ボタンを追加してみよう

Android アプリケーション開発　ハンズオンセミナー

ユーザー操作を受け取るために、画面上にボタンを作ってみましょう。ここでは、Android の Eclipse プラグインを用いてボタンを定義します。

日本 Android の会　木南英夫

2009/08/05

2009 年 8 月 5 日

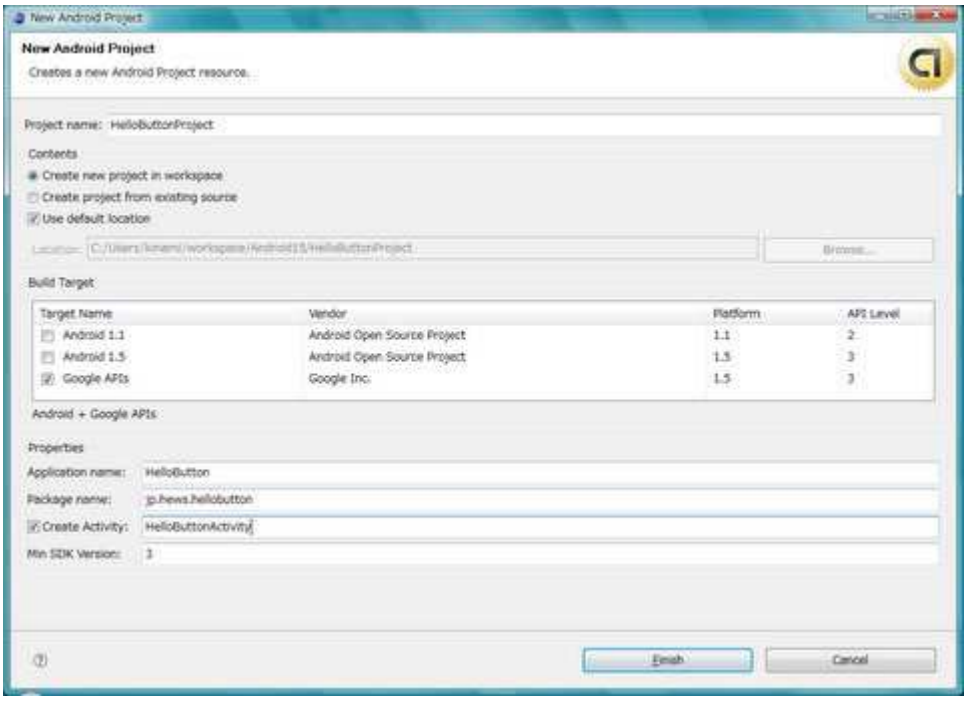
ボタンを追加してみよう

Android アプリケーション開発　ハンズオンセミナー

プロジェクトを作成する

必要に応じて、File > New > Android Project で新規のプロジェクトを作成します。

ここでは、以下のようなプロジェクトを作成してみます。



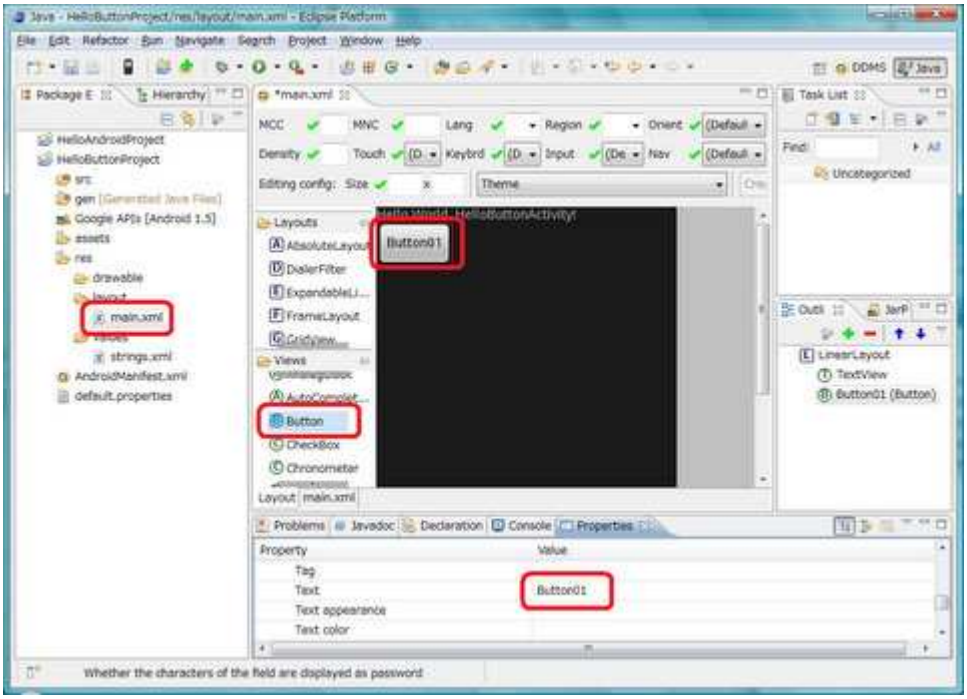
Project Name	HelloButtonProject
Build Target	Google APIs
Application Name	HelloButton
Package Name	jp.hews.hellobutton
Create Activity	HelloButtonActivity
Min SDK Version	3（Build Target を指定すると自動的に設定される）

画面にボタンを配置する

画面にボタンを配置するには、レイアウトを定義した XML ファイルを編集します。

まず、レイアウトを定義した res/layout/main.xml を開いて、画面上にボタンを配置します。

この画面でボタンを配置するには、GUI で挿入する方法と、下部の「main.xml」タブから直接 XML を編集する方法があります。ここでは、GUI を用いて定義します。

