アプリで未来をつくるスマートフォン・ワールド

Interface増刊

Smartphone World

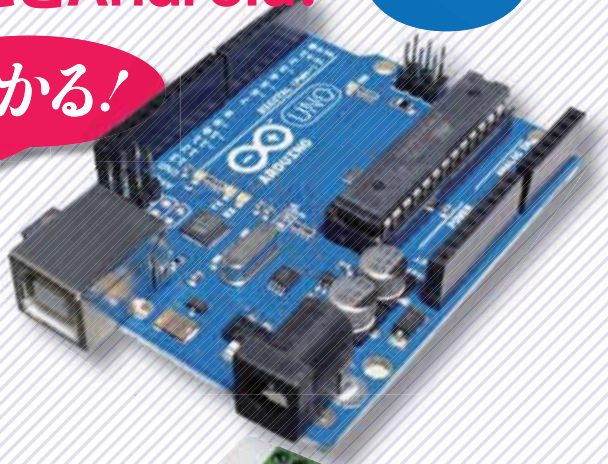
Volume. 3

http://smart.cqpub.co.jp/

表紙の基板を読者プレゼント！

twitter account / @sw_cq

1冊まるごとAndroid！



わかる！

Android × Arduino 即効プログラミング

ADKでハードウェア連携アプリを作る
Twitterでサーボ・モータを動かす

第2特集

Androidとマイコン・ボードをつなぐテクニック

どんな端末ともつながる！MicroBridgeの底力
Bluetooth/Wi-FiでCortex-M3ボードとつなぐ

スマホで作る**脈波測定器**

脱ビギナーの**ゲーム開発**

ARアプリに3DCGを取り込む

知って得するAndroidのツボ

見本

CQ出版社

時代に遅れるな！

Androidとクラウドとハードウェアの未来

編集部　協力 アールティ

ADKの衝撃

2011年5月11日，アメリカ・サンフランシスコで開催されたGoogle社の開発者向けカンファレンス「Google I/O」で，Androidの世界を広げる新しい概念が発表されました．その名も「Android Open Accessory Development Kit」．通称「ADK」です．

これを利用すると，Android端末とほかのハードウェアをつなげて，新しいアイデアを容易に表現できるといいます．カンファレンスでは，Androidタブレットと迷路の模型をつなぎ，それをクラウド経由で操作するというイメージ・デモも披露されました（写真1）．

さらにインパクトのあるアイデアとして，迷路を人間が乗れる大きさに巨大化し，それをタブレット端末で操作するというビデオ（写真2）も公開されました．

ADKの概念は，Androidとハードウェアを接続し，さらにクラウドとつなげて新しいサービスを作る（図1）というものです．現時点では，Androidとハードウェアの接続はUSBでの通信に限られていますが，将来はBluetoothやWi-Fi（無線LAN）などでも通信できるようです．また，Google社としては，ハードウェアをクラウドに直接接続できる環境を開発することも考えている模様です．

ADKを利用してアプリとハードウェアを開発すれば，IoT（Internet of Thing，モノのインターネット化）や，M2M（Machine to Machine）を簡単に実現できるとして，開発者や企業が注目しています．アプリの世界で活躍していた多くの企業も注目しており，従来の「組み込み機器」専業のメーカの発想を飛び越えた新しいサービスを生み出すかもしれません．

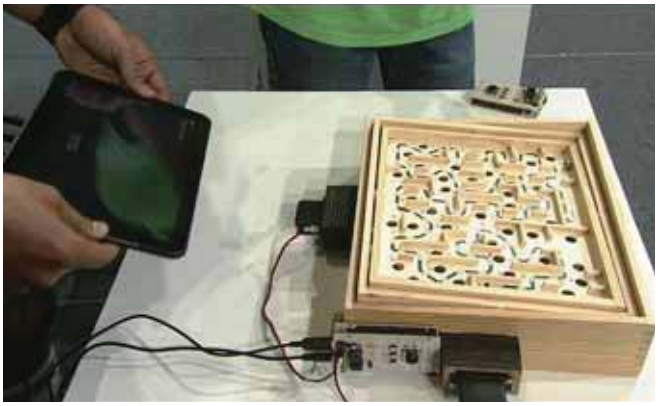


写真1　Google I/O 2011でADKを使ったデモが披露された
YouTube「Android: Momentum, Mobile and More at Google I/O」Google I/O 2011: Keynote Day One」より．
http://www.google.com/events/io/2011/sessions/android-momentum-mobile-and-more-at-google-i-o.html

写真2　Google I/O 2011では巨大迷路をタブレットで操作するビデオも紹介された
YouTube「Android: Momentum, Mobile and More at Google I/O」Google I/O 2011: Keynote Day One」より．
http://www.google.com/events/io/2011/sessions/android-momentum-mobile-and-more-at-google-i-o.html

