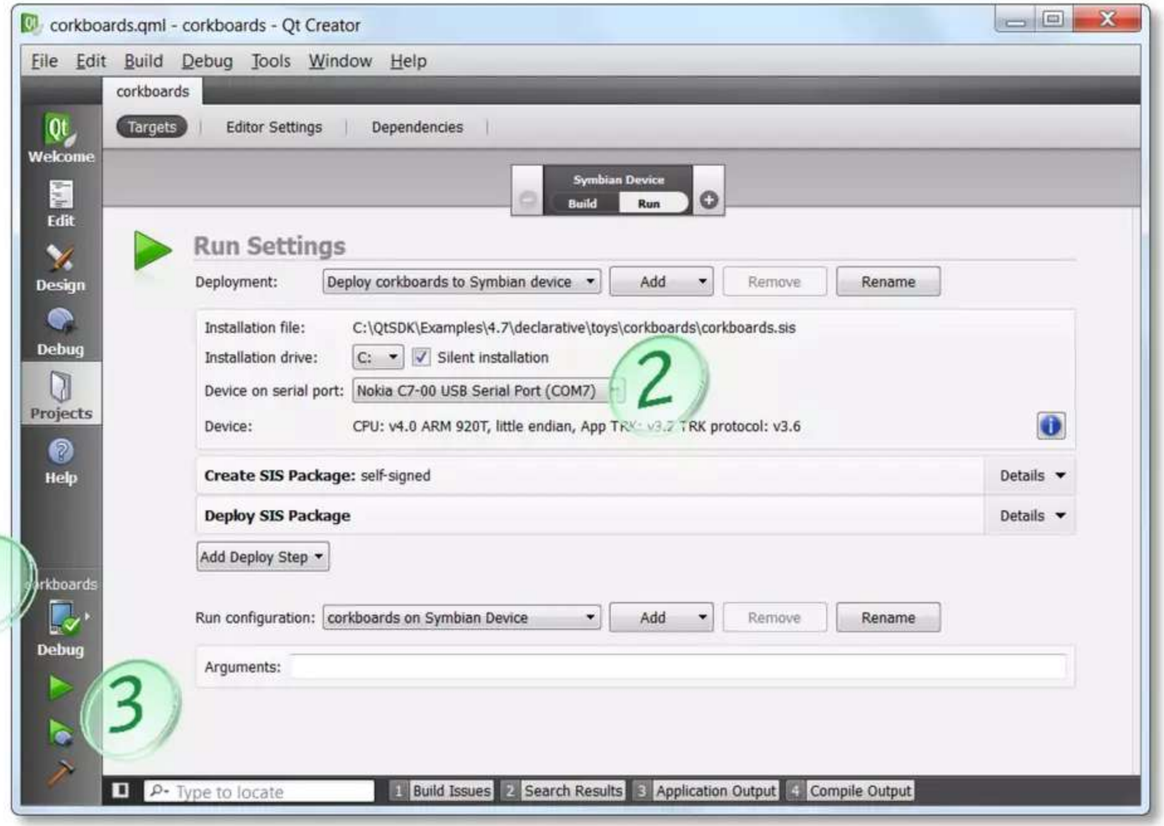
Choosing Targets

- Select at least the Symbian Anna target
  - Qt 4.7.4 & Qt Mobility 1.2 is required

