BlackBerry Jam Americas

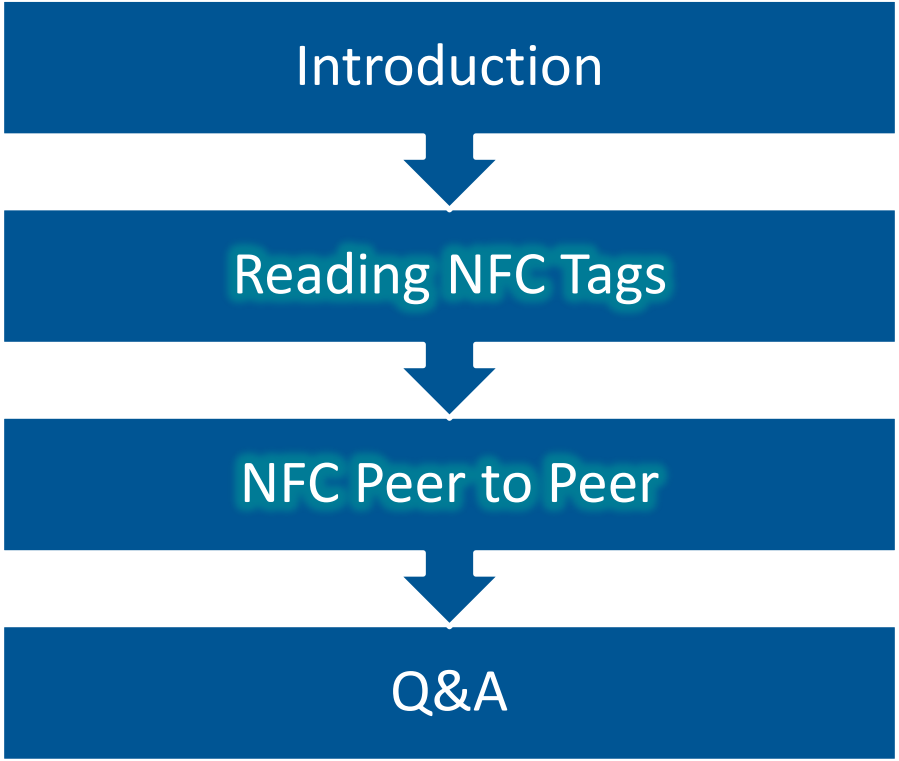
Spice up your Apps with NFC

JAM14

Martin Woolley and Rob Williams

September 25-27, 2012

Agenda



Introduction

Reading NFC Tags

NFC Peer to Peer

Q&A

Overview

Near Field Communications is a magical technology

Just by touching NFC devices....

Trigger actions	Transfer state	Transfer objects	Execute transactions
Find bus arrival times	*Games - your turn now!*	*Photos, contacts etc*	*Payment, physical access*

