

日本Androidの会 浜松支部 第4回ミーティング Aug 13, 2011

Activity と Intent

有山圭二（日本Androidの会 関西支部）

■ MainActivityのコードを確認する

- どこにもHello Worldとは書かれていない

```
package org.sample.androidproject;

import android.app.Activity;
import android.os.Bundle;

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

■ Hello World の在処

- res/layout/main.xml

```
<?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"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
/>
</LinearLayout>
```

- res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, MainActivity!</string>
    <string name="app_name">AndroidProject</string>
</resources>
```

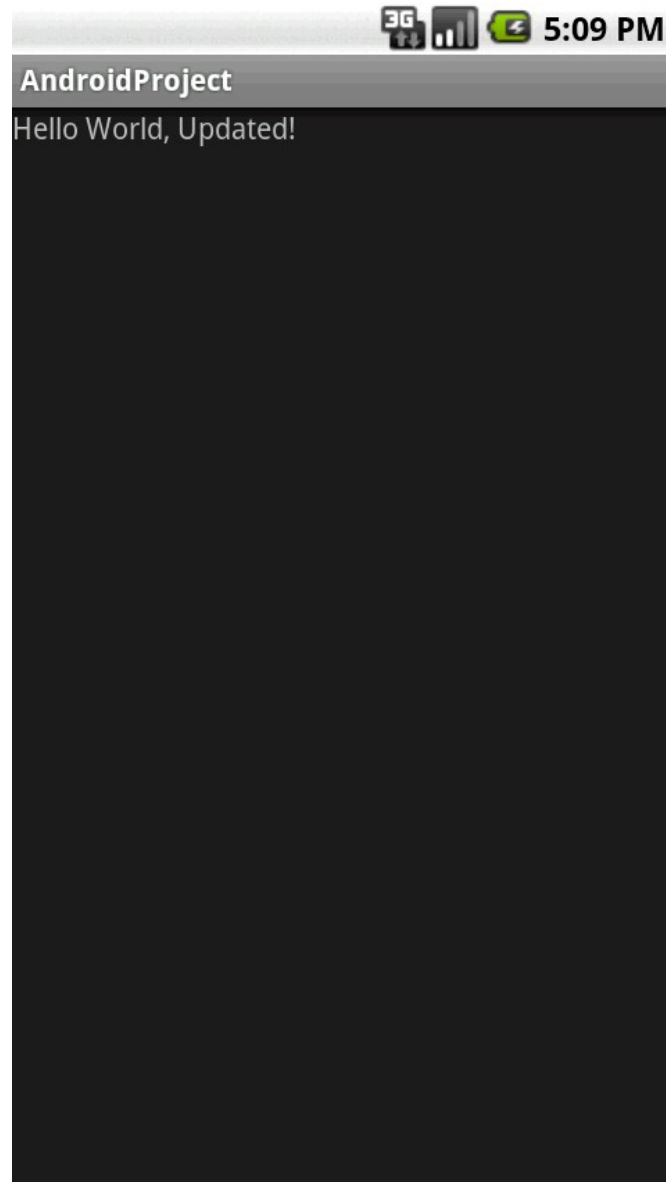
■ 文字列を変更する

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="hello">Hello World, Updated!</string>
  <string name="app_name">AndroidProject</string>
</resources>
```

変更

- 。 変更後、実行してみる

■ 実行結果



■ ボタンを追加する

- res/layout/main.xml

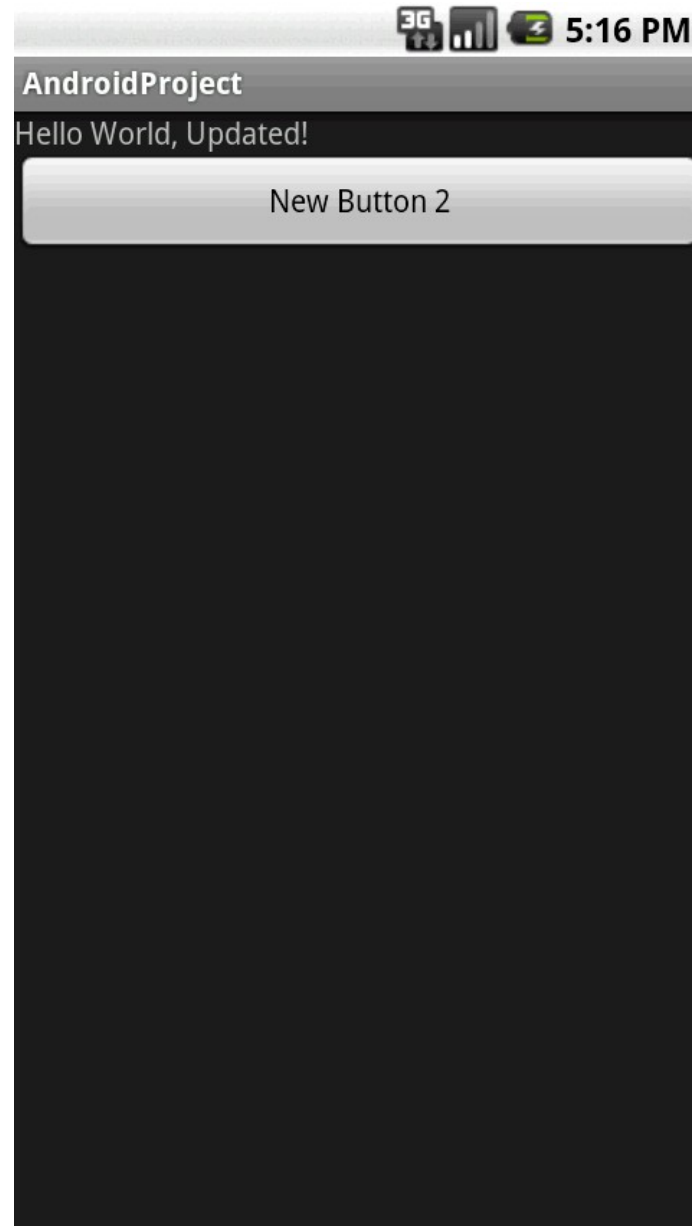
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello2"
    />
</LinearLayout>
```

