ボタンイベントアプリ

イベント処理を含むアプリとして，ボタンをもち，ボタンを押すと文字列を表示するアプリを作る．このアプリは，HelloWorld アプリを改造して作成するため，アプリ作成の途中からの手順を示す．

1. ボタンの設置

(1) レイアウトにボタンを追加する

パレットの「フォーム・ウィジェット」からボタンのアイコンをドラッグして，ワークスペースにドロップする．

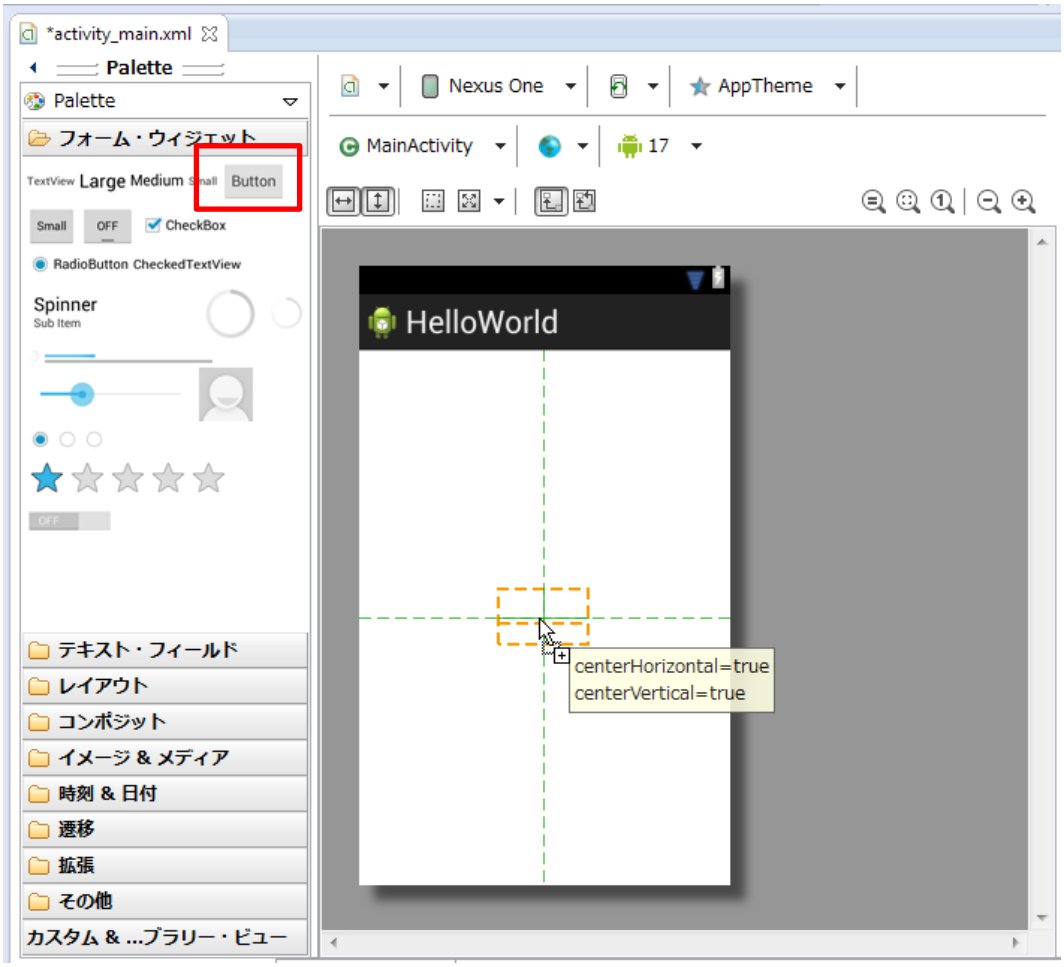

図 1　ボタンの追加

2. 文字列を定義する

追加したボタンの右端に，黄色い警告マークが表示されているのがわかる．警告マークにカーソルを合わせると，以下のような説明が表示される．

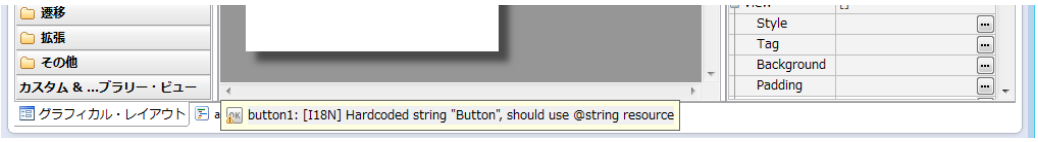

図 2 警告の表示

これは，表示される文字列を，レイアウトファイルに直に記載していることが原因である．文字列を直接書き入れると，多言語化を行うときに言語ファイルによる差し替えが困難になることや，文字列を修正する際に，手間がかかるなど，メンテナンスや拡張を考慮していない状態になる．そこで，文字列の定義は，別のファイルでまとめて行うことが推奨されている．

そこで，文字列を専用の XML ファイルに定義する．

