# 共通項目

## memo

クリスにflutter を使えと言われていた気がしたが。どうしてなんだっけ？20.07.10

## 通信関連

### Bluetooth

Bluetoothのパーミッションを宣言する為に，app\manifests\AndroidManifest.xmlに以下を追加

<manifest ... >

<uses-permission android:name="android.permission.BLUETOOTH" />

<uses-permission android:name="android.permission.BLUETOOTH\_ADMIN" />

<!-- If your app targets Android 9 or lower, you can declare

ACCESS\_COARSE\_LOCATION instead. -->

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

...

</manifest>

コード

import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

BluetoothAdapter mBluetoothAdapter;

mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

if (mBluetoothAdapter == null) {

// エラー: Bluetooth なし.

}

<https://moewe-net.com/android/connect-bluetooth>

### Ethernet

app\manifests\AndroidManifest.xml に以下を追記

...

</application>

<uses-permission android:name="android.permission.INTERNET"/>

</manifest>

コード例（送信）：

try{

socket = new Socket("192.168.1.1",1234); // IP(host名)とポート番号

PrintWriter pw = new PrintWriter(socket.getOutputStream(), true);

String text = "hello from android";

pw.println(text);

}

catch(Exception e){

e.printStackTrace();

}

#### エラー対応（Ethernet）

java.net.UnknownHostException

tryを使わずにソケット通信を行おうとするとこのエラーが出る。

単純にtryを使う事でエラーは無くなった。20.07.17

最初はアンドロイドはhostsファイルに記載のないサーバーとの通信は行えないようにできているかと思ったが，そういう事では無さそう。

## 開発環境

大きくXamarinとAndroid Studioがある

|  |  |  |
| --- | --- | --- |
|  | Android Studio | Visual Studio(Xamarin) |
| 開発会社 | Android純正 | マイクロソフト（買収） |
| 使用言語 | Java，Kotlin | C# |
| 画面デザイン | .xml デザイナ有り | .amal　xmlのようなもの  　例えばactivity\_main.axmlと言うファイルがResourcesにある |
| 端末（スマホ）  エミュレータ | 付属（AVGマネージャー → ） | 付属  ※但し上手く動かない？ |

## Manifests

フォルダ構成：app\manifests\AndroidManifest.xml

xml形式で記述されている。

### application

### activity

指定アプリケーションの動作を指定？

intent-filter 親コンポーネントの機能を宣言

action

android:name="android.intent.action.MAIN" // 指定アクティビティをエントリポイントにする。

android:name= "android.intent.category.LAUNCHER" // 指定アクティビティをエントリポイントにする。

# flutter

## 概要

単純にAndroid 開発を楽にしてくれる

flutterを使う意義を一番感じた情報：[Androidネイティブアプリ開発者から見たFlutter](https://medium.com/@itometeam/androidネイティブアプリ開発者から見たflutter-e85248b29ed1)

## インストール方法

・ダウンロード：<https://flutter.dev/docs/get-started/install/windows>

⇒ どうやらGitのファイルになっているらしい。つまりGitでもダウンロード可能。

⇒ 英語だが，このサイトがチュートリアルにもなっている。基本的にこの指示に従えば良い。

適当なディレクトリに配置。　※C:\Program Files　のような管理者権限の必要なディレクトリは避ける。

・PATH：flutter\bin　までのパスを通す必要がある。 ⇒ flutterコマンドが使えるようになる

・Power Shellかcmdでflutter doctor を実行 ⇒ [!] が付いた部分があれば，修正する必要がある。

# power shell

PS C:\Users\mtc170001> flutter doctor

[!] Android toolchain

X Android SDK file not found: adb. の場合

⇒　Android Studio をインストールすればよい。

*※あるバージョンのAndroid Studioでは /tools ディレクトリが作成されず，Android SDK file not found:が出続ける。*

*対応：Configure → SDK Manager → SDK Toolsタブ*

*→ Hide Obsolete packagesのチェックを外す → Android SDK Tools (obsolete) をインストール*　　[情報元](https://stackoverflow.com/questions/60462340/android-sdk-is-missing-a-tools-folder-meaing-no-sdkmanager-meaning-android-li)

X Android licenses not accepted.