追記

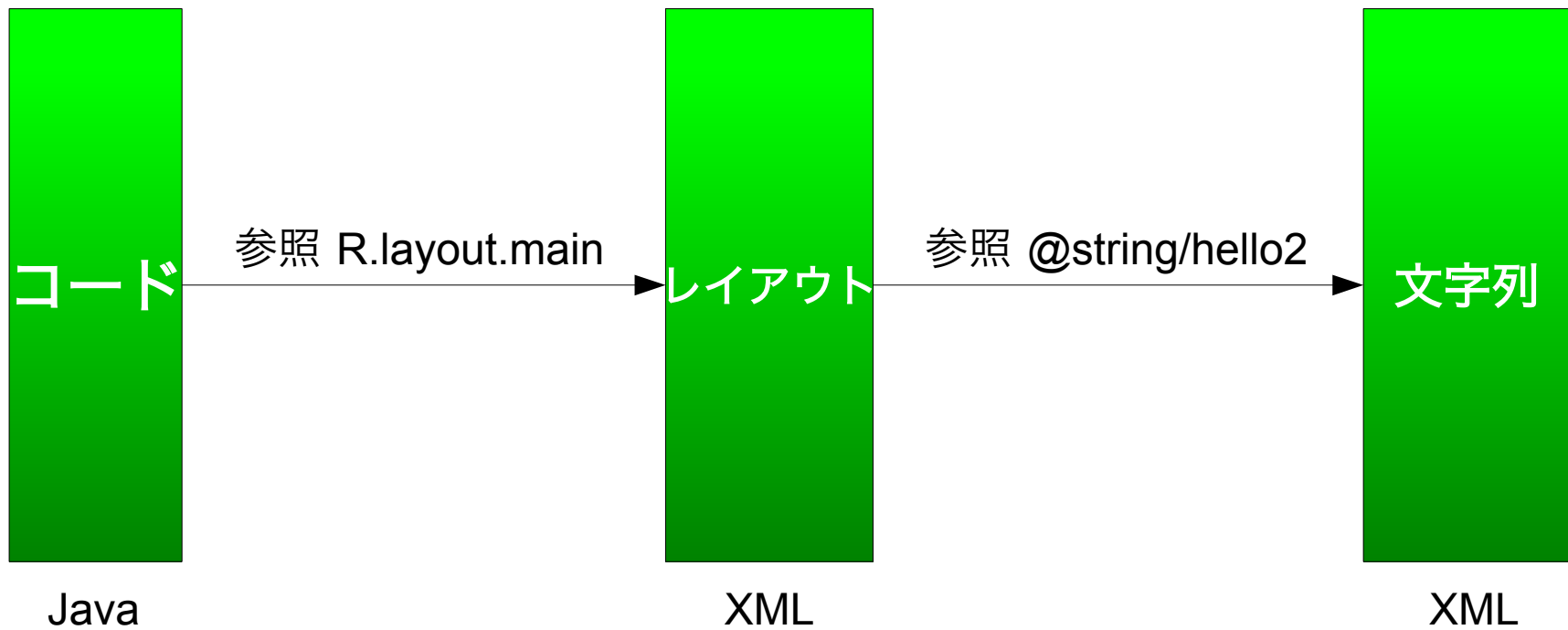
- res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, Updated!</string>
    <string name="hello2">New Button</string>
    <string name="app_name">AndroidProject</string>
</resources>
```

■ 実行結果



■ 参照関係





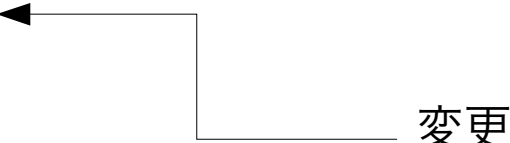
「コード」と「リソース」の分離

- 処理をする「コード」と、「リソース」を分離する
- リソースは、画面レイアウトや文字等の「値」、または画像等、コードとは切り離せる「データ」
- リソースは、resディレクトリ以下に配置
 - 画面レイアウト → layout
 - 値→values
 - 文字列 → string
 - 配列（数値や選択肢で利用する） → array
 - 画像 → drawable
 - アニメーション → anim
 - 音声等・その他データ → raw
 - etc...

■ 表示内容を変えてみる

- res/layout/main.xml

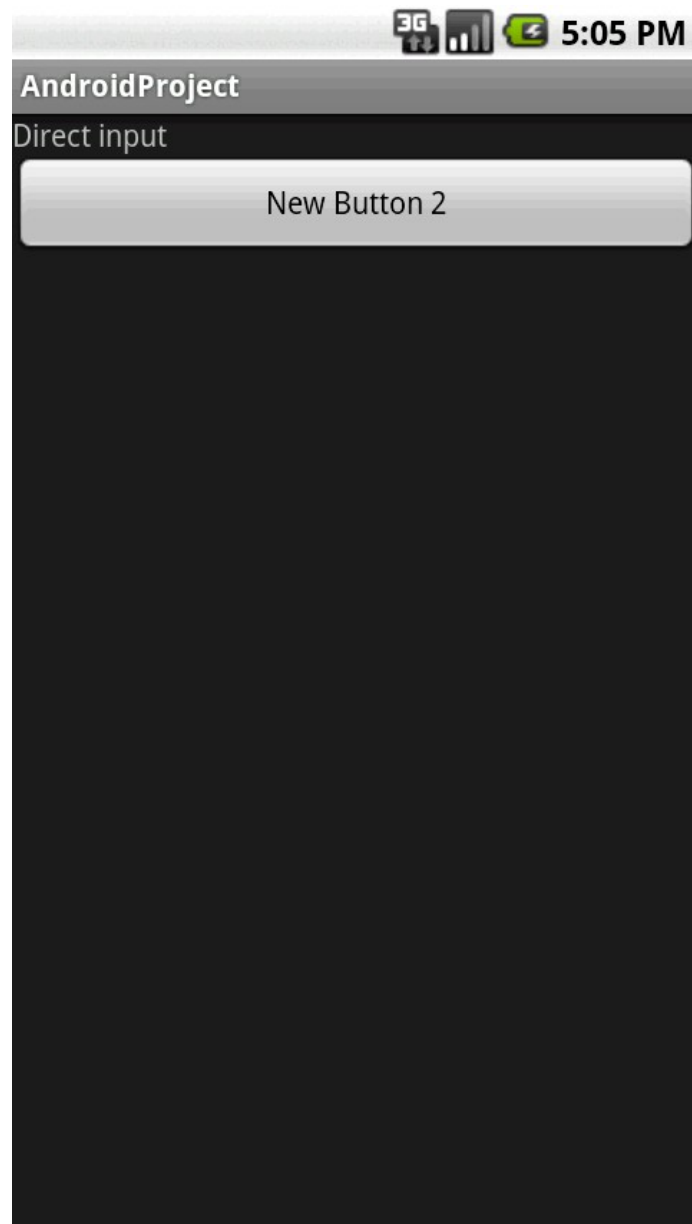
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Direct input"
/>
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello2"
/>
</LinearLayout>
```



変更

- res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, Updated!</string>
    <string name="hello2">New Button 2</string>
    <string name="app_name">AndroidProject</string>
</resources>
```



直接書けるじゃない？

■ 世界に羽ばたくAndroid

- Android端末は世界中で販売されている
- Android Marketを通じてアプリを配布可能な国は世界48カ国（2010/9/27 現在）

多言語対応、どうしますか？

■ Androidアプリの多言語対応

- res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, Updated!</string>
    <string name="hello2">New Button 2</string>
    <string name="app_name">AndroidProject</string>
</resources>
```

