

3G通信技術を活用した
IoT教材キット V4.0

(オープンソースハードウェアArduinoによる試作開発期間短縮技術)

テキスト R4.1



Arduino UNO R3



Genuino101



Arduino M0 Pro



IoTABシールドV4.0

本ドキュメントについて

- ▶ 本ドキュメントは、株式会社タブレインが独自に作成してきたIoTデバイスの試作・開発教育向け教材です。
- ▶ 本マニュアルの無断複写はお断り致します。
- ▶ 学校法人内でご利用の場合で、かつタブレイン直販による事前に購入者が分かっている場合には、問題ありませんが、代理店を通じてご購入され複写ご利用される時は、事前に info@tabrain.jp へお問い合わせお願いいたします。

【もくじ】

- 第Ⅰ編 IoT基本技術
- 第Ⅱ編 3GIM基本技術
- 第Ⅲ編 IoT試作演習
- 第Ⅳ編 IoT展開事例
- 第Ⅴ編 補足資料



IoTABシールドV4.0



3GIM V2.2



Genuino101

IoT教材キット V4.0 マニュアル R4.1 (No.2)

作成・著作権 株式会社タブレイン

はじめに

- ▶ 本マニュアルは、Arduino UNO R3 + 3 GIM + IoTABシールドを組み合わせたIoT教材キットの使い方を紹介したものです。
- ▶ 本マニュアルでは、分かり易さを重視し、さらに短時間で高機能が学べるものと考え、サンプルプログラムを豊富に提示しています。
- ▶ 多くのサンプルプログラムを稼働し、その動きや流れを理解することで、新たな変更・追加へと進み、創造的なシステムへと成長させていくことができます。
- ▶ IoTシステムは新しい概念ですが、基本は遠隔監視（モニタリング）と遠隔制御の2つの考えによって成り立っています。
- ▶ 本マニュアルでは、入力電子部品であるセンサ値を取得する遠隔監視と、出力電子部品であるLCDやLED、スピーカなどの遠隔制御を短時間で学ぶようになっています。
- ▶ 本マニュアルには、最初に3 GIMのコマンド群およびライブラリ群を紹介しています。これらはリファレンスとして読み飛ばし、後で分からないときに利用することをお勧めします。