Opportunity

Enrich your apps with NFC

Tactile

Frictionless

Convenient

Fun

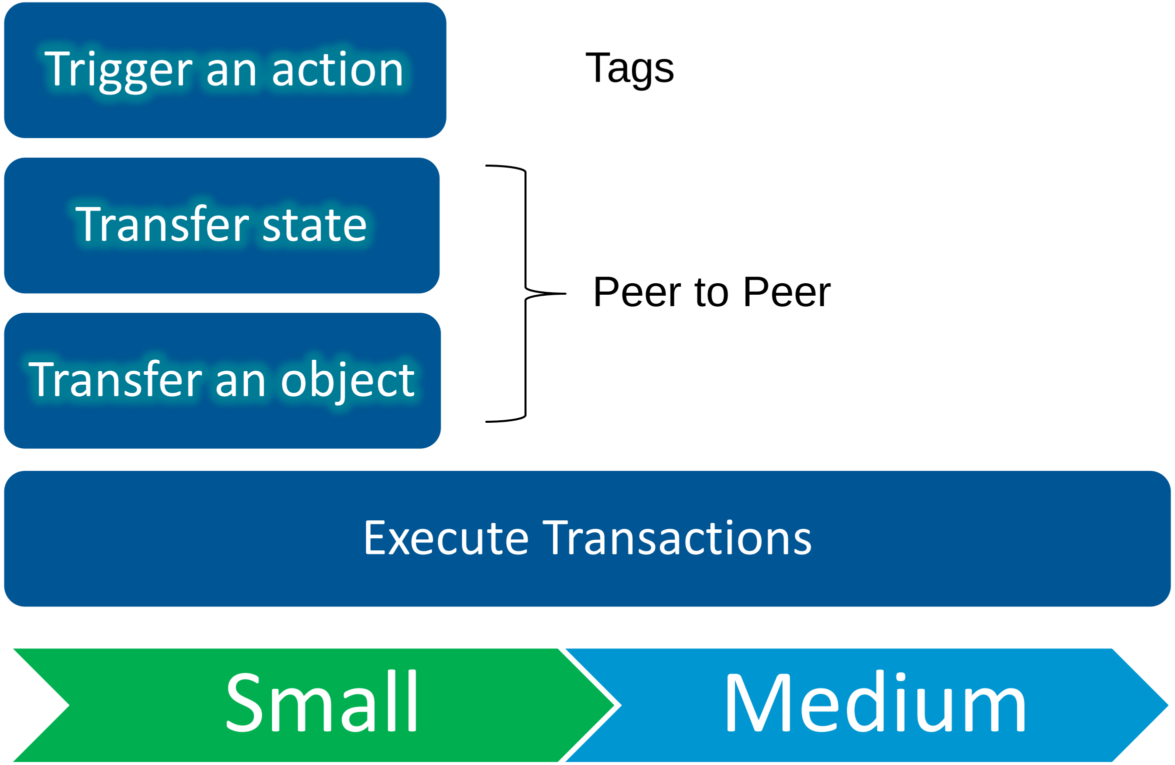
Secure

"Mobile NFC ..... users expected to grow from 16.7M to over 250M in 4 years (from 2011 – 2015)"

Forecast from Juniper Research (April 2011)

Effort to Enrich Your Apps

BlackBerry Jam Americas



Trigger an action

Tags

Transfer state

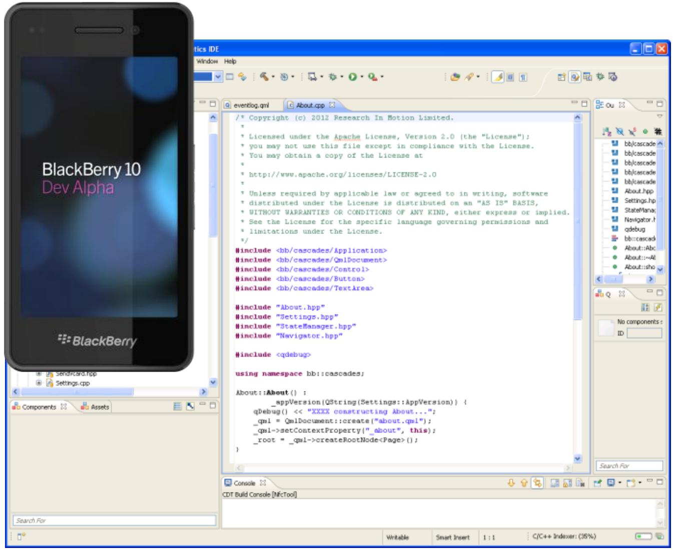
Transfer an object

Peer to Peer

Execute Transactions

Small

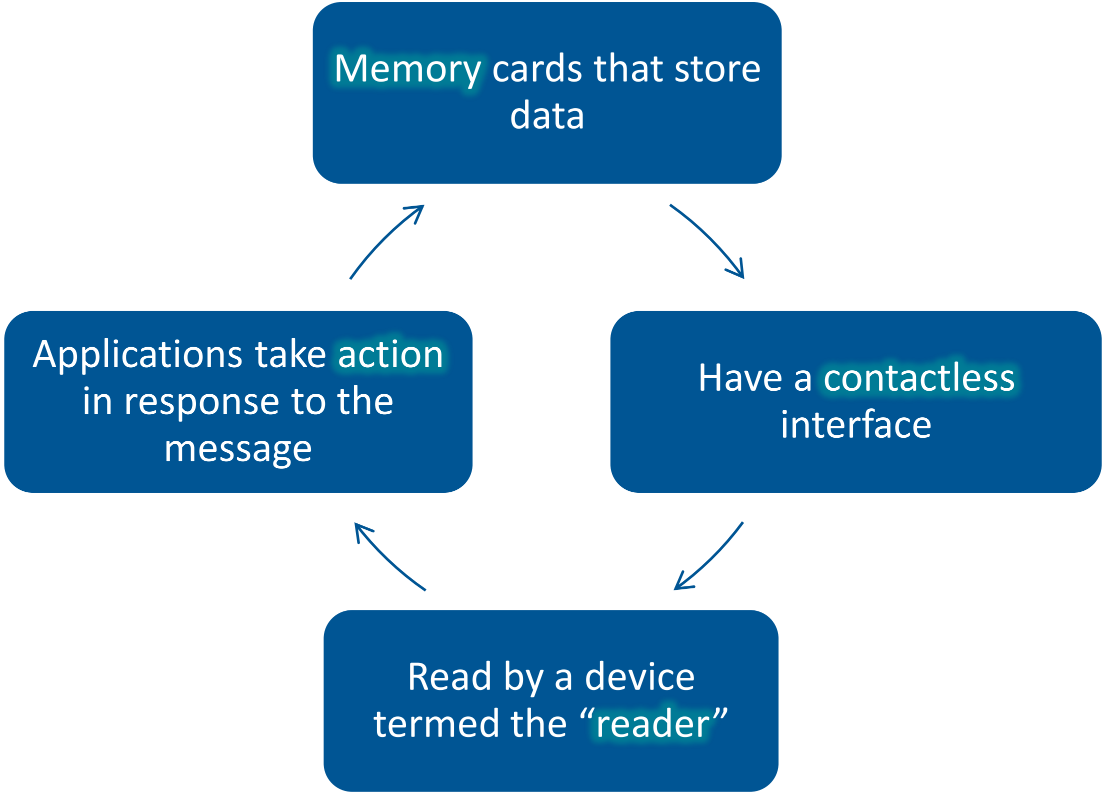
Medium



```
/* Copyright (c) 2012 Research In Motion Limited.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
#include <bb/cascades/Application>
#include <bb/cascades/QmlDocument>
#include <bb/cascades/Control>
#include <bb/cascades/Button>
#include <bb/cascades/TextArea>

#include "About.hpp"
#include "Settings.hpp"
#include "StateManager.hpp"
#include "Navigator.hpp"

#include <qdebug>

using namespace bb::cascades;

About::About() :
        _appVersion(QString(Settings::AppVersion)) {
    qDebug() << "XXXX constructing About...";
    _qml = QmlDocument::create("about.qml");
    _qml->setContextProperty("_about", this);
    _root = _qml->createRootNode<Page>();
}
```

