Android Hello World チュートリアル

[Android][Eclipse][Java]

- http://developer.android.com/resources/tutorials/hello-world.html
- http://www.youtube.com/watch?v=cw5GrRK4AAg
- Android SDK のインストール

プラットフォームのインストール

- Hello World アプリケーションを動かすためには少なくとも Andoroid のプラットフォームの一つは SDK 環境にインストールしておく必要がある。以下を参照。

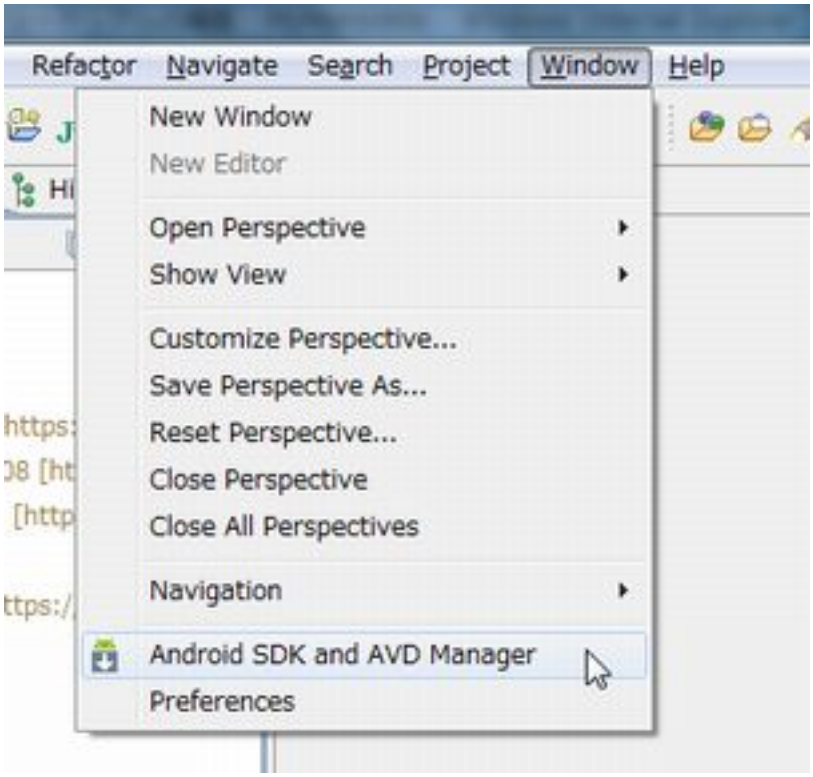
- SDK のインストール
- ADT Eclipse プラグイン

AVD の作成

- http://developer.android.com/guide/developing/tools/avd.html

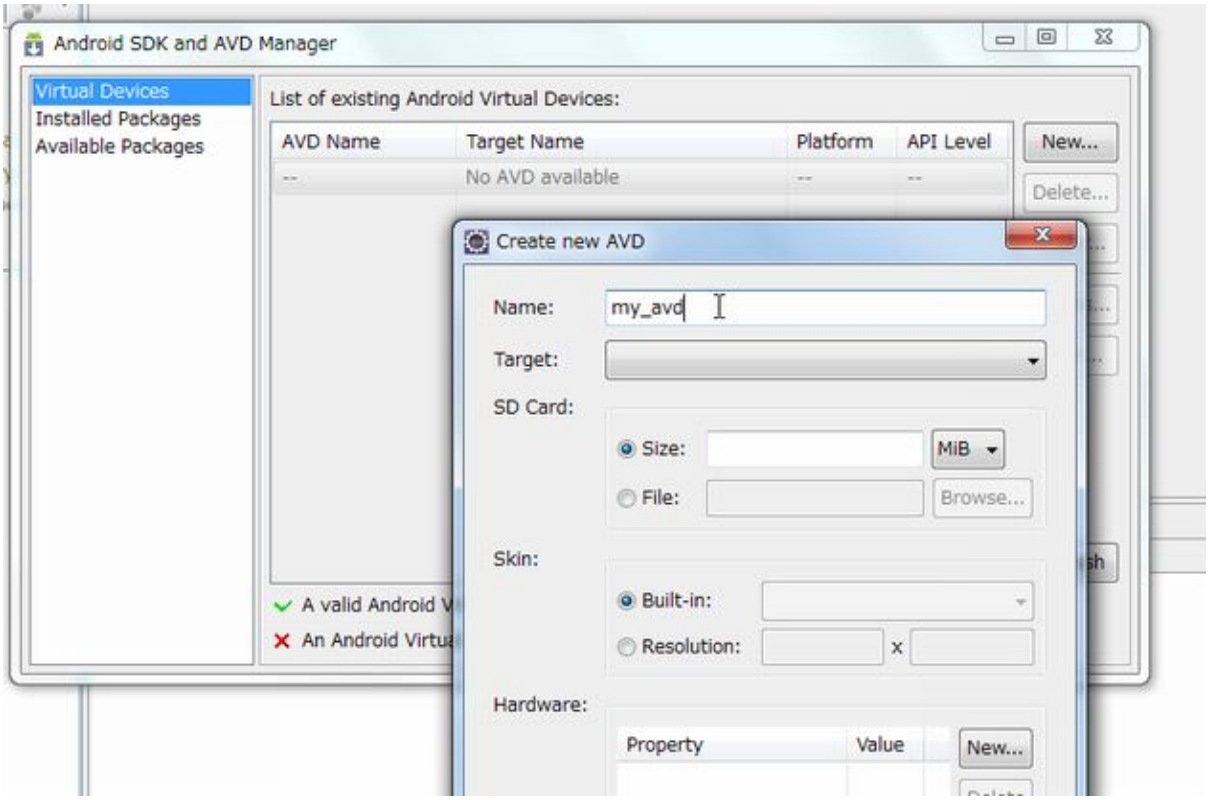
- このチュートリアルでは、アプリケーションを Android エミュレータで走らせる。
- エミュレータを起動する前に、Android Virtual Device(AVD) を作成する必要がある。
- AVD はシステムイメージおよびデバイスのセッティングをエミュレータを利用して定義する。