　　⇒ flutter doctor --android-licenses を実行。

[!] Android Studio (version \*.\*)

⇒　Android StudioでFlutterとDartのプラグインをインストールする

　　Configure → Plugins　　　　　※１度適当にプロジェクトを作成しないと表示されないかも

　　→　 とDartを選択。（Flutter入れる時にDartも選択できる）

[!] No devices available

⇒ スマホなどをUSB接続すればよい。エミュレータでも可。

参考：<https://qiita.com/satodayo/items/84d369fd2dc90f8cda99>

## プロジェクトの作成

Android　Studioで（プロジェクトを閉じて　→）「Start a new flutter project」

またはメニュー：New → d → d → d → d

Flutter Application

Flutter SDK path：最初にzip解凍ファイル（Gitファイル）を配置したパスを指定する。

Include Kotlin supportや Include Swift support はお好みで。

Use android.\*artifacts

Android10対応の新しいライブラリで，Goolgeは全てのプロジェクトでの使用を推奨している。

最近ではUse legacy Android support librariesと言うものに変わっているらしい。20.07.15

意味こそ正反対だが，意味合いはほぼ同じ。

AndroidManifest.xml could not be found. への対応：

# Android　Studio

## 概要

### 箇条書き

プロジェクトを開いてから実際に使用可能になるまで少し時間が必要らしい

エミュレータはあくまで別アプリ，と言った感じ。デバッグまえに起動しておく必要がある。

ステップ実行はF8

F9が継続実行？

File → ExportToZip

View.getid

### フォルダ構成

app

├ manifests エントリポイントなど，このアプリケーションの基本設定が記述される。

│

├ java 各イベントトリガに対する，実行コードが記述される。

│

└ res リソースを管理する。（開発アプリで使用する画像ファイルなど。）

　　├ drawable ？

　　├ layout GUIの画面デザイン構成が記述されている。

　　│　└ *activity\_name*.xml 表示する部品やその詳細を記述する。１つのactivity≒１つの画面。

　　├　mipmap

　　└ values

　　　　├ colors.xml

　　　　├ strings.xml アプリケーションで使う文字列を扱う。多言語化も想定し，常にここで文字列を設定する。

　　　　└ styles.xml

⇒ 割と，html+CSS的な発想に思われる。20.07.06

この中で編集の頻度が特に高いのは res\values\strings，layout\activity\_name.xml，javaフォルダ内のファイルの３つ。

### R.java

app.R ? 20.07.15 android.R もあるらしい

アプリケーションのリソースを管理しているRクラスを抽象化したもの。

プロジェクトの最初のビルドで作成され，その後ビルド毎に自動的に更新される。

基本的に各リソースはIDで管理される。

Cannot resolve symbol ‘R’ エラー対応

大体，layoutやres\stringsの .xmlファイルの記述ミス

## インストール

#### Linux

64bit Linuxの場合は先に以下を実行

sudo apt install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.0:i386

AndroStudioのtar.gzファイルをダウンロード。解凍して配置

cd android-studio/bin

bash studio.sh # 起動

/homeディレクトリがいやな場合

sudo cp -r android-studio /usr/local

bash studio.sh # 起動

デスクトップエントリの作成

Android Studioを起動 →  → Create Desktop Entry

　もちろん[公式](https://wiki.archlinux.jp/index.php/デスクトップエントリ)の方法でも良い。

公式：[https://developer.android.com/studio/install?hl=ja#linux](https://developer.android.com/studio/install?hl=ja" \l "linux)

## 簡単なアプリのチュートリアル

### 新しいProject

|  |  |
| --- | --- |
| 画面 | 操作内容／値 |
| Start a new Android Studio project | Welcome to Android Studio画面にある |
| Select a Project Templete | Empty Activity　　空っぽ。最初はこれから始めればよい。 |
| Configure Your Project | Name： プロジェクト名。アルファベットを使う。Android端末で表示される文字列とは別。  Use legacy Android support libraries：  Support Libraryと言うのはAndroid10(X)以前のライブラリの事。  新機能を古いAndroid OSでも使えるようにした下位互換。  必要ない。少なくとも，このチュートリアルには必要ない。 |
|  |  |
|  |  |

### 部品(Wedget)の配置

#### 補足情報

補足１．　xmlns

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

xmlnsはxml上での名前空間の指定。この宣言以降，android: *hoge*="*fuga*" が使用できる。

これは以下の様に捉えて良いらしい。

"http://schemas.android.com/apk/res/android":layout\_width="match\_parent"

<!-- ↓↓↓ -->

android:layout\_width="match\_parent"

これを「マークアップ言語」と呼ぶらしい。

補足２．　not constrained エラー

This view is not constrained. It only has designtime positions, so it will jump to (0,0) at runtime unless you add the constraints