Arduino UNO + IoTABシールド

第 I 編 IoT基本技術

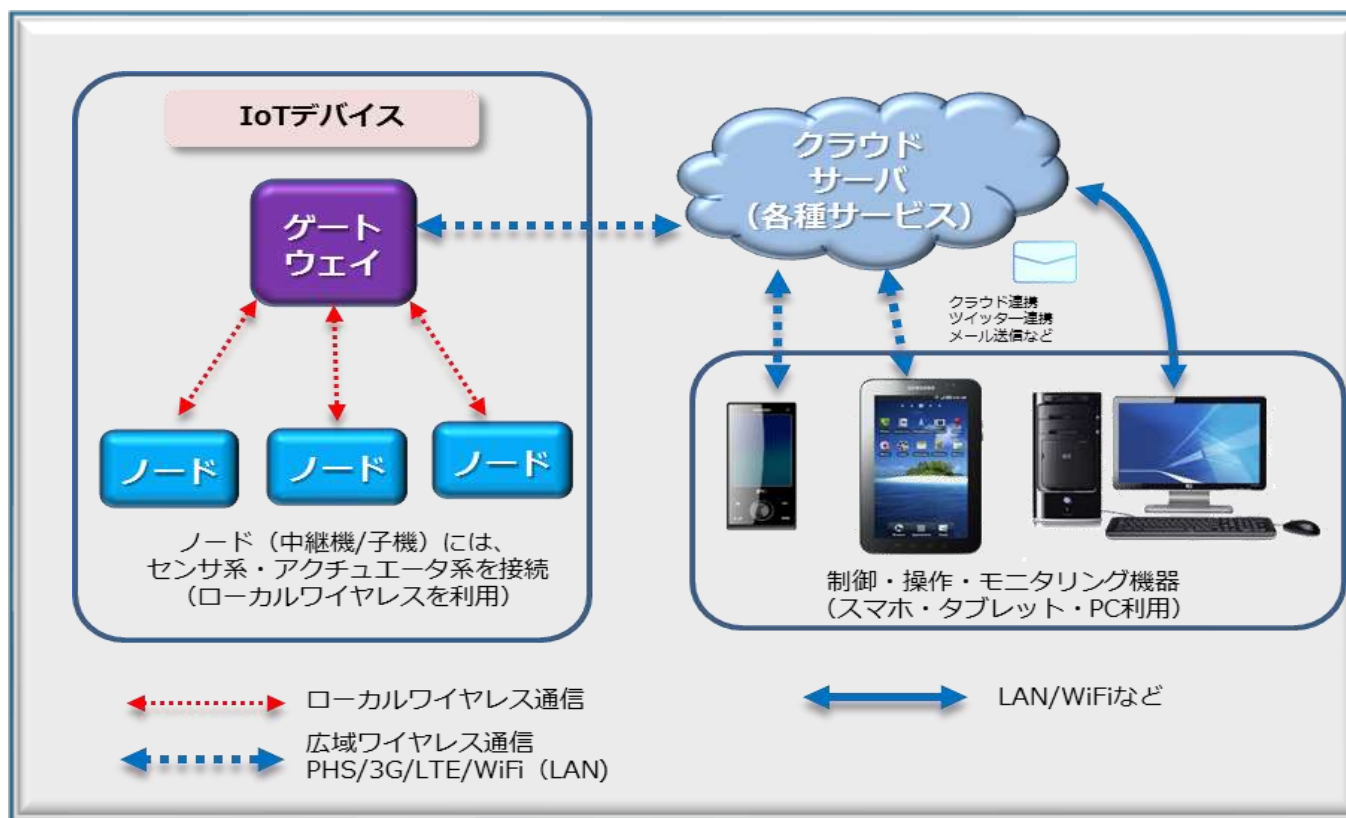
第1章 はじめに

1. IoTセンサ・ネットワークについて

ここでは、3 GIMを使ったIoTセンサ・ネットワークの構成についてご紹介します。

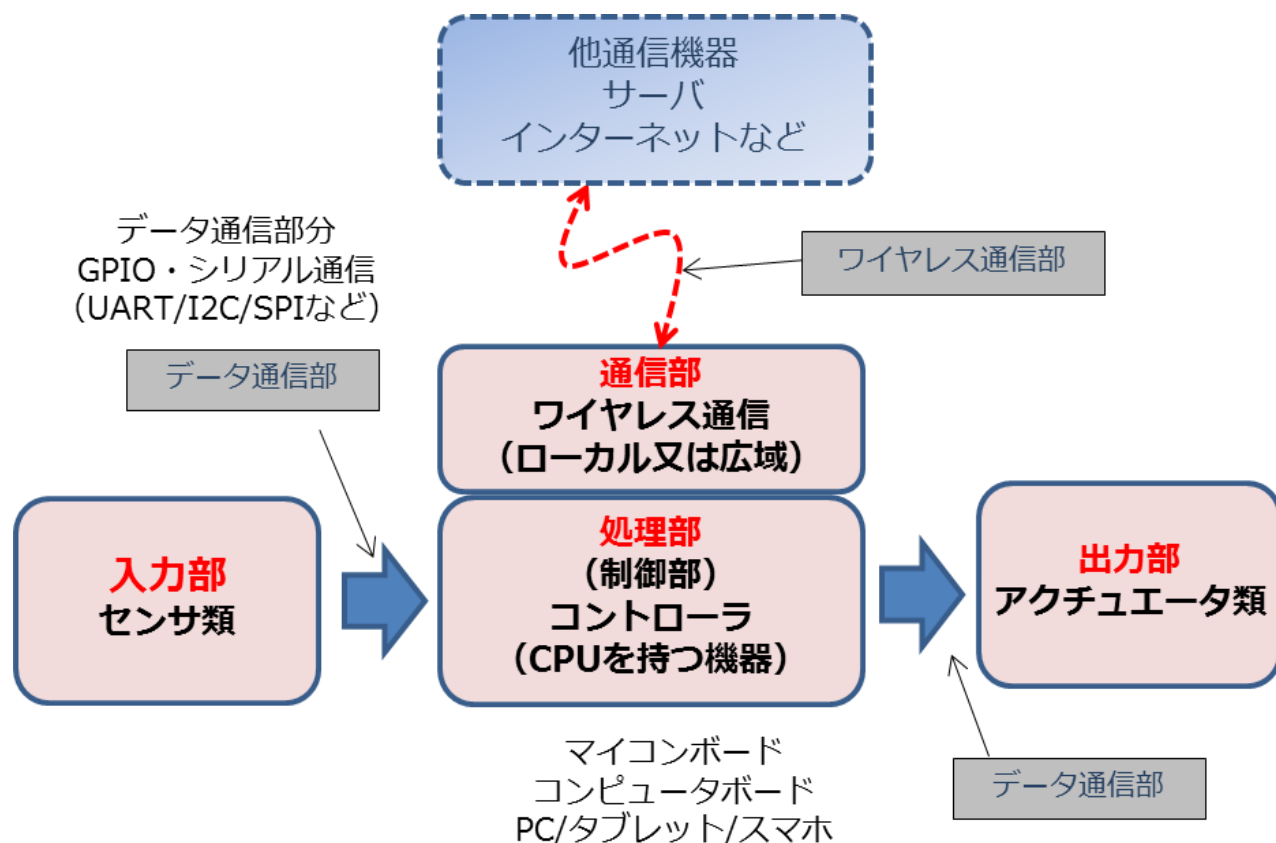
IoTセンサ・ネットワークとは、IoTデバイス（またはIoTノード）がセンサ・デバイスとして、さまざまなセンサ値を取得し、それをIoTゲートウェイを通じてインターネット上のクラウドまたはサーバに送信・蓄積することとなります。