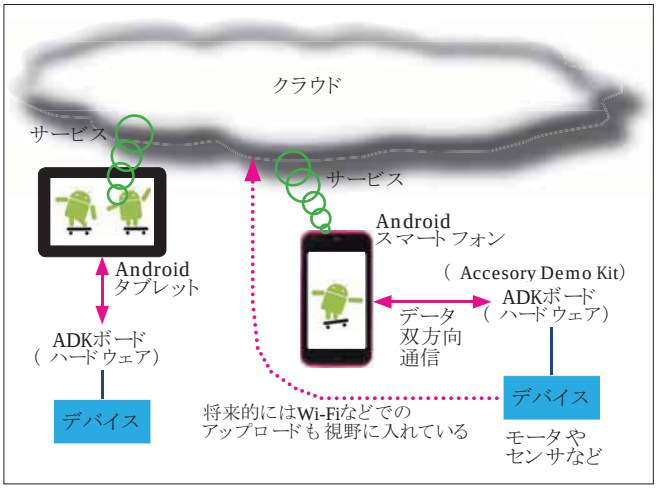


図1　クラウドとデバイスがつながる

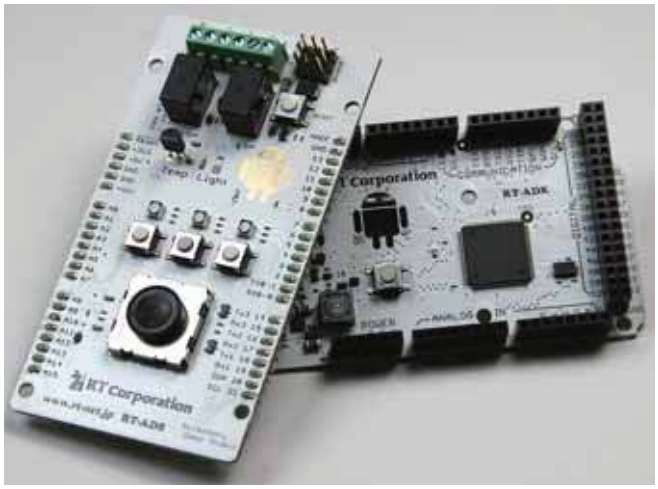


写真3　アールティの「RT-ADK&RT-ADS」

ADKの開発には日本企業が！

このADKの開発には，ロボット関連の日本企業であるアールティが携わっています．

同社は，ADK対応ボード（Accesory Demokit）をGoogleと同時発売し，前述のGoogle I/O 2011で発表しました．ArduinoベースのADK評価基板で，「RT-ADK&RT-ADS」という名前で販売しています（写真3）．

同社によるADKを利用した応用例としては，写真4の「HUG」（RIC90 Android ADKバージョン）があります．Samsung社とコラボレーションしたロボットで，2011年12月に六本木ヒルズの「Galaxyカフェ」で公開されました．頭部は「Galaxy Tab」タブレットを，胸部は「Galaxy SⅡ」を搭載し，これらを利用をしてロボットの制御をしています．動作を指示すると，来場者とハグをします．その際に頭部のタブレット搭載カメラで来場者の笑顔を撮影し，クラウド経由でイベント用のWebサイトにアップロードするという仕組みを構成しています．

なお，同社がADKの開発に関わったのは，2009年から同社がAndroidを搭載したロボット（写真5）を公開し，それがGoogle社の目に留まったのがきっかけということです．このロボットは世界各地で展示されました．

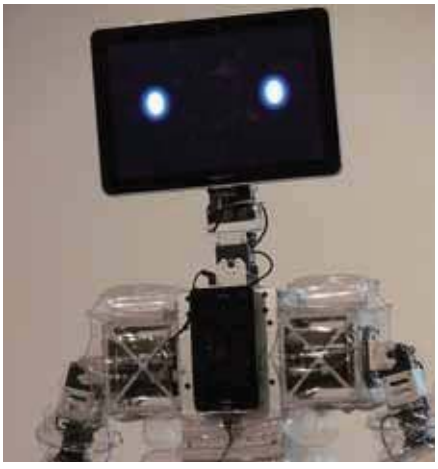
ADKを技術から考える

それでは，具体的にはADKとはどんなものでしょうか？

ADKとは，大まかに言えばAndroid開発環境のライブラリ（API）や，ADK対応ハードウェア（Accesory DemoKitと呼ばれるもの）で構成され，それらがUSBで通信できるもの，と言えるでしょう．

これだけ聞くと，Android端末以外のハードウェアと通信できるだけで目新しさも何もない，と思う方もいるかもしれません．しかし，ADKが注目されるのは，その技術的な発想にもよるのです．

Android端末とハードウェアはUSB



（a）タブレットのカメラで来場者を撮影

（b）アールティの「RIC 90 Android」ADKバージョンがベース

（c）ハグ動作はADKを利用してAndroidで制御している

（d）頭部の「Galaxy Tab」

（e）胸部の「Galaxy SⅡ」

写真4　Galaxyシリーズとコラボしたロボット“Hug”

写真5　アールティの「RIC android」（Google Developers Day 2011 Tokyoにて）

ADKボードのアプリを作ってみよう

飯塚 康至

写真1 ボードをつないだときに起動する

ボードをつないだときにアプリが起動し，ADKボードからの入力や出力を確認できる簡単なアプリを作ります．さらに，ジョイスティックで画像イメージを動かしたり，リレーを使ったタイマのアプリを作成したりします．

本章では，ADKの使い方のポイントを理解していきます．まず，ボードをつないだときにアプリが起動するようにします．さらに，ボードからの入力値を確認するアプリや，簡単な出力のアプリを作ります．さらに，ジョイスティックで画像を動かしたり，タイマのアプリを作成したりします．