プロジェクトのresディレクトリ以下に、新しく "values-ja" ディレクトリを作成。
"values/strings.xml"をそのままコピーする。

- res/values-ja/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">更新したボタン</string>
    <string name="hello2">新しいボタン</string>
    <string name="app_name">日本語版 AndroidProject</string>
</resources>
```

ロケールの変更



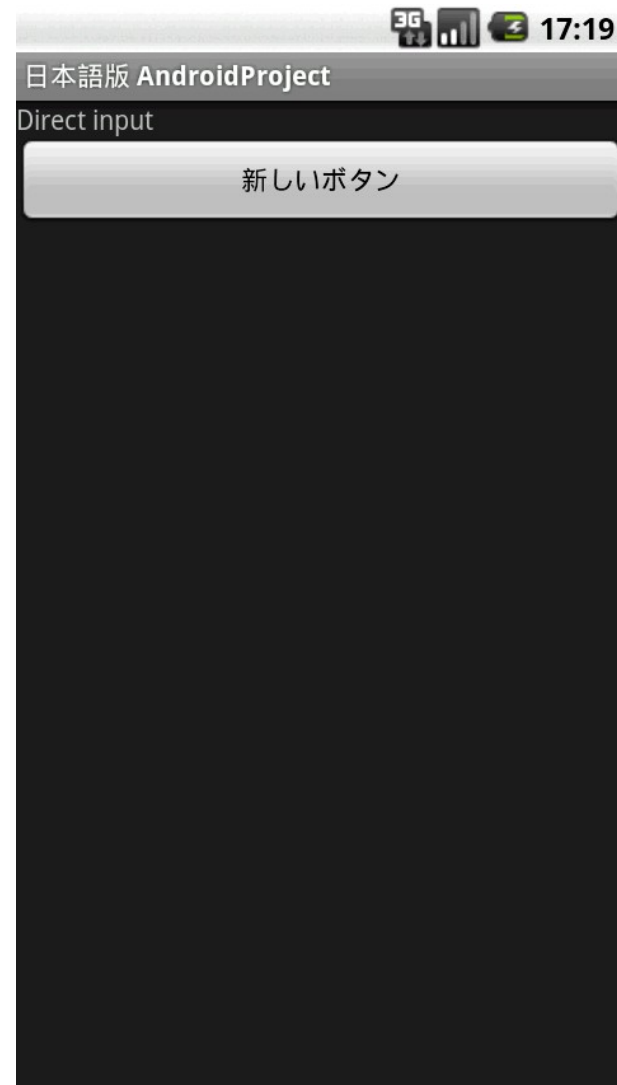
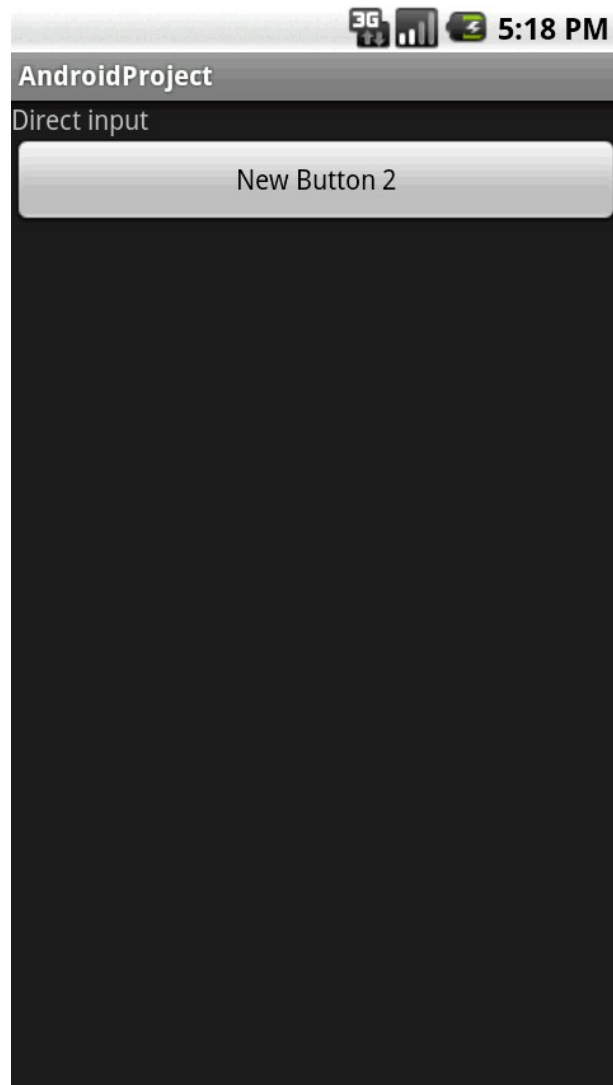
- Custom Locale を実行する

Custom Locale



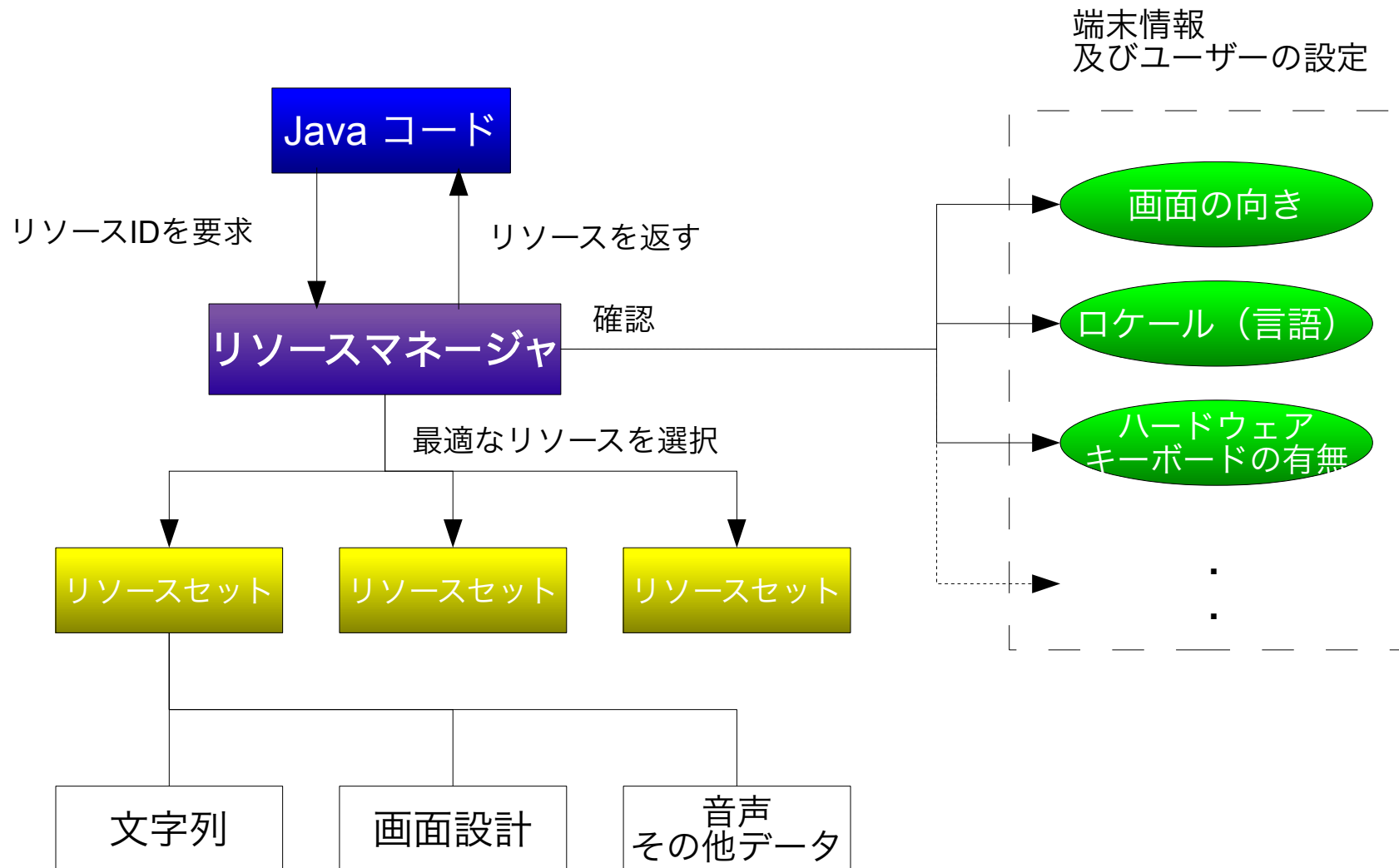
- 。 リストをスクロールして、
 - ja
 - ja-JPを、選択して
ロングタップ（押し続ける）
- 。 ダイアログが表示されるので、
 - Applyを、選択する

■ ロケールの変更後



選択する言語によって、自動的に表示が切り替わる

■ コードとリソースの分離



■ コードから画面のテキストを変更する

- 操作したいViewに、android:id属性を設定する
- res/layout/main.xml

