# 本書の想定する開発環境

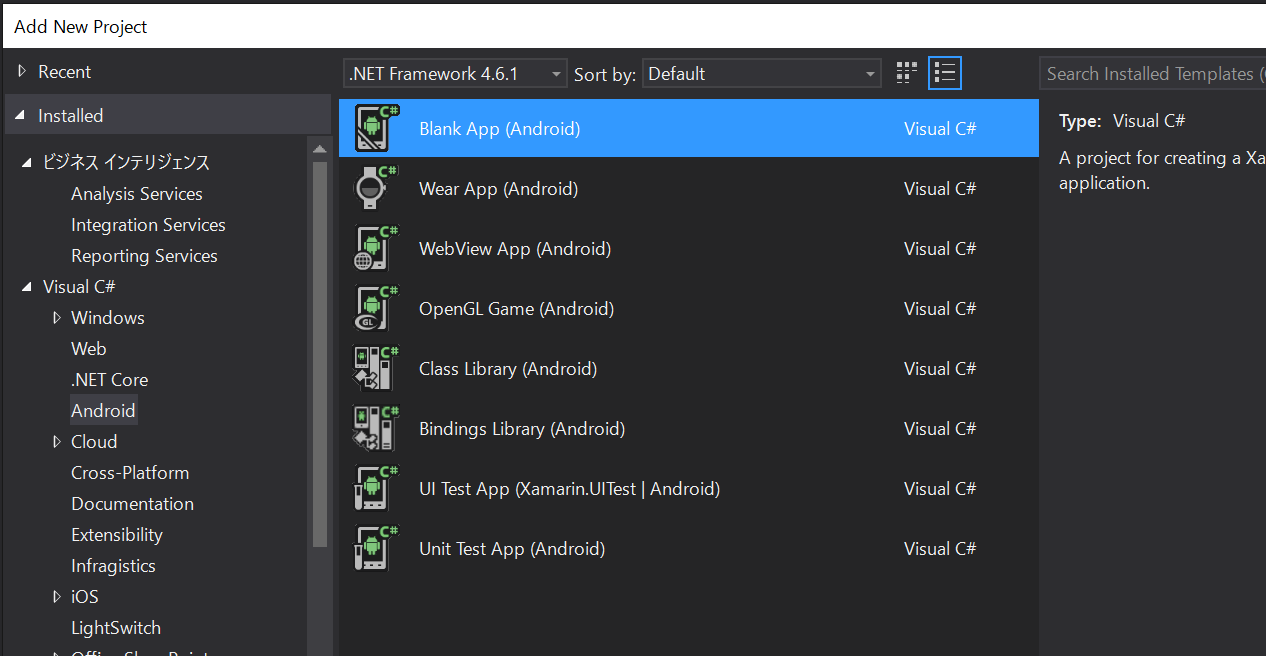
本書では、Xamarin.Androidを使ったアプリケーション開発について解説します。著者も入門しながらなので間違い等あるかもしれませんが、その場合はご指摘ください。以下の開発環境・言語を想定しています。

* Visual Studio 2015（英語の言語パックを当てています）
* Xamarin.Android
* C# 6.0
* Android 6.0（APIレベル23）

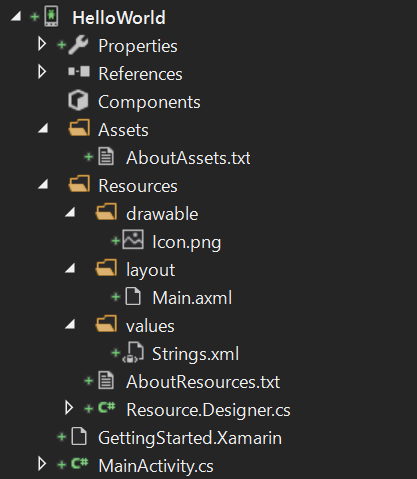
環境構築の方法やIDEの基本的な使い方や、C#についての基本的な文法は知っているものとして解説を行います。これらについて前提知識が十分でないかたは別の書籍やWeb上のサイトなどで情報を得てください。

# ハローワールド

ハローワールドを通じて、簡単なアプリケーションの開発の流れを見てみようと思います。Visual Studioのプロジェクトの新規作成から「Android」→「Blank App（Android）」を選択します。