- ① 3 GIMがインターネット接続できることからIoTデバイス（ゲートウェイ）として構築
- ② 3 GIMとローカル通信ができるようにし、親子関係でのネットワークを構築



2. IoTデバイスについて

ここで述べているIoTデバイスは、主として① 制御部となるコンピュータを持つ機能と、②センサなどの入力機能、③LCD・LED・スピーカなどの出力機能、④ インターネット接続するワイヤレス通信機能 を持つものとしします。
①から③までは、本マニュアルのNo.1で説明した内容となります。



3. 必要となるIoT技術

IoTシステムを構築する上では、以下のような技術が必要となります。必ずしも全てが必要というわけではありませんが、ここでのIoT教材キットでは、多くのIoT技術を学ぶことができるようになっています。

1) センサ技術

センサは、技術進化が早く、新しい製品が多く誕生するようなところですが、ただ一方では成熟したセンサ群もあり、小型で安価で省エネのものも増えてきています。また一方では、センサの精度は価格差によっても大きく異なるようになっています。

2) マイコン制御技術

IoTデバイスを制御する上では、マイコンが必須となります。最近では、Arduinoの普及によって、互換機も登場し、安価で、高機能なものが出てきて、しかも開発環境も高機能なものが無償で提供されるようになっています。

3) ワイヤレス通信技術

ワイヤレス通信技術は、従来ですとATコマンドによる制御が必要で、高度な技術者でさえ難しいとされてきた分野です。弊社が提供する3GIMは、中学生でも分かるようにライブラリ群を紹介し、通信プロトコルの理解も深く追求することなく利用できるようにしました。

4) インターネット接続技術

インターネット接続においては、httpGETやhttpPOST、それにTCP/IPなどがあり、そのプロトコルは、クライアントとサーバとの関係で、難しいところが多くあります。特にセキュリティ対応においては、さまざまな対応も必要となり、その技術習得はさらにハードルが高くなります。

5) クラウド（サーバ）構築技術

センサデバイスから取得データは、インターネットを通じてクラウド（サーバ）にアップされます。その時、単に蓄積するだけではなく、さまざまなデータ蓄積方法があり、またその解析や分析などを行い、さらにデバイスの制御も行う場合もあります。その他にセキュリティ対策も必要となり、高度で多岐に渡る技術が必要となっています。

6) データ分析技術

センサ・ネットワーク技術は、蓄積したビックデータを有効活用することによって、価値を生むものです。解析や分析を行い、そこでの知的な仕組みをもって対応することで、より付加価値のあるものへとつながっていきます。この中にはデータ解析技術だけではなく、より高度な処理となる人工知能的な処理も必要となっていきます。（初心者には「R-Studio」などを推奨します）

7) その他

IoTセンサ・ネットワークは、まだまだ実績が少ない分野となります。現場での新たな課題も多く出てくるもので、実践での知識蓄積が必要になっていくものと考えます。豊富な見地と知識と判断によって、新たな課題を解決していくことが大事になっていきます。そのためにも既存の知識・知恵も大事にしながら日夜挑戦していく気概が必要と考えます。

4. IoTネットワーク

- ▶ 2つのネットワーク
 - ▶ エリアネットワーク
 - ▶ デバイス間、デバイス～ゲートウェイ間を結ぶローカルなネットワーク
 - ▶ アクセスネットワーク(コアネットワーク)
 - ▶ ゲートウェイ・デバイス～コアネットワーク間を結ぶワイドエリアネットワーク
- ▶ IoTネットワークの要件
 - ▶ 多様なIoTアプリケーションの特性に合わせて、適切なネットワークを選択する必要がある：
 - ▶ 信頼性（確実性）
 - ▶ 可用性（広いカバーエリア・通信距離、省電力性）
 - ▶ 実時間性（リアルタイム性、応答時間）
 - ▶ セキュリティ（認証、暗号化）
 - ▶ スループット（通信速度、ネットワーク帯域）
 - ▶ コスト（普及度、ライセンス）
 - ▶ サイズ（フットプリント、アンテナを含めた物理的なサイズ）

5. IoTネットワーク

▶ 2つのネットワーク

▶ エリアネットワーク

プロトコル名	周波数帯	特長
BLE (Bluetooth Low Energy)	2.4GHz	スマホに搭載されており、最も広く普及
ZigBee	2.4GHz	欧州・北米で広く普及
IEEE802.15.4	2.4GHz	ZigBeeのベースとなるシンプルなプロトコル
Z-Wave	920MHz	ホームオートメーション等で利用
WiSUN	920MHz	日本発の標準規格でスマートメータに採用
enOcean	920MHz	独が策定した規格で、超低消費電力