ADKをつなぐとアプリが起動するようにする

ADKのボードをつなぐとアプリが起動するようにします．**表1**のようにAndroidプロジェクトを作成しましょう．

表1 Androidプロジェクトを作成

項 目	値
プロジェクト名	ADKSample02
ターゲット名	GoogleAPIs 2.3.3
アプリケーション名	ADKSample01
パッケージ名	任意
アクティビティ名	ADKSampleActivity

intent-filterを設定すると，ボードをつないだときにアプリが起動します（**図1**，**写真1**）．そのために，**リスト1**を参考にしてAndroidManifest.xmlファイルを編集します．

ADKはすべての端末でサポートされているわけではないので，**リスト1**（**a**）を追加します．次に，拡張ライブラリの利用を**リスト1**（**b**）のように宣言します．そして，intent-filterを**リスト1**（**c**）のようにセットし，ADKが接続されたときのintentを受けられるようにします．また，このintent-filterで利用するmeta-dataを**リスト1**（**d**）のようにセットします．

android:resourceはXMLファイルとしてres/xml内に，accessory_filter.xmlを**リスト2**のように指定します．

以上で，ボードを接続したときにアプリケーションの起動選択画面が表示されるようになります．最終的なAndroidManifest.xmlは**リスト3**のとおりです．

図1 ボードをつなぐとアプリが起動する

入力値を確認する

ADKボードからの入力値を確認するアプリを作ってみましょう．

まず，画面を作成します．各センサ

リスト1 AndroidManifest.xmlの編集内容

```
<uses-feature android:name="android.hardware.future.usb.accessory" />
```

（a）usb.accessory対応の追加

```
<uses-library android:name="com.android.future.usb.accessory" />
```

（b）拡張ライブラリの利用の宣言

```
<action android:name="android.hardware.usb.action.USB_ACCESSORY_
ATTACHED" />
```

（c）intent-filterのセット

```
<meta-data android:name="android.hardware.usb.action.
                                        USB_ACCESSORY_ATTACHED"
      android:resource="@xml/accessory_filter" />
```

（d）intent-filterで利用するmeta-dataをセット

リスト2 accessory_filter.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <usb-accessory manufacturer="Google,
Inc." model="DemoKit" version="1.0" />
</resources>
```

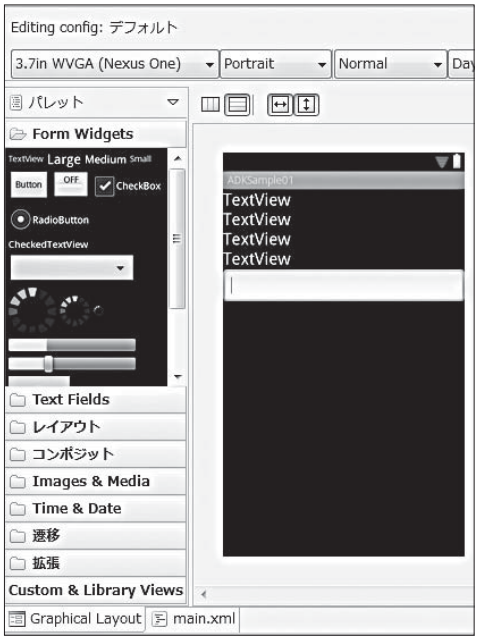


図2 4個のTextViewを配置

リスト3 最終的なAndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/
                                                    android"
      package="org.soho_style.ADKSample01"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-sdk android:minSdkVersion="10" />

    <application android:icon="@drawable/icon"
android:label="@string/app_name">
    <uses-library android:name="com.android.future.usb.
                                              accessory" />
        <activity android:name=".ADKSampleActivity"
                  android:label="@string/app_name">
            <intent-filter>
    <action  android:name="android.hardware.usb.action.
                                  USB_ACCESSORY_ATTACHED" />
            </intent-filter>
     <meta-data android:name="android.hardware.usb.action.
                                    USB_ACCESSORY_ATTACHED"
                android:resource="@xml/accessory_filter" />
        </activity>
    </application>
  <uses-feature android:name="android.hardware.future.usb.
                                              accessory" />
</manifest>
```

リスト4 ADKSampleActivityのフィールドに追加する内容

```
private static final String ACTION_USB_PERMISSION =
"com.google.android.DemoKit.action.USB_PERMISSION";
  private static final int MESSAGE_SWITCH = 1;
  private static final int MESSAGE_TEMPERATURE = 2;
  private static final int MESSAGE_LIGHT = 3;
  private static final int MESSAGE_JOY = 4;
  UsbManager mUsbManager;
  ParcelFileDescriptor mFileDescriptor;
  FileInputStream mInputStream;
  FileOutputStream mOutputStream;
  UsbAccessory mAccessory;
  TextView tv1, tv2, tv3, tv4;
```

の値を表示できるように，4個のTextViewを配置します(**図2**)．TextViewの下に配置したEditTextは，ログを取得するLogCatを確認するためにテスト的に配置したものです(p.30参照)．

続いて，Activityの記述を行います．ADKSampleActivityのフィールドに**リスト4**を追加します．

ACTION_USB_PERMISSIONはIntent Filterで利用するためのaction名です．次に，どのセンサが反応したかを分かりやすくするために定数を定義します．UsbManegerはUSBホストを管理するオブジェクトです．ParcelFileDescriptorは接続されたアクセサリからの入出力を担当するFileDescriptorです．FileInputStreamとFileOutputStreamは入出力のためのストリームになります．UsbAccessoryは接続されたアクセサリを表すオブジェクトです．