「HelloWorld」とプロジェクト名をつけてプロジェクトを作成します。そうすると、以下のような構造をもったプロジェクトが作成されます。



MainActivity.csがメインの画面を表すクラスになります。Main.axmlが、MainActivityで使用されている画面のレイアウトを定義したファイルになります。Strings.xmlが、アプリ内で使用する文字列を定義したファイルになります。Resource.Designer.csファイルはMain.axmlやStrings.xmlなどから自動生成される、プログラム内から各種リソースにアクセスするためのIDが定義されたクラスになります。MainActivityクラスは以下のようになっています。

using System;

using Android.App;

using Android.Content;

using Android.Runtime;

using Android.Views;

using Android.Widget;

using Android.OS;

namespace HelloWorld

{

[Activity(Label = "HelloWorld", MainLauncher = true, Icon = "@drawable/icon")]

public class MainActivity : Activity

{

int count = 1;

protected override void OnCreate(Bundle bundle)

{

base.OnCreate(bundle);

// Set our view from the "main" layout resource

SetContentView(Resource.Layout.Main);

// Get our button from the layout resource,

// and attach an event to it

Button button = FindViewById<Button>(Resource.Id.MyButton);

button.Click += delegate { button.Text = string.Format("{0} clicks!", count++); };

}

}

}

まず、ポイントとなるのがActivityクラスを継承している点です。このクラスを継承したものがAndroidでは画面として使えます。次に、このクラスが初期起動時に呼び出されるActivityであることを定義するために属性が定義されています。

[Activity(Label = "HelloWorld", MainLauncher = true, Icon = "@drawable/icon")]

MainLauncher = trueの記述がそれになります。LabelはラベルでIconはアイコンです。Activityが生成されたときにフレームワークから呼び出されるOnCreateメソッドで初期化処理が行われています。SetContentViewメソッドが画面定義ファイルとActivityの紐づけを行うメソッドになります。Resourceクラスが、先ほど説明したResource.Designer.csファイルで自動生成されたファイルでMain.axmlを指し示すためのIDにResource.Layout.Mainでアクセス出来るようになっています。これでMain.axmlで定義された見た目が使用されるようになります。Main.axmlは、以下のように定義されています。

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

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

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<Button

android:id="@+id/MyButton"

android:layout\_width="match\_parent"

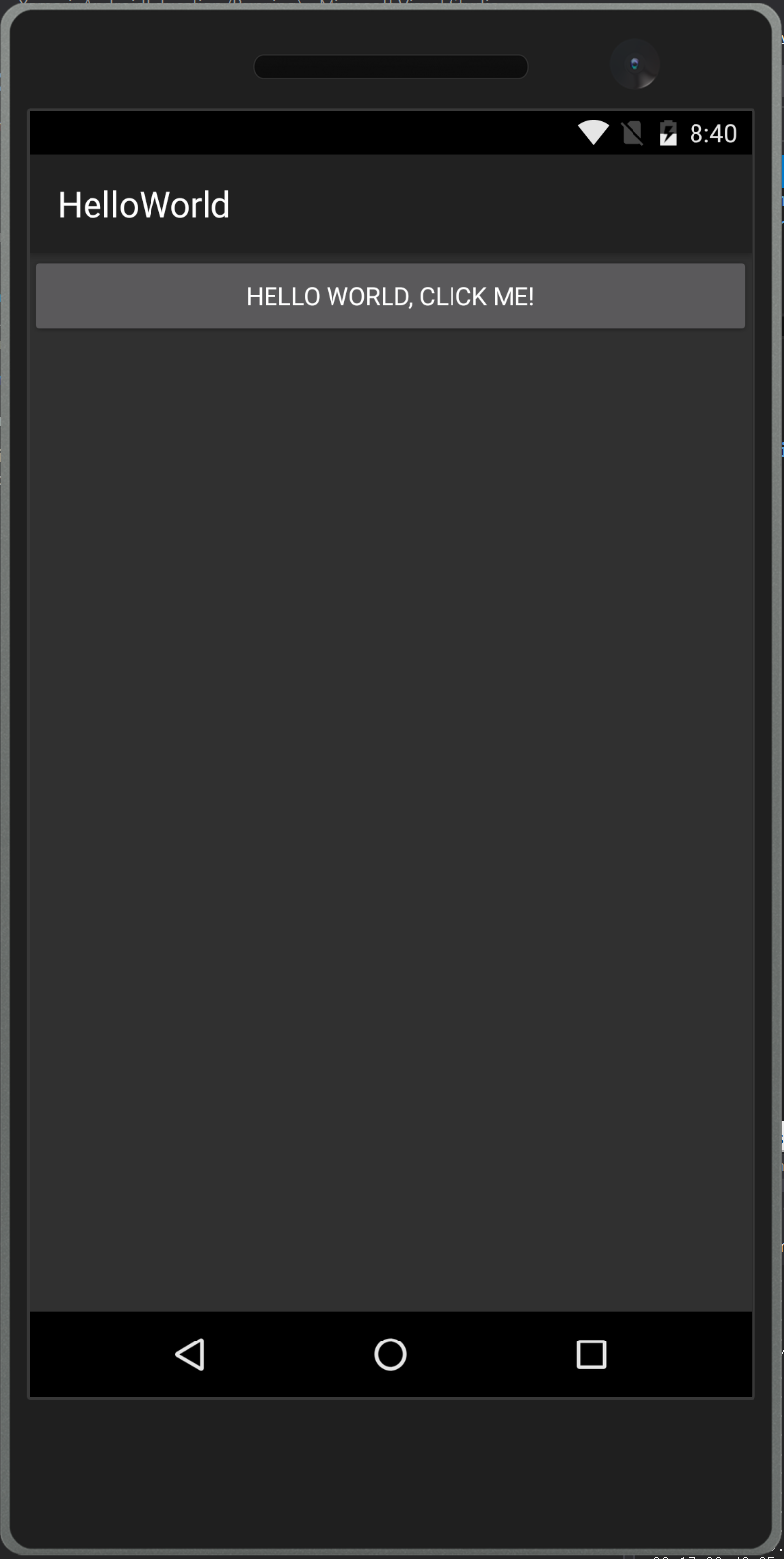
android:layout\_height="wrap\_content"