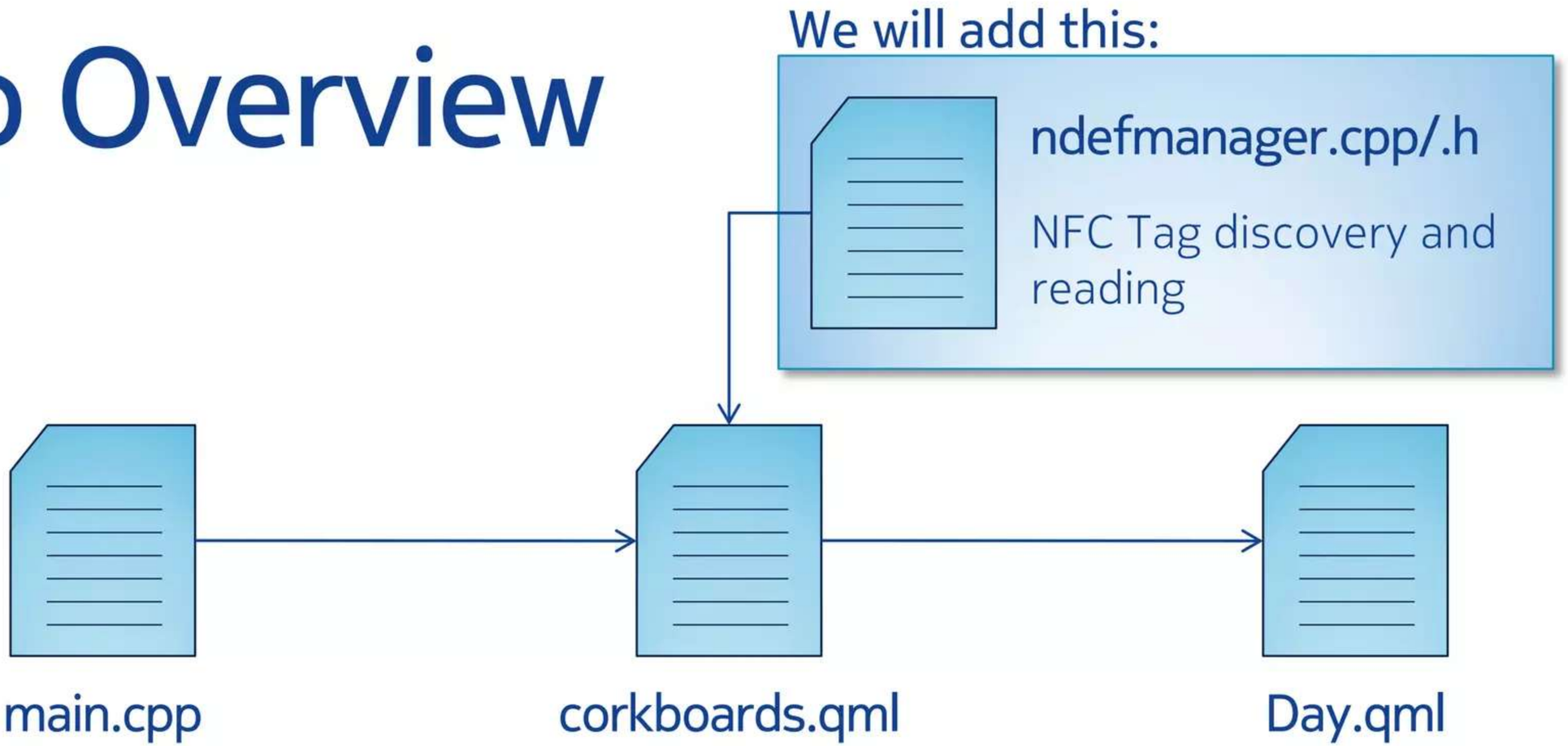


Test!

- Connect phone via USB
  - Ovi/Nokia Suite mode
  - Start CODA on device
- Check
  1. Selected Symbian target
  2. Device is recognized
3. Play to compile and deploy!