(1) res/values/strings.xmlを開く

「リソース」タブで開くと，GUI エディタで編集することができる．「strings.xml」タブで開くと，XML ファイルを直接編集することができる．

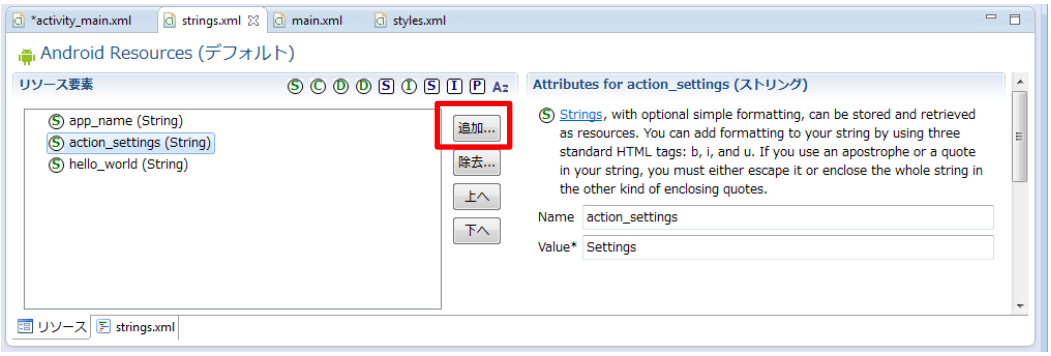

図 3 リソース定義の一覧

(2) 項目を追加する

「追加」ボタンを押すと，の画面が現われるので，「String」を選択する．

図 4 項目を選択する画面

(3) NameとValueを入力する

Name の欄に「button_name」, Value の欄に「押す」と入力する.

図 5 定義する内容を記入する

入力した後には，保存することで,「リソース要素」に反映される.

(4) ボタンの表示に文字列を適用する

activity_main.xml を開き，レイアウトされたボタンを選ぶ.

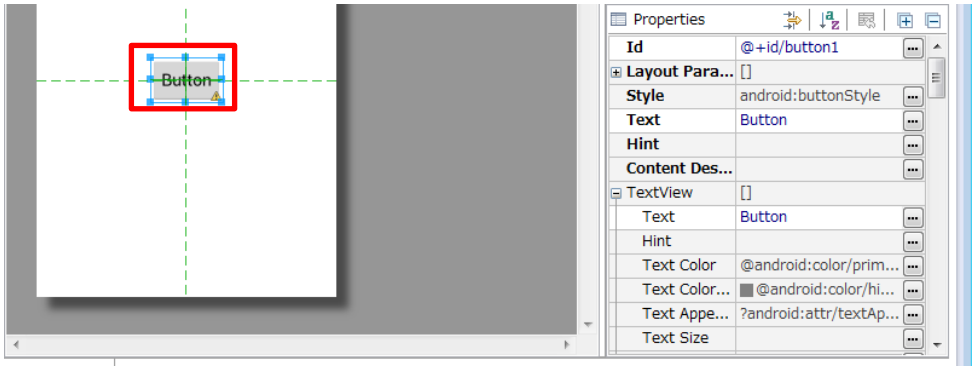

図 6　リソースを指定するボタンを選ぶ

Properties ウィンドウのなかに,「Text」項目の右に「…」と表示されているボタンがあるので，それを押すと，ウィンドウが開くので，先ほど設定した,「button_name」を選ぶ.