```
<?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"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Direct input"
    />
<Button android:id="@+id/main_btn_test"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello2"
    />
</LinearLayout>
```

追記

■ コードから画面のテキストを変更する

- findViewById()で画面に表示されているオブジェクトの一つを特定して取得する。
- findViewByIdで取得した時点では、全てViewクラスとして扱われるので、適切なクラスにキャストする（この場合はButtonクラス）してから、setTextで文字列を設定する

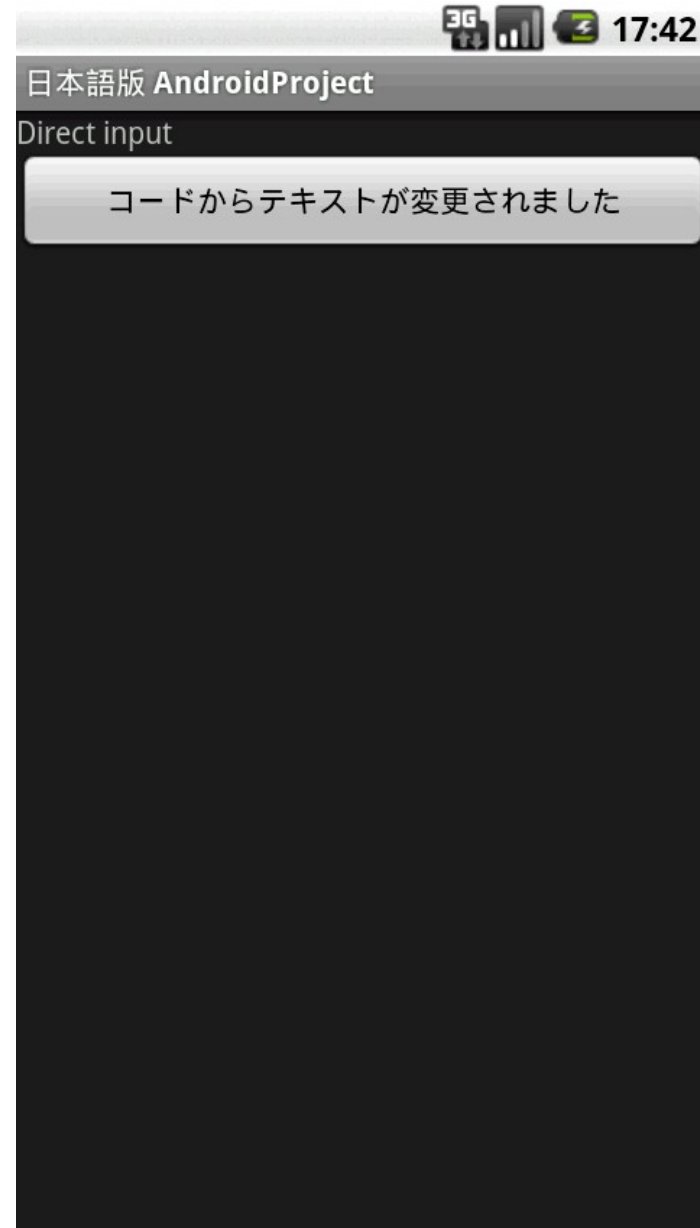
```
package org.sample.androidproject;

import android.app.Activity;
import android.os.Bundle;