そして，onCreateメソッドで**リスト5**のようにUsbManagerの取得とBroad

リスト5 UsbManagerの取得とBroadcastReceiverの登録

```
public void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.main);
mUsbManager = UsbManager.getInstance(this);
IntentFilter filter = new IntentFilter
                          (ACTION_USB_PERMISSION);
filter.addAction(UsbManager
                .ACTION_USB_ACCESSORY_DETACHED);
 registerReceiver(mUsbReceiver, filter);
 tv1 = (TextView) findViewById(R.id.textView1);
 tv2 = (TextView) findViewById(R.id.textView2);
 tv3 = (TextView) findViewById(R.id.textView3);
 tv4 = (TextView) findViewById(R.id.textView4);
 }
```

リスト6 BroadcastReceiver

```
private final BroadcastReceiver mUsbReceiver =
                        new BroadcastReceiver() {
  @Override
  public void onReceive(Context context,
                                Intent intent) {
    String action = intent.getAction();
    if (ACTION_USB_PERMISSION.equals(action)) {
      synchronized (this) {
        UsbAccessory accessory = UsbManager.
                            getAccessory(intent);
          openAccessory(accessory);
        }
      }
    }
  };
```

Android×Arduinoでプロトタイピング

佐々木 陽

Androidとハードウェアをつなぐには，ArduinoでUSBやBluetoothのシールドを利用する方法があります．一番簡単にアプリとハードウェアの接続を可能にする方法として，Arudino MEGA ADKを使ってADKの仕組みを解説します．

ここではIoT（Internet of Things）のプロトタイピングの第一歩として，AndroidとArduinoの連携という形で実現してみます．

AndroidとArduinoボードを連携させる基礎を紹介します．具体的な開発の例として，Arduinoにつなげたスイッチを押すと画面が光ったり，アプリのボタンにタッチすると基板に接続したLEDが光ったりするような簡単なサンプルを解説します．

表1　必要な機材

ハードウェア
Android 2.3.3以降のスマートフォン もしくはAndroid 3.1以降のタブレット
Arduino MEGA ADK
ソフトウェア
Arduino 1.0
Eclipse + ADT + Android SDK
Google APIs by Google Inc. API Version 10 Revision 2以降のライブラリ

必要なハードウェアとソフトウェア

必要な端末とハードウェアとソフトウェアを**表1**に示します．

まず，Android端末が必要です．執筆時点の2011年11月現在，ADKが動作する機種には，Nexus S，Nexus One，Samsung Galaxy Tab 10.1，Motorola Xoomなどがあります．つまり，Android 2.3.3以降を搭載したスマートフォンまたはAndroid 3.1以降を搭載したタブレットです．

しかし，新規で発売されるAndroid 2.3.3以降の端末でも，ADKに非対応という機種がまだ多く存在しています．これは，ADKへの対応が必須ということではなく，各メーカがオプションで対応しているためです．

ハードウェアとしては，Arduinoが必要です．使用するArduino MEGA ADK（**写真1**）[※1]は，Arduino MEGA 2560を拡張し，USBホストを搭載したADK対応の基板です．Arduino MEGA ADKは，スイッチサイエンスや秋葉原の千石電商などで，7,000円～8,000円で購入できます．

Arduinoのソフトウェアは「Arduino 1.0[※2]」で開発します．

Androidの開発には，標準的なアプリを開発できる環境を用意します．本誌「Smartphone World」Volume.1などを参考にしながら，統合開発環境Eclipse[※3]や，Android開発に必要なADT[※4]，Android SDK[※5]をインストールし，開発環境を構築してください．

このほかに，ライブラリ「Google APIs by Google Inc. API 10 Rev. 2」も必要です．Android SDK Managerを起動し，インストールします（**図1**）．本誌p.14～に詳しい手順を解説して

※1　Arduino MEGA ADK　http://www.arduino.cc/en/Main/ArduinoBoardADK
※2　Arduino 1.0　http://arduino.cc/en/Main/Software
　　Arduino MEGA ADKは，Arduino 1.0で正式に対応端末として追加された．
※3　Eclipse　http://www.eclipse.org/downloads/
※4　Android Development Tools　http://developer.android.com/sdk/eclipse-adt.html
※5　Android SDK　http://developer.android.com/sdk/index.html

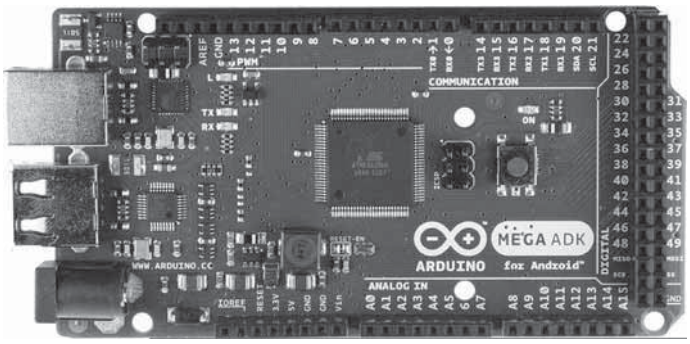


写真1　Arduino MEGA ADK

図1　API 10 Rev.2をインストール

表2　各フォルダのインストール先

Macの場合	Contents>Resources>Java>libraries>AndroidAccessory
	Contents>Resources>Java>libraries>USB_Host_Shield
	Contents>Resources>Java>libraries>CapSense
Windowsの場合	<arduinoをインストールしたフォルダ>¥libraries¥AndroidAccessory
	<arduinoをインストールしたフォルダ>¥libraries¥USB_Host_Shield
	<arduinoをインストールしたフォルダ>¥libraries¥CapSense