android:text="@string/Hello" />

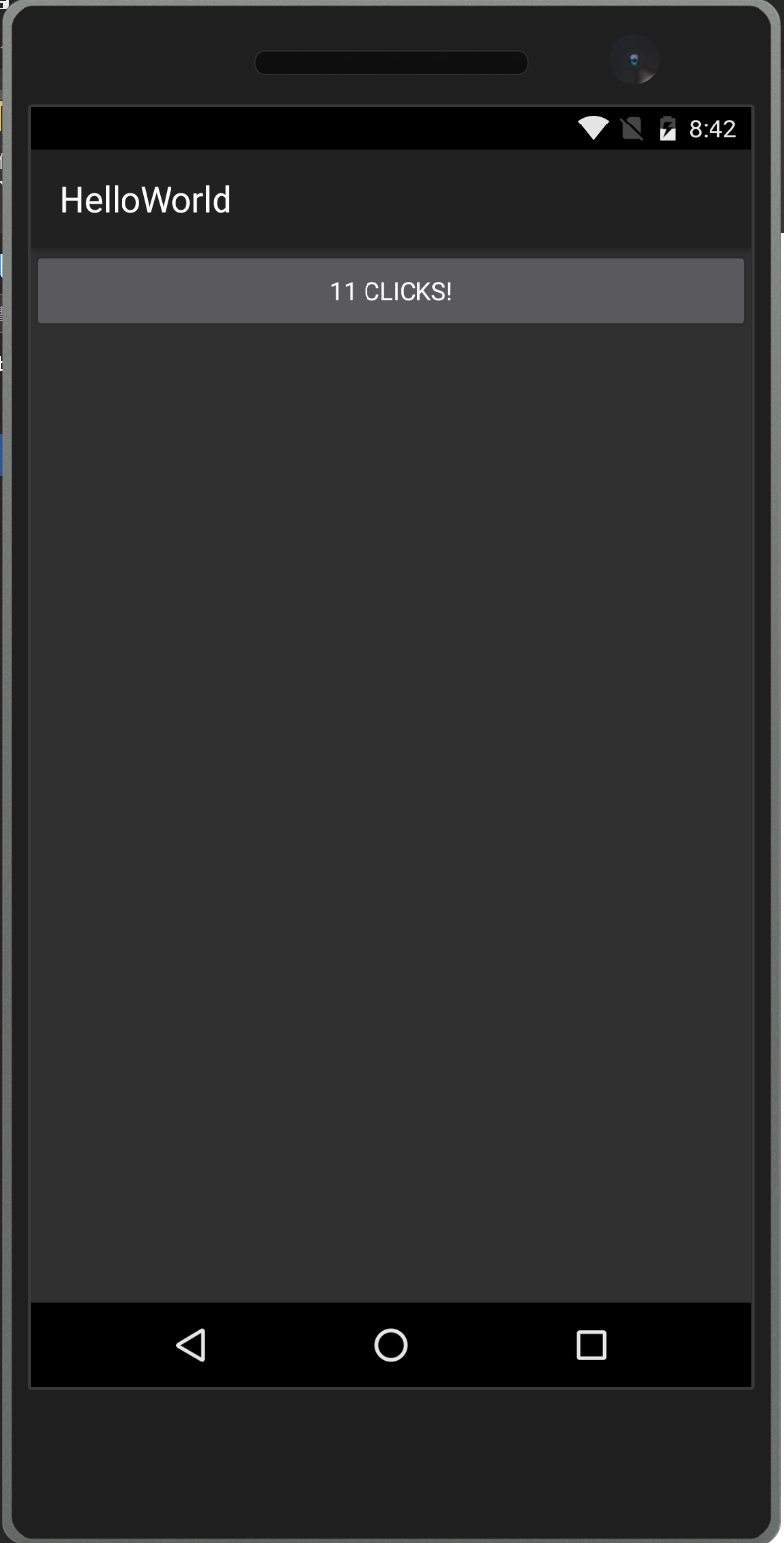
</LinearLayout>

xmlns:android=”http://schemas.android.com/apk/res/android”名前空間が定義されたXMLファイルになります。LinearLayoutという要素を縦や横に並べるレイアウトの中に、Buttonが置かれているというくらいの理解で今は大丈夫（というかこれを書いてる時点で、その程度の理解です）だと思います。ポイントとして、押さえておきたいのが、Buttonに対して定義されているandroid:id=”@+id/MyButton”という属性です。これはMyButtonという名前のIDをボタンに割り当ててることになります。

ではMainActivityに戻ります。MainActivityでは、FindViewByIdメソッドを使って画面内のコントロールにアクセス出来ます。ここでは、MyButtonというIDのコントロール（先ほど確認したMain.axml内で定義されてるやつですね）を取得してイベントを登録しています。処理自体はcount変数をカウントアップしているだけのシンプルなものですね。では、実行してみましょう。



ボタンを押すとカウントアップされた値が表示されます。



## BMIを計算してみよう

これを少しいじってBMIを計算するアプリを作ってみようと思います。身長と体重を入力するボックスを作ります。Main.axmlを以下のように変更します。

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

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

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<Button

android:id="@+id/MyButton"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/Hello" />

<EditText

android:id="@+id/EditTextHeight"

android:hint="@string/Height"

android:inputType="numberDecimal"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/EditTextWeight"

android:hint="@string/Weight"

android:inputType="numberDecimal"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</LinearLayout>

EditTextがAndroidのテキスト入力用コントロールになります。android:hint属性は、ウォーターマークをだすための属性になります。@string/Heightや@string/Weightは、strings.xmlで以下のように定義している文字列を参照する書き方です。BmiMessageは、プログラム内から使用する予定の文字列になります。