Reading NFC Tags

About NFC Tags



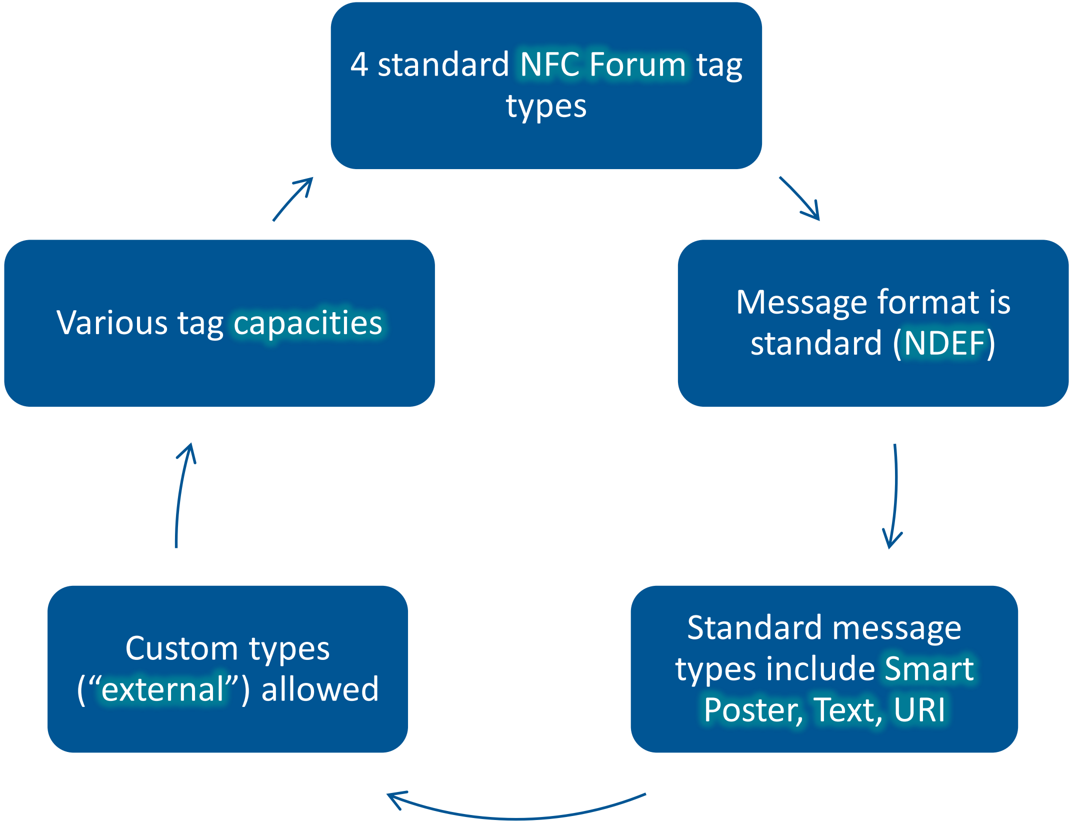
Memory cards that store data

Have a contactless interface

Read by a device termed the “reader”