App Overview



Application startup

Loads and shows corkboards.qml

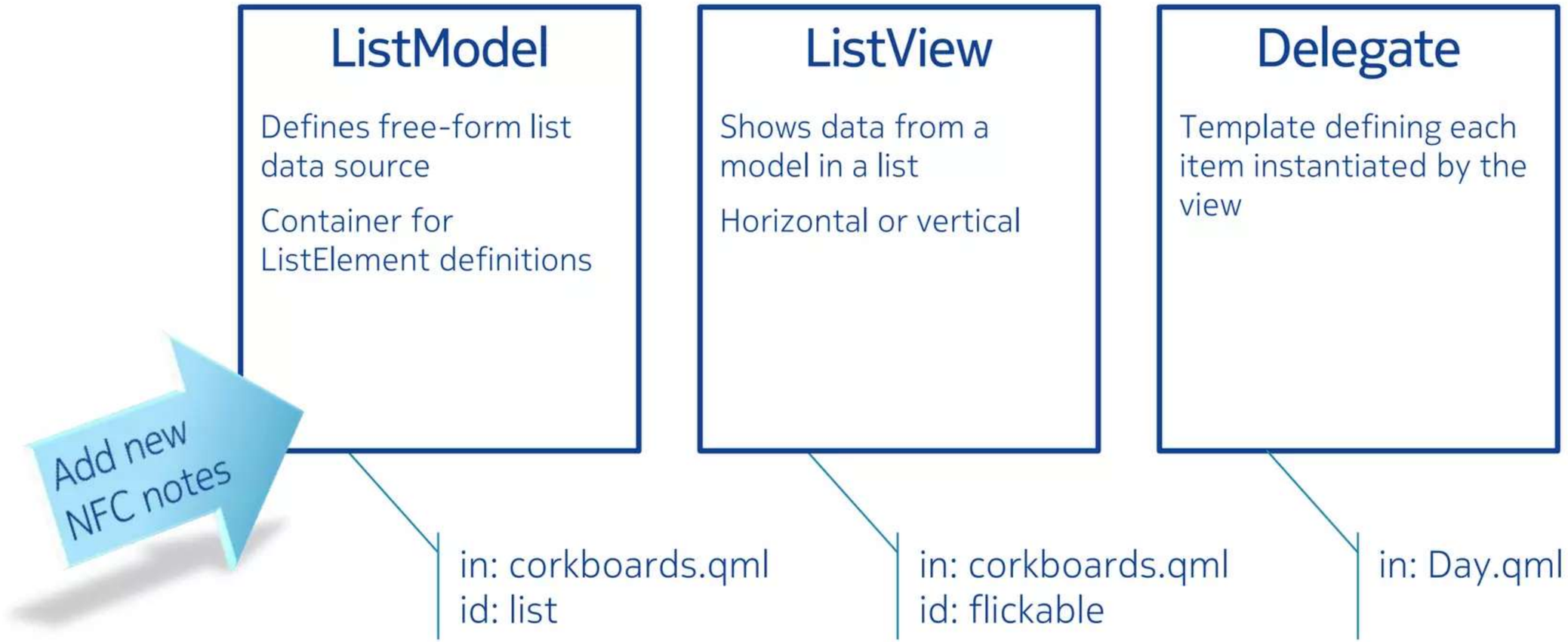
UI definition

Defines data model and view

Delegate for drawing the UI for a single day

Interaction for notes

Model, View & Delegate



Add Connectivity

- Edit the Qt project file (corkboards.pro)
  - Uncomment & add mobility component: connectivity

```
CONFIG += mobility
MOBILITY += connectivity
```

  - Add LocalServices Capability (for Symbian)

```
symbian:TARGET.CAPABILITY += NetworkServices LocalServices
```

Allows Internet access

Allows NFC

Good to Know: Symbian Security *

* Qt apps are native apps; therefore, the security model of the target operating system applies.

1. Determine the required security / privacy features
   - Most common: *(See: developer.nokia.com/Community/Wiki/Capabilities)*

Feature	Capability
Internet access, telephony, messaging	NetworkServices
Access location (GPS, etc.)	Location
Camera, record audio	UserEnvironment
Contacts, Calendar	ReadUserData / WriteUserData
Bluetooth, NFC	LocalServices
IMEI, model name, battery status	ReadDeviceData

Basic / User Capabilities

Extended / System Capabilities

2. Add to Qt project file *(.pro)*

```
symbian:TARGET.CAPABILITY += Location LocalServices
```

Good to Know: Symbian Security

3. Get the right certificate during development

No / Basic Capabilities

Use auto-generated self signed certificate
(default, no action required)

Extended Capabilities

Sign up for Nokia Publish account (€1)
publish.nokia.com

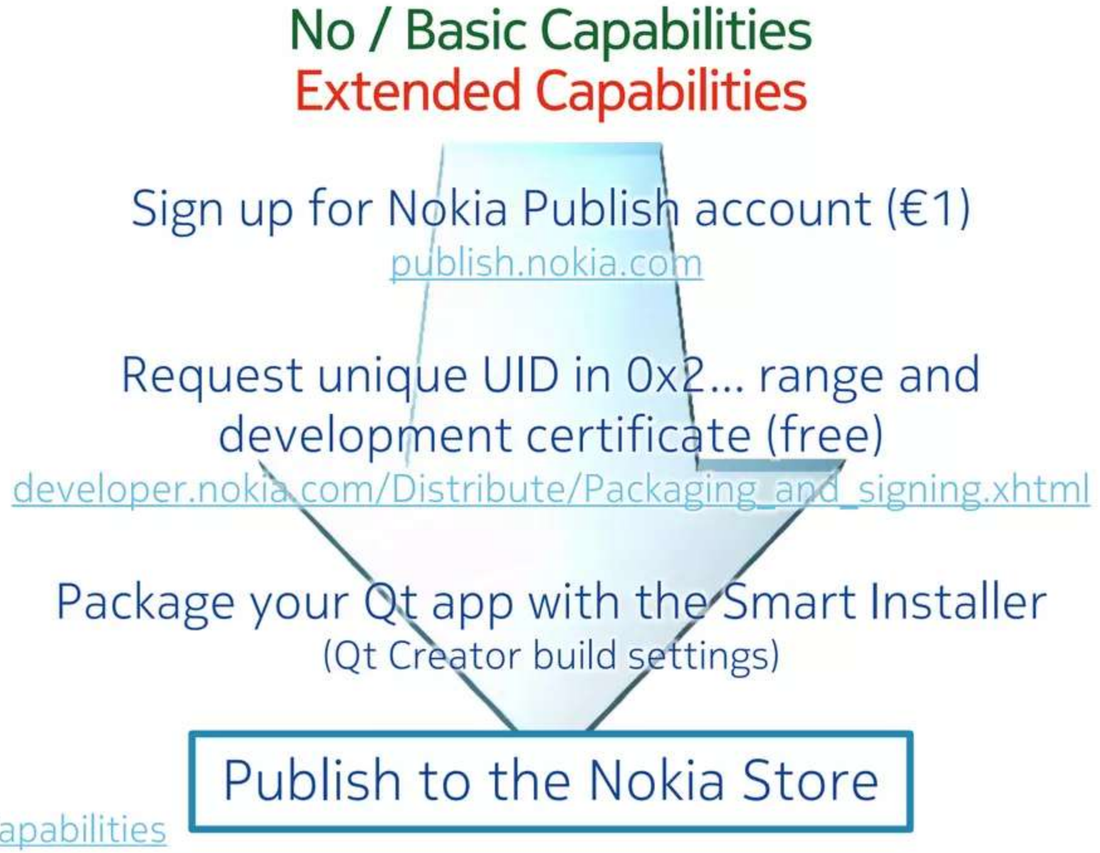
Request development certificate (free)
developer.nokia.com/Distribute/Packaging_and_signing.xhtml