このエラーの意味：layout\_constraint 関連の指定が無く，どこを基準にWedgetを配置して良いか分からない状態。

### ２つ目の画面（Activity）

java.com.example.*app\_name*. を右クリック

→ New → Activity → EmptyActivity

→ Activity Name を編集　→ 

また，Activityを追加した後にはManifest.xmlの更新が必要になる。

（Android Studioが自動で追記してくれるが）

public void onNextButtonClick(View view) {

Intent intent = new Intent(MainActivity.this,*SubActivity*.class);

// 呼び出し先の Intent（画面）に引数を渡す場合

intent.putExtra("MenuName", "hoge");

startActivity(intent);

}

### 実機への書き込み

## View

### Layout

memo

こんな表記を見つけた data タグ？　　20.07.17

<data>

<variable

name="textString"

type="String" />

<variable

name="textColor"

type="Integer" />

</data>

#### 基本

フォルダ構成：app\res\*activity\_name*.xml

xml形式で記述されている。htmlとcssの中間のような感じで，画面デザインがxml形式で記述されている。

Layoutと言う枠の中に１つ１つのWedget を配置していく。



画面の切り換えについて：

右画像のボタンを押す事で編集画面を切り替える事ができる。

Design 専用デザイナで直感的に編集

Code xmlファイルを直接編集

Split 画面を半分に分けてデザイナとxmlを同時に表示する

⇒　深い理解の為にCode画面での編集を標準とし，その都度Design画面で確認を行うようにする。

@+id R.javaにidを追加する

@id 既にR.javaに存在しているidのリソースを使用

@string res\strings.xml から文字列を取得する。@string/*id\_name*

wrap\_content 各部品に必要なサイズが自動的に割り振られる。

match\_parent 親部品で有効な範囲いっぱいまで広げる。

#### 単位について

スマホ（Android）は機種によって画面解像度が異なる為にpixelという単位を使うには不向き。

dp(density-independent pixels)とは密度非依存ピクセルのことで、dipとも呼ばれる。

sp フォント向きサイズ？

px(pixel) 但し前述のとおり，ほとんど使う事は無い。

#### Layoutの種類

Constraintlayout

比較的新しく追加されたLayout

FrameLayout

単純に複数のWedget をグループ化して，内部的なViewを形成する。

RelativeLayout

Wedgetを相対的に配置する

LinearLayout

横一列，または縦一列に並べてくれるLayout。何もせずに使うと，CSSを使わないhtmlの様に一列に並んで殺風景になるので，大体，入れ子にして使用される。

専用Attribute

android:orientation="vertical" 内にあるWedgetの並び方向 値：vertical/ horizontal

TableLayout

専用Attribute

android: shrinkColumns 折り返してでも狭いスペースでコンテンツを表現しようとする

android: stretchColumns 行の開いているスペースいっぱいの空間を取ろうとする

値：左端が0から始まる文字列数値で対象の列を指定する。

TableRow TableLayoutの子Wedget

専用Attribute

style="@style/CodeFont"

Wedgetの関連Attribute

layout\_span="2" セルの結合。htmlで言うところのcolspan

layout\_column="1" 列番号の指定

GridLayout

#### 再生可能なレイアウト

例えばYesNoダイアログみたいな何度も使う画面や，位置固定のメニューバーなどが作れる。

１．新しいLayout用の .xml ファイルを作成する。

※最初の情報ではFrameLayoutを使っていたが，Layoutはなんでも良いらしい 20.07.17

２．利用する側のLayout用 .xmlファイルで layoutを指定する。

　必須ではないが，idも一意のものを指定する。

<include layout=”@layout/*custom\_layout* android:id="@+id/*included\_view1*" />

（公式情報）<https://developer.android.com/training/improving-layouts/reusing-layouts?hl=ja>