Applications take action in response to the message

About NFC Tags



4 standard NFC Forum tag types

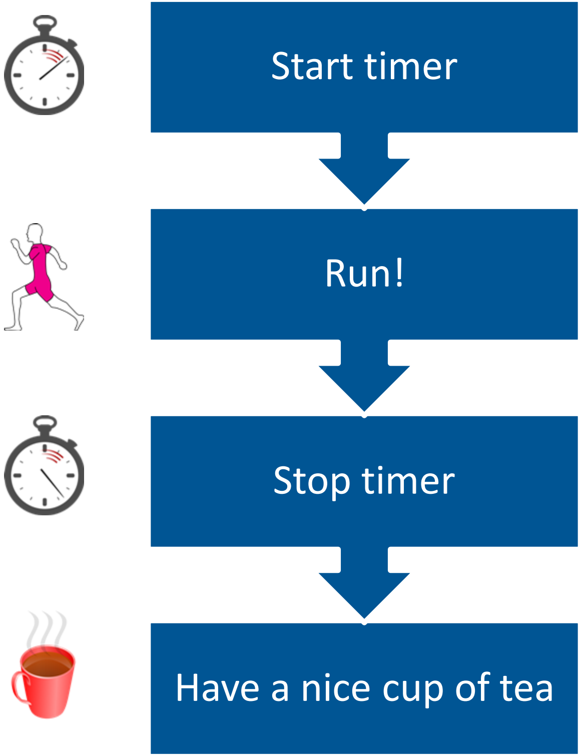
Message format is standard (NDEF)