Sign your app
(Qt Creator build settings)

Test on your develoment device(s)

Good to Know: Symbian Security

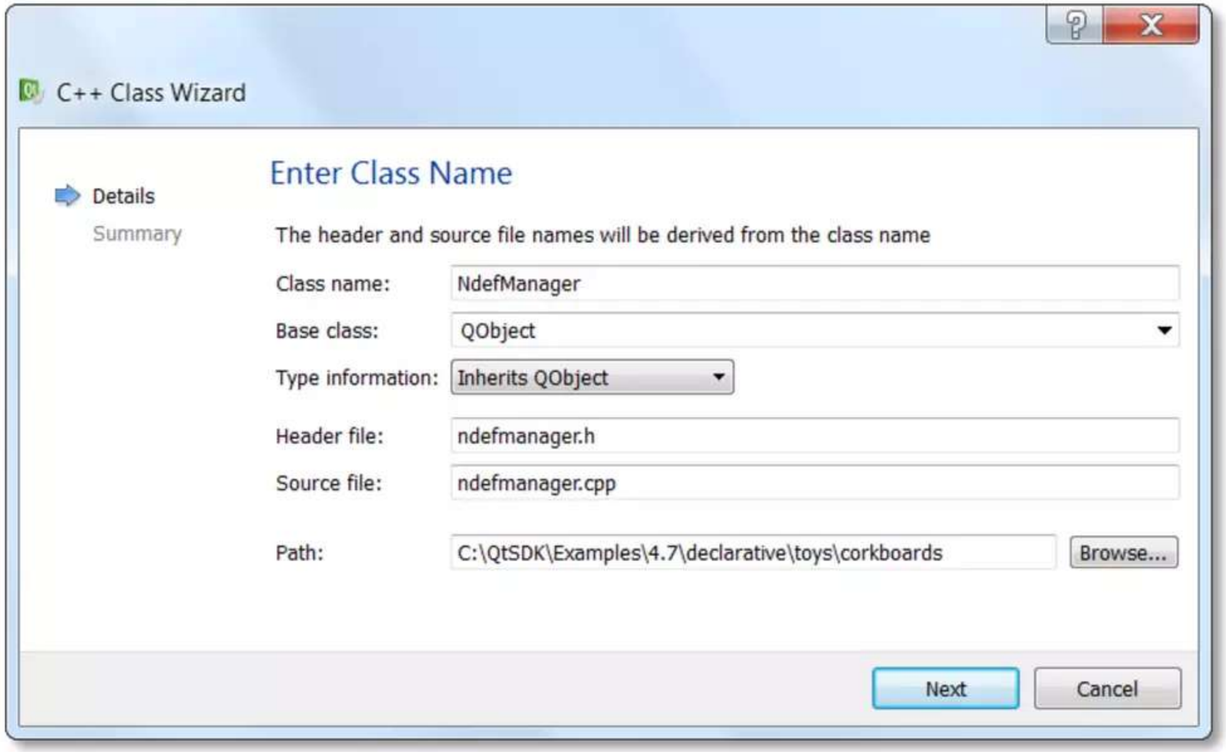
4. Publish



Need Certified Signed capabilities?
developer.nokia.com/Community/Wiki/Capabilities
-> Purchase a Publisher ID ($200 / year)
Submit to Certified Signed process @ www.symbiansigned.com

NdefManager

- New C++ class
  - Right-click the project
    - → Add New ...
    - → C++ Class
  - Class name: NdefManager
  - Base class: QObject



Qt Meta-Object System

- C++ extended with QObject
  - **Introspection:** meta-information at runtime
  - **Inter-object communication:** signals & slots
  - **Parent / Child hierarchy:** easier memory management

Include

- Headers, define member variable

ndefmanager.h

```
#include <qnearfieldmanager.h>
#include <qndeffilter.h>
#include <qnearfieldtarget.h>
#include <qndefmessage.h>
#include <qndefrecord.h>
#include <qndefnfcurirecord.h>
#include <QUrl>
#include <QDebug>  // we will test first!
QTM_USE_NAMESPACE  // Use Qt Mobility namespace

[...]
private:
    QNearFieldManager *nfcManager;
```

