

## 補足資料

### Intentによるアプリケーションとアクティビティの呼出し

#### Android アプリのキモとなるIntentとは何？

「Intent」(呼び出し要求)とは Android 独自の機能です。簡単にいえばアプリケーションや他のアクティビティを呼び出す機能ですが、他のアプリケーションを機能や扱うデータ型式で“検索”して呼び出すことができます。

たとえば、ウェブブラウザを呼び出したい場合、「View」(データの表示)というアクションと URL というデータを指定します。ウェブブラウザの名前などを指定する必要がありません。このIntentが実行されると、Android は、URL を扱うことができ、かつそれを表示できる自身が持つアプリケーションの中から探し、そのアプリケーションに URL を渡して表示を行なわせます。もし、複数のアプリケーションがある場合は、ユーザーがどのアプリケーションを使うのかを選ぶダイアログが表示されます。

	<p>Android 端末で追加のウェブブラウザをインストールしたのちに、何らかのアプリケーションで URL をタッチすると、アプリの選択画面が表示される。これは Intent によって、ウェブブラウザを呼び出している例だ</p>
---	---

このようにアプリケーションで\_intentを使うと、コード中にブラウザの名前を入れる必要もないし、実際の実行時に別のブラウザに置き換わっていたり、複数のブラウザが端末にあっても問題ないのです。

また逆にブラウザを起動するアプリケーションがあったとしても、ユーザーは別のブラウザをインストールしたら、そちらを起動するようにも指定できます。このようにアプリケーション同士を直接相互依存させるのではなく、実際に行なう作業(アクションといいます)を指定して、外部のアプリケーションと間接的に関係させることができるわけです。(これを「**暗黙的intent**」と呼ぶ)

また、intentは1つのアプリケーションの中で、**アクティビティを切り替えるのにも**利用します。アプリケーションは、必ずアクティビティを含むため、intentはアクティビティを呼び出しているともいえます。つまり、アプリケーション内での自作した別のアクティビティの呼び出しは、intentの特殊な形と考えることができるわけです。この場合は、機能などでなく、直接別のアクティビティ名を使い、呼び出します。(これを「**明示的intent**」と呼ぶ)

## まずは簡単な例(暗黙的intent)を作成してみる

今回は、壁紙を変更するアクションを呼び出すソースコードを用意しました。

### ●AnmokuIntentDemo.java

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.content.Intent;

/**
 * 暗黙的intentの一例。壁紙を変更するアクションを呼び出す
 */
public class AnmokuIntentDemo extends Activity
    implements OnClickListener {

    /** 起動時処理 */
```

```

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // main.xml レイアウトファイルを読み込み、画面に設定
    setContentView(R.layout.main);

    /* main.xml ファイル内のidを指定してボタンを取得し、
    イベント処理を設定*/
    ImageButton button =
        (ImageButton) this.findViewById(R.id.Button);
    button.setOnClickListener(this);
}

/** ボタンクリック時のイベント処理内容 */
public void onClick(View view) {
    /* 壁紙を変更するアクション処理要求 (intent) を作成
    とくに今回は必要がないのでデータは送らない。該当する任意
    のアプリケーションが起動 */
    Intent intent = new Intent(Intent.ACTION_SET_WALLPAPER);
    // intentを送信
    startActivity(intent);
}
} //classの終わり

```

### ●res/layout/main.xml (レイアウト指定 XML ファイル)

```

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:orientation="vertical"

    android:layout_width="fill_parent"

    android:layout_height="fill_parent"

    >

```

<!-- 画像ファイルを扱う場合は res/drawable/ 以下に画像ファイル xxx を配置し、@drawable/xxx で指定すれば良い。 -->

```
<ImageButton android:id="@+id/Button"

    android:src="@drawable/icon"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content">

</ImageButton>
```

```
</LinearLayout>
```

Intent の最も簡単な例は、自作したアプリケーション(アクティビティ)から別アプリケーションを呼び出す場合です。まずは Android の画像系を扱うアプリケーションを呼び出してみましょう。

Android 自身が用意している Intent に関しては、Android Developer の Reference に情報が 있습니다。ただし、一カ所にすべてまとまっているわけではなく、Intent クラスで規定する標準のアクションと、他のクラス(たとえば、設定関連のアクションは、Settings クラスなど)が持つアクション(アクティビティの表示)に分かれています。今回は Intent クラスに定数定義があります。

```
Intent intent = new Intent( Intent.ACTION_SET_WALLPAPER );
startActivity(intent);
```

1 行目は **Intent オブジェクトの作成**で、このとき引数としてアクションを渡します。今回のアクション(Intent.ACTION\_SET\_WALLPAPER)は、任意の背景画像を設定するもので、データを別途必要としないので、とくにになにもせずに startActivity でアクティビティを表示させることができます。

これが暗黙的 Intent の一例です。このようにスマートフォン内の他のアプリを簡単に起動することができます。

以下は、URL を指定してブラウザを起動する暗黙的インテントの例です。

#### ●BrouserIntentDemo.java