リスト1　AndroidAccessory.h

```
70  int endTransmission(void);
71  size_t write(uint8_t b);
72  int write(void *buff, int len);
```

リスト2　AndroidAccessory.c

```
280  size_t AndroidAccessory::write(uint8_t b)
281  {
282     outBuff[outBuffIndex++] = b;
283  }
```

リスト3　Max_LCD.cpp

```
257  inline size_t Max_LCD::write(uint8_t value) {
258     LCD_sendchar(value);
259  }
```

リスト4　Max_LCD.h

```
100  void setCursor(uint8_t, uint8_t);
101  size_t write(uint8_t);
102  void command(uint8_t);
```

リスト5　Arduino側サンプル・ソース

```
1  #include <Max3421e.h>
2  #include <Usb.h>
3  #include <AndroidAccessory.h>
4
5  AndroidAccessory acc("Google, Inc.", // 組織名
6              "DemoKit", // アプリ名
7              "DemoKit Arduino Board", // アプリ説明
8              "1.0", // バージョン
9              "http:///www.google.com", // URL
10            "0000000012345678"); // シリアル
11 void setup();
12 void loop();
13
14 void setup()
15 {
16   acc.powerOn();
17 }
18
19 void loop()
20 {
21  if(acc.isConnected()){
22
23    // データの送受信処理
24    }
25
26    delay(10);
27 }
```

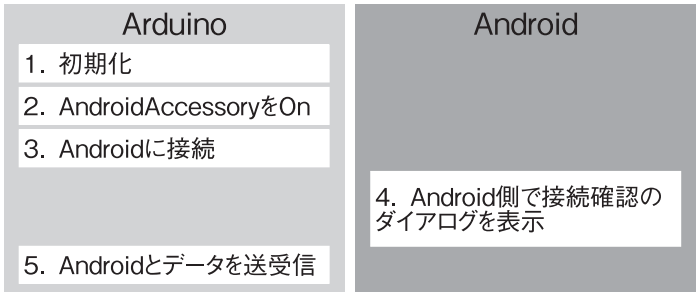


図2　AndroidとArduinoを接続する手順

います．

Arduinoの開発環境の構築

今回は，リリースされたばかりのArduino 1.0を使用しますが，Arduino 1.0でADKの開発をしようとすると，ADKのライブラリにエラーが4カ所発生します．それを修正します．

① AndroidAccessoryとUSB_Host_Shieldのコピー

Arduinoのライブラリは，arduinoのlabsのバージョンを使用します．http://labs.arduino.cc/uploads/ADK/GettingStarted/arduino_bundle_ADK.zipからライブラリをダウンロードします．解凍すると，AndroidAccessoryフォルダとUSB_Host_Shieldフォルダが生成されるので，**表2**の各フォルダにコピーします．

② CanSenseのコピー

次に，http://www.arduino.cc/playground/Main/CapSenseからCapSense04.zipをダウンロードします．解凍するとCapSenseフォルダが生成されるので，こちらも**表2**のフォルダにコピーします．

③ ソースコードの修正

Arduino 1.0からは，void write()がsize_t write()に変更になっているので，返り値の型をsize_tに書き換えます．

具体的には，libraries¥AndroidAccessoryフォルダのAndroidAccessory.hとAndroidAccessory.c，libraries¥USB_Host_Shieldフォルダの Max_LCD.hとMax_LCD.cppを**リスト1**～**リスト4**のように書き換えます．

ADKのライフサイクル

AndroidとArduinoの接続は，**図2**の手順で行います．acc.isConnected()が呼ばれるとAndroidとの接続が開始されます．最初の接続時には，Android端末上に接続の開始を確認するダイアログ・ボックスが表示されます．確認メッセージの画面で，OKを押すとArduinoとAndroid の接続が開始されます．

Arduino側のサンプル・ソースコードは**リスト5**です．ソースコードの作成が終わったら，ArduinoとPCとAndroidを

第2特集

ADKを使わないでボードと接続する（その1）

AndroidとArduinoが簡単につながるMicroBridge

Android端末とArduinoボードを接続する，つまりアプリとハードをつなぐ方法はADK（Android Developers Kit）だけではありません．MicroBridgeというライブラリを使えば，ほとんどのAndroid端末と，すべてのArduinoとをつなげられるのです．

いわたん

Arduinoとシールド

ArduinoとはAVRマイコン，マイコンにつながる入出力ポートや電源，USBシリアル・インターフェースなどを搭載した基板（**写真1**）と，マイコンの開発環境としてC/C++ライクなプログラミング言語，統合開発環境をまとめたシステムの名称です．Arduinoはもともと教育向けに安価なマイコン環境を構築することを目的に開発され，回路図，ブートローダ，ファームウェア，開発環境などがすべて無償で公開されています．そのため，世の中には本家のArduinoを模した互換ボードが多数あります．

Arduinoの特徴として，「シールド」と呼ばれる拡張ボードがあります．これはArduino本体の入出力ポートをコネクタとして別回路を“ガッチャンコ”と載せることで，さまざまな機能を簡単に使用できるようにしてくれます．今回紹介するMicroBridgeも，USBホスト・シールドと呼ばれるシールドを使用することで簡単にArduinoからUSBホスト機能を使用できます．

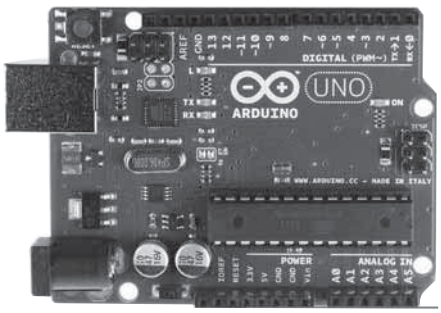


写真1　Arduino基板「Arduino UNO」

Androidとつながる! MicroBridgeとは?