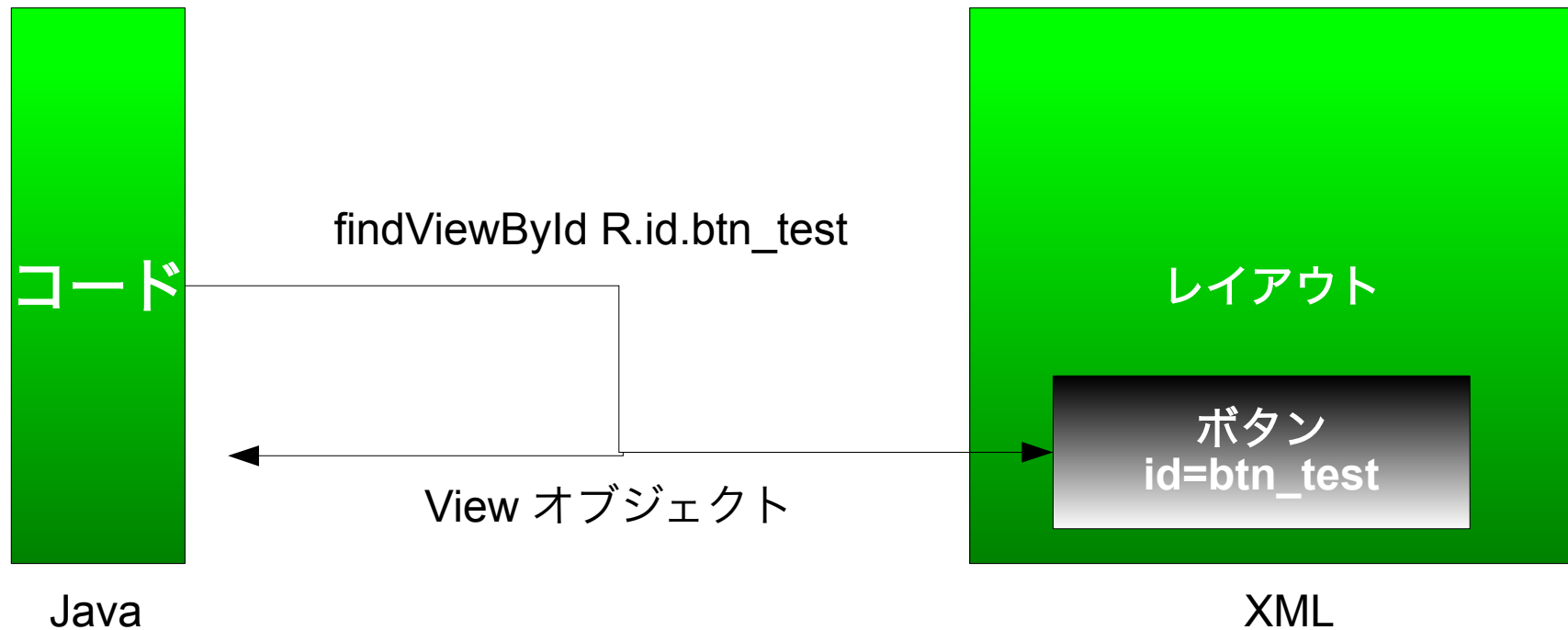
public class MainActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button btnTest = (Button) findViewById(R.id.main_btn_test);
        btnTest.setText("コードからテキストが変更されました");
    }
}
```

- 変更後、実行してみる

■ 実行結果



■ 参照関係



■ ボタンにクリックイベントを設定する

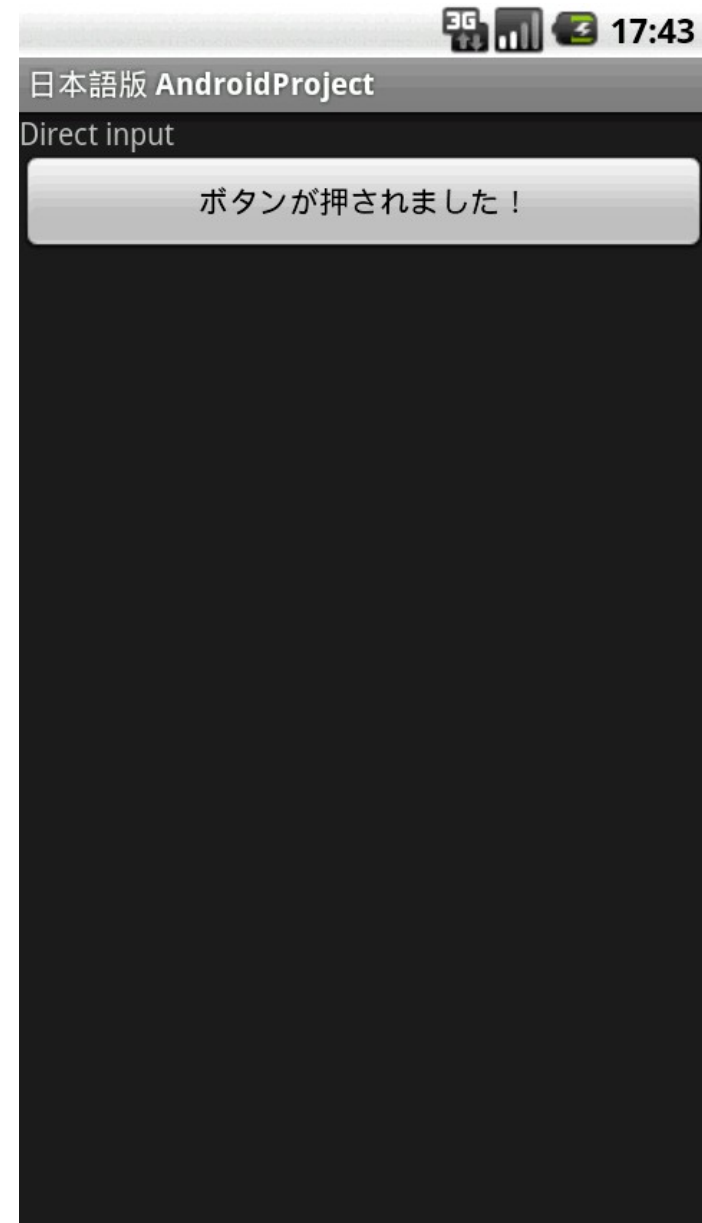
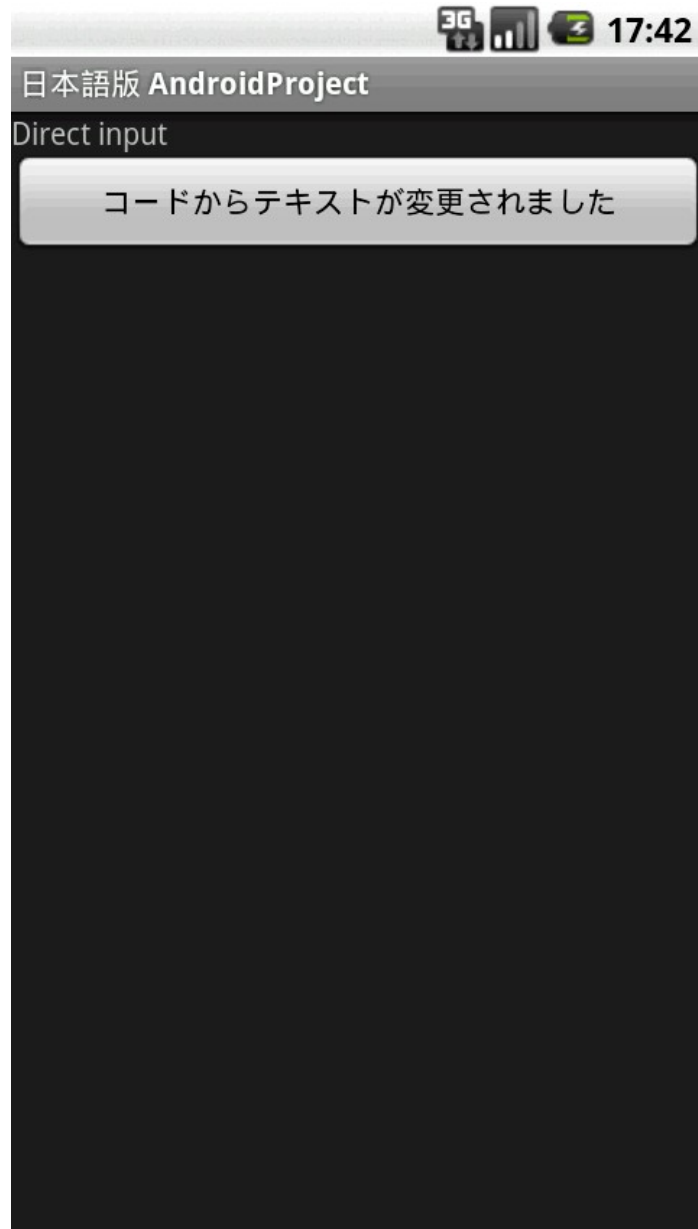
- ボタンが押されたときの処理を実装したOnClickListenerを、setOnClickListener()に渡す

```
package org.sample.androidproject;

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

public class MainActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button btnTest = (Button) findViewById(R.id.main_btn_test);
        btnTest.setText("コードからテキストが変更されました");
        btnTest.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                ((Button)v).setText("ボタンが押されました!");
            }
        });
    }
}
```

■ 実行結果



■ MainActivityのコードを確認する

```
package org.sample.androidproject;

import android.app.Activity;
import android.os.Bundle;

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

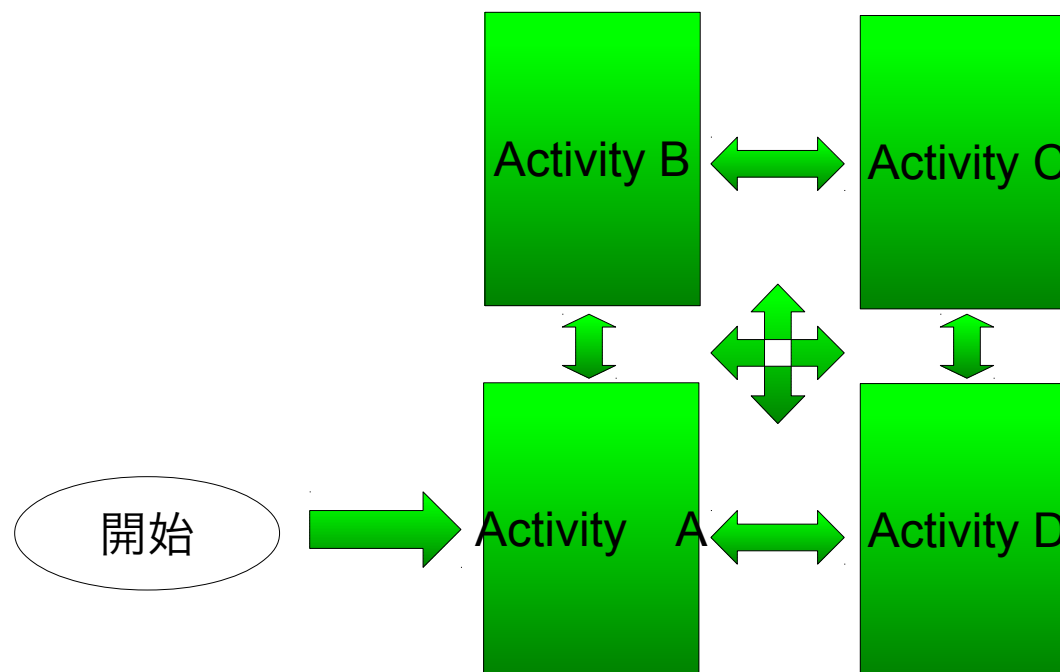
- MainActivityは、Activityクラスを継承している
- Activityクラスは、Androidのフレームワークで定義されている「一つの画面を表すコンポーネント」