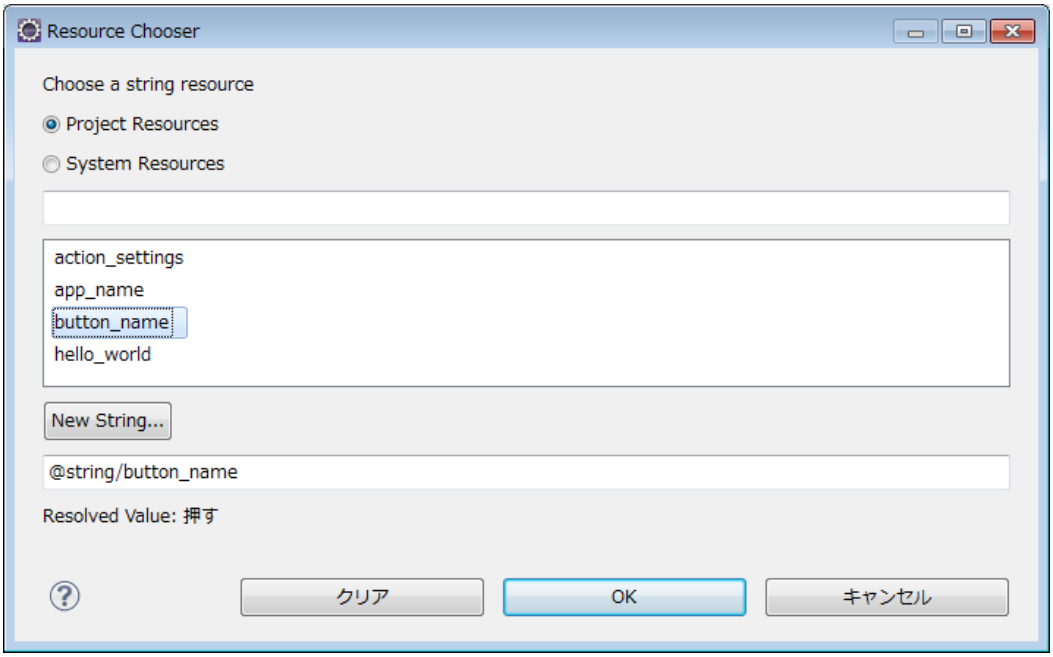

図 7 設定するリソースを選ぶ

そうすると，ボタンの表示が「押す」に変わり，警告マークが消えている．

3. イベント処理

ボタンを押すという「イベント」が発生したときに一定の処理を行わせるための方法には，主に 2 種類ある．ここでは，レイアウトファイルで設定する方法を示す．

(1) 準備

(a) 文字列を表示させるためのTextViewを準備する

パレットウィンドウの「TextView」をストラクチャーウィンドウの「RelativeLayout」に，ドラッグアンドドロップする．

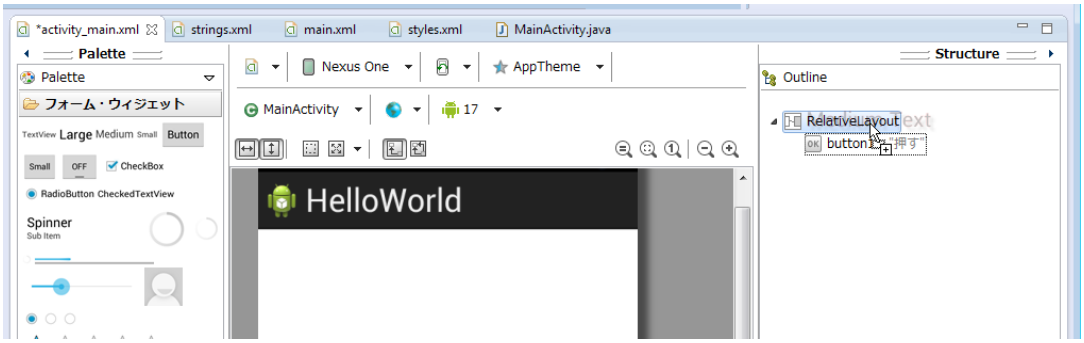

図 8 TextView の追加

(b) 表示する文字列を準備する

res/values/strings.xml を開き，「追加」ボタンを押す．

2 種類の文字列を定義するので，文字列配列（String Array）を選ぶ．

図 9 String Array を選択する

追加した String Array に名前をつける．ここでは，「messages」とする．

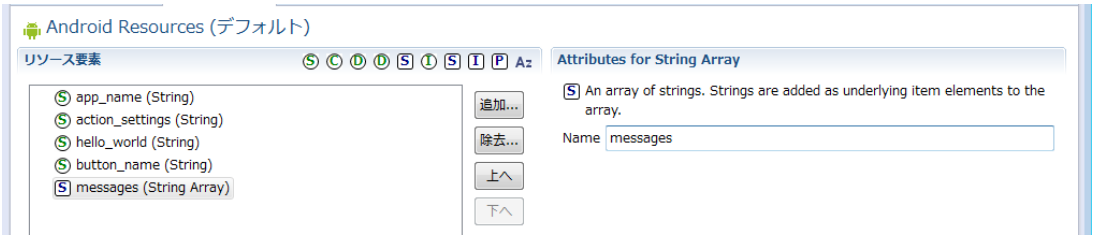

図 10 Array に名前をつける

作成した String Array はまだ空っぽなので，要素を追加する．messages(String Array)を選んだ状態で「追加」ボタンを押して，リストから Item を選ぶ．

このとき，「Create a new element in the selected element, message(String Array)」のラジオボタンが選ばれていることを確認する．これが選ばれていることで，先に定義した配列の要素を追加することができる．