MicroBridgeは，Android Debug Bridge（ADB）をArduino用に実装したライブラリです．ADBを利用するので，Android端末のroot化が不要で，バージョンもAndroid 1.5以降から使用できます．おそらく，2011年11月現在，国内キャリアから販売されている端末のほぼすべてで使用できるでしょう．

MicroBridgeでは，マイコンからAndroid端末上でのコマンドの実行，ファイルの転送，デバッグ・ログ（logcat）の取得，TCP/IPの通信ができます．TCP/IP通信を利用することで，ArduinoとAndroidアプリ間の通信が可能になります．Android端末上で「amコマンド」を駆使することで，アプリがインストールされていなかったらマーケットへ遷移させるなどの複雑な動作もできます．使い方次第で可能性が無限に広がります．

MicroBridge以外のAndroidとマイコンを接続する方法としては，本誌の第1特集で紹介しているUSB Accessoryモードがあります．第1特集で紹介しているため詳細は省きます．

開発環境構築

MicroBridgeの魅力を十分に知ったところで，開発環境を構築していきましょう．

●ハードウェア

MicroBridgeを使用するために必要なハードウェアは以下です．

・パソコン（以下，PC）
・Android端末（バージョン1.5以降）
・Arduino本体
・USBホスト・シールド
・USBホスト・シールドをArduino本体とつなぐ連結用のピン
・外部電源（6～12V）

まずは，Arduinoの環境構築から始めましょう．MicroBridgeは，「Arduino UNO」，「Arduino MEGA」，「Arduino Deicimila」や，そのほかの互換ボードでも動きます．つまり，すでに何かしらのArduino本体があれば，それを使

用できます．ポート数が必要な場合はArduino MEGAがお勧めですが，値段や入手性を考えるとArduino UNOをお勧めします．Arduino UNOであれば，USBホスト・シールドと連結用のピンをあわせて，約6000円ほどで用意できます．今回は，Arduino UNOの使用を前提として環境構築，ソース・コードの解説を行います．

USBホスト・シールド（**写真2**）は，ArduinoにUSBホスト機能を持たせるためのシールドです．スイッチサイエンスや千石電商などで購入できます．連結用のピンはスイッチサイエンスで購入できます．連結用のピンを使用すると，USBホスト・シールドを接続した状態でもさらに入出力ポートからジャンパを引き出したり，さらにシールドを取り付けたりすることが可能になります．

USBホスト・シールドはVIN（Arduinoの電源電圧）に入ってくる5Vより高い電圧を5Vへ変換し，さらに3.3Vへ変換して電源供給を行います．USBから給電すると5VがVINに供給されますが，これでは，VINから5Vへ変換することができずシールドが動作しません．Arduinoの5Vピンをシールド上のレギュレータへ接続するか，Arduinoへの外部電源を5Vを超えたものにする必要があります．今回は単純にするために外部電源を使用していきます．

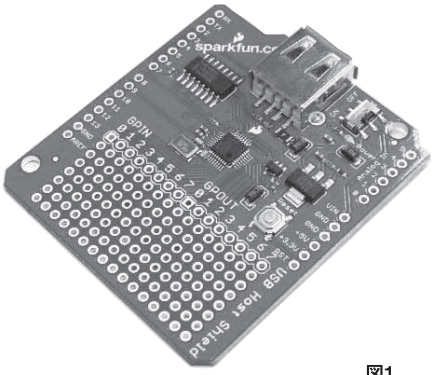


写真2　USBホスト・シールド

図1
Mac OS版Arduino IDEの画面

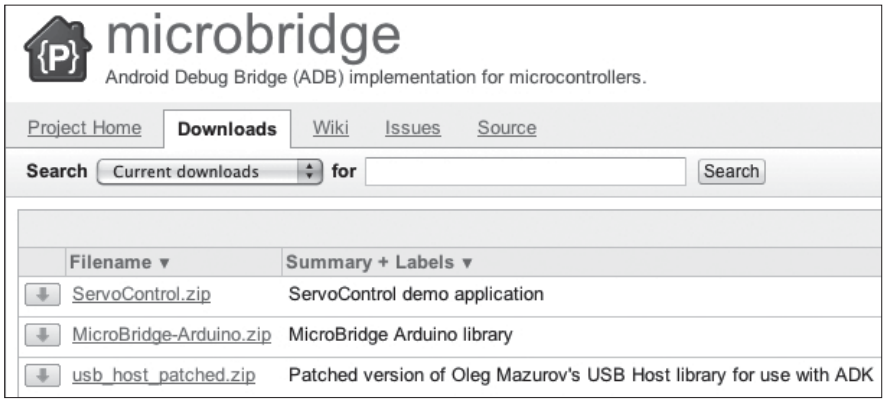


図2　MicroBridge（http://code.google.com/p/microbridge/）でMicroBridge-Arduino.zipをダウンロード