■ アプリケーションを構成するコンポーネント

- Activity
- Service
- ContentProvider
- BroadcastReceiver

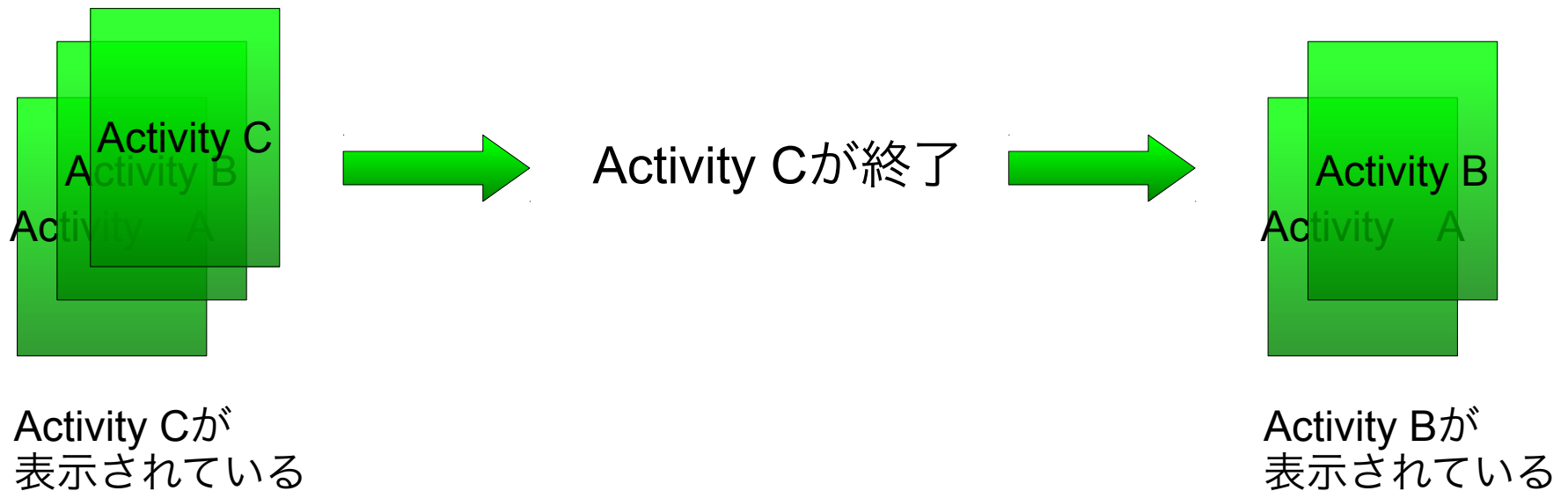
■ Activity

- 画面表示のコンポーネント
 - ユーザーからの入力処理し、サービスやその他の処理結果を出力する
- Activityを相互に呼び出す事で、画面の遷移を行う



■ Activity

- システムが一度に表示するActivityは一つのみ
- 呼び出した順番に重なって、一番上の画面が終了すると、次の画面が表示される



■ 画面の遷移

- Activityを継承したクラスの追加
- Activityで表示する画面レイアウトの定義
- Activityの実装
- AndroidManifest.xmlへ追加
- Activityの呼び出し

■ Activityを継承したクラスの追加

プロジェクトのsrcディレクトリ以下、パッケージ”org.sample.androidproject”に、新しく ”NextActivity.java” クラスを作成する。

- src/org.sample.androidproject/NextActivity.java

```
package org.sample.androidproject;  
  
import android.app.Activity;  
  
public class NextActivity extends Activity {  
  
}
```

■ Activityで表示するレイアウトの定義

プロジェクトのres/layoutディレクトリ以下に、
新しく"next_view.xml"を作成する

- res/layout/next_view.xml

```
<?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"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
</LinearLayout>
```

■ Activityの実装

- src/org.sample.androidproject/NextActivity.java

```
package org.sample.androidproject;

import android.app.Activity;
import android.os.Bundle;

public class NextActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.next_view);
    }

}
```

追記

- onCreateメソッドをオーバーライド
- setContentViewで、next_viewレイアウトを設定

■ AndroidManifest.xmlへ追加

- AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.sample.androidproject"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon"
android:label="@string/app_name">
        <activity android:name=".MainActivity"
            android:label="@string/app_name"
            >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".NextActivity" />
    </application>
    <uses-sdk android:minSdkVersion="4" />
</manifest>
```

追記

■ Activityの呼び出し

◦ MainActivity.java

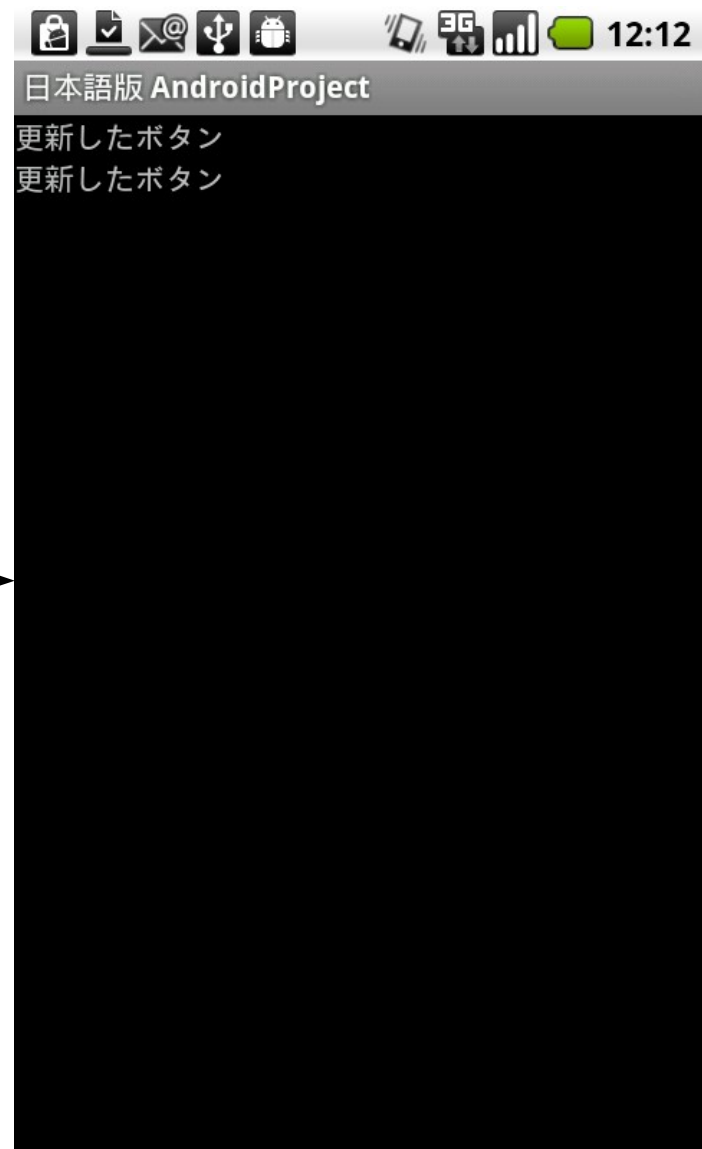
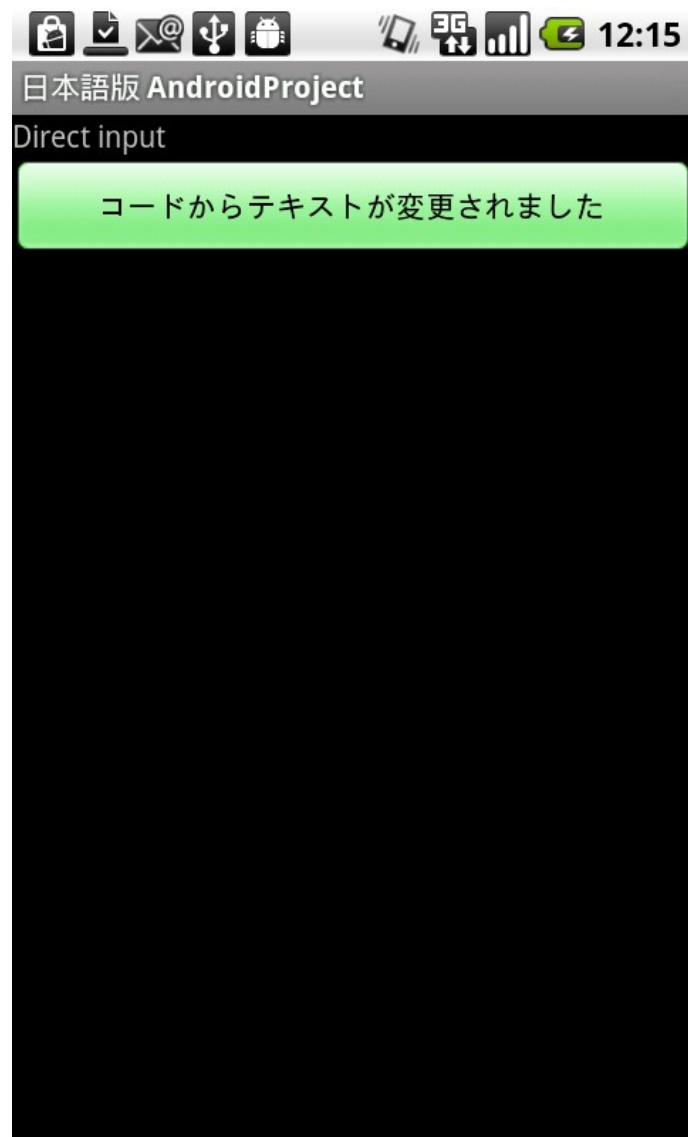
```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    Button btnTest = (Button) findViewById(R.id.main_btn_test);
    btnTest.setText("コードからテキストが変更されました");
    btnTest.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
            nextActivity();
        }
    });
}
```

変更