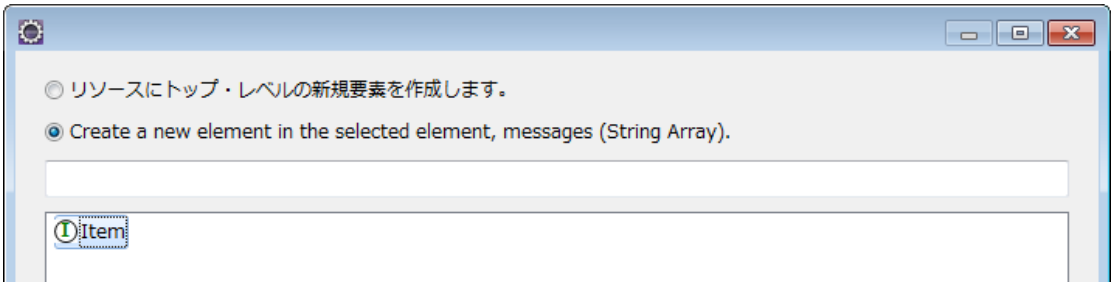

図 11 String Array に Item を追加する

Item を作成して，「Value」の項目に値として，「Hello World」を入れる．

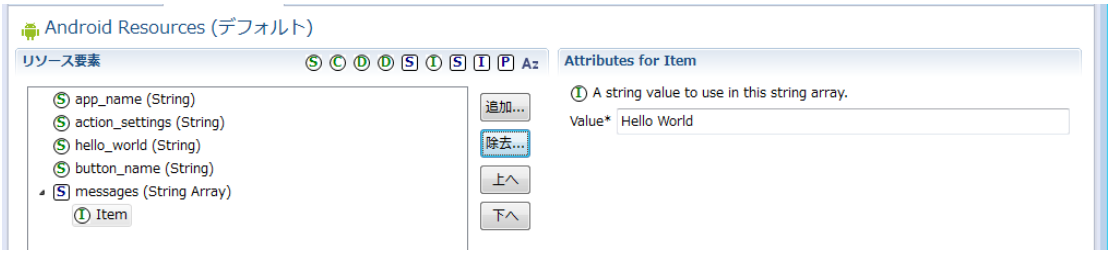

図 12　Item の内容を入力する

同様に，messages の Item として，Value「Nice to meet you!」を追加する．

(2) ボタンにハンドラを関連づける

(a) activity_main.xmlを開き，ボタンを選択する．

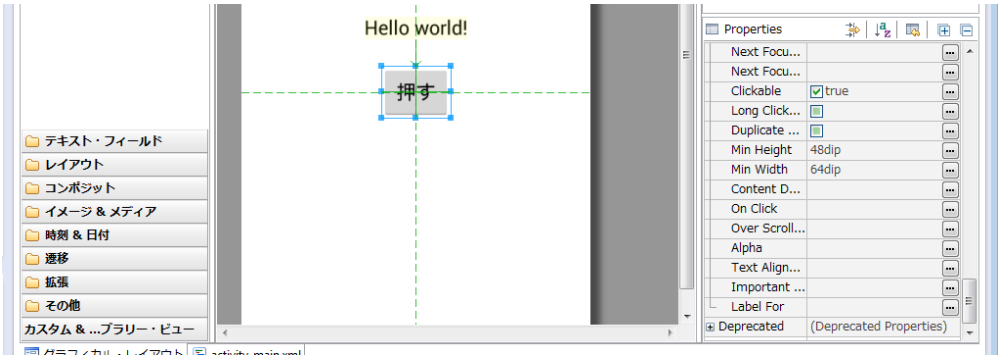

図 13　ボタンを選んだ様子

(b) Propertiesにて，ボタンをクリックしたときのハンドラとして動作するメソッドの名前「onClickHanlder」を「On Click」の値として指定する．