Start Waiting

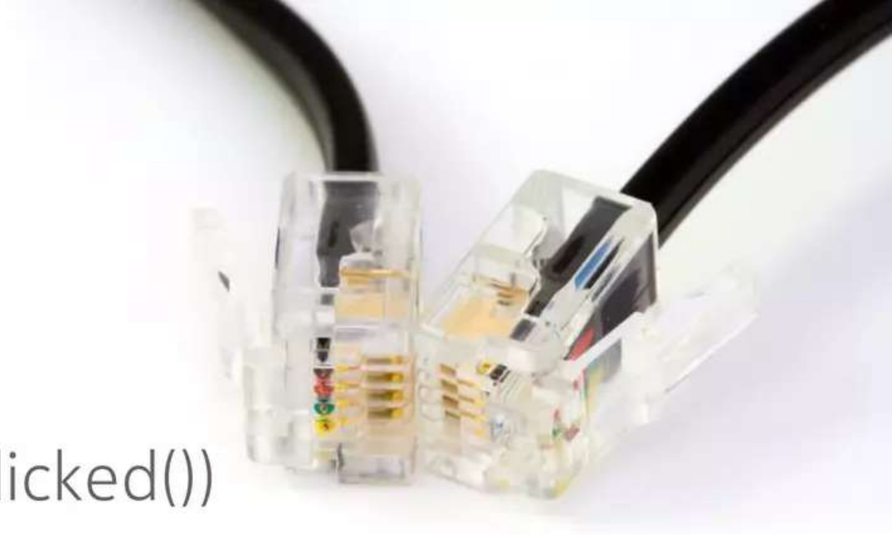
ndefmanager.cpp

```
NdefManager::NdefManager(QObject *parent) : QObject(parent)
{
    // NdefManager (this) is the parent; will automatically delete nfcManager
    nfcManager = new QNearFieldManager(this);
    nfcManager->setTargetAccessModes(QNearFieldManager::NdefReadTargetAccess);

    // React only to Uri records (NfcRtd, "U")
    QNdefFilter filter;
    filter.appendRecord<QNdefNfcUriRecord>();
    nfcManager->registerNdefMessageHandler(filter,
        this, SLOT(targetDetected(QNdefMessage,QNearFieldTarget*)));

    // Get notified when the tag gets out of range
    connect(nfcManager, SIGNAL(targetLost(QNearFieldTarget*)),
        this, SLOT(targetLost(QNearFieldTarget*)));
}
```

Signals & Slots



- Signal
  - **Emitted when a particular event occurs** (e.g., clicked())
  - Qt objects: predefined signals
  - Also create your own signals
- Slot
  - **Function called in response to a signal**
  - Qt objects: predefined slots (e.g., quit())
  - Also create your own slots
- **Connection signals ↔ slots established by developer, handled by Qt framework**

Signals & Slots

- Type safe
  - Signal signature must match signature of receiving slot
  - (Slot might also have shorter signature and ignore the rest of the arguments)
- Loosely coupled
  - Emitter doesn't know or care which slots receive signal
  - Information encapsulation
- Integrated, independent components
  - Slots are normal C++ member functions
  - Don't know if signals are connected to them

Lost & Found

- Methods to handle lost & found tags = Slots

ndefmanager.h

```
private slots:
    void targetDetected(const QNdefMessage &message,
                        QNearFieldTarget *target);
    void targetLost(QNearFieldTarget *target);
```

- Tip
  - Alt + Enter, and Qt Creator writes the .cpp method definition!

```
public slots:
    void targetDetected(const QNdefMessage &message, QNearFieldTarget *target);
    void targetLost(QNearFieldTarget *target);
                Add Definition in ndefmanager.cpp
```

React

- New target: write debug output to console for now

ndefmanager.cpp

```
void NdefManager::targetDetected(const QNdefMessage &message,
                                 QNearFieldTarget *target)
{
    qDebug() << "Uri Target detected!";
}
```

- Target out of range: delayed object delete

ndefmanager.cpp

```
void NdefManager::targetLost(QNearFieldTarget *target)
{
     target->deleteLater();
}
```

Register C++ with QML

- Make your C++ class available in QML
  - QML Name: NdefManager
  - QML Library: Nfc 1.0

main.cpp

```
#include <QtDeclarative>
#include "ndefmanager.h"

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);
    qmlRegisterType<NdefManager>("Nfc", 1, 0, "NdefManager");
    [...]
}
```

