

NFCセミナー資料(講演会)

2013/10/05

早瀬

自己紹介

- 氏名：早瀬 潤也
- 住処：愛媛県松山市在住
- 所属：GDG四国 ※コミュニティ
- 仕事：プログラマ/SE

コミュニティ紹介 - GDG四国

四国地方をカバーする

Google Developer Group です。

Googleのテクノロジー全般について、
四国4県を巡りながら勉強会を開催しています。

- Android
- Google App Engine
- Googleが提供している各種APIなど

HTML5もくもく会@松山



DevFest (Google I/O報告会)



本題の前に...

Q1.NFCをご存知の方？

Q2.Androidアプリ開発方法を
ご存知の方？

Q3. プログラミング (Java) が
できる方？

自己紹介タイム

- 自己紹介
- 参加動機、得たいこと
- 何か告知したいことがあればどうぞ

目的

- NFCについて知る(1日目)
- “NFC歌留多”のアイデア出し(1日目)
- Android + NFCについて知る(2日目)

Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

“NFC歌留多”とは

- 四国情報通信懇談会 平成25年度調査研究事業
- NFCタグシールを張り付けたカードのこと
- 弱視者や高齢者などの情報弱者も含め、
だれもが参加可能な情報社会を実現するための
有用な情報端末としての「タブレット」や「スマートフォン」の
操作を簡単に行える手段として開発する

「NFC歌留多」によるスマートフォン操作の簡単化と スマートハウスアプリへの応用研究

き 気温、今
何度？
(点字版)



ね 熱中症に
ご用心



ま 孫に電話



て テレビの
スイッチ、
毎朝8時



本研究では

- ・流通している、NFCタグシールとアプリケーションランチャを使って、スマホの操作簡単化を実現
- ・情報弱者の身の回りを気づかっている人が、本人の代わりに設定してあげられるようになるための、教材開発と講習会を開催
- ・健常者も、使って嬉しいスマートハウスアプリへの応用研究

かるたにタッチして、特定アプリだけを起動

→ 簡単に操作できるスマホ/タブをおじいちゃん、おばあちゃんにプレゼント

考えてみよう！

- NFCを使って操作を楽にできないか？
- NFCを使って遊べないか？



Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

NFCとは

Near Field Communication

近距離無線通信

NFCとは

NFC通信規格対応の”ICチップ”を搭載している
機器やカードを、近距離で「かざす」だけで
データの読み書きや通信が可能

NFCとは

- 近距離 = 10cm程度

※ただし周り(筐体など)の影響を受けます

- 周波数帯 = 13.56MHz

N-mark

NFCサービスが利用可能な場所を示す




NFCの搭載が色々な機器に進めば、
お互いの機器を近づけ「かざす」だけで
さまざまなデータのやりとりが行えるようになります。

- デジタルカメラ
- デジタルサイネージ
- プリンター
- スマートデバイス(携帯電話等)
- パソコン
- テレビ

NFCとは - NFCの利点(QRコードと比較)

	QRコード	NFCタグ
読み取り方法	1.端末の電源をON 2.カメラを起動 3.カメラでQRコードを読み取る (URLを取得)	1.端末の電源をON 2.タッチ (URL取得)
読み取り時間	短い	短い
ユーザビリティ	カメラの起動が必要	タッチするだけ
コスト	印刷代のみ	タグが1枚80円以上
弱点	文字のみしか入力できない。 水などで滲むと読み取れない。	金属の近くにあると読み取りできない場合がある 直接水がかかると壊れる可能性がある
カバー率(携帯)	高い(カメラ機能があれば良い)	QRコードに比べると低い (NFC対応機種である必要がある)

NFCとは - NFCの利点

- デザインがカスタマイズできる
(QRコードは見た目が...) → 
- カメラが利用できない場所(暗所など)でも利用可能
- 文字列以外の情報(画像など)も載せることが可能
※ただし容量に注意

NFCとは - NFCの動作原理

1. リーダ／ライタのアンテナを使って微弱電波を発生



NFCリーダ／ライタ搭載端末



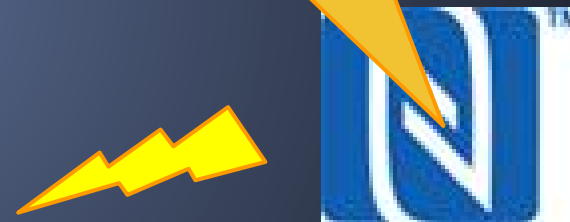
NFC(タグ等)