Window ― Android SDK and AVD Manager を開く



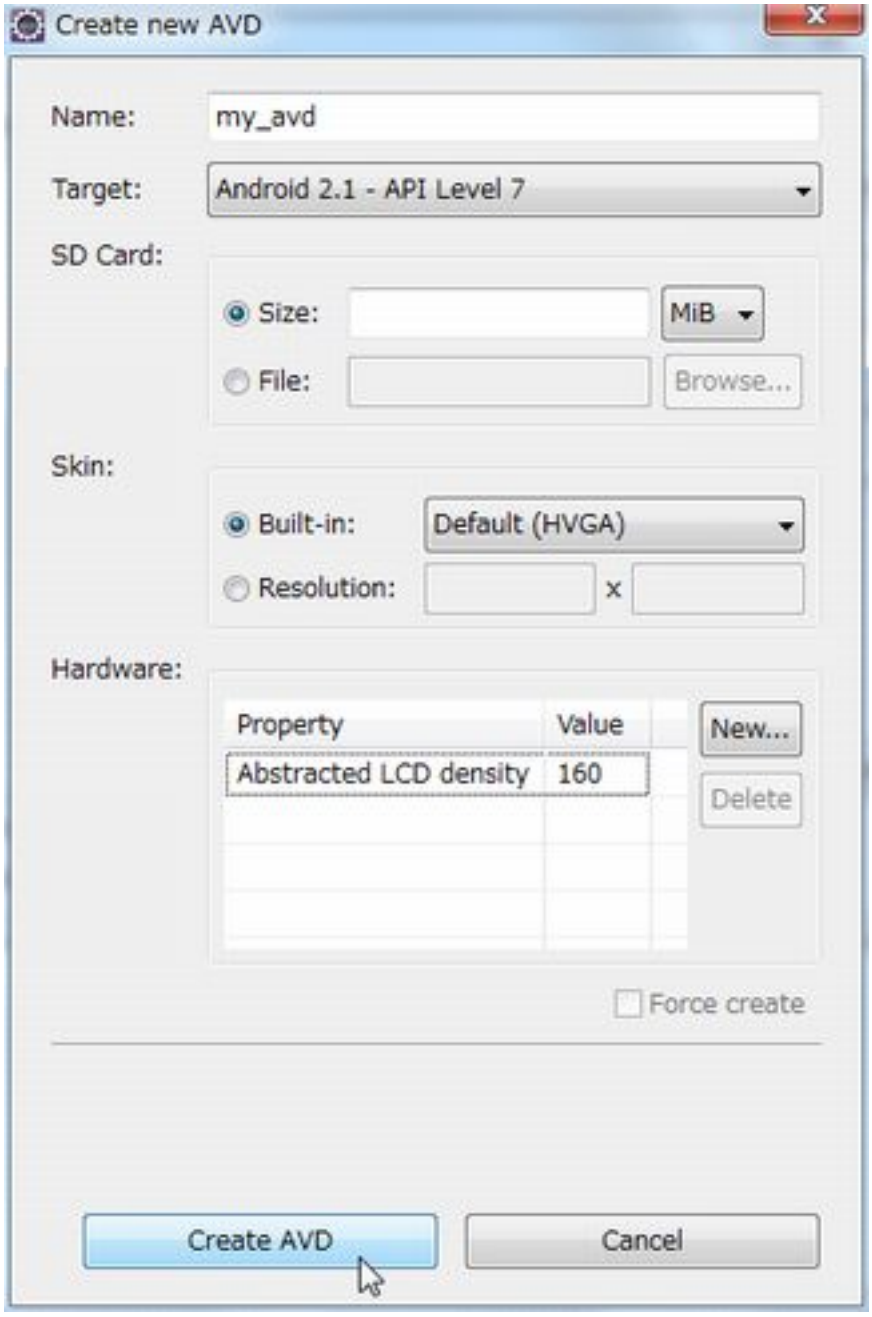
Select Virtual Devices を選択し New を押下

Create New AVD ダイアログが開くので、my_avd のような名前を設定



プラットフォームターゲットを選択し、Create AVD 押下