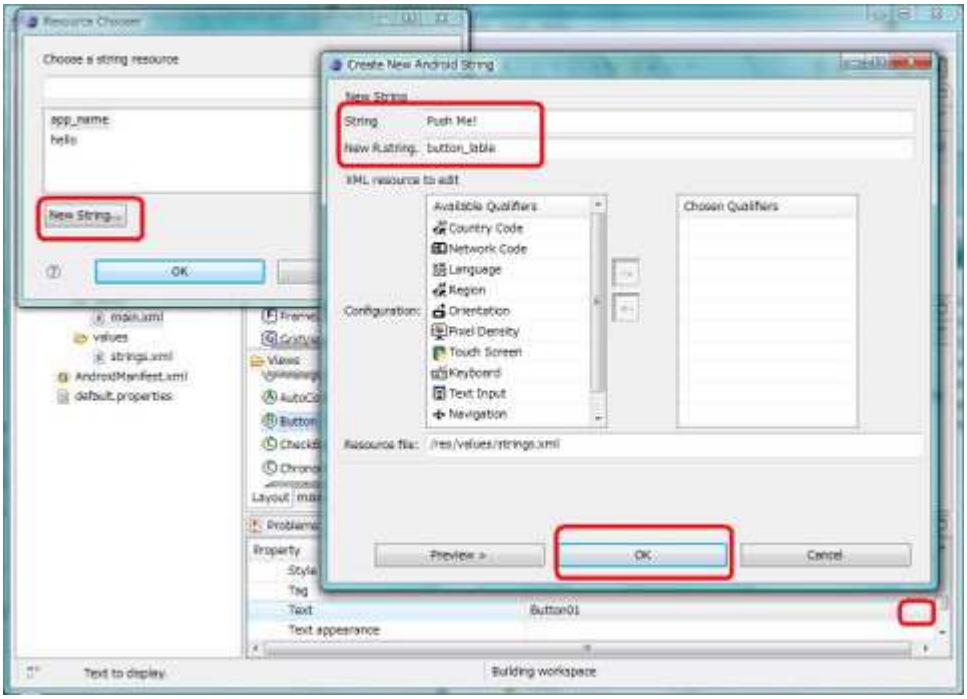


- Package Explorer で res/layout/main.xml をダブルクリックして main.xml を開きます。
- Views のフォルダーのなかの「Button」要素をドラッグして、画面に挿入します。
- Button01 という文字の表示されたボタンが配置されます

ボタンに表示するラベルを定義する

ボタンのラベルは、res/values/strings.xml で定義します。ここでは、ボタンのプロパティから開かれる Resource Chooser を使用して、文字列を定義します。

- 下部の Properties タブで、Text 要素を選択して、Resource Chooser を選択します。



- 下部の「New String」を選択すると Create Android String の画面が開きます。
- String に表示する文字として「Push Me!」を入力します
- New R.string に、定義した文字列の ID として、「button_label」を入力します
- Preview を押して文字列が string.xml に反映されることを確認します。
- OK を押して、Resource Chooser の画面に戻ります
- いま定義した「button_lable」を選択して、「OK」を押します

ボタンが押された時の処理を追加する

ボタンクラス（android.widget.Button）にリスナーを登録することで、ボタンが押された時の処理を追加することができます。

ボタンが押された時に呼び出されるリスナーの型は、android.view.View.OnClickListener です。

ここでは、メインのアクティビティクラスである HelloButtonActivity にリスナーを実装します。

- HelloButtonActivity をダブルクリックして開きます
- 先頭のインポート文の最後に実装に必要なボタンクラスとリスナークラスをインポートします。

```
import android.view.View;
import android.widget.Button;
```

- HelloButtonActivity クラスに「implements View.OnClickListener」を追加して、リスナーの実装を宣言します。

```
public class HelloButtonActivity extends Activity implements View.OnClickListener {
```

- HelloButtonActivity クラスに「public void onClick(View v)」メソッドを追加して、リスナーを実装します。
  - ✧ リスナーのメソッドの中では、finish()を呼び出します。このメソッドを呼び出すと、そのアクティビティを終了します。

```
  public void onClick(View v) {
    finish();
  }
```

- HelloButtonActivity クラスの onCreate メソッドで、ボタンにリスナーを登録します。
  - ✧ setContentView メソッドで、main.xml で定義した要素を表示します
  - ✧ findViewById メソッドで、main.xml で定義したボタンのインスタンスを取り出します。
  - ✧ 取り出したボタンクラスの setOnClickListener メソッドに this を渡すことで、リスナーを登録します

```
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.main);

        Button button = (Button)findViewById(R.id.Button01);
        button.setOnClickListener(this);
    }
```

- 作成したソースコードは、以下のようになっています。

```
package jp.hews.hellobutton;

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

public class HelloButtonActivity extends Activity implements View.OnClickListener {

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

        Button button = (Button)findViewById(R.id.Button01);
        button.setOnClickListener(this);
    }

    public void onClick(View v) {
        finish();
    }
}
```

プログラムを実行する

修正したプログラムを実行して、表示されたボタンを押すことで、アクティビティが終了することが確認できます。

- Run メニューの Run で Run As 画面を表示する
- 「Android Application」を選択することで、アプリケーションが実行される
- 「Push Me!」というボタンを押すと画面が閉じて、ランチャーの画面が表示される

演習問題

- ボタンをふたつ横に並べて配置してみましょう。
- 二つのボタンのクリックを一つのリスナーのメソッドで受けて、どちらのボタンが押されたのかを判定してみましょう。
- イメージボタンや、ラジオボタンなど、別の種類のボタンを作ってみましょう。