NFCとは - NFCの動作原理

2. 発生した微弱電波から電磁誘導によって電力を取得、
取得電力が規定値を上回った時点で
NFCタグ内の IC チップを起動

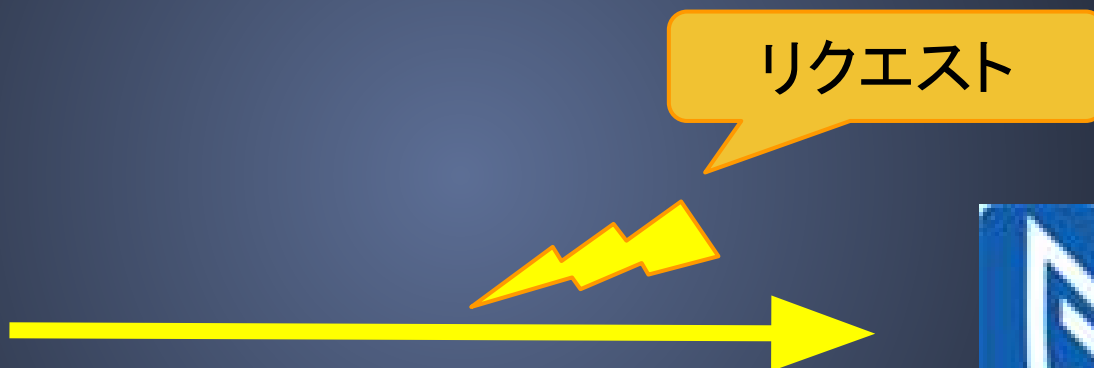


ICチップ起動



NFCとは - NFCの動作原理

3. リーダ／ライタは微弱電波に変調をかけて、
カードにリクエストデータを送信



NFCとは - NFCの動作原理

4. リーダ／ライタからのデータを受信したNFCタグは、リクエスト・データを処理し、リーダ／ライタにレスポンス・データを返信(負荷変動)する。



レスポンス

NFCとは - NFCの定義

【狭義のNFC】

国際標準規格として承認された13.56MHzの近距離無線通信技術のこと。

ISO/IEC18092 (NFC IP-1)、ISO/IEC21418 (NFC IP-2) など

既存のICカードとRFID用無線通信技術の集合体として定義されている。

【広義のNFC】

狭義のNFCを用いたシステム構築に必要な「機能」や「仕組み全般」を示す。

NFCとは - 規格

- NfcA -

ISO/IEC 14443, Type Aに準拠しているタグ

通信距離は10cm以下(近接型)

(MIFARE) NXPセミコンダクターズ社が開発

世界的にもっとも普及している。日本ではTaspoなど

- NfcF -

JIS 6319-4に準拠しているタグ

通信距離は10cm以下(近接型)

SuicaやPASMO、Edyなど。ソニーが開発

日本では使われているカードはほとんどがこの形式

高価であるが、他に比べ高速通信／高セキュリティを持つ

NFCとは - 規格

- NfcB -

ISO/IEC 14443, Type Bに準拠しているタグ
通信距離は10cm以下(近接型)

モトローラ社が開発

日本では免許証や住民基本台帳カード、パスポートなど
海外では電子マネー等に利用されている

- NfcV -

ISO/IEC 15693に準拠しているタグ
通信距離は70cm以下(近傍型)

RFIDとして、物流の管理やゲームセンターのカードに
利用されている

NFCとは - NFCフォーラム

NFC技術の利用価値向上を
ミッションとした業界標準団体

NFCとは - NFCフォーラムの活動

- デバイス, サービス間の互換性を実現するための仕様策定
- NFCフォーラム仕様を活用した商品開発のプロモーション
- NFC技術に関するグローバルな史上への啓蒙活動
- NFC機能を謳った商品のNFCフォーラムの準拠確認