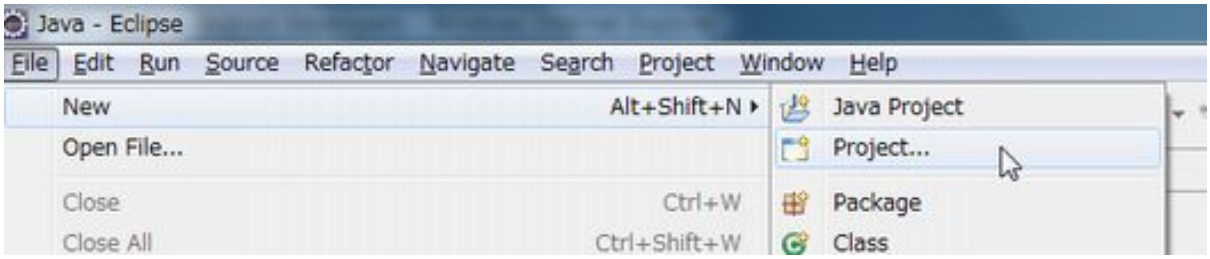
・プラットフォームを選択する。残りのフィールドは現時点では無視してよい。



Android プロジェクトの新規作成

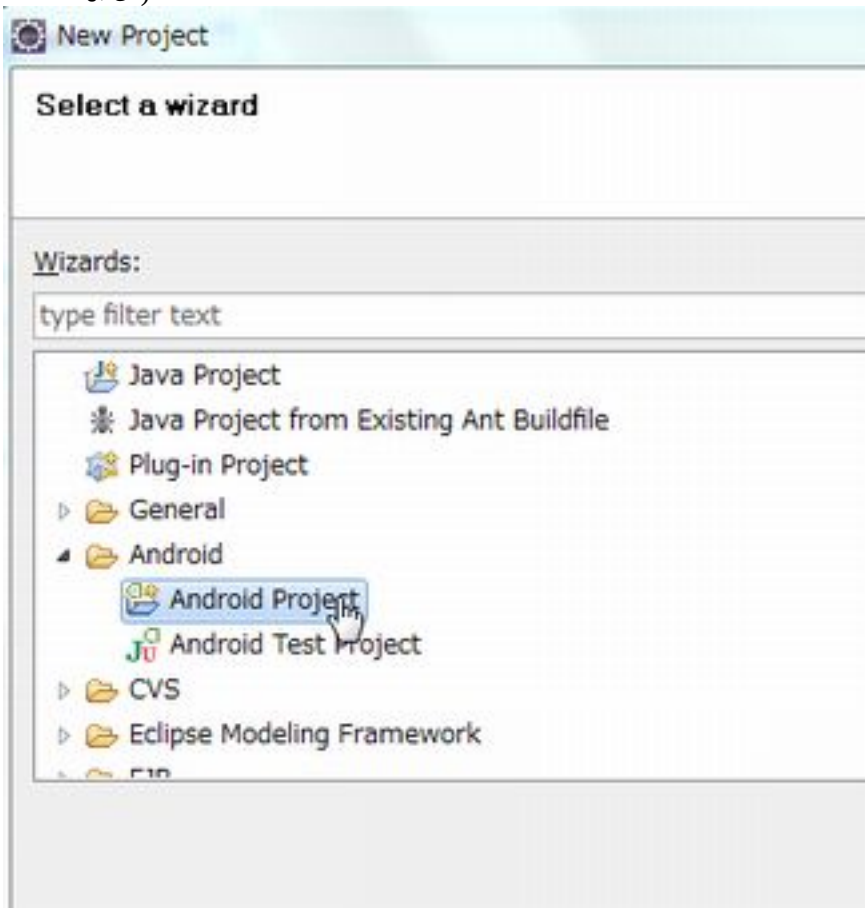
http://www.youtube.com/watch?v=Q1SoGRAuVRs