```
import android.app.Activity;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.content.Intent;

/**
 * 暗黙的インテントの一例。ブラウザを起動するアクションを呼び出す
 */
public class BrouserIntentDemo extends Activity
    implements OnClickListener {

    /** 起動時処理 */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // main.xml レイアウトファイルを読み込み、画面に設定
        setContentView(R.layout.main);

        // main.xml ファイル内のidを指定してボタンを取得
        Button button =
            (Button) this.findViewById(R.id.startBrouserButton);
        button.setOnClickListener(this);
    }

    /** ボタンクリック時のイベント処理内容 */
    public void onClick(View view) {
        //データの場所を特定するオブジェクトを生成
        Uri uri = Uri.parse("http://www.google.co.jp");
        //アクションは指定場所のデータをユーザに表示する
        Intent i = new Intent(Intent.ACTION_VIEW, uri);
    }
}
```

```

        //起動
        startActivity(i);
    }
} //class の終わり

```

### ●res/layout/main.xml (レイアウト指定 XML ファイル)

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <Button android:text="検索サイトへ行く"
        android:id="@+id/startBrowserButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

    </Button>

</LinearLayout>

```

---

## 暗黙的インテントに設定する ACTION\_\* ★★★★

暗黙的インテントがアプリ内部から発行されると、そのアクションに対応できるアクティビティをもつアプリが選択されて、それが一つの場合は起動、複数の場合は選択ダイアログが表示される。

### ACTION\_\*の種類

---

- ACTION\_CALL
- ACTION\_VIEW

- ACTION\_SEND
- ACTION\_SET\_WALLPAPER
- ACTION\_WEB\_SEARCH

など多数の種類がある。

<http://developer.android.com/intl/ja/reference/android/content/Intent.html>

#### メールを起動させる場合

---

```
Intent it = new Intent();
it.setAction(Intent.ACTION_SENDTO);
it.setData(Uri.parse("mailto:hoge@hage.com"));
it.putExtra(Intent.EXTRA_SUBJECT, "サンプルコードの件");
it.putExtra(Intent.EXTRA_TEXT, "あけましておめでとうございます。");
startActivity(it);
```

メールが、ACTION\_SENDTO に対応できるアクティビティをもっていれば、そのメールアプリが起動または選択肢の一つとして表示される。

#### ブラウザを起動させる場合

---

```
String text = inputText.getText().toString();
Uri url = Uri.parse(text);
Intent it = new Intent();
it.setAction(Intent.ACTION_VIEW);
it.setData(url);
startActivity(it);
```

ブラウザが、ACTION\_VIEW に対応できるアクティビティをもっていれば、そのブラウザアプリが起動または選択肢の一つとして表示される。

起動されるアプリ側ではどんな記述をされているか。

---

「あなたは、ACTION\_VIEW が暗黙的インテントで発行されたら反応しなさい。」

アプリから発行された「暗黙的Intent」の ACTION\_\* の種類は、起動される(受け取る)アプリ内ではどう記述されているか。

AndroidManifest.xml

```
<activity android:name=".ReceiveApplication"
          android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCHER" />
    </intent-filter>

    <intent-filter android:label="@string/app_name">
        <action android:name="android.intent.action.SEND" />
        <data android:mimeType="text/plain" />
        <category
android:name="android.intent.category.DEFAULT" />
    </intent-filter>

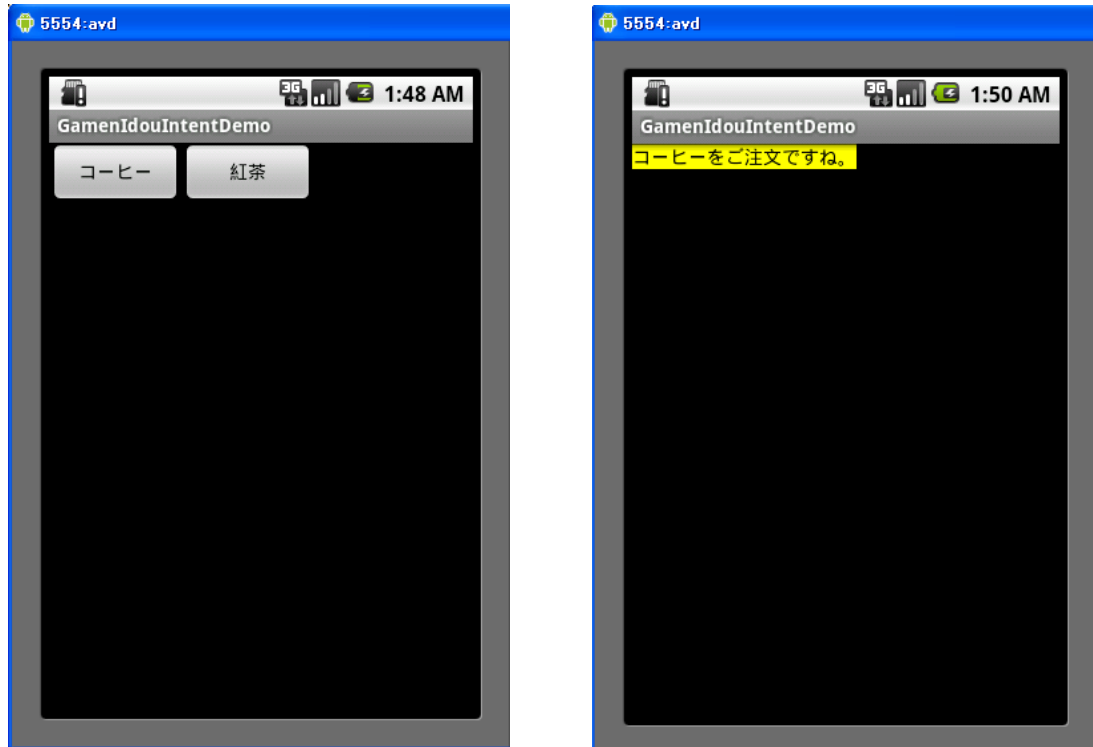
</activity>
```