NFCとは - NFCフォーラム

- website

<http://www.nfc-forum.org/home/>

- 仕様書

http://www.nfc-forum.org/specs/spec_list/

NFCとは - NFCのコンセプト

1. 国際的な通信互換性を提供すること
2. 端末に形状の自由度を持たせること

NFCとは - NFCのコンセプト

1. 国際的な通信互換性を提供すること

世界各地で利用されている異なる非接触ICカードの技術をまとめることによって、国際的な互換性を実現し、

- ・ユーザーの利便性の向上
- ・市場の拡大
- ・アプリケーション開発の促進

などを図る。

NFCとは - NFCのコンセプト

2. 端末に形状の自由度を持たせること

従来はカードとリーダー／ライター端末(店舗用端末など)に限定されていた非接触ICカード製品の形状に自由度を持たせ、携帯電話・PC・テレビといった民生機器に搭載することや、ID-1(クレジットカードのサイズ)以外の形状のシールタグとして提供することも、NFCの狙いの一つ

NFCとは - 形状の自由度

- ・シールタイプ
- ・アクセサリタイプ
- ・カードタイプ
- ・機器内蔵タイプ

※公演時は画像付きで紹介しましたが本資料からは画像削除

Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

NFCでできること

1. リーダライタ機能
2. カードエミュレーション機能
3. 端末間通信機能(P2P)

NFCでできること①

【リーダライタ機能】

NFCが搭載された電子機器は「決済端末」や「ハンディリーダー」として様々な非接触ICカードやタグの”読み書き”ができます。

NFCでできること① - リーダライタ機能

- クーポンやキャンペーン情報を表示する
- 地図情報を表示する
- メールを送信する
- 画像を表示する
- 特定のアプリケーションを開く
- 電話をかける

NFCでできること②

【カードエミュレーション機能】

これまで非接触ICカードが実現してきたクレジットや電子マネー決済など、カードのエミュレーション(「Mifare」や「FeliCa」のような動作)を行う。

NFCは様々なICカード規格と互換性があるため、複数の規格のICカードサービスを包含した展開が可能になります。

NFCでできること③

【端末間通信機能(P2P)】

NFC搭載の機器同士を「かざす」だけで、
「メールアドレス」「スケジュール」「画像データ」などを
交換(端末同士のデータ送受信)することができます。

Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

NFCの事例紹介①

学生証で健康管理

http://www.cseltd.co.jp/news/130805press_ku.htm

NFCの事例紹介②

ドアやスマホのロックを開けられる指輪

<http://news.mynavi.jp/news/2013/08/14/175/index.html>

※動画参照(54秒)

NFCの事例紹介③

街情報(店紹介/ルート案内)を得られる

<http://weekly.ascii.jp/elem/000/000/159/159059/>

※ URL参照

NFCの事例紹介④

イベント参加者情報／入退店管理

<http://www.brilliantservice.co.jp/product/machikon.html>

スタンプラリー

<http://www.brilliantservice.co.jp/product/nfcquest.html>

※ URL参照

NFCの事例紹介⑤

店舗にて遊べるゲーム

<http://www.mif-design.com/blog/2013/08/27-074345.php>

※ URL参照(動画 1:47秒)

NFCの事例紹介⑥

NFCを活用しているカメラ

https://www.sony.jp/ichigan/products/NEX-5T/feature_4.html#L1_390

※ URL参照

NFCの事例紹介⑦

NFCを活用しているプリンタ

http://k-tai.impress.co.jp/docs/news/20130826_612599.html

※ URL参照

Agenda

- “NFC歌留多”とは (目的)
- NFCについて (説明)
 1. NFCとは
 2. NFCでできること
 3. NFCの事例紹介
- “NFC歌留多”デモ (体験)

“NFC歌留多”デモ

本日は3種類のサンプルを用意しました。

1. 電話発信
2. 天気予報読み上げ
3. リレー制御(LED ON/OFF)