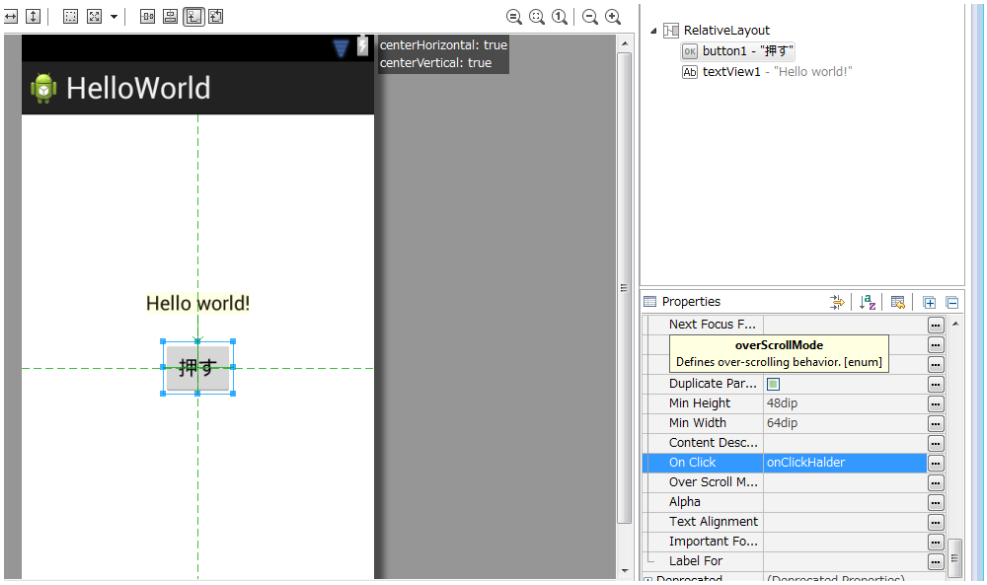

図 14　On Click の欄に onClickHandler を指定する

(3) 処理内容を書く

(a) src/MainActivity.java を開く

(b) onClickHandlerを追加する

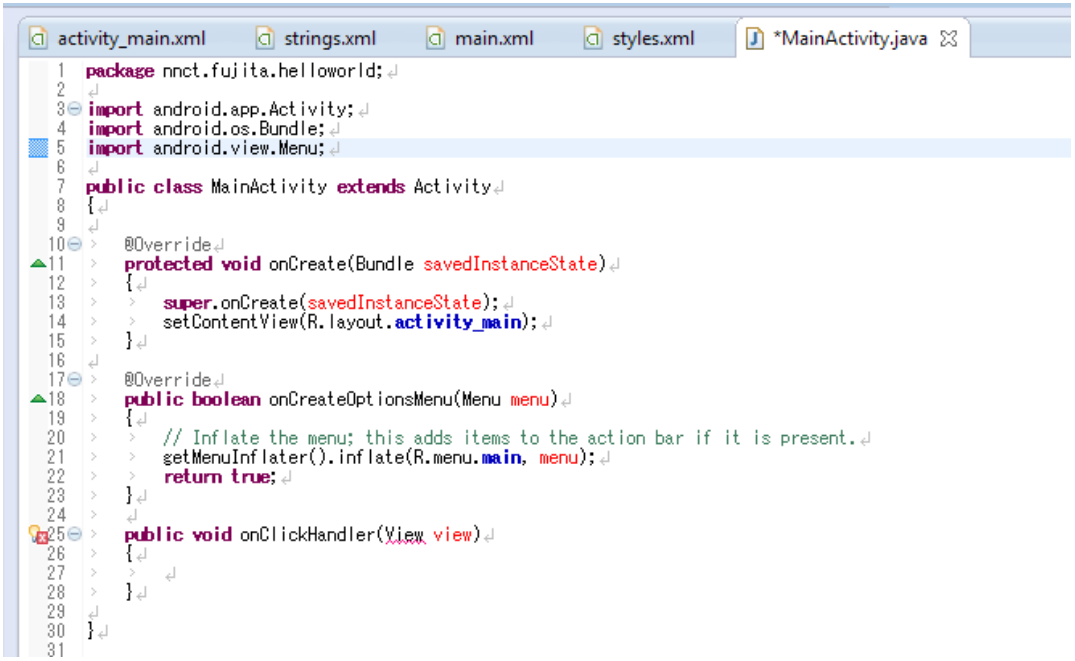

図 15　onClickHanlder メソッドを追加する

(c) onClickHandlerに処理内容を追加する

リスト 1　MainActivity クラスの onClickHandler メソッド

```
1  public void onClickHandler(View view)
2  {
3    /* Messages をリソースから得て messages[]配列に入れる */
4    String[] messages = getResources().getStringArray(R.array.messages);
5    /* textView1 の id からビューを得て，tv とする */
6    TextView tv = (TextView)findViewById(R.id.textView1);
7    /* tv に書かれている文字列と messages[0] の文字列が同じなら */
8    if(tv.getText().toString().equals(messages[0]))
9    {
10       /* tv に messages [1] の文字列を書く */
11       tv.setText(messages[1]);
12   }
13   else
14   {
15       /* tv に messages [0] の文字列を書く */
16       tv.setText(messages[0]);
17   }
18 }
```

(d) 必要なクラスをインポートする

TextView に赤い下線が現われる．これは，定義されていないクラスを使おうとしていることを警告している．TextView クラスを使うために，定義されているクラスをインポートしなければならない．そこで，この下線部分にカーソルを合わせて[Ctrl+ 1]を押すと，

ポップアップがあらわれ，解決方法を提案してくれる．今回は，「“TextView”をインポートします（android.widget)」が適しているため，これをクリックすると，

```
import android.widget.TextView;
```

を自動で追加してくれる．

図 16　クラスをインポートする

【補足】R.xxx.yyy　とはなにか.

gen/パッケージ名/R.java　で定義されている R クラスの中にある，xxx クラスで定義されているフィールド yyy の値を参照している．複数の内部クラスによって構成されているため．ドット「.」によって，クラス内のフィールドにアクセスする．

```
package nnct.fujita.helloworld;

public final class R {
    public static final class array {
        public static final int messages=0x7f060000;
    }
    public static final class attr {
    }
    public static final class dimen {
        /** Default screen margins, per the Android Design guidelines.

         Customize dimensions originally defined in res/values/dimens.xml (such as
         screen margins) for sw720dp devices (e.g. 10" tablets) in landscape here.

         */
        public static final int activity_horizontal_margin=0x7f040000;
        public static final int activity_vertical_margin=0x7f040001;
    }
    public static final class drawable {
        public static final int ic_launcher=0x7f020000;
    }
    public static final class id {
        public static final int action_settings=0x7f090002;
        public static final int button1=0x7f090000;
        public static final int textView1=0x7f090001;
    }
    public static final class layout {
        public static final int activity_main=0x7f030000;
    }
    public static final class menu {
        public static final int main=0x7f080000;
    }
    public static final class string {
        public static final int action_settings=0x7f050001;
        public static final int app_name=0x7f050000;
        public static final int button_name=0x7f050003;
        public static final int hello_world=0x7f050002;
    }
    public static final class style {
        /**
```