### Wedget

各部品の設定値はAttributeと呼ばれる。（Visual Studioなどでプロパティと呼ばれるもの）

#### 共通アトリビュート

android:layout\_gravity

右寄せ，上寄せなど。

値：right, left, top, bottom, center, center\_vertical, center\_horizontal

layout\_constraintの違いは？

android:layout\_weight

個々のWedgetの幅比率

LinearLayoutやTableLayout で個々のWedgetの表示比率を変える

（関連）weightSum ※親Layoutで設定し，要素は２つだけと1:1:1とかで並べたい時に使う。

android:layout\_width 部品の横幅。単位はピクセルではなく，dip。　　(see also) layout\_height

wrap\_content 各部品に必要なサイズが自動的に割り振られる。

match\_parent 親部品で有効な範囲いっぱいまで広げる。

android:layout\_constraintTop\_toTopOf

自身の上端と指定したViewの上端を揃える。つまりは上寄せ。

（関連）toRightOf, toLeftOf, toBottomOf

（補足）

toRightOf とtoLeftOf を同時に使う事で中央配置が可能

android:layout\_constraintStart\_toStartOf

自身の左端と指定したViewの左端を揃える・

（関連）layout\_constraintEnd\_toEndOf

⇒ Left とStartの違い：

android:layout\_constraintHorizontal\_bias

3:7など比率を持たせて表示する

android:textStyle 太字，イタリック　などの設定

style

各Wedgetの表示（描写）スタイルを指定する。

TableLayoutなどで，内部のレイアウトのスタイルを指定してくる警告に対応する時に使ったりもする。

#### Button

##### OnClickの処理

タイプ１：匿名クラス

初心者向きサンプルとしては一番多い気がする。

protected void onCreate(Bundle savedInstanceState) {

...

btTest.setOnClickListener(new View.OnClickListener() {

public void onClick(View view) {

TextView (TextView)txView = findViewById(R.id.text\_view\_msg);

txView.setText("hoge");

}

});

}

Androidのバージョンによっては (TextView)txView = findViewById() の型変換記述は不要（自動）。

タイプ２：変数

protected void onCreate(Bundle savedInstanceState) {

...

final Button btTest = findViewById(R.id.button\_test);

btTest.setOnClickListener(buttonClick);

}

private View.OnClickListener buttonClick = new View.OnClickListener() {

@Override

public void onClick(View view) {

switch (view.getId()) {

case R.id.button\_test:

TextView txView = findViewById(R.id.text\_view\_msg);

txView.setText("hoge");

break;

}

}

};

タイプ３：implements

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

final Button btTest = findViewById(R.id.button\_test);

btTest.setOnClickListener(this);

}

public void onClick(View view) {

if (view.getId() == R.id.button\_test) {

TextView txView = findViewById(R.id.text\_view\_msg);

txView.setText("ggg");

}

}

}

以タイプ４：layoutに記述する

activity\_main.xml

<Button

android:onClick="onTestButtonClick"

/>

MainActivity.java

public void onTestButtonClick(View view) {

// 処理

}

この方法が一番スッキリするし，デザイナとの連携も取れる気がする。

JavaScript的な発想な気がする。どっちが先だったんだろう。

#### ListView

entries 選択候補の文字列配列

#### RadioButton

以下の様にxmlを記述する。

<RadioGroup>

<RadioButton

<!-- 設定値　-->

>

</RadioGroup>

#### TextEdit

アトリビュート(android:)

digit 指定した範囲内の文字しか使用できなくする入力規制

（例）

android:digits="0123456789." # IPアドレス用の入力規制

inputType 入力を受け付ける文字列種類を指定する。例）number, textMultiLine, textUri など。

#### TextView

概要

文字列を表示する。.NETのLabelのような機能。

メソッド：

setText() 表示文字列の変更。通常idで指定する。

EditText

#### Spinner

CheckBox

SeekBar

RatingBar

Switch

#### .NET対比表（C#ではコントロールと呼ばれている）

|  |  |  |  |
| --- | --- | --- | --- |
| 名前 | 機能 | C# |  |
| CheckBox |  | 同名 |  |
| EditText | 文字列の入力 | Textbox |  |
| ListView | Spinner との違い？ | 同名 |  |
| ImageView |  | ImageBox |  |
| RadioGroup |  | RadioButton |  |
| RatingBar |  |  |  |
| SeekBar |  |  |  |
| Spinner | ドロップボックス型の複数選択 | ComboBox |  |
| Switch |  |  |  |
| TextView | 文字列の表示 | Label |  |
|  |  |  |  |

### レイアウト例

（補足）ConstraintLayoutが出る前の情報だとLinerLayoutやRelativeLayoutを入れ子にする例が多いが，まずはConstrainLayoutだけで対応可能かを考える事にする。

### エラー対応（View）

This view is not constrained…

layout\_constraint による配置の制約の記述がない為に配置ができない。

verticallyとかhorizontallyとかなっている場合は，文字通りその方向だけ記述がない。

## Activity

ファイルパス：app\java\com.example.