▶ アクセスネットワーク(コアネットワーク)

プロトコル名	特長	通信速度
3G	広く普及した、廉価なアクセスネットワーク	数百K～数十M bps
LTE	今後のエリア拡大が期待される低遅延のアクセスネットワーク	数十Mbps
WiMAX	都市部中心の高速なアクセスネットワーク	同上
FTTH	もっとも高速な固定公衆網	1Gbps
ADSL	広く普及している固定公衆網	数十Mbps

[ブレイク] IoT技術テキスト (リックテレコム出版)

IoT技術テキスト -MCPC IoTシステム技術検定 対応

- 単行本（ソフトカバー） - 2016/10/19（第2版 2018/10/11）

内容紹介

IoTの基礎から、システム、通信、デバイス等のインフラ知識、さらにデータ活用、プロトタイピング、セキュリティなど、実務の課題に直結した知識が体系的にまとめられています。

本書1冊で、IoTに関連する技術の全容を概括でき、まとまった知識が得られます。

◆「MCPC IoTシステム技術検定(中級)」の公式テキストです。

内容詳細

■ 本書1冊で、検定試験に合格可能!

本書は、MCPCのIoTシステム技術検定(中級)の出題カテゴリに準拠しており、試験の対象分野全般をカバーしています。
このため、検定に挑戦する方にとって最適な教材となっており、本書1冊で合格レベルまでの知識を得ることができます。

■ グローバルに通用するIoTエンジニアへの第一歩!

IoTシステムは、多種多様な技術、製品の組合せによって実現されますが、現状ではシステムの企画、構築、運用ができるエンジニアが大幅に不足しています。本書は、複数の分野に渡るIoTのシステム技術を、特定の分野に偏らず、過不足なく学べるようバランスに配慮してまとめています。これからの時代に求められるスキル養成のための入門書です。

■ 新しい技術動向もキャッチ!

変化の激しい国際標準化の動向や、注目されている新技術の動向なども追跡し、IoTの新しい波を積極的に取り入れています。



タブレイン（第6章「IoTシステムの
プロトタイピング開発」執筆）

もくじ	
第1章	3 GIMとは
第2章	UART \$ コマンド機能概説
第3章	コマンド・サンプルスケッチ
第4章	ライブラリ概説
第5章	ライブラリ・サンプルスケッチ



3 GIM V2.2

第Ⅱ編 3 GIM基本技術

もくじ

1. 3 GIM誕生の経緯
2. 3 GIM V2.2とは
3. 世界最小クラス 3 G通信機器 3 GIMについて
4. 3 GIM + IoTTABシールドの構成
5. 3 GIM a3gimライブラリの概要
6. 3 GIM V2.2の主な機能
7. 3 GIM V2.2 UART\$コマンド一覧
8. 3 GIM a3gim.h ライブラリ一覧
9. A3gim R4.3 ライブラリ一覧



第1章 3 GIMとは

1. 3 GIM誕生の経緯

- ・ 2010年度、総務省の雇用創出 ICT絆プロジェクトにて応募し、採択「モバイル人材育成のための教材」を開発。2010年度末に完遂
- ・ 2011年夏に、オープンソースハードウェアArduino上での展開を思考
- ・ 2011年秋、会津大学 IT秋フォーラムにて、試作（評価）版を発表
- ・ 2011年12月19日 オープン ワイヤレス アライアンス発足セミナー開催
- ・ 2012年1月 任意団体オープン ワイヤレス アライアンスにて活動開始
- ・ 2012年7月 IEM評価版 3 GIM貸出開始
- ・ 2012年8月28日 教育向け 3 GIM紹介セミナー開催
- ・ 2012年10月 IEM製品版 3 GIM販売開始

その他

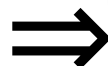
2012.05 東京都立小石川中等教育学校
サイエンスカフェで講演
2012.05 ワイヤレスジャパン展で出展紹介
2012.06 日本情報科教育学科で研究発表
＜優秀研究賞を授与頂く＞
2012.07 高度ポリテクセンターで講義 1
2012.10 高度ポリテクセンターで講義 2
2013.02 日本技術情報センターで講義
2013.03 NCPCの中のモバイルM2M-WG
にてArduino+ 3 GIM紹介