NdefManager in QML

- Import module

corkboards.qml

```
import Nfc 1.0
```

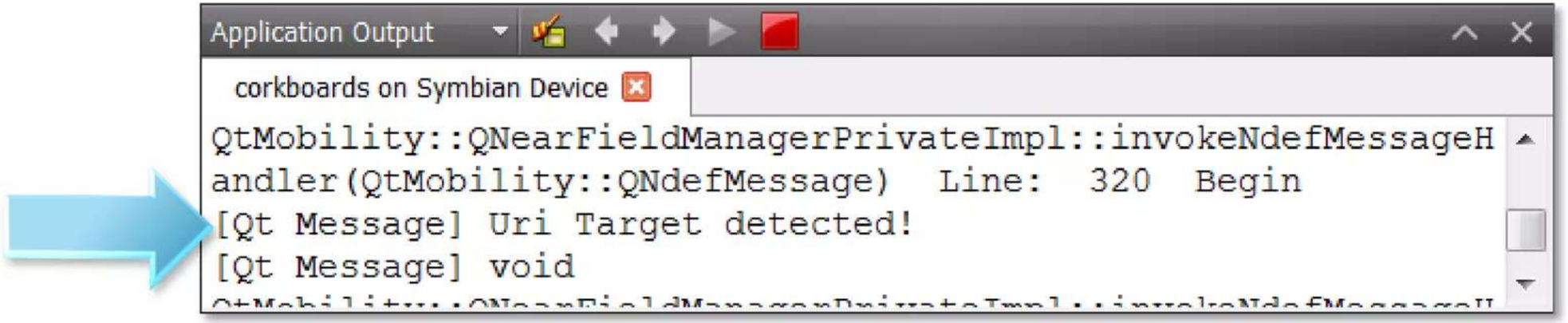
- Create Instance

corkboards.qml

```
NdefManager {
    id: ndefManager
}
```

Test! #2

- Run the app
- Touch a URI formatted NDEF tag
- Observe the Application Output



Parse Tag & Emit Contents

- Revisit and extend the target detection slot

ndefmanager.cpp

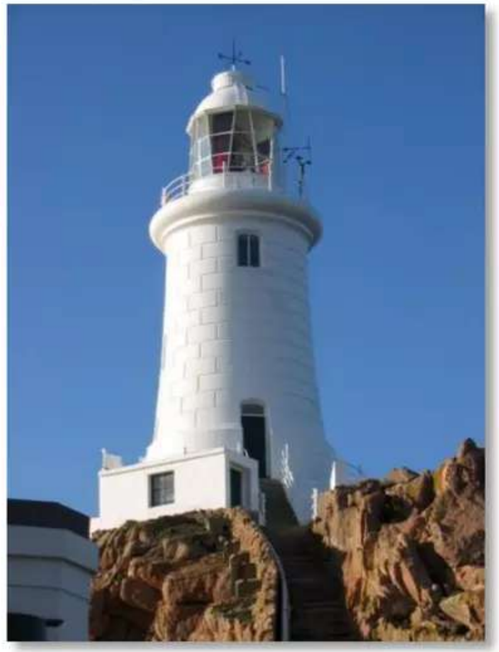
```
void NdefManager::targetDetected(const QNdefMessage &message,
                                 QNearFieldTarget *target) {
    qDebug() << "Uri Target detected!";
    // Go through all records in the message
    foreach (const QNdefRecord &record, message) {
        // Check type again, just to make sure
        if (record.isRecordType<QNdefNfcUriRecord>()) {
            // Convert to the specialized URI record class
            QNdefNfcUriRecord uriRecord(record);
            // Emit a signal with the URI
            emit nfcReadTagUri(uriRecord.uri());
        }
    }
}
```

Define the Signal

- Signals have no implementation
  - They represent information, not a method to execute