派生クラス：

AppCompatActivity ポートライブラリを利用する為の基本となるクラス？

ActionBarActivity

### メソッド

onCreate() .NETで言うとLoadに近い？

これがアプリケーションのエントリポイントになる事が多い。

onStart() .NETで言うとActivatedに近い？

onRestat() 画面が非表示（Stop）になった後，再表示される時に発生する。

onResume() 他の画面が表示（Pause）された後，再表示される時に発生する。

### エントリーポイント

以下の記述があるActivityがエントリーポイントになる

<intent-filter>

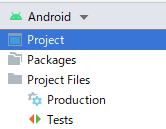
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

## 操作

### プロジェクトビューワ

一番左上のAndroid のところをProjectにするとプロジェクトビューワに切り替わる。（右図参照）

外部ライブラリの取り込みなどを行う時に使う。　⇒ app\libs

（基本的にプロジェクトビューワの方が見やすいような 20.07.29）

もとに戻す時はAndroidに戻せばよい。

## GUI関連

### オブジェクトの取得

以下の様に行う。

Button button1 = (Button)findViewById(R.id.button\_a);

Androidのバージョンによっては() を用いた型変換が不要

たかが１つの文字列でも，values\strings.xmlを利用すると良い。

### 定義へ移動

対象メンバなどを右クリック → GoTo

Implementation 定義している行へ

Superclass 親クラスのソースコードへ

strings.xml

配列にしたい時は<string-array>

# Xamarin

SDKをXamarinと共有

1.Xamarin for Visual Studioをインストールする

2.Android Studioをインストールする

この際、Android SDKはXamarinの物とは別に「スペースも日本語も含まれないパスへ」インストールしてください

3.Visual Studioを起動し、「オプション」->「ツール」->「Xamarin」から、Android SDKのパスを2.でインストールした箇所へインストールする

4.「コントロールパネル」->「プログラムのアンインストール」から「Android SDK Tools」をアンインストールする

ツール->オプション->Xamarin->Androidの設定

よりパスをAndroid Studioで入れたものに変更

デフォルトだと C:\Users\xxxx\AppData\Local\Android\sdk

20.07.06 Visual StudioとAVD Manager が何度やっても上手く連動しない為，Xamarinはしばらく保留。

# エミュレータ(AVD)

普通のスマホと同じ様に，言語設定ができるので，表記を日本語にしておいた方が良い。

# Java

箇条書き

世界中でクラス名が一意になるように，PojectName.ClassName となっている。

## 予約語

super 継承元の親クラスの事。≒C言語系のbase。

final （恐らく）継承不可にする

extends クラス定義の際の継承元クラスの指定。親クラスを引き継ぐ。

implements interfaceの実装を記述。要するにinterfaceの継承。

## アットマーク（なん呼ぶか分からない）

@Nullable

null 許容型の事と思われる

protected void onCreate(@Nullable Bundle savedInstanceState) {

}

@Override

メソッドに付けて，superクラスのメソッドをオーバーライド（上書き）する。

## プログラミング標準

### ループ

#### forループ

static void printList(List<?> list) {

for (Object elem : list) {

System.out.println(elem);

}

}

### ジェネリクス

|  |  |  |
| --- | --- | --- |
| 書き方 | 名称 | 詳細 |
| List<?> list | 非境界 | 全ての型対応 |
| List<? extends Number> list | 上限境界 | 指定のクラスを継承した型のみ |
| List<? super Number> list | 下限境界 |  |

## 構造化

### クラス

・あるクラスの内部だけで使うクラスはprivate で定義する

例）

public class TestMainClass{

private class TestSubClass {

}

}

・publicクラスは .java ファイル１つにつき１つしか書けない。

## 雑多情報

### 非同期

AsyncTask やAsyncTaskLoaderは非推奨。運営であるGoogleが非推奨と言っている。

java.util.concurrent を使うべき（？）。　⇒ 20.07.17現在，ほとんど情報がない

#### AsyncTask

上にも書いたが運営であるGoogleが非推奨と言っているので，他の情報を探すよう意識する。

doInBackground(Params…) ◦バックグラウンドでの非同期処理、何か重い処理を記述する

◦ここに渡されるパラメータの型は1つめのParams

onProgressUpdate(Progress…) ◦途中経過をメインスレッドで表示する場合

◦doInBackground内でpublishProgressを設定するとメインスレッドから呼ばれ、パラメータの型は2つめのProgress

onPostExecute(Result)

コード例（実装）：

private class TestTask extends AsyncTask<String,String,String> {

　　// コンストラクタ

public TestTask () {

}

@Override

protected String doInBackground(String... objects) {

try{

// 処理

}

catch(Exception e){

e.printStackTrace();

}

return null;