File ― New ― Project.



Android Project を選択し、Next を押下

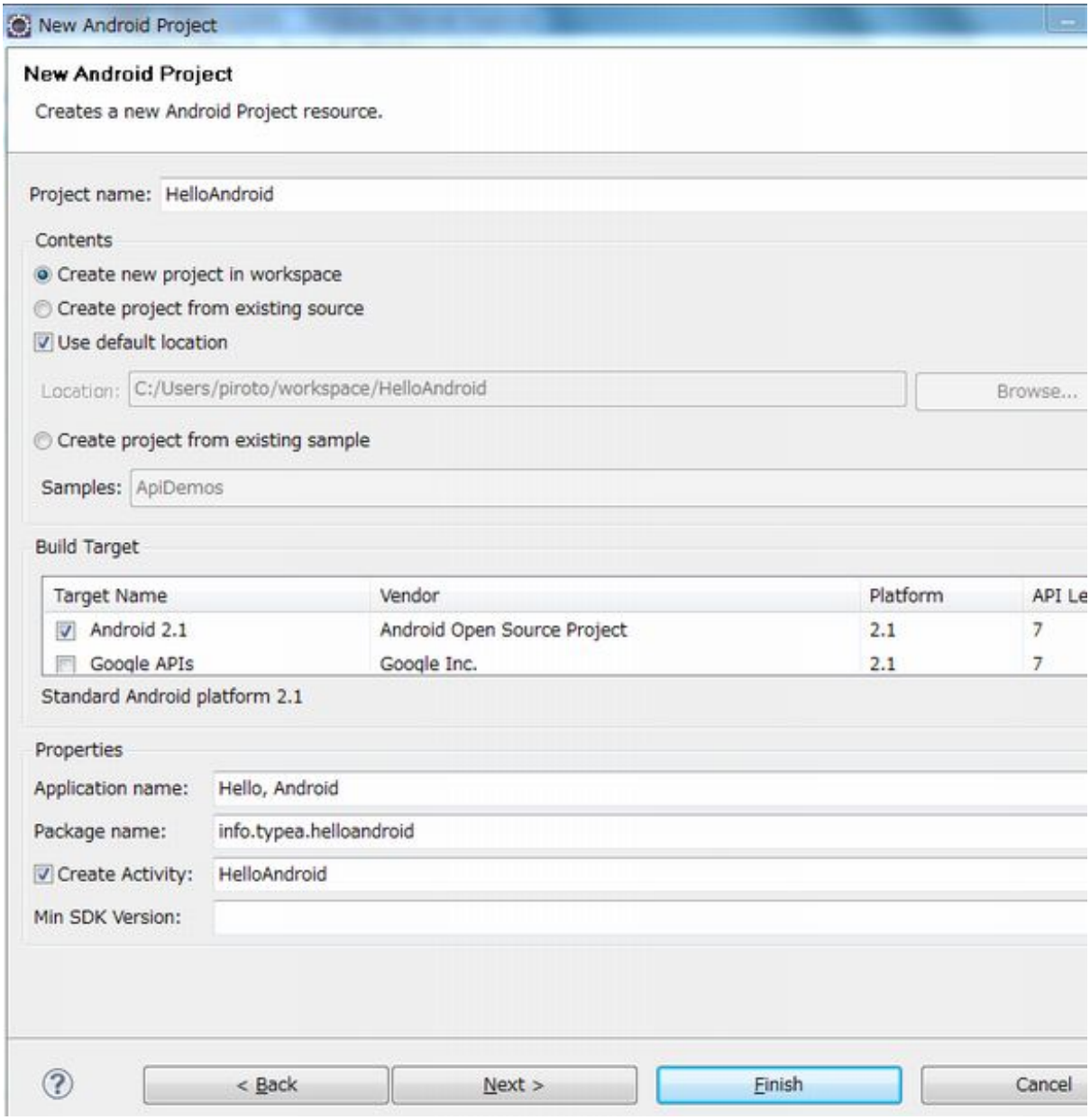
・ADT プラグインのインストールが成功していれば、Android フォルダに Android Project が存在する。(1 つ以上のプロジェクトを作成したのち、Android XML File エントリも有効になる)