**総務省 2010年度
雇用創出「ICT絆プロジェクト」
として支援を受け開発**



2010年末に本プロジェクトで採択され、
2011年1月～3月で、モバイル教育教材を
開発・完成させる。

貸出用の
IEM評価版 3 GIM



2011年夏ごろから関係者の知恵に
よって3 GIMが出てくる⇒ **2011年
10月13日** 会津大学 IT秋フォーラム
にて、デモ試作品をプレゼン紹介。

**教育現場でもっと簡単に利用できる
モバイル教材として製造・販売**

オープンソースハードウェア
のコンセプトで展開



IEM版 3 Gシールド



Arduino

3 Gシールドの
コンセプトは
①シンプルで、
②高機能を活用、
③応用展開が容易

2. 3 GIM V2.2とは ①

- ▶ Arduino上で簡単に3G通信を行うことができる 3 G通信モジュール
 - ▶ FOMA系の通信をサポート（docomo、MVNOのIIJmio、IIJmobile、So-netモバイルLTE、b-mobile、DTI、Soracom等）
 - ▶ クアルコム設計・AnyData製造の**IEMモジュール版**（既販売）と、**USB dongle版**(予定) を使った2つの3 GIMを用意
- ▶ 低いCPU性能で少ないメモリを持つArduino上で利用できるよう、高機能なAPIをライブラリ“a3gim”として提供
 - （※海外でも3 GIMが販売されているが、ATコマンドで利用：技術ハードルは高い）
 - ▶ http://tabrain.jp/3GIM_V2.2/3GIM%20V2.2R01manual.pdf
- ▶ ライブラリが提供する予定の機能は、下記の通り：
 - ▶ Web通信(HTTP GET/POST)
 - ▶ GPS位置取得(GPS Standalone, AGPS)
 - ▶ TCP/IP通信(connect, disconnect, read, write)
 - ▶ その他（時刻やIEM取得等）（※TCP/IPなどを利用してメール送受信なども可能）

2. 3GIM V2.2とは②

▶ 世界最小サイズの3G通信モジュール・ブレイクアウトボード

- ▶ シェラワイヤレス社の「HL8548-G」（JATE/TELEC 取得済）・NTTドコモ（IOT取得済）を採用
- ▶ サイズは 35mm × 25mm × 7mm , 重量は7.5 g と超小型な3G通信モジュール
- ▶ 32ビットARMマイコン（LPC812M101JTB16）を搭載、独自のファームウェアを開発できる
- ▶ GPS/AGPSが利用可能
- ▶ さまざまなIoTデバイスやゲートウェイとして利用できる携帯向けで、消費電力が低い

HL8548-Gの主な仕様	
UMTS	Band 1/6/19
EDGE/GPRS/GSM	850/900/1800/1900 MHz
GPS	GPS (1575.42MHz) GLONASS (1602MHz) 、 Assisted GPS
Speed	7.2Mbps(Download)/5.76Mbps(Upload)
その他	JATE 取得済み(docomo IOT取得済み)
サイズ	23mm×22mm×2.5mm
動作温度	-30℃ ～ 70℃