上記に加えて、

既存アプリケーションの紹介を行います。

既存ソフトウェアを利用する場合

- NFCタスクランチャー

NFCタグへの各種書き込みが簡単に行える

- NFC TagInfo

NFCタグの内容読み込みができる

※公演時は操作方法のスライドを用意しましたが本資料からは削除

デモ変更ネタ - 1

電話発信(NfcReaderActivity)の宛先(電話番号)を変更してみる。

電話発信(NfcReaderActivity)の resolveIntent処理を変更する。

```
private void resolveIntent(Intent intent) {  
    // NDEF のペイロードが含まれるタグがスキャンされたかどうかを判定する。  
    if (NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())) {  
        // タグからパースされた NDEF メッセージの配列を取得する。ACTION_NDEF_DISCOVEREDの場合は必須で送信される。  
        Parcelable[] ndefMessageData = intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);  
        if (ndefMessageData == null) {  
            Log.w(LOG_TAG, "NDEFメッセージが取得できませんでした。");  
            return;  
        }  
  
        String messageData = extractMessageData(ndefMessageData);  
        if ("biz.kokolab.telephone".equals(messageData)) {  
            // 1.電話をかけるIntentを投げる  
            startActivity(new Intent(Intent.ACTION_CALL, Uri.parse("tel:09099999999")));  
        } else if ("biz.kokolab.weather".equals(messageData)) {  
            startActivity(new Intent(this, WeatherSpeechActivity.class));  
        } else if ("biz.kokolab.relay".equals(messageData)) {  
            startActivity(new Intent(this, RelayControlActivity.class));  
        } else if ("biz.kokolab.ghimenews".equals(messageData)) {  
            startActivity(new Intent(this, EhimeNewsSpeechActivity.class));  
        } else {  
            Log.i(LOG_TAG, "想定外のタグデータ受信です: data = " + messageData);  
        }  
    } else {  
        Log.i(LOG_TAG, "想定外のIntent受信です: action = " + intent.getAction());  
    }  
}
```

電話発信(NfcReaderActivity)の resolveIntent処理を変更する。

```
private void resolveIntent(Intent intent) {  
    // NDEF のペイロードが含まれるタグがスキャンされた  
    if (NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())) {  
        // タグからパースされた NDEF メッセージの配列  
        Parcelable[] ndefMessageData = intent.getParcelableArrayListExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);  
        if (ndefMessageData == null) {  
            Log.w(LOG_TAG, "NDEFメッセージが取得できませんでした");  
            return;  
        }  
  
        String messageData = extractMessageData(ndefMessageData[0]);  
        if ("biz.kokolab.telephone".equals(messageData)) {  
            // 1.電話をかけるIntentを投げる  
            startActivity(new Intent(Intent.ACTION_CALL, Uri.parse("tel:09099999999")));  
        } else if ("biz.kokolab.weather".equals(messageData)) {  
            startActivity(new Intent(this, WeatherSpeechActivity.class));  
        } else if ("biz.kokolab.relay".equals(messageData)) {  
            startActivity(new Intent(this, RelayControlActivity.class));  
        } else if ("biz.kokolab.ahimehnews".equals(messageData)) {  
            startActivity(new Intent(this, AhimehNewsSpeechActivity.class));  
        } else {  
            Log.i(LOG_TAG, "想定外のタグデータ受信です: data = " + messageData);  
        }  
    } else {  
        Log.i(LOG_TAG, "想定外のIntent受信です: action = " + intent.getAction());  
    }  
}
```


startActivity(他Activityの呼び出し)にて
自分が開発したクラスではなく、
このような形式**(暗黙的Intent発行)**で
指定することで、電話アプリを起動している。

電話発信の場合、 tel:電話番号 という形式

変更例

tel:の電話番号を変更

```
private void resolveIntent(Intent intent) {  
    // NDEF のペイロードが含まれるタグがスキャンされたかどうかを判定する。  
    if (NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())) {  
        // タグからパースされた NDEF メッセージの配列を取得する。ACTION_NDEF_DISCOVEREDの場合は必須で送信される。  
        Parcelable[] ndefMessageData = intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);  
        if (ndefMessageData == null) {  
            Log.w(LOG_TAG, "NDEFメッセージが取得できませんでした。");  
            return;  
        }  
  
        String messageData = extractMessageData(ndefMessageData);  
        if ("biz.kokolab.telephone".equals(messageData)) {  
            // 1.電話をかけるIntentを投げる  
            startActivity(new Intent(Intent.ACTION_CALL, Uri.parse("tel:08070268315")));  
        } else if ("biz.kokolab.weather".equals(messageData)) {  
            startActivity(new Intent(this, WeatherSpeechActivity.class));  
        } else if ("biz.kokolab.relay".equals(messageData)) {  
            startActivity(new Intent(this, RelayControlActivity.class));  
        } else if ("biz.kokolab.ahimehnews".equals(messageData)) {  
            startActivity(new Intent(this, AhimehNewsSpeechActivity.class));  
        } else {  
            Log.i(LOG_TAG, "想定外のタグデータ受信です: data = " + messageData);  
        }  
    } else {  
        Log.i(LOG_TAG, "想定外のIntent受信です: action = " + intent.getAction());  
    }  
}
```