ACTION\_SEND に対応して反応するように該当アクティビティにIntentフィルターを記述しておく。

次に明示的Intentの例を作成してみる



自作した別のアクティビティを呼び出す簡単な例を用意しました。最初の画面でコーヒーのボタンを選択すると、注文名が次画面に渡され、表示されます。



#### ●MainGamen. java

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.content.Intent;

public class MainGamen extends Activity implements OnClickListener {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

```

// main.xmlファイル内のidを指定してボタンを取得し、イベント処理を設定
Button coffeeButton =
    (Button) this.findViewById(R.id.coffeeButton);
coffeeButton.setOnClickListener(this);
Button teeButton =
    (Button) this.findViewById(R.id.teeButton);
teeButton.setOnClickListener(this);

}

public void onClick(View v) {
// このクラスオブジェクトと次画面のクラスオブジェクトをつなぐ
Intent orderIntent = new Intent(this, OrderGame.class);

//選択されたボタンを取得
Button orderButton = (Button)v;

//選択されたボタン上の文字列を取得
String orderName = (String) orderButton.getText();

//キーと値を付加情報としてセット
orderIntent.putExtra("注文", orderName);

//intentを送信
startActivity(orderIntent);

}
} //終わり

```

### ●res/layout/main.xml (レイアウト指定 XML ファイル)

```
<?xml version="1.0" encoding="utf-8"?>
```

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <Button android:text="コーヒー" android:id="@+id/coffeeButton"
        android:layout_width="100px"
        android:layout_height="wrap_content"></Button>

    <Button android:text="紅茶" android:id="@+id/teeButton"
        android:layout_width="100px"
        android:layout_height="wrap_content"></Button>

</LinearLayout>

```

---

次に呼び出される注文画面クラスを作ります。

### ● OrderGamen.java

```

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

public class OrderGamen extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.order);

        // main.xml ファイル内のidを指定してボタンを取得し、イベント処理を設定
        TextView orderTextView =
            (TextView) this.findViewById(R.id.orderTextView);
    }
}

```

```

        // getIntentでintentの中身を取得
        Intent intent = this.getIntent();

        // キーを使って送信された付加情報の値を取得し、表示
        String orderName = intent.getStringExtra("注文");
        orderTextView.setText(orderName + "をご注文ですね!");
    }
} // 終わり

```

### ●res/layout/order.xml (レイアウト指定 XML ファイル)

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <TextView
        android:background="#ffff00"
        android:textColor="#000000"
        android:id="@+id/orderTextView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="" />

</LinearLayout>

```

---

ここでは複数のアクティビティを使用しますが、アプリケーション中にアクティビティを追加する際にはアプリケーションのマニフェストファイル (**AndroidManifest.xml**) にアクティビティを登録する必要があります。

(メインのアクティビティについては **eclipse** に よって自動的に登録されているため、登録する必要はありません。)

以下がそのマニフェストファイルです。

## ●AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="intentedemo.GamenIdouIntentDemo" android:versionCode="1"
    android:versionName="1.0">
    <application
        android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity
            android:name=".MainGamen"
            android:label="@string/app_name">
            <intent-filter>
                <action
                    android:name="android.intent.action.MAIN" />
                <category
                    android:name="android.intent.category.LAUNCHER"
                    " />
            </intent-filter>
        </activity>

        <!-- 以下を追加 android:name にはアクティビティのクラス名を記載する。 -->
        <activity android:name=".OrderGamen">
        </activity>

    </application>
    <uses-sdk android:minSdkVersion="3" />

</manifest>
```

---