```
private void nextActivity() {
    Intent intent = new Intent(this, NextActivity.class);
    startActivity(intent);
}
```

追記

■ 実行

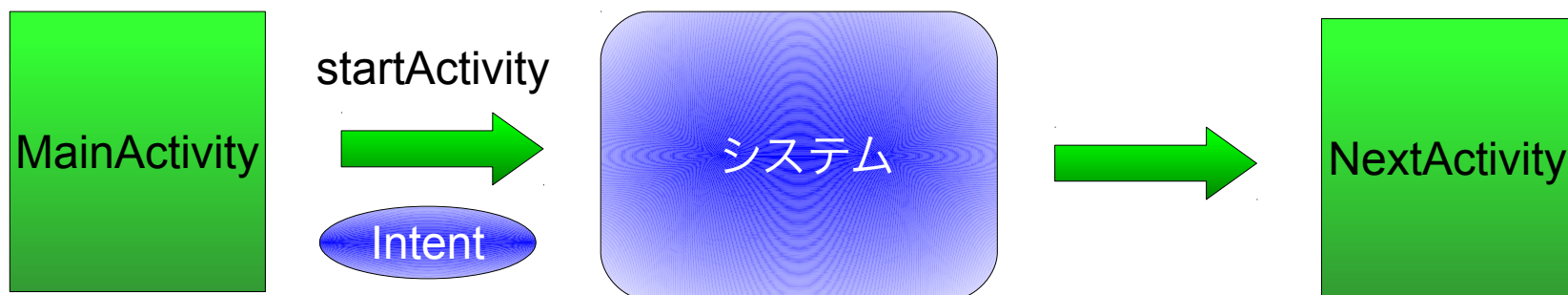


■ Intent

```
private void nextActivity() {  
    Intent intent = new Intent(this, NextActivity.class);  
    startActivity(intent);  
}
```

- Intent
[名]((形式))1 [U]意図, 意向, (…する) 意志((to do)); 目的; 《法律》意思; 故意.
⇒INTENTION[類語] true intent真意 with intent to ...
(Goo 英和辞書)

システムに「やりたいこと（意図）の内容」を伝える



■ 様々なIntent

- MainActivity.java

```
private void showMap() {  
    Uri uri = Uri.parse("geo:34.7096014, 137.7346837");  
    Intent intent = new Intent(Intent.ACTION_VIEW, uri);  
    startActivity(intent);  
}
```

■ 実行結果



?

■ Intentの種類

```
private void nextActivity() {  
    Intent intent = new Intent(this, NextActivity.class);  
    startActivity(intent);  
}
```

```
private void nextActivity() {  
    Intent intent = new Intent();  
    intent.setClassName("org.sample.androidproject", "NextActivity");  
    startActivity(intent);  
}
```

開始したいActivityのパッケージ名とクラス名を指定する = 明示的Intent

```
private void showMap() {  
    Uri uri = Uri.parse("geo:34.7096014, 137.7346837");  
    Intent intent = new Intent(Intent.ACTION_VIEW, uri);  
    startActivity(intent);  
}
```

開始したいActivityに渡すデータのみを指定する = 暗黙的Intent

■ Intentに指定できる内容

- **ACTION** : 何をするのか
 - ACTION_VIEW
 - ACTION_EDIT
- **CATEGORY** : DEFAULT / LAUNCHER / INFO
 - CATEGORY_DEFAULT
 - CATEGORY_LAUNCHER
- **DATA** : URIで表現 etc...
 - content://contacts/people/1
- **TYPE** : DATAの値の種別を、MIMEで指定
 - jpeg/image
- **FLAG** : 起動に関する付加情報

■ AndroidManifest.xml

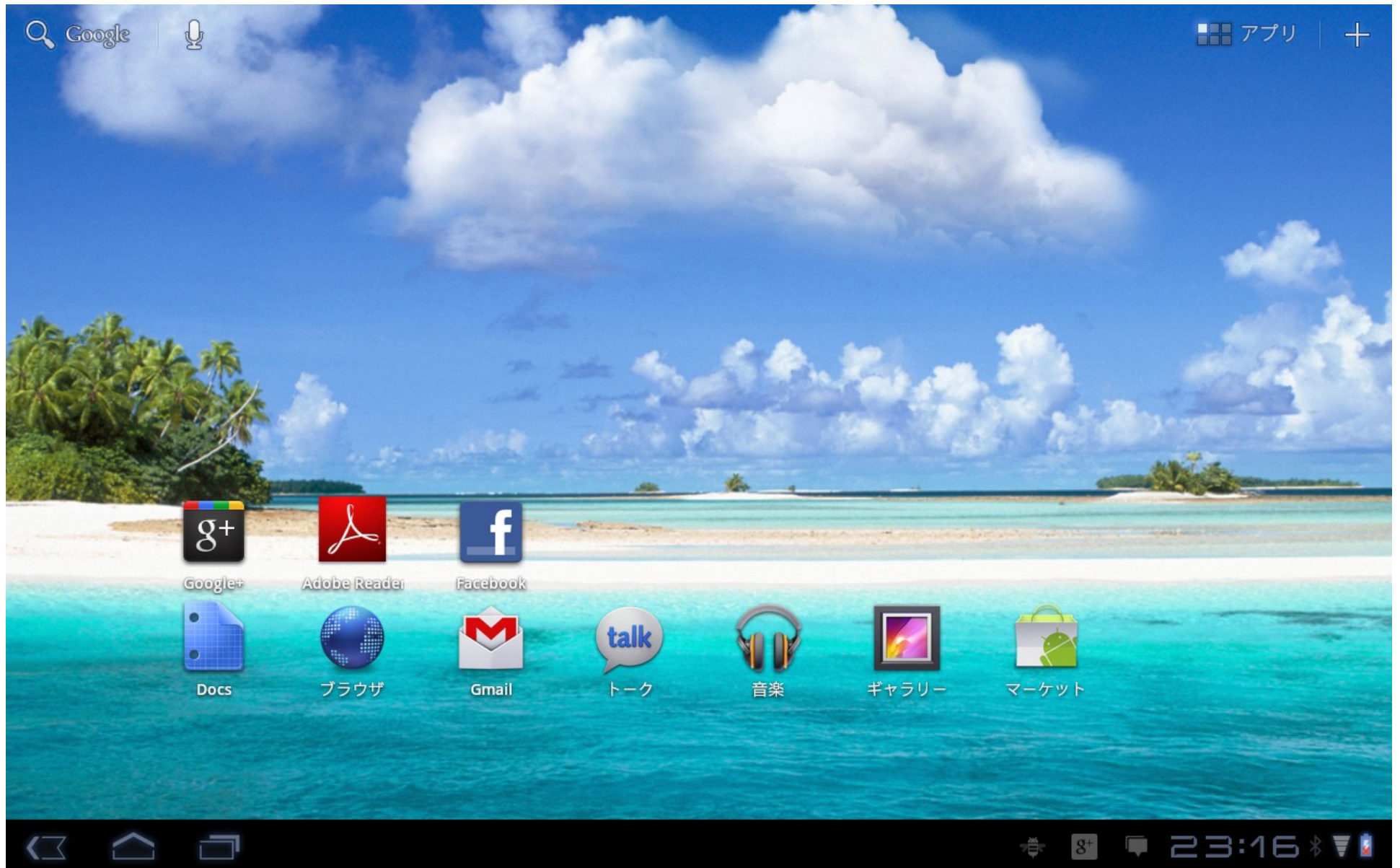
- 各アプリケーションにどのようなコンポーネントがあるのか、コンポーネントがどのようなIntent进行处理するか、システム側で把握している
 - AndroidManifest.xmlに記載
 - intent-filterの定義

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.sample.androidproject"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".MainActivity"
            android:label="@string/app_name"
            >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".NextActivity" />
    </application>
    <uses-sdk android:minSdkVersion="4" />
</manifest>
```