暗黙的Intentの例

ブラウザを開く

```
startActivity(new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.google.com")));
```

Mapを開く

```
startActivity(new Intent(Intent.ACTION_VIEW, Uri.parse("geo:0,0?q=Tokyo")));
```

カメラを開く

```
startActivity(new Intent(MediaStore.ACTION_IMAGE_CAPTURE));
```


※「カメラを開く」について、エラーがでた場合(赤文字)は
ポップアップ表示時に**Alt+Enter**で**import**を追加

変更例

ブラウザを開く

```
private void resolveIntent(Intent intent) {
    // NDEF のペイロードが含まれるタグがスキャンされたかどうかを判定する。
    if (NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())) {
        // タグからパースされた NDEF メッセージの配列を取得する。ACTION_NDEF_DISCOVEREDの場合は必須で送信される。
        Parcelable[] ndefMessageData = intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
        if (ndefMessageData == null) {
            Log.w(LOG_TAG, "NDEFメッセージが取得できませんでした。");
            return;
        }

        String messageData = extractMessageData(ndefMessageData);
        if ("biz.kokolab.telephone".equals(messageData)) {
            // 1.電話をかけるIntentを投げる
            startActivity(new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.google.co.jp/")));
        } else if ("biz.kokolab.weather".equals(messageData)) {
            startActivity(new Intent(this, WeatherSpeechActivity.class));
        } else if ("biz.kokolab.relay".equals(messageData)) {
            startActivity(new Intent(this, RelayControlActivity.class));
        } else if ("biz.kokolab.ehimenews".equals(messageData)) {
            startActivity(new Intent(this, EhimeNewsSpeechActivity.class));
        } else {
            Log.i(LOG_TAG, "想定外のタグデータ受信です: data = " + messageData);
        }
    } else {
        Log.i(LOG_TAG, "想定外のIntent受信です: action = " + intent.getAction());
    }
}
```



デモ変更ネタ - 2

天気読みあげ(WeatherSpeechActivity)の
読みあげるテキストを天気以外に変更してみる。

読みあげ(WeatherSpeechActivity)の doInBackground処理を変更する。

```
protected Map<String, Object> doInBackground(Void... voids) {
    try {
        final String speechText = findWeatherSpeechText();
        byte[] speechMediaData = findSpeechMediaData(speechText);

        final File ttsTempFile = File.createTempFile("google tts", "mp3", WeatherSpeechActivity.this.getCacheDir());
        ttsTempFile.deleteOnExit();

        writeDataToTmpFile(ttsTempFile, speechMediaData);

        return new HashMap<String, Object>() {{
            put("text", speechText);
            put("file", ttsTempFile);
        }};
    } catch (Throwable e) {
        mBackgroundError = e;
    }

    return null;
}
```

この行(findWeatherSpeechText())で
お天気情報を取得している。
String = 文字列型。
speechTextに取得した文字列が設定される

```
protected Map<String, Object> doInBackground(Void... voids) {
    try {
        final String speechText = findWeatherSpeechText();
        byte[] speechMediaData = findSpeechMediaData(speechText);

        final File ttsTempFile = File.createTempFile("googletts", "mp3", WeatherSpeechActivity.this.getCacheDir());
        ttsTempFile.deleteOnExit();

        writeToTmpFile(ttsTempFile, speechMediaData);

        return new HashMap<String, Object>() {{
            put("text", speechText);
            put("file", ttsTempFile);
        }};
    } catch (Throwable e) {
        mBackgroundError = e;
    }

    return null;
}
```