●Arduinoの開発環境をインストール

ソフトウェアは以下を用意します．

・Android SDK
・Arduino IDE（開発環境）
・MicroBridgeライブラリ

Android SDKは，すでにインストールされていることを前提とします．まだ，インストールしていない場合は，本誌「Smartphone World」Volume.1やWeb上の情報を参考にしてください．

Arduino IDEは，http://arduino.cc/から入手し，任意の場所に展開します．WindowsマシンかLinuxマシンであれば，これで開発環境のインストールは終了です．Mac OS版では，アプリケーション・フォルダにコピーします．Arduino 1.0の環境が2011年11月末に公開されましたが，バージョン1.0からMac OS版ではdmg形式からzip形式に変更され，Applicationsへのショートカットが入っていない状態になりました．

インストール後にArduino IDEが起動するかを確認します（**図1**）．なお，図はMac OS版のため，メニューが表示されていません．

●MicroBridgeライブラリを追加

MicroBridgeの開発は，Google Code上のプロジェクト（http://code.google.com/p/microbridge/）でオープンソースで行われています．

MicroBridgeのライブラリを入手します．上記のページで［Download］を選択し，表示されるリストから［MicroBridge-Arduino.zip］をダウンロードします（**図2**）．

ダウンロード完了後に，圧縮ファイルを解凍します．解凍したファイル内にあるAdbフォルダをArduinoの開発環境を展開したフォルダ内のlibrariesフォルダにフォルダごとコピーします．この際にArduino IDEが起動している場合は終了しておいてください．

Mac OS版では，ライブラリはArdui

センシングとスマホは好相性

スマートフォンで脈波測定器を作る

スマートフォンを利用して脈波測定器を作ると，小型で持ち運びに便利な道具ができあがります．USBやBluetoothでの通信ではなく，音声入出力端子を使うことでハードルを下げた例を紹介します．マイク端子を使うと，多くの可能性があることに気付きます．

吉川 浩，巻口 誉宗

本稿では，スマートフォンを用いた脈波測定器の作成と，音声入出力端子を用いて脈波以外の信号をセンシングする応用例を紹介します．

脈波とは何か?

脈波とは，末梢の血管の容積が血流や血圧によって時間的に変化する様子を波形として捉えたものです(**図1**)．

脈波は心臓や血管の影響を受けるため，脈波から様々な情報を得ることができます．たとえば，心拍の変化，血管の硬さ，ストレスなどです．このような脈波から人間の心身の状態を解析する研究は数多く存在します．また近年，コミュニケーションやヒューマン・インターフェースなどに脈波などの生体情報を利用する研究も行われています．

研究を進める上では，安静時だけでなく活動時の脈波も測定できると都合がよいとされています．歩行中や，運動をしているとき，車の運転をしているときなど日常の脈波を取得できれば，人間の生態をより詳しく把握できるはずです．しかし，従来の脈波測定器は持ち運びが不便で，室内で安静状態の脈波しか測定できませんでした．

これらの課題を解決するため，筆者の研究室ではスマートフォンを利用したハンディな脈波測定器(**写真1**，**図2**)を開発し，日本バーチャルリアリティ学会大会でデモンストレーションを行いました．