■ Honeycomb(3.0) 以降

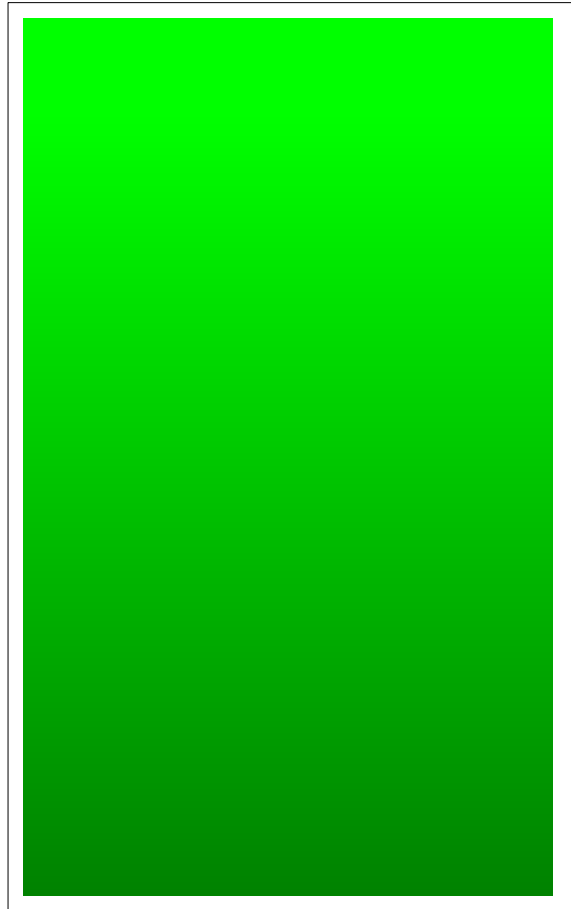


■ Honeycombとは

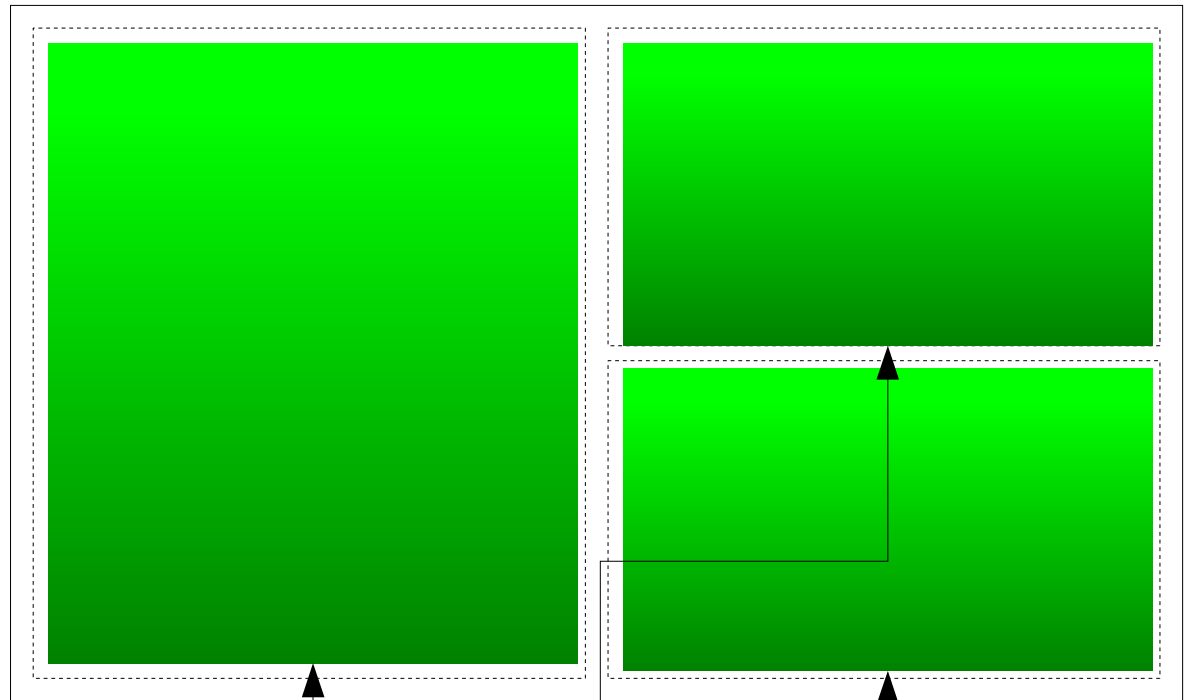


■ ActivityとFragment

Activity



Activity



Fragment

■ 詳しくは



日経Linux 2011年 9月号
「特別連載 最新Android開発」
<http://itpro.nikkeibp.co.jp/article/MAG/20110805/363747/>

質疑応答

本資料は、[有限会社シーリス](#)の著作物であり、
クリエイティブコモンズの表示-非営利-継承 3.0 Unported ライセンスの元で公開しています。



本資料の内容の一部は、Googleが作成、提供しているコンテンツを複製したもので、
クリエイティブコモンズの表示 2.5 ライセンスに記載の条件に従って使用しています。