プロジェクト詳細設定

・Project name: HelloAndroid
・Application name: Hello, Android
・Package name: info.typea.helloandroid (プライベートのネームスペース)
・Create Activity: HelloAndroid

上記を設定し、Finish



・フィールドの説明

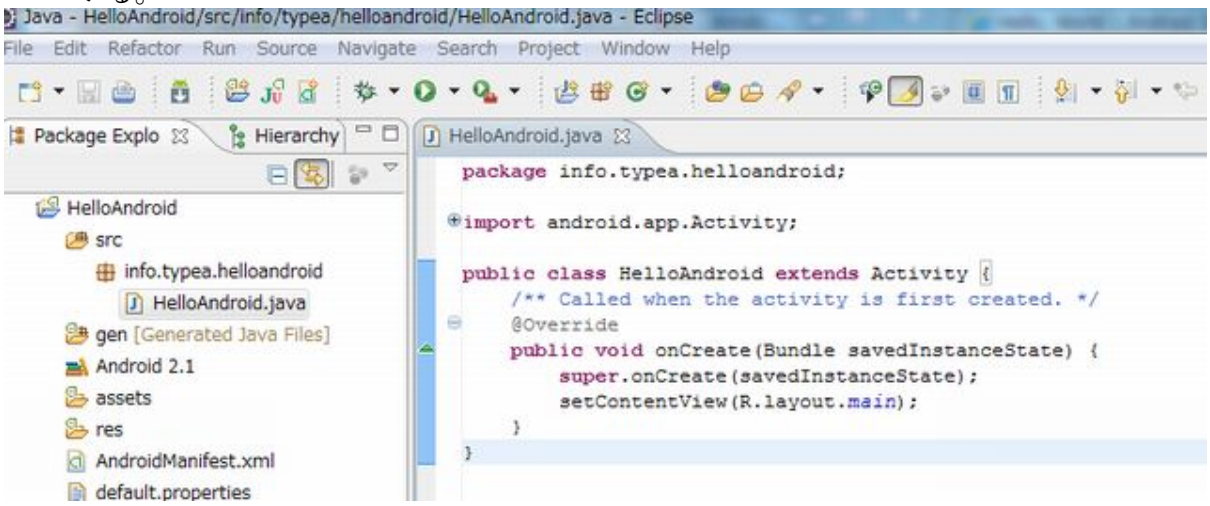
フィールド	内容
Project Name	Eclipse のプロジェクト名。同名ディレクトリはプロジェクトのファイルを含む
Application Name	アプリケーションの人間が読むためのタイトル。Android デバイスに表示される。
Package Name	パッケージのネームスペース (Java でのルールに従う)。パッケージ名を生成されるスタブアクティビティの配下にセットする。パッケージ名はインストールされた Android システムでユニークである必要がある。

Create Activity	プラグインによって生成されるスタブクラスの名前。Android の Activetiy のサブクラス。Activity は 起動して作業できるシンプルなクラス。必須ではないが、選択すれば UI を生成できる。チェックはオプションだが、Activity はアプリケーションの基盤として多くの場合に利用される。
Min SDK Version	アプリケーションの最低限の API レベルを指定する。Android API レベル参照。
Use default location	チェックにより、プロジェクトのファイルの置き場所を変更する。
Build Target	アプリケーションのコンパイルターゲットとなるプラットフォーム