Standard message types include Smart Poster, Text, URI

Custom types (“external”) allowed

Various tag capacities

Demo - NFC Fun Run

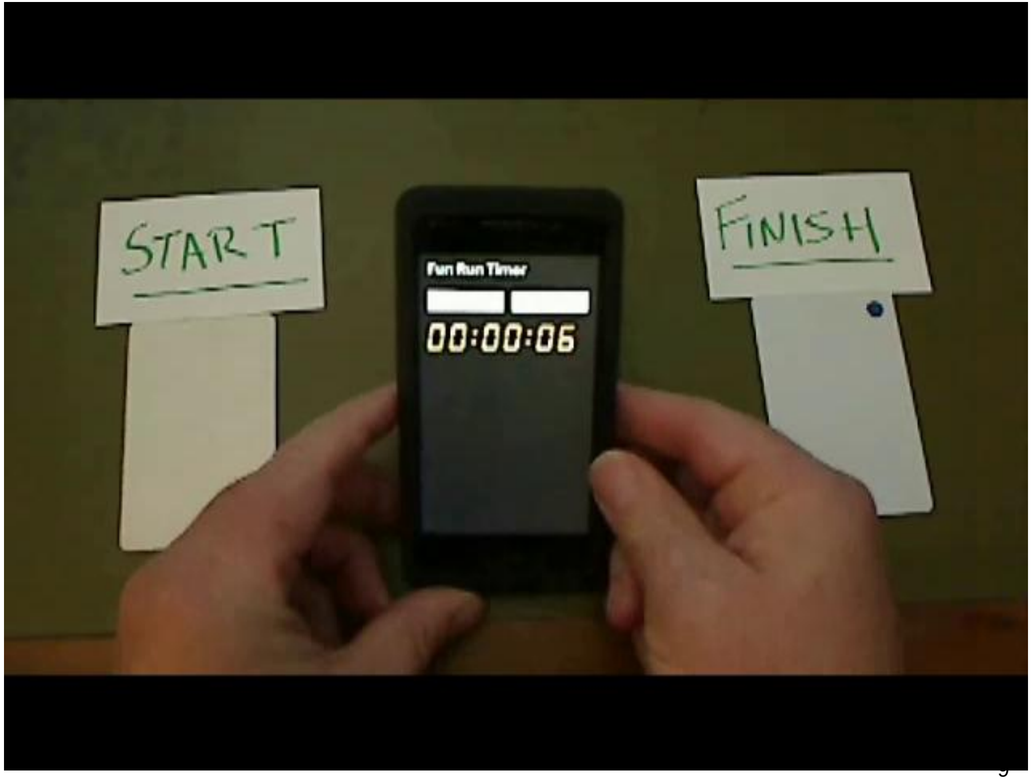


Start timer

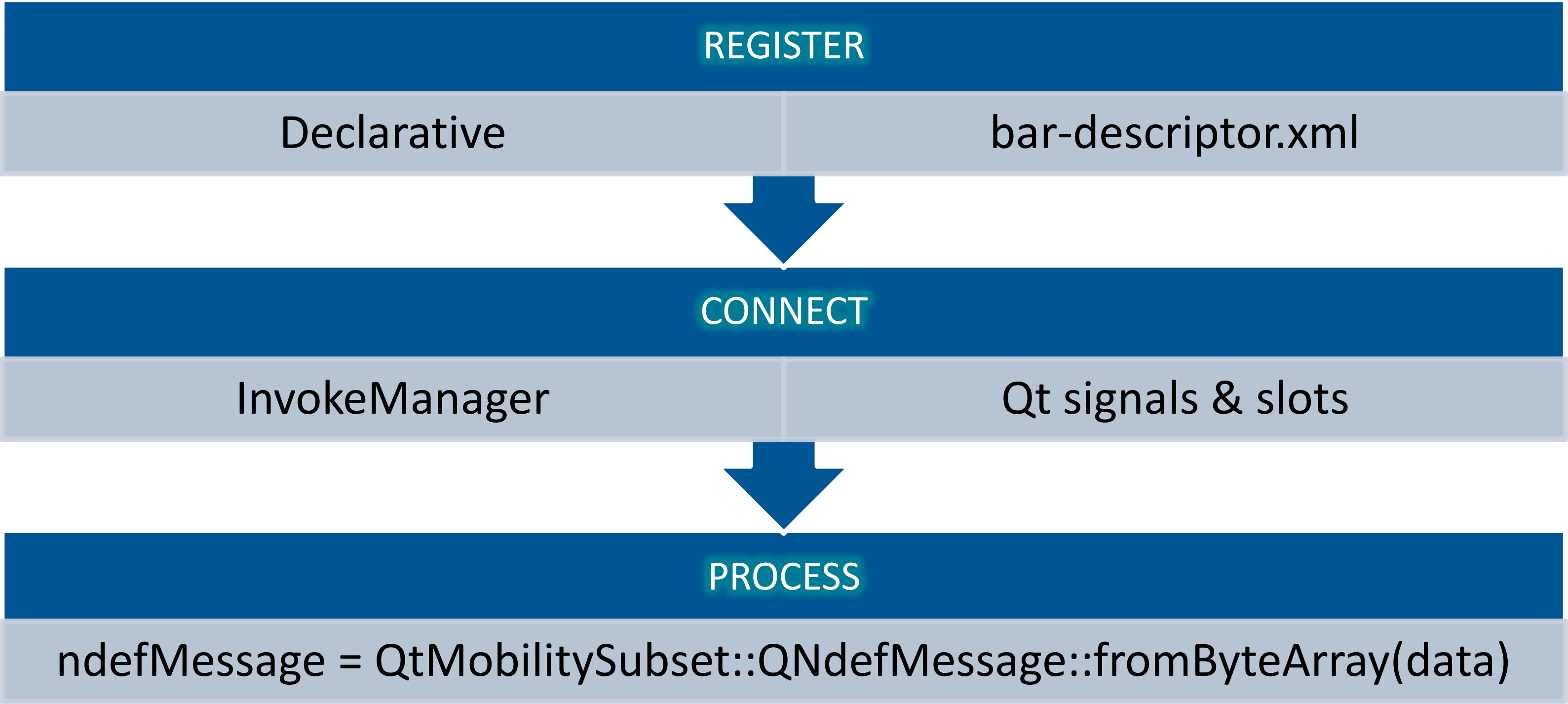
Run!

Stop timer

Have a nice cup of tea



The 3 Step Process



REGISTER

Declarative | bar-descriptor.xml

CONNECT

InvokeManager | Qt signals & slots

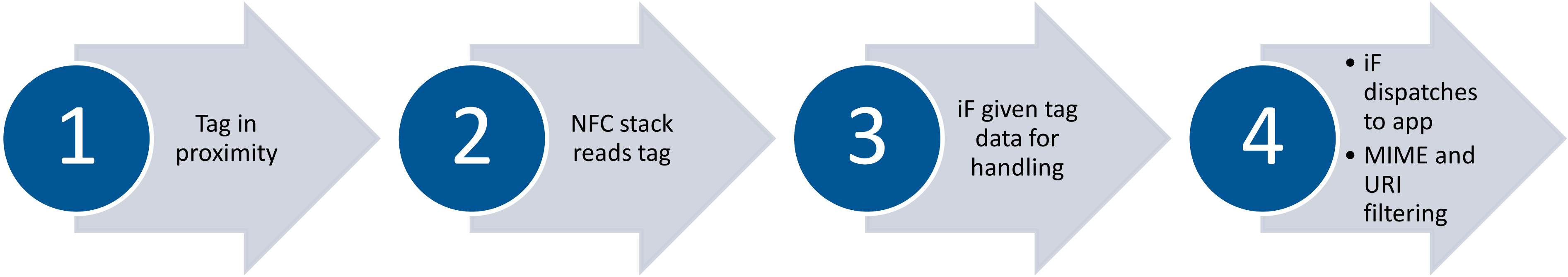
PROCESS

ndefMessage = QtMobilitySubset::QNdefMessage::fromByteArray(data)