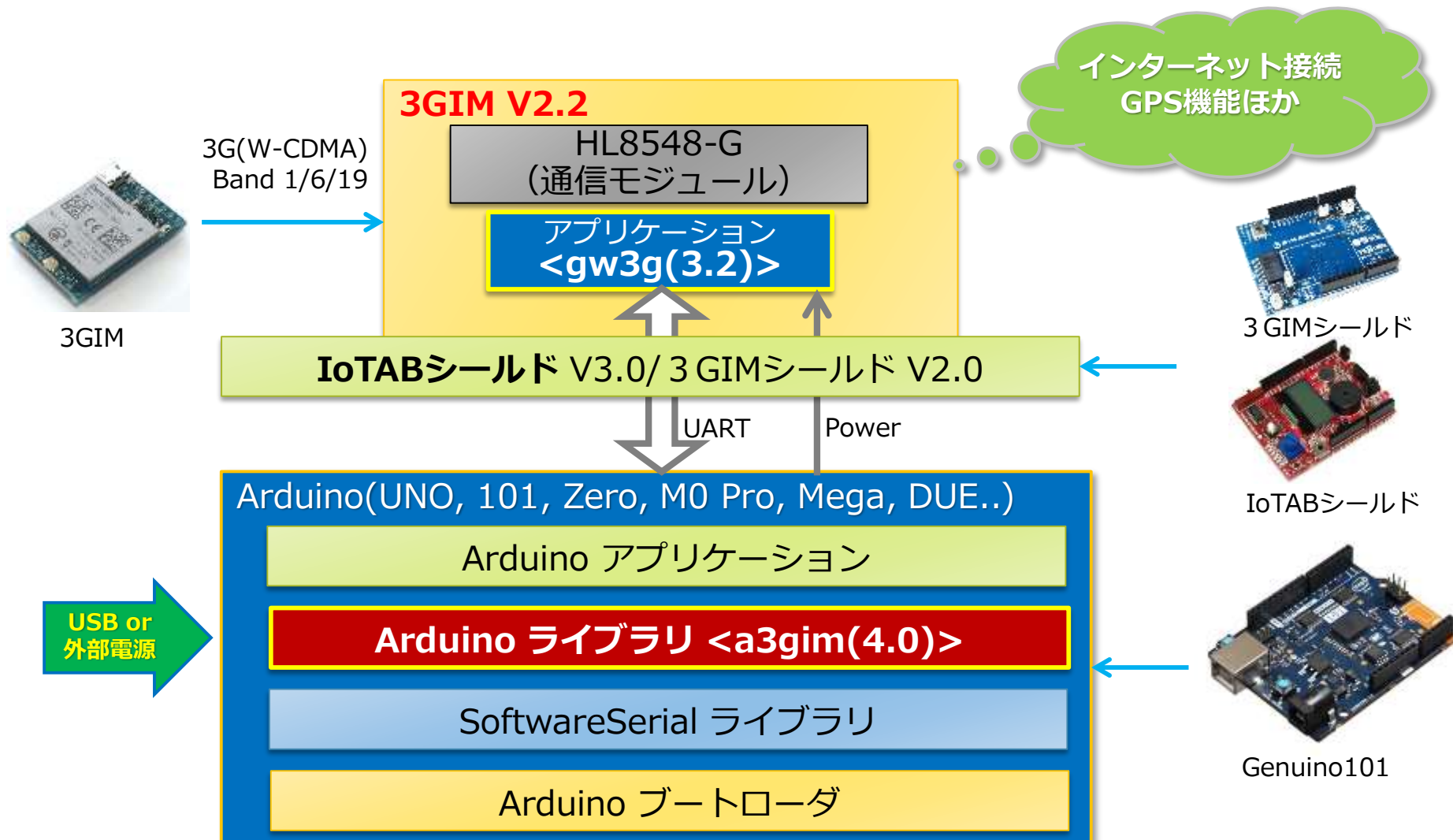
3. 世界最小クラス 3G通信機器 3GIMについて

- ▶ 25mm×35mmの3G通信機器



- ▶ 3GIMと同じ機能がそのまま利用できる。
- ▶ Arduinoをはじめ、mbed、RaspberryPiなどにも利用可能（UART通信）
- ▶ 2014年11月から販売開始。既に大学や企業で利用中。
- ▶ 現在最新版は、3GIM V2.2（通信モジュール：HL8548-G搭載）
- ▶ Assisted GPS（GLONASS併用）および アクティブGPSアンテナ対応

4. 3 GIM + IoTABシールドの構成



5. 3 GIM用 a3gimライブラリの概要

HL8548-Gとは

HL8548-Gは、小型の3 G通信モジュールで、その特徴は
▼ シェアラワイヤレス社の3 G通信モジュール（JATE/TELEC取得済）
サイズは 23mm × 22mm × 2.5mm、重量は4.5g（超小型）
携帯向けに設計されたモジュールであり、消費電力が低い
シェアワイヤレス モジュール製品 - 株式会社アルティマ

HL8548-Gの主な仕様	
無線周波数	800/850/900/1900/2100 MHz
GPS	Standalone GPS, AGPS
Speed	(UL)5.76Mbps/(DL)7.2Mbps
その他	JATE/TELEC 取得済み
動作温度	-45℃ ~ 85℃

ライブラリ概要

3 GIMでは、Arduino側から利用する主なライブラリ群
を以下に分類分けして紹介します。
（基本は、Arduinoライブラリ標準のものをベースとしています）

機能分類	機能概要	補足
コントロール関連	3 GIMの電源制御、初期化等	
ショートメッセージ関連	SMS（ショートメッセージ）の送受信	SIMカード限定による利用
Web関連	GET/POSTのメソッド発行、Tweet	HTTP GET/POST
現在位置取得（GPS）関連	GPS位置情報取得	GPS, AGPS
プロフィール関連	プロフィールの読み書き	
通信その他機能	電波強度、時計、サービス関連	

【注意】 a3gimライブラリは継続的にバージョンアップを重ねていく予定ですので、
インターネットなどによって最新の情報を取得するようにしてください。

Arduino用ピン接続

ピン	用途	補足
Vin	電源供給	電源切替SWにより切り替え可能
Vcc	参照電圧	
GND	グラウンド	
Tx	UARTのTxD	ソフトウェアシリアルとして使用
Rx	UARTのRxD	同上

動作環境

項目	動作環境	補足
Arduino	UNO	
	Leonard	特殊設定
	Pro(5V)	
	Pro(3.3V)	
IDE	Mega 2560/ADK	特殊設定
	バージョン 1.6 以降	1.6.8以上を推奨
電源	USB	800mA以上の供給能力が必要
	ACアダプタ(5V用)	7~9Vで1A以上のものを推奨
	ACアダプタ(3.3V用)	5~6Vで1A以上のものを推奨