「Build Target」 選択したプラットフォームが Android 1.1 プラットフォームを使うということは、アプリケーションが Android 1.1 プラットフォームライブラリに向けてコンパイルされることを意味する。もし、Android 1.5 プラットフォーム上で、1.1 で作成された AVD を走らせる場合、Build Target をあわせる必要はない Android アプリケーションは、前方互換性を持っており、1.1 向けにビルドされたアプリケーションは、通常 1.5 でも起動する。逆は、成り立つわけではない。

プロジェクト準備完了

・プロジェクトの準備は以上で感 r ひょう。
・パッケージエクスプローラーで、HelloAndroid.java ファイルを開くと、以下の様になっている。



・このクラスは Activity を基底クラスとしている。
・Activity はプラットフォームのアクションから利用される 一つのアプリケーションエンティティ。
・あるアプリケーションは、複数に分割されたアクティビティをもっているが、ユーザーが一度に対話するのはそのうちの一つ。
・onCreate() メソッドは Android システムから Activity が開始したときに呼び出される。
・このメソッドは、すべての初期化および UI のセットアップを実行すべき場所。
・あるアクティビティはユーザーインターフェースが必要な訳では無いが通常利用する。

UI の構築

```
package info.typea.helloandroid;

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

public class HelloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(this);
        tv.setText("Hello, Android");
        setContentView(tv);
    }
}
```

■ View

・Android のユーザーインターフェースは View から呼ばれたオブジェクト階層から組み立てられている。
・View は、UI レイアウトの要素で、描画可能なオブジェクトで、ボタン、イメージ、テキスト、ラベルなど
・これらのオブジェクトは、View のサブクラスでテキストを扱うのは TextView

■ Context

・この変更で、TextView をコンストラクタで生成する。コンストラクタは、Android Context インスタンスをパラメーターとして受け取る。
・Context はシステムを処理する。
  ・リソースの解決などサービスの提供
  ・データベースやプリファレンスへのアクセス

■ setContentView

・最後に、Activity UI の中心に表示するために TextView を setContentView() に渡す。
・Activity がこのメソッドを呼び出さないと、UI は存在せず、システムはブランクスクリーンを表示する。

・Activity クラスは、Context を継承している。HelloAndroid クラスは Activity のサブクラス (もちろん Context のサブクラスでもある)
・なので、HelloAndroid の参照を Context として、TextView に渡すことができる。

アプリケーションの起動

1.Eclipse プラグインは自動的にプロジェクトの起動設定を生成し、エミュレータを起動する。
2. 環境に依存するが、Android エミュレータは、完全に起動するのに数分かかるので、これに耐える。
3. エミュレータが起動したら、Eclipse プラグインは、アプリケーションをインストールしてデフォルトの Activitiy を起動する
4. 以下の画面が表示される



- "Hello, Android" とグレーのバーに表示されているところは、実際はアプリケーションのタイトル。
- Eclipse プラグインは、これを自動生成する (この文字列は res/values/strings.xml で定義されており、AndroidManifest.xml から参照されている)
- タイトルの下が、TextView オブジェクトに設定した内容。

UI を XML レイアウトに変更する

プログラミングによる UI レイアウト

- 上記例は、プログラミングによる UI レイアウトで、アプリケーションの UI をソースコードで直接構築する手法。
- UI のプログラミングを数多く経験しているなら、プログラミングによるアプローチが、少しのレイアウトの変更がソースコード上で大きな頭痛の種となど、どれだけ不安定かになじみがあるだろう。

XML による UI レイアウト

- 上記が、UI の生成に Android では XML によるレイアウトファイルという代替方式を提供する理由

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

- Android XML レイアウトファイルの一般的な構造は簡単

・XML 要素のツリーとなっている。その中のノードそれぞれは View クラスの名前となっている。
・自ら作成したどんな名前のクラスでも、View を継承していれば XML レイアウトの要素として利用できる
・この構造で、簡単にすばやく UI を作成でき、プログラミングするよりシンプルな構造と文法を利用できる。
・このモデルは、Web 開発モデルに触発されており、アプリケーションのプレゼンテーション (UI) をロジックから分離する。