Register

```
<invoke-target id="com.example.NfcRaceTime">
   <require-source-permissions>invoke_across_perimeters</require-
   source-permissions>
   <type>APPLICATION</type>
   <filter>
        <action>bb.action.OPEN</action>
        <mime-type>application/vnd.rim.nfc.ndef</mime-type>
        <property var="uris"
        value="ndef://4/my.rim.com/myrecordtype"/>
   </filter>
</invoke-target>
```

Invocation Framework (iF)



1 Tag in proximity

2 NFC stack reads tag

3 iF given tag data for handling

4
- iF dispatches to app
- MIME and URI filtering

Connect

```
bb::system::InvokeManager* _invokeManager =
   new bb::system::InvokeManager();


QObject::connect(_invokeManager,
        SIGNAL(
               invoked(const bb::system::InvokeRequest&)),
                     this,
        SLOT(
               receivedInvokeRequest(
                      const bb::system::InvokeRequest&)));
```

Process

```
// SLOT method
void App::receivedInvokeRequest(
    const bb::system::InvokeRequest& request) {

    QByteArray data = request.data();

    QtMobilitySubset::QNdefMessage ndefMessage =
        QtMobilitySubset::QNdefMessage::fromByteArray(data);

    handleNdefRequest(ndefMessage);
}
```

Process

```
// look for NDEF records of our special type
QList<QtMobilitySubset::QNdefRecord>::const_iterator ndefRecord;

for ( ndefRecord = ndefMessage.begin(); ndefRecord !=
ndefMessage.end(); ndefRecord++) {

    if (ndefRecord->typeNameFormat() ==
        QtMobilitySubset::QNdefRecord::ExternalRtd) {

        if (QString(
        ndefRecord->type()).compare("my.rim.com:myrecordtype") == 0 ){
                emit raceTagDetected(QString(ndefRecord->payload()));
        }
    }
}
```

NFC Peer to Peer Mode

The Opportunity

Transfer data from device to device

Frictionless

Shared

Interactive

Share just by touching

Create physical, interactive experiences

Protocol Options

SNEP

Simple NDEF Exchange Protocol

Message Oriented

Higher level

LLCP

Logical Link Control Protocol

Connection Oriented

Lower level

API Options

Invocation Framework (iF)

- Less NFC coding required
- Other app provides service to yours
- iF may provide UI for user selections

Direct API Usage

- Some NFC API calls in event loop
- All NFC ops in your app
- Complete control over UI

SHARE with iF

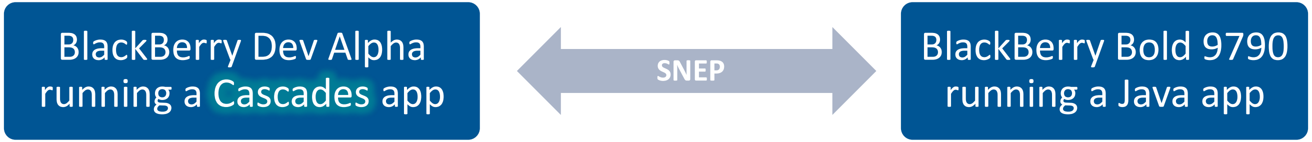
```
Button {
               id: shareButton
               text: "Share vCard - Press and Hold!"
           contextActions: [
               ActionSet {
                   title: "Share NDEF Data"
               InvokeActionItem {
                   id: sharedNdefData
                   query {
                       mimeType: "application/vnd.rim.nfc.ndef"
                       invokeActionId: "bb.action.SHARE"
                   }
               }
           }
           ]
```

SHARE with iF

```
Button {
....
  contextMenuHandler: ContextMenuHandler {
    onShowing: {
      // make NDEF message as byte array in C++
      sharedNdefData.data =
        _ndefFactory.getNdefVcardMessage(sendVcard.ndefFirstName,
        sendVcard.ndefLastName, sendVcard.ndefAddress,
        sendVcard.ndefEmail, sendVcard.ndefMobile);
    }
  }
}
```

Demo - Proximity Gaming

Touch Tic Tac Toe
Moves: 0
Reset
Place your BlackBerry smart phones back to back to start a NEW GAME



BlackBerry Dev Alpha running a Cascades app

SNEP

BlackBerry Bold 9790 running a Java app

What did we just see?

