

# プログラミングの手引き

- ここではAndroid上で動作するHVC-C用のクラスライブラリの使い方について説明します。

# 主なクラスの概要

クラス名	説明
HVC	HVCのsuperクラス
HVC_BLE	HVC-Cに対応するクラス ・ HVC-Cの操作はこのクラスで行います。
HVC_PRM	HVCの設定パラメータ ・ HVC.setParam()の引数として渡されます。
HVC_RES	検出結果を格納するクラス
HVC_BleCallback	処理完了後に呼び出されるコールバックをまとめたクラス ・ HVCクラスのメソッドは非同期で実行され、 処理完了後にコールバック関数が呼び出されます。

- 詳細なクラス図はHVC-C\_Androidクラス図をご参照下さい。

# HVCクラスの使い方①

## <メイン処理フロー抜粋>

```
hvcBle = new HVC_BLE();  
hvcPrm = new HVC_PRM();  
hvcRes = new HVC_RES();
```



クラスの作成

```
BluetoothDevice device = SelectHVCDevice("OMRON_HVC.* | omron_hvc.*");
```

BLEデバイス取得

```
hvcBle.setCallBack(hvcCallback);
```

コールバックを登録

```
hvcBle.connect(getApplicationContext(), device);
```

HVCとBLE接続

```
hvcPrm.face.MinSize = 60;
```

```
hvcBle.setParam(hvcPrm);
```



パラメータ設定

```
hvcBle.execute(HVC.HVC_ACTIV_FACE_DETECTION |
```

検出実行

```
        HVC.HVC_ACTIV_FACE_DIRECTION, hvcRes);
```

```
hvcBle.disconnect();
```

BLE接続を切断

※本処理はサンプルプログラムのMainActivity内のrunメソッドを参考にしてください。

※検出結果の取得は次ページ参照

# HVCクラスの使い方②

## <検出結果の取得>

```
@Override
public void onPostExecute(int nRet, byte outStatus) {
    for (FaceResult faceResult : hvcRes.face) {
        int size = faceResult.size;
        int posX = faceResult.posX;
        int posY = faceResult.posY;
        int conf = faceResult.confidence;
    }
}
```

検出結果はonPostExecute()にてアクセスして下さい。

※本処理はサンプルプログラムのMainActivity内のonPostExecute()メソッドを参考にしてください。