※Arduino 互換機の GR-SAKURA（ルネサス製）でも稼働

関数例 (tweet)

int tweet (const char* token, const char* msg)

機能概要	Twitterへ投稿する
引数	token : アクセスに必要なトークン msg : 投稿するメッセージ
戻り値	0 : 正常に投稿できた時 0以外 : 投稿できなかった時

※tweet関数は、Web関連の関数群の中のひとつの機能となります。

6. 3 GIM V2.2の主な機能

Arduino上のセンサや
アクチュエータなどとの連携

クラウド連携（M2Xは無料で利用可能）

センサ類との
接続は無限大

インターネット
接続は無限大

クラウドへの
データアップ

センサ値の
ツイート

画像データ
ネットサーバ
へアップ

ツイート連携

画像データアップ

GPS（位置
情報）取得

センサ値
メール受信

メール受信

GPS機能

その他：SMS、メール送受信も可能
HTTP技術やTCP/IP技術を応用することで、可能性は無限大

7. 3GIM V2.2 UART \$コマンド一覧

3GIM V2.2 のUART経由で利用できる \$ コマンドを下表に示します

No	分類	機能	コマンド	頁	機能概要	補足 (V1.0との比較)
1	System	Version	\$YV		gw3gアプリのバージョン情報の取得	
2		RSSI	\$YR		電波受信強度(RSSI)の取得	
3		Service	\$YS		利用可能サービスの取得	
4		IMEI	\$YI		IMEIの取得	
5		LED	\$YL		LED (RUN) の状態の取得、設定	
6		Baudrate	\$YB		UARTの通信速度の変更	【拡張】リセット後に有効
7		Reset	\$YE		リセット (初期化)	
8		Time	\$YT		日時の取得	
9		Airplane mode	\$YP		エアプレーン (機内) モードの切り替え	
10		ATcommand	\$YA		ATコマンドパススルーモード切換え	【新規】
11	GPS	Get Location	\$LG		位置情報の取得、ロケーションサービスの停止	【拡張】GPS/GLONASS利用
12		Set/Get param	\$LX		位置情報取得のタイムアウト値の取得・設定	
13	Web	Get	\$WG		GETリクエストの送出、レスポンスの取得	ヘッダ指定可(R2.0から)
14		Post	\$WP		POSTリクエストの送出、レスポンスの取得	
15	TCP/IP	Read	\$TR		TCP/IPコネクションからのデータからの読み出し	バイナリデータも取扱可
16		Write	\$TW		TCP/IPコネクションへのデータの書出し	同上 (1 Kバイト以下)
17		Connect	\$TC		TCP/IPコネクションの接続	
18		Disconnect	\$TD		TCP/IPコネクションの切断	
19		Status	\$TS		TCP/IPコネクションの状態の取得、設定	【変更】
20		Get sockname	\$TN		ソケットのIPアドレスとポート番号を取得	接続時のみ有効
21		Tunnel Write	\$TT		現在のコネクションへのデータ直接書出し	32 Kバイト以下
22		Set/Get Param	\$TX		TCP/IP関連タイムアウト時間を取得・設定	
23	Profile	Set/Get Profile	\$PS		デフォルトプロファイル番号の設定	【拡張】

3GIMからの応答 (レスポンス) は、各コマンドの機能紹介にて説明していきます。

注意) V2.2では、SMS関係のコマンドは使えなくなりました。

8. 3 GIM a3gim.h ライブラリー一覧

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

* 無償サービス「<http://arduino-tweet.appspot.com/>」を利用（要Twitterの登録）

分類	メソッド名	機能概要	補足
コントロール関係	getStatus※	3GIMの状態取得	
	begin※	ライブラリの初期化	
	end※	ライブラリの終了	
	restart※	3GIMのリセット	
	start※	3GIMの電源ON	
	shutdown※	3GIMの電源OFF	
	getIMEI	IMEIの取得	携帯端末固有番号
	setLED	緑色LEDのON/OFF	
	setBaudrate	通信速度の設定	
	setAirplaneMode	エアプレーン（機内）モードのON/OFF	
	setResult	通信結果を取得	
インターネット関係 (Web)	httpGET※	GETメソッドの要求	https取得も可能
	httpPOST	POSTメソッドの要求	
	tweet※	Twitterへの投稿	*
インターネット関係 (TCP)	connectTCP※	TCPコネクションを接続する	
	disconnectTCP※	TCPコネクションを切断する	
	writeBegin	シリアル通信で直接書込み	
	read※	データを読み込む	
	write※	データを書き出す	
位置情報取得（GPS）関係	getLocation	現在位置の取得	内蔵GPSを使用
	getLocation2	Assisted GPSを使った現在位置の取得	
その他ライブラリ	getServices	利用可能サービスの取得	
	getRSSI	電波強度の取得	
	getTime	現在時刻の取得	日付・時刻形式
	getTime2	現在時刻の取得	通算秒形式
	getVersion	3GIM(gw3gアプリ)のバージョンの取得	
プロファイル関係	setDefaultProfile	デフォルトプロファイル設定	
	getDefaultProfile	デフォルトプロファイル取得	
ATコマンドパススルー	enterAT	ATコマンドパススルーモード	