Start a new game

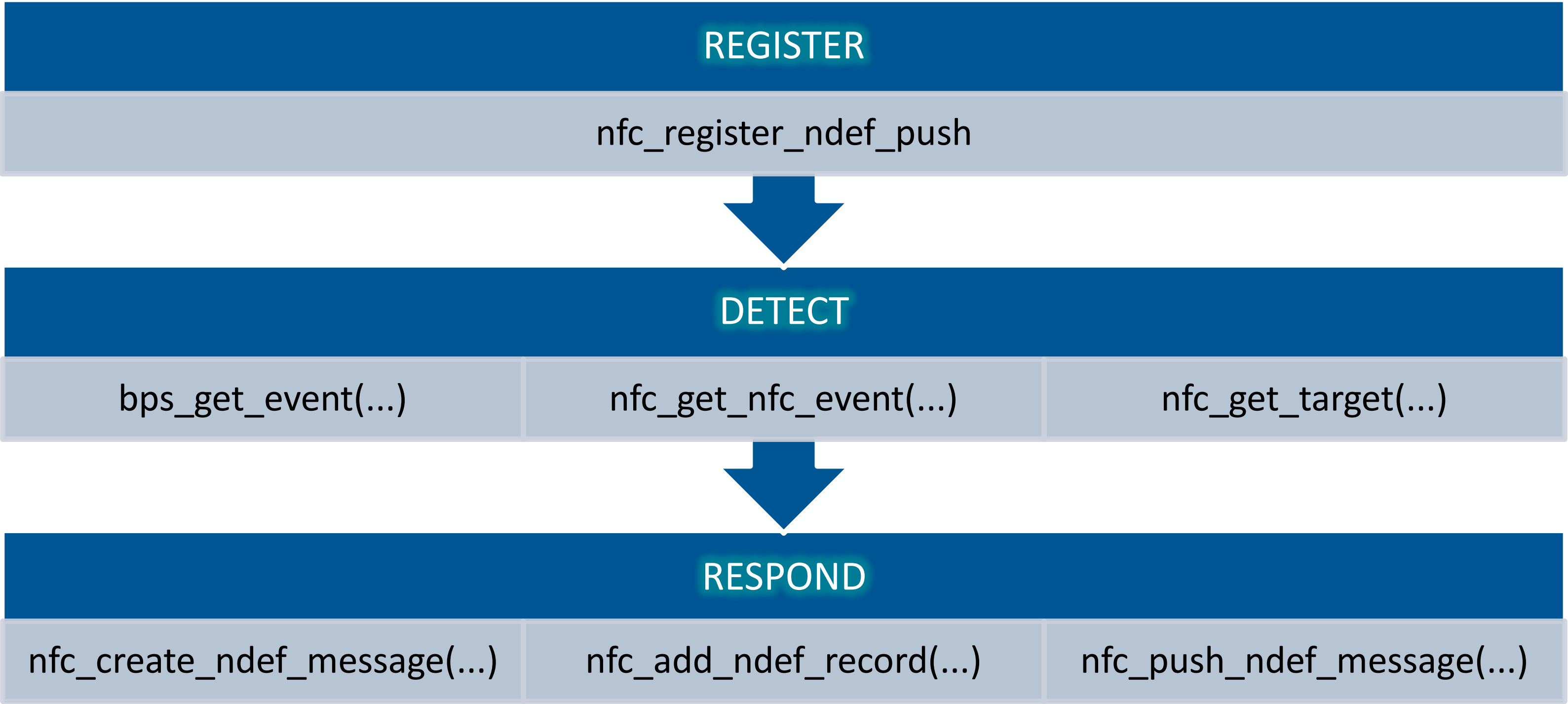
Pass turn

Just by touching devices

State transferred over SNEP

Interoperability

The 3 Step Process



REGISTER

nfc_register_ndef_push

DETECT

bps_get_event(...)

nfc_get_nfc_event(...)

nfc_get_target(...)

RESPOND

nfc_create_ndef_message(...)

nfc_add_ndef_record(...)

nfc_push_ndef_message(...)

Register

```
void NfcWorker::registerSnepClient() {

    int rc = nfc_register_snep_client();

    if (rc != BPS_SUCCESS) {
        // error handling
    }

}
```

Detect

```
// we have an event loop reading events from BPS
while (!_timeToDie) {
       bps_event_t *event;

       rc = bps_get_event(&event, BPS_EVENT_TIMEOUT);

       if (!rc && event) {
               domain = bps_event_get_domain(event);

               if (nfc_get_domain() == domain) {
                   // SNEP events handled here
                   handleNfcEvent(event);
```

Detect

```
void NfcWorker::handleNfcEvent(bps_event_t *event) {

  uint16_t code = bps_event_get_code(event);

   if (NFC_SNEP_CONNECTION_EVENT == code) {
        sendTurnData(event);
   } else if (NFC_NDEF_PUSH_SUCCEED_EVENT == code) {
       // success! Our message was pushed to the other device
   } else if (NFC_NDEF_PUSH_FAILURE_IO_ERROR_EVENT == code) {
       // failure! Suggest the user tries again
   }
}
```