この装置の特徴は，スマートフォン

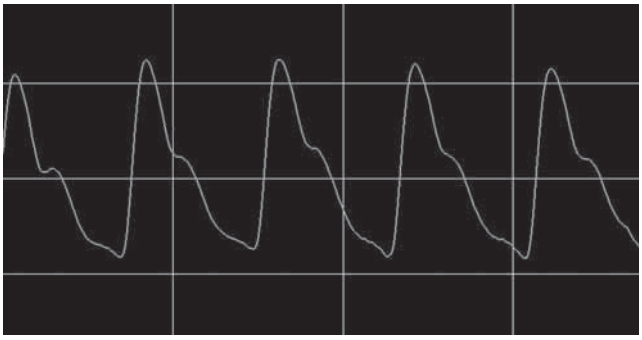


図1　脈波のグラフ

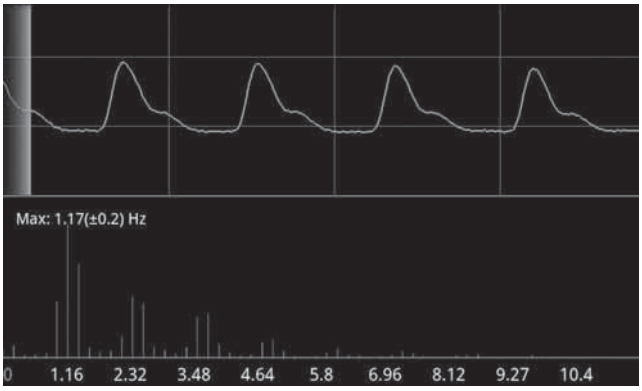


図2　脈波測定器のアプリ画面

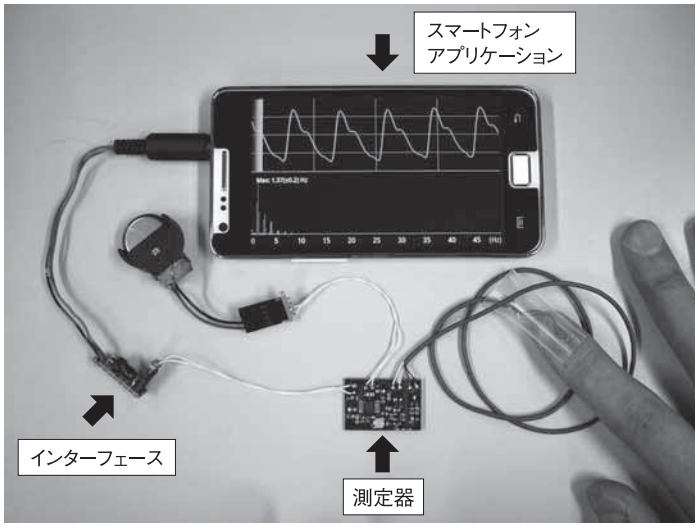


写真1　開発した脈波測定器

NDKとCGソフトウェアで本格的なARを実現

CGキャラクタを写し込むARカメラを作る

AndroidのARアプリを作ります．3DCGソフトウェア「Metasequoia」を利用して3Dモデルを作り，OpenGLを利用してNDKで作成したARカメラに写し込みます．今回，筆者が特別に用意したライブラリを使って，プログラミングを理解していきましょう．

山本 隆一郎

本誌Volume.2 p.50 ～では，OpenGL ESをNDKで高速に描画する方法を紹介しました（**図1**）．今回は，3DCG（3次元のコンピュータ・グラフィックス）ソフトウェアで作成した複雑なCGを表示するアプリを作成します．さらに，CGと一緒に写真が撮れる「AR（Augument Reality）カメラ」を作成します．

さまざまな3Dフォーマット

前回作成した3Dのキューブを構成する各頂点は，アプリ内に直接コーディングされています．アプリを実行すると，頂点の変更は一切できません．複雑な形状を表現するために手動でプログラミングするのは大変です．

3DCGのプログラミングでは一般的に，複雑な形状の表現は，デザイン専用のソフトウェアで形状をデザインし，その出力結果をアプリケーションに読み込ませて表示します．専用のソフトウェアは，操作性もよく，作成した形状の変更も容易です．表示側にとっても，複雑な頂点を毎回プログラミングしたり，ビルドする手間が省けます．

しかし，一つ問題があります．それはCGのデータ形式に標準や統一規格がなく，いくつかの形式が存在していることです．例えば，古くからある機械設計CADのDXFは有名です．Web黎明期のころから存在するVRMLや，3DCGソフトウェアとして高機能な「Solid Works」のsldprtなど，多くの形式が混在しています．これら全てのデータに対応するのは現実的ではありません．実際に3DCGを利用する場合は，自分の使いやすい形式を選べばよいでしょう．データの互換性が欲しい場合は，3DCGソフトウェアで対応できます．

メタセコイアを使う

今回は，3DCGソフトウェアとして「Metasequoia（メタセコイア）」（**図2**）を利用します．非常に使い勝手が良く，人気があります．CGの初心者だけでなく上級者もこれでモデリング（形状のデザイン）を行い，そのデータをほかのソフトウェアで加工したり，レンダリングしたりすることも多いようです．機能制限付きの無償版もあり，メタセコイア形式のデータなら自由に読み

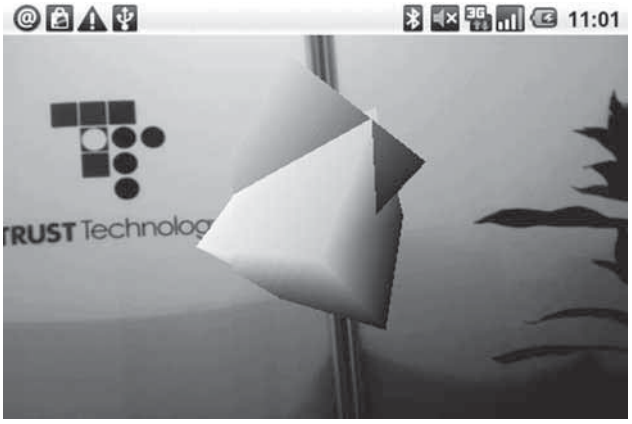


図1 本誌 Volume.2で作成したNDK版のARViewアプリ

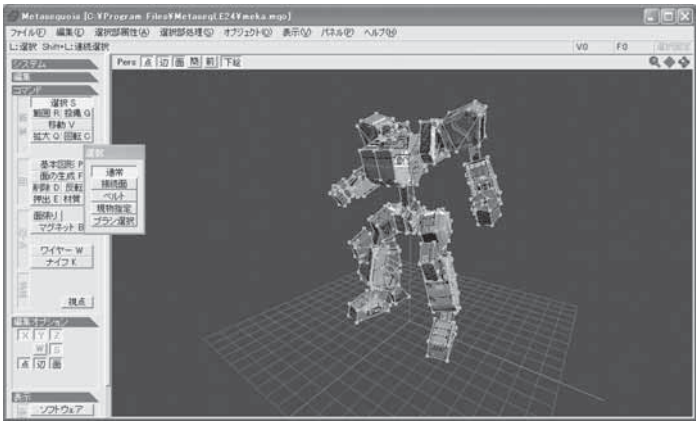


図2 メタセコイアの実行画面（付属のサンプル・データを表示）

お求めは全国書店または『CQ出版WebShop』をご利用ください.
広告ページは表示していません

Interface増刊 Smartphone World スマートフォン・ワールド Volume.3
CQ出版社 〒170-8461 東京都豊島区巣鴨1-14-2 TEL.03-5395-2141（販売部）
定価1,300円 本体1,238円