この行(findSpeechMediaData)で
文字列(天気情報)を音声に変換している。
⇒ speechTextを変更すれば
読みあげる文章を変更できる

```
protected Map<String, Object> doInBackground(Void... voids) {
    try {
        final String speechText = findWeatherSpeechText();
        byte[] speechMediaData = findSpeechMediaData(speechText);

        final File ttsTempFile = File.createTempFile("google_tts", "mp3", WeatherSpeechActivity.this.getCacheDir());
        ttsTempFile.deleteOnExit();

        writeToTmpFile(ttsTempFile, speechMediaData);


        return new HashMap<String, Object>() {{
            put("text", speechText);
            put("file", ttsTempFile);
        }};
    } catch (Throwable e) {
        mBackgroundError = e;
    }

    return null;
}
```

変更例

変数speechTextの文字列を
読みあげたい文字列(固定)に変更

```
protected Map<String, Object> doInBackground(Void... voids) {  
    try {  
        final String speechText = "本日は晴れです";  
        byte[] speechMediaData = findSpeechMediaData(speechText);  
  
        final File ttsTempFile = File.createTempFile("googletts", "mp3", WeatherSpeechActivity.this.getCacheDir());  
        ttsTempFile.deleteOnExit();  
  
        writeDataToTmpFile(ttsTempFile, speechMediaData);  
  
        return new HashMap<String, Object>() {{  
            put("text", speechText);  
            put("file", ttsTempFile);  
        }};  
    } catch (Throwable e) {  
        mBackgroundError = e;  
    }  
  
    return null;  
}
```



デモ変更ネタ - 3

リレー制御(ReleayControlActivity)の
挙動を変更してみる。

リレー制御(RelayControlActivity)の SendRelayControlCommand処理を変更する。

```
private class SendRelayControlCommand implements Runnable {

    /**
     * {@inheritDoc}
     */
    @Override
    public void run() {
        byte value = (byte) (isOn.get() ? 0x1 : 0x0);
        byte[] buffer = new byte[1];

        // 1バイトのみのプロトコルデータ
        buffer[0] = value;
        try {

            //出力ストリームにbuffer[]配列データを書き込む(?)
            mUsbOut.write(buffer);

            isOn.getAndSet(!isOn.get());
        } catch (IOException e) {
        }
    }
}
```

```
private class SendRelayCont
```

```
/**
```

```
 * {@inheritDoc}
```

```
 */
```

```
@Override
```

```
public void run() {
```

```
    byte value = (byte) (isOn.get() ? 0x1 : 0x0);
```

```
    byte[] buffer = new byte[1];
```

```
    // 1バイトのみのプロトコルデータ
```

```
    buffer[0] = value;
```

```
    try {
```

```
        //出力ストリームにbuffer[]配列データを書き込む(7)
```

```
        mUsbOut.write(buffer);
```

```
        isOn.getAndSet(!isOn.get());
```

```
    } catch (IOException e) {
```

```
    }
```

```
}
```

```
}
```

NFCタグをタッチするたびに本処理を行う。
valueには1 か 0の値が設定される。

value = 1 : Arduino側の端子をHIGH

value = 0 : Arduino側の端子をLOW

```

private class SendRelayControlCommand implements Runnable {

    /**
     * {@inheritDoc}
     */
    @Override
    public void run() {
        byte value = (byte) (isOn ? 1 : 0);
        byte[] buffer = new byte[1];

        // 1バイトのみのプロトコルデータ
        buffer[0] = value;
        try {

            //出力ストリームにbuffer[]配列データを書き込む(7)
            mUsbOut.write(buffer);

            isOn.getAndSet(!isOn.get());
        } catch (IOException e) {
        }
    }
}

```

mUsbOut.write()でUSBを利用して
Arduino側にデータ(0 or 1)を送信する

変更例

HIGH -> 5000ミリ秒待機 -> LOWと切り替える。

```
private class SendRelayControlCommand implements Runnable {  
  
    /**  
     * {@inheritDoc}  
     */  
    @Override  
    public void run() {  
        byte[] buffer = new byte[1];  
  
        try {  
            // HIGH  
            buffer[0] = 0x01;  
            mUsbOut.write(buffer);  
  
            // 5000ミリ秒待機(HIGHのまま)  
            try {  
                Thread.sleep(5000);  
            } catch (InterruptedException e) {}  
  
            // LOW  
            buffer[0] = 0x00;  
            mUsbOut.write(buffer);  
        } catch (IOException e) {}  
    }  
}
```

Thread.sleep(...);
⇒ ...ミリ秒だけ処理を待機する