Detect

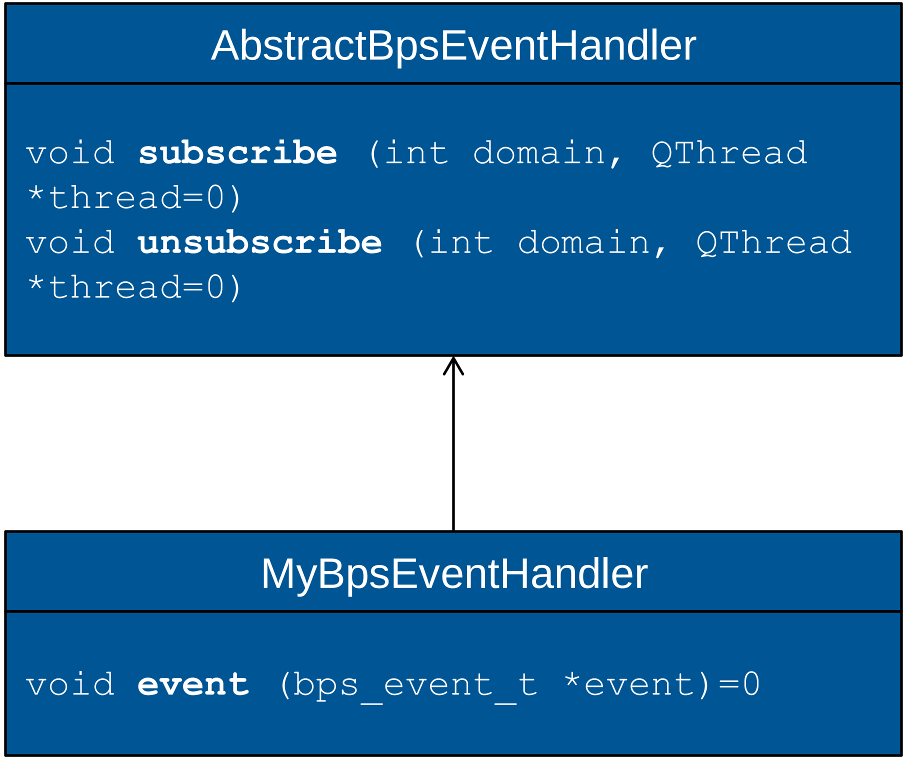
```
void NfcWorker::sendTurnData(bps_event_t *bps_event) {

    // get the NFC event object from the BPS event object
    nfc_get_nfc_event(bps_event, &nfc_event);

    // target is like a reference to the other device
    // once we have a target, we can push data to it over SNEP

    nfc_get_target(nfc_event, &target);
```

Respond

```
// make our message payload
QByteArray ba;
ba.resize(9);
ba[0] = 2; // turn over message ID
......
nfc_create_ndef_record(NDEF_TNF_EXTERNAL,
   domain_plus_type.toUtf8().constData(), payload, 9, 0, &record);

nfc_create_ndef_message(&myNdefMessage);

nfc_add_ndef_record(myNdefMessage, myNdefRecord));

nfc_push_ndef_message(target, myNdefMessage);
```

Last but not least….

- AbstractBpsEventHandler makes the “detect step” easier
- Create a sub-class
- subscribe to particular event domains such as nfc_get_domain()
- Receive BPS events via call backs to your event method



AbstractBpsEventHandler

```
void subscribe (int domain, QThread
*thread=0)
void unsubscribe (int domain, QThread
*thread=0)
```

MyBpsEventHandler

```
void event (bps_event_t *event)=0
```

Closing Comments

Resources

Articles

- http://tinyurl.com/cqfzf7y

Code

- https://github.com/blackberry/Cascades-Community-Samples

Contacts

- Martin Woolley @mdwrim
- John Murray @jcmrim
- Rob P Williams @robbieDubya

Let’s Talk More

BlackBerry Jam Americas

- Join the roundtable for this session at 6pm tonight in Exhibit Hall 1
- Continue the conversation on Twitter using the hashtag #BBJam and the session ID, #JAM14

Don’t Forget

BlackBerry Jam Americas

- Download the BlackBerry Jam Americas Mobile Conference Guide from BlackBerry App World
- The Appreciation Event is tonight at 7pm across the street at the Tech Museum
- Give us your Reasons to Believe at the 10k Reasons to Believe booth in the Jam Space

BlackBerry Jam Americas

THANK YOU

JAM14

Martin Woolley and Rob Williams

September 25-27, 2012