ndefmanager.h

```
signals:
    void nfcReadTagUri(const QUrl& nfcTagUri);
```



Create new Notes

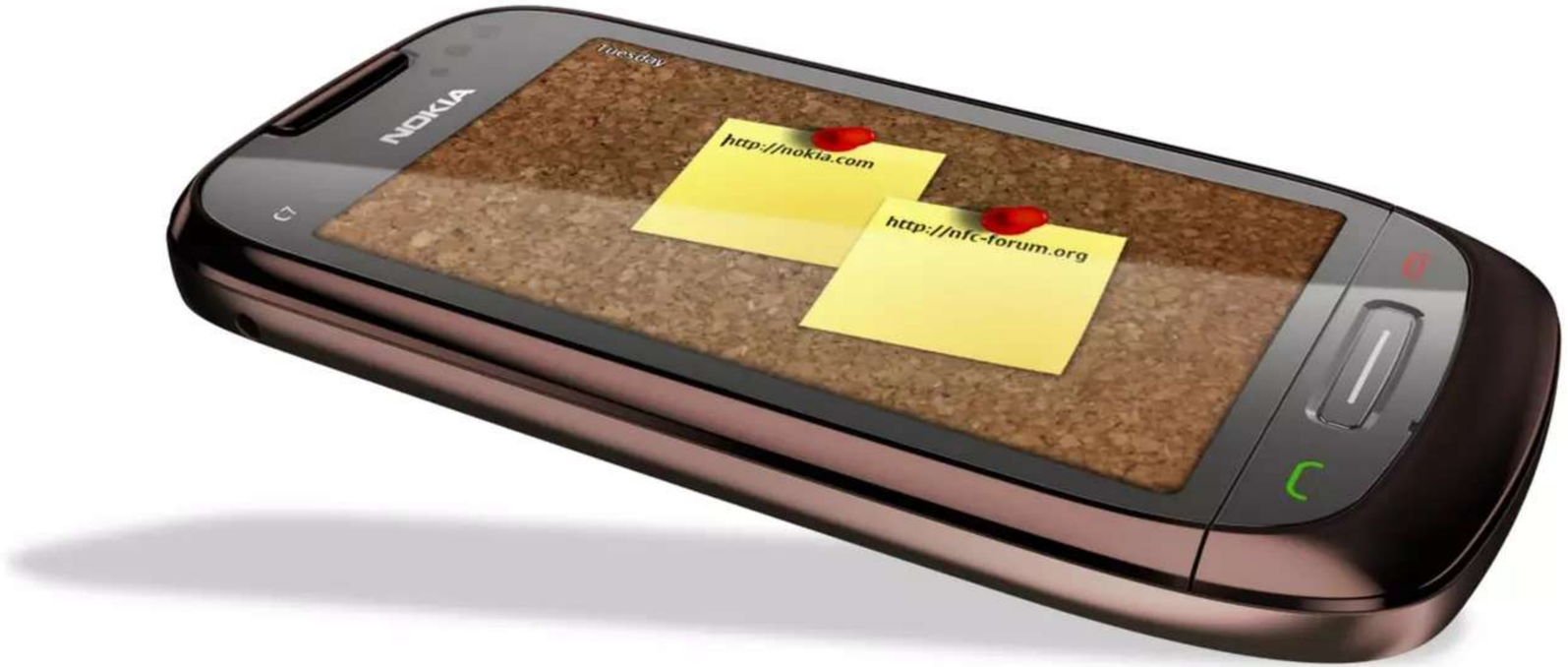
- Handle our new signal
  - on *<signal>* : *<javascript code>*
  - Get current corkboard page → append new notes element → key: “noteText”, value: URI from NDEF message

corkboards.qml

```
NdefManager {
    id: ndefManager
    onNfcReadTagUri:
        list.get(flickable.currentIndex).notes.append({"noteText":nfcTagUri})
}
```

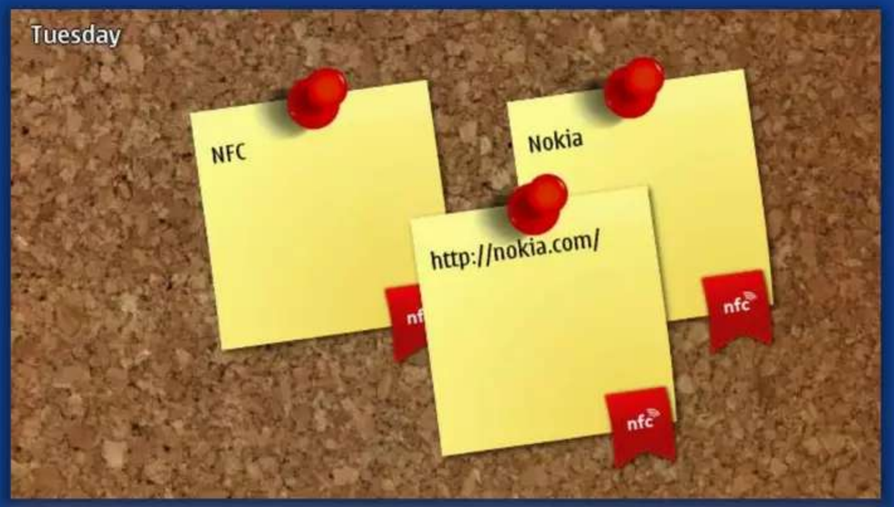
Test! #3

- Touch Uri-formatted NDEF NFC tags to create notes on the current page!