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

<resources>

<string name="Hello">Hello World, Click Me!</string>

<string name="ApplicationName">HelloWorld</string>

<string name="Weight">体重</string>

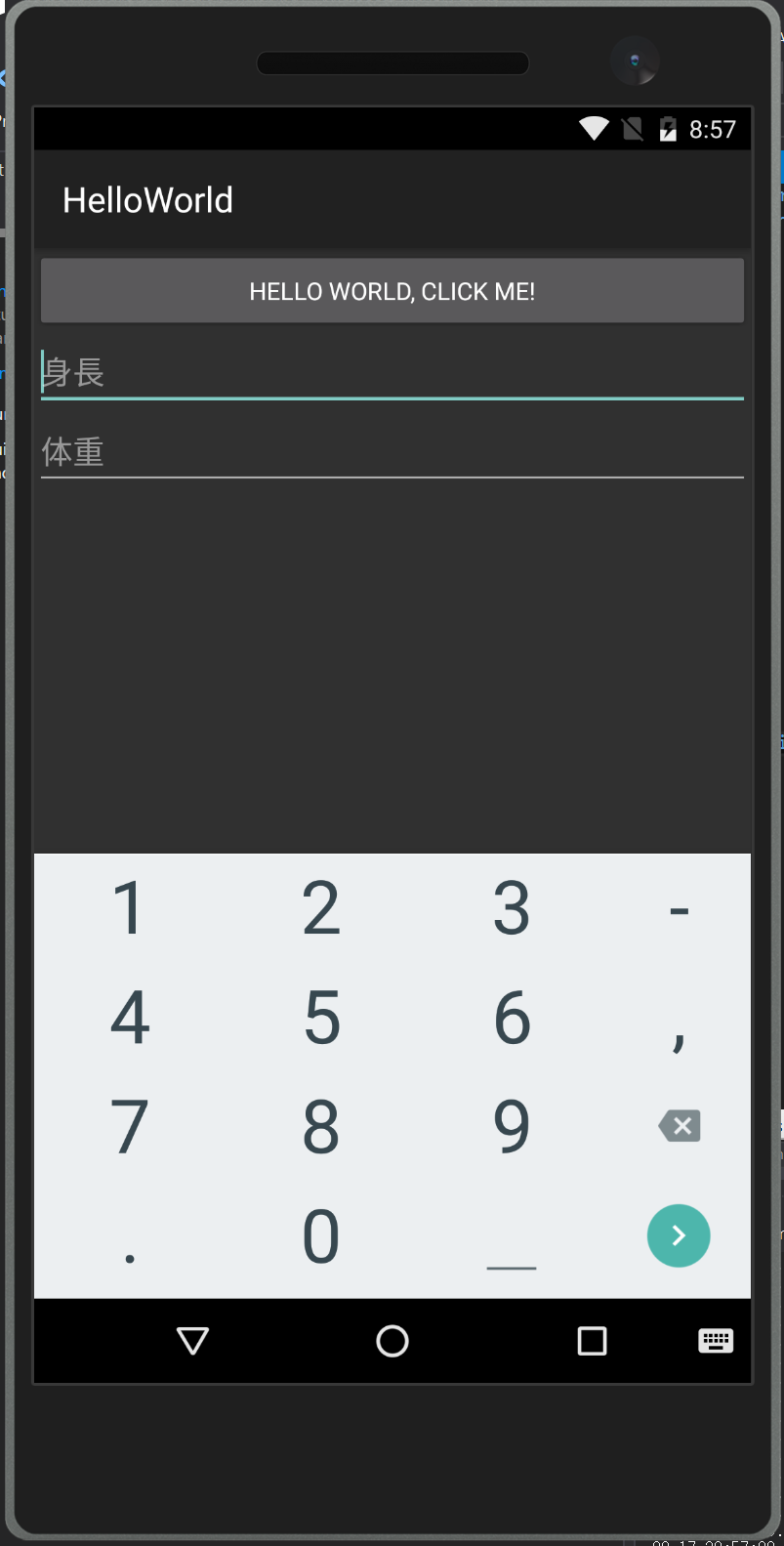
<string name="Height">身長</string>

<string name="BmiMessage">BMIは%fです</string>

</resources>

android:inputType=”numberDecimal”は、数値を入力するための設定になります。android:layout\_widthとandorid:layout\_heightは、幅と高さをどうするかを指定します。今回は横幅は親に合わせて、縦幅はコンテンツに合わせるように指定しています。（つまり横長）

実行すると、以下のように表示されます。



身長、体重のウォーターマークが出ている点と、入力欄にフォーカスを合わせたときのソフトウェアキーボードが数字になっているので、先ほどの設定が効いていることがわかります。

最後に、処理を書いていきます。BMIの計算式は「体重(kg) ÷ {身長(m) Ｘ 身長(m)}」なので、ボタンが押されたときに、そう表示されるようにしています。アプリの仕様的にわかりにくいですが、今回は体重はkgで、身長はmで入力されてるものとして処理しています。

using Android.App;

using Android.OS;

using Android.Widget;

namespace HelloWorld

{

[Activity(Label = "HelloWorld", MainLauncher = true, Icon = "@drawable/icon")]

public class MainActivity : Activity

{

protected override void OnCreate(Bundle bundle)

{

base.OnCreate(bundle);

this.SetContentView(Resource.Layout.Main);

var button = this.FindViewById<Button>(Resource.Id.MyButton);

var height = this.FindViewById<EditText>(Resource.Id.EditTextHeight);

var weight = this.FindViewById<EditText>(Resource.Id.EditTextWeight);

button.Click += (\_, \_\_) =>

{

double h;

if (!double.TryParse(height.Text, out h))

{

return;

}

double w;

if (!double.TryParse(weight.Text, out w))

{

return;

}

// 体重(kg) ÷ {身長(m) Ｘ 身長(m)}

var bmi = w / (h \* h);

button.Text = string.Format(this.Resources.GetString(Resource.String.BmiMessage ,bmi));

};

}

}

}

ActivityにResourcesというプロパティがあり、これのGetStringメソッドを使うことでStrings.xmlに定義した文字列をプログラム中から使うことが出来ます。今回はBMIの計算結果を表示する文字列として使っています。

実行して入力してボタンを押すと以下のようになります。