}

}

コード例（呼び出し）：

public void onButtonClick(View view) {

TestTask testTask = new TestTask ();

testTask.execute("hoge", "fuga", "piyo");

}

20.07.17現在，この３つのジェネリック型が何を指すのか良く分かっていない。

Android 3.1(タブレット用)や4以上の端末はUSBホスト機能がある。

### シリアル通信

#### 概要

Android 3.1(タブレット用)や4以上の端末はUSBホスト機能がある

Javaは、シリアル通信用のライブラリが含まれてない。

Communications APIと互換性があるRXTXライブラリと呼ばれるものがある。

<https://tomosoft.jp/design/?p=2987>

Google Play に公開されているアプリで232C通信できたので，ハード的には間違いなくできる。20.08.05

テスト環境

Android（タブレット） ：ASUS ZenPad 3 8.0 (Z581KL)

Androidのバージョン ：

232c to USB ：Prolific PL-2303

USB typeAへの変換 ：Cable Matters USB C USB A 変換アダプタ

アプリ：Serial USB Terminal 　，USB Serial Console　

Androidアクセサリ：<http://www.picfun.com/android/androidframe.html>

USBホスト：<https://developer.android.com/guide/topics/connectivity/usb/host>

com.felhr.usbserial　結局これをつかっている？？

20.07.29

FTDI社のFT232RL用ドライバがあるらしい。但し同社のUSB変換を使う必要があるのでは？

#### ライブラリを使う方法

usb-serial-for-android-v010.jar を使う

[Githubダウンロード先] <https://github.com/mik3y/usb-serial-for-android>

（参考）<http://shokai.org/blog/archives/6962>

方法１：USBケーブルを使用：

アンドロイドで232C通信が必要な時は直接type-c と232Cを変換するケーブルを使った方が不具合が発生しにくいらしい。

<https://www.ratocsystems.com/products/subpage/usb60fc.html>

方法２：

計量器の類をブルートゥース対応のものを選定する，とした方がスマホ的だとは思うが，ブルートゥース対応のものは大体高い。そこでラズパイの様なマイコンを使って232C通信させ，wifiまたはブルートゥースでデータを送信する手がある。

（232C ↔ ブルートゥース変換デバイスも存在するが，やはり高い；大体２万円以上）

こんなものを使うと232Cブルートゥース変換基板を作成できる。

<http://akizukidenshi.com/catalog/g/gK-07378/> \2,400

（補足）SPP(シリアルポートプロファイル) と言うシリアル通信を無線化する専用プロトコルが存在する為，wifiを用いる場合よりも無線化しやすい。

# ライブラリ・リファレンス

（オンライン資料）<https://developer.android.com/reference/classes?>

## パッケージ(package)

androidx.appcompat.app

android.widget Button，TextView，AdapterViewなど

android.view View

## クラス(android.widget)

注意：WedgetやLayoutの情報は[こちら](#_View)にまとめてある。

### Adapter

### AdapterView

#### AdapterView.OnItemClickListener

public　static　interface　AdapterView.OnItemClickListener

概要：対象のAdapterView内である項目クリックされた際のコールバックメソッドonItemClick() を提供する。

## クラス（その他：仮）

### Context

通常Activityインスタンスが指定される。.NETのobjectの様な物？それともControl?

### CharSequence

20.07.16 恐らく文字列の事

### Intent

別アプリからのアクションをリクエストする時に使用するメッセージオブジェクト。

### Listener

HelloListener

ClickListener

Log

#### メソッド

|  |  |  |
| --- | --- | --- |
| .wtf |  | Assertを記述：開発者にとって絶対に発生してはいけない問題 |
| .e |  | Error： |
| .i() |  | ログレベルInfoでログを記述 |
| .d |  | Debug： |
| .v |  | Verbose：詳細情報 |
| .w |  | Warn：エラーとは言えないが潜在的な問題 |

### Service

別スレッドで動く処理を管理してくれるクラス。

### Socket

名前空間：java.net.Socket

概要：ethernet によるソケット通信を行う

特徴：UIスレッドで使う事は出来ないので，別スレッドを用意する必要がある。

### Toast

メソッド

makeText(Context context, CharSequence text, int duration)

# 用語

Support Library

なお，名前空間は android.support.v7.app. AppCompatActivity などとなっている

AndroidX

Support Libraryの後継でAndroid10.0のライブラリ（らしい）

なお，名前空間は androidx.appcompat.app.AppCompatActivity などとなっている

IntentFilter

暗黙的 Intent を受信することが出来る。