XML 属性

属性	内容
xmlns:android	XML 名前空間の定義。Android tools に Android 名前空間で定義された公開属性を参照させる。すべての Android レイアウトファイルにおいて一番外側のタグにて必ず持つ必要がある属性。
android:layout_width	スクリーン上でどれだけの幅をこの View が使用すべきかを定義。例では、ただひとつの View しかないため、すべてのスクリーンを占めたい。"fill_parent" はそういう場合に指定。
android:layout_height	スクリーン上での高さをあらわす以外は、android:layout_width と同様。
android:text	TextView が表示するテキストをセットする。例では、文字列リソースをハードコーディングする代わりに利用している。文字列は res/values/strings.xml ファイルに定義されている。他言語へのローカライズをレイアウトファイルを修正する必要がないため、アプリケーションに文字列を挿入する場合、この方法を推奨。

res ディレクトリ

・これらの XML レイアウトファイルは、res/layout/ に属している。"res" は、"resources" の略で、このディレクトリはコード以外のアプリケーション資産 (レイアウトファイルに加えイメージ、サウンド、ローカライズ用文字列など) すべてを含む

■既に存在するアプリケーションを XML レイアウトを使用するように変更

・Eclipse プラグインは、自動的にレイアウトファイル (main.xml) を生成する。
・作成した "Hello World" アプリケーションでは、レイアウトをプログラムから生成し、このファイルは無視された。
・しかし、ほとんどの場合にはコードではなく XML ファイルにレイアウトを定義すべき。
・以下では、どのように既に存在するアプリケーションを XML レイアウトを使用するように変更するのかを説明。

■ main.xml 内容の置き換え

・Eclipse のパッケージエクスプローラーの /res/layout フォルダーを展開し、main.xml を開く (開いたら、ウィンドウ下部の mail.xml タブをクリックすると XML ソースが見える)
・内容を以下の XML に置き換える。

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

■ strings.xml の編集

・res/values フォルダの中の strings.xml を開く
・これは、ユーザーインターフェースのすべてのデフォルトテキスト文字列を保存すべき場所
・Eclipse を使っているなら、ADT が既に 2 つの文字列を hello と app_name を持ってる。
・hello を "Hello, Android! I am a string resource!" 変更すると、以下のようになる。

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello, Android! I am a string resource!</string>
    <string name="app_name">Hello, Android</string>
</resources>
```

■ HelloAndroid クラスの修正

・HelloAndroid クラスを XML レイアウトを利用するように、以下の様に修正

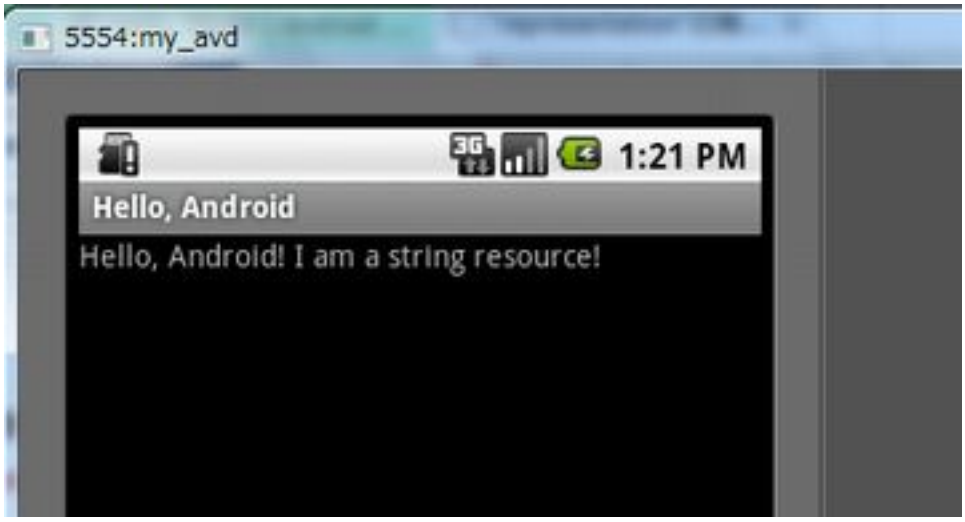
```
package info.typea.helloandroid;

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

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

・setContentView に View オブジェクトの代わりに レイアウトリソースのリファレンスを渡す。リソースは、R.layout.main として特定されるが、実際は /res/layout/main.xml で定義したレイアウトが解析されコンパイルされたオブジェクトを表す。
・Eclipse プラグインは自動でこの R.java クラスへ参照を生成する。
・Eclipse を使わない場合、Ant でアプリケーションをビルドすると生成される。

・アプリケーションを再度実行する。


5554:my_avd
1:21 PM
Hello, Android
Hello, Android! I am a string resource!