Interactive Nfc Corkboard

https://projects.developer.nokia.com/nfccorkboards



Tasks

- Extend previous example
- React to any kind of tags
- Write tags by pressing the red flag while touching a tag

Overview – Before

Register NDEF filters and detect targets

```
QNdefFilter filter;
filter.appendRecord<QNdefNfcUriRecord>();
nfcManager->registerNdefMessageHandler(filter, this,
            SLOT(targetDetected(QNdefMessage,QNearFieldTarget*)));
```

Signal is only emitted when
a tag containing all records defined
in the filter are found

The slot directly gets the NDEF message

```
void targetDetected(const QNdefMessage &message,
                    QNearFieldTarget *target);
```

Overview – After

Detect any kind of targets

```
connect(nfcManager, SIGNAL(targetDetected(QNearFieldTarget*)),
        this, SLOT(targetDetected(QNearFieldTarget*)));
nfcManager->startTargetDetection();
```

Signals is emitted for every target that is detected

```
void targetDetected(QNearFieldTarget *target); {
    const bool hasNdefMessage = target->hasNdefMessage();
    if (hasNdefMessage) {
        connect(target, SIGNAL(ndefMessageRead(QNdefMessage)),
                this, SLOT(ndefMessageRead(QNdefMessage)));
        target->readNdefMessages();
    }
}
```

Another signal is emitted when reading the message from the target is finished

```
void NdefManager::ndefMessageRead(const QNdefMessage &message) ;
```

React to all targets

- Generic target detection instead of listening to specific tag message(s)
  - Remove previous targetDetected() with two parameters

ndefmanager.h

```
public slots:
    void nfcWriteTag(const QString& nfcTagText);          // New
private slots:
    void targetDetected(QNearFieldTarget *target);        // Modified from prev.
    void ndefMessageRead(const QNdefMessage &message);    // New
private:
    QNearFieldTarget *cachedTarget;                       // New
```

Start Generic Target Detection

ndefmanager.cpp

```
NdefManager::NdefManager(QObject *parent) : QObject(parent)
{
    // [...]
    // React on
    QNdefFilter
    filter.appendRecord<QN
    nfcManager->
        this, SL
    // Get notified when the tag gets out of range
    connect(nfcManager, SIGNAL(targetLost(QNearFieldTarget*)),
        this, SLOT(targetLost(QNearFieldTarget*)));
    connect(nfcManager, SIGNAL(targetDetected(QNearFieldTarget*)),
        this, SLOT(targetDetected(QNearFieldTarget*)));
    nfcManager->startTargetDetection();
}
```

Read NDEF Message

- Detected a target? Read its message.

ndefmanager.cpp

```
void NdefManager::targetDetected(QNearFieldTarget *target)
{
    qDebug() << "Target detected!";
    const bool hasNdefMessage = target->hasNdefMessage();
    if (hasNdefMessage) {
        connect(target, SIGNAL(ndefMessageRead(QNdefMessage)),
                this, SLOT(ndefMessageRead(QNdefMessage)));
        target->readNdefMessages();
    }
    // Cache target to member variable for writing
    cachedTarget = target;
}
```

Target Lost

- Set cached target to NULL

ndefmanager.cpp

```
void NdefManager::targetLost(QNearFieldTarget *target) {
    cachedTarget = NULL;    // New
    target->deleteLater();
}
```

Parse NDEF Message

- Same code as before, different method

ndefmanager.cpp

```
void NdefManager::ndefMessageRead(const QNdefMessage &message) {
    // Go through all records in the message
    foreach (const QNdefRecord &record, message) {
        // Check type
        if (record.isRecordType<QNdefNfcUriRecord>()) {
            // Convert to the specialized URI record class
            QNdefNfcUriRecord uriRecord(record);
            // Emit a signal with the URI
            emit nfcReadTagUri(uriRecord.uri());
        }
    }
}
```

Write URI Ndef Message

ndefmanager.cpp

```
void NdefManager::nfcWriteTag(const QString &nfcTagText) {
    // The device currently has a target in reach
    if (cachedTarget) {
        // Convert text to a URI, adding missing http:// if necessary
        QUrl convertedUrl = QUrl::fromUserInput(nfcTagText);
        // Check if it is a valid URL
        if (convertedUrl.isValid() && nfcTagText.contains('.')) {
            QNdefMessage message;
            // Create a URI NDEF record
            QNdefNfcUriRecord uriRecord;
            uriRecord.setUri(convertedUrl);
            message.append(uriRecord);
            qDebug() << "Writing URI ...";
            // Write NDEF message to the tag
            cachedTarget->writeNdefMessages(QList<QNdefMessage>() << message);
        } else {
            qDebug() << "No valid URI";
        }
    } else {
        qDebug() << "No cached target available";
    }
}
```

Extend UI with Write Button

Day.qml, between lines 122 / 123 (below MouseArea with id: mouse)

```
Image {
    id: writeButton
    source: "NfcFlag.png"
    rotation: -8 // Note image itself is rotated
    anchors { bottom: parent.bottom; right:parent.right }
    MouseArea {
        anchors.fill: parent
        onClicked: ndefManager.nfcWriteTag(myText.text)
    }
}
```

NfcFlag.png

<project dir>\qml\

Test!

https://projects.developer.nokia.com/nfccorkboards

- Touch a tag, at the same time press the NFC flag of a note
  - Writes the note text to the tag
  - Remember to enter a valid URI!

The extended solution on Nokia Developer Projects handles additional record types and potential errors.