9. A3gim R4.3 ライブラリー一覧

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

* 追加 : V2.2で追加となったもの

* 仕様変更 : V2.2で以前のものから仕様変更されたもの

分類	メソッド名	機能概要	補足
コントロール (Control)	begin※	ライブラリの初期化	
	end※	ライブラリの終了	
	restart※	3GIMのリセット	
	start※	3GIMの電源ON	
	shutdown※	3GIMの電源OFF	
	getServices	現在使用できる通信サービス	
	getIMEI	IMEI-IDの取得	
	setBaudrate	UARTの通信速度の設定	初期は9600bps
	setLED	LEDのON/OFF	
	setAirplaneMode	エアプレーンモードのON/OFF	
インターネット関係 (Web)	httpGET※	GETメソッドの要求	http/https利用可能
	httpPOST	POSTメソッドの要求	同上
	tweet※	Twitterへの投稿	
通信機能その他	getRSSI	電波強度の取得	
	getTime	現在時刻の取得	日付・時刻形式
	getTime2	現在時刻の取得	通算秒形式
	getVersion	3GIMのバージョンの取得	
	getResult	3GIMからの応答値受信	
	sendCommand	3GIMへのコマンド送信	
	sendData	3GIMへのデータ送信	
	enterAT	ATコマンドパススルーモード	*仕様変更
	discardUntil	3GIMからの文字の受信	
現在位置取得 (GPS)	getLocation	現在位置の取得	緯度経度情報
	getLocation2	現在位置の取得2	緯度経度他情報
	setLocationParams	GPS機能関連設定	*追加
その他ライブラリ	connectTCP※	TCPコネクションを接続	
	disconnectTCP※	TCPコネクションを切断	
	getStatusTCP	TCPコネクション最新状況取得	
	setTCPParams	TCPパラメータ設定	*追加
	writeBegin	シリアル通信で直接書き込み	
	read※	データの読み込み	2つのバリエーション有
	write※	データの書き出し	3つのバリエーション有
プロフィール	setDefaultProfile	デフォルトプロフィールを設定	※仕様変更
	getDefaultProfile	デフォルトプロフィールを取得	

第2章 UART\$コマンド機能概説



1. システム関連

1. SYSTEM VERSION ①

■ 3GIM V2.1（通信）モジュールに設定されたファームウェアのバージョン取得

通信モジュールに設定されているファームウェア（gw3g）のバージョンを取得する

項目	値など	説明	補足
機能分類	System		
機能名	VERSION	ファームウェア（gw3g）のバージョンを取得する	
コマンド形式		\$YV¥n	
引数	-		
応答値	【正常時】	\$YV=OK version¥n	
	version	"9.9"形式のバージョン（整数桁がメジャ番号、小数以下がマイナ番号）※	
	【エラー時】	\$YV=NG errno ¥n	
	errno	141：コマンドの形式に誤りがある	
前提条件			
補足事項	①	本コマンドで取得できるバージョン情報は、ファームウェアgw3gのバージョン情報である。3GIMのハードウェアのバージョン情報とは異なる。	

※ 2017年8月の出荷時点ファームウェアバージョンは「3.3」である。

1. SYSTEM VERSION ②

■ 事例：バージョン取得サンプルプログラム

スケッチ名：system_version.ino

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
void setup() {
  Serial.begin(9600);
  Serial3G.begin(9600);
  Serial.println("begin SYSTEM version");
  pinMode(7,OUTPUT);
  digitalWrite(7,HIGH);
  delay(100);
  digitalWrite(7,LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YV");
  while(!Serial3G.available());
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
  pinMode(7,OUTPUT);digitalWrite(7,HIGH);
}

void loop() {}
```

【補足】ここでの呼出し関数群について

- SoftwareSerial は、ソフトシリアル通信利用宣言
- Serial3G.begin(通信速度)は、ボーレート宣言
- delay(num)は、numミリ秒の待機時間
- Serial3G.listen()は、受信状態占有関数
- Serial3G.available()は、受信バイト数を返す
- Serial3G.readStringUntil('\n')は、リターン値までの文字列読み込み関数

■ 実行モニタ画面（サンプル）

```
begin SYSTEM Version
$YV=OK 3.3
end
```

応答受信

返信が無い場合は、以下のようなことが考えられる

- 1) 通信モジュールに電源（橙色LED点灯）が入っていない
- 2) 通信ボーレートが間違っている
- 3) 配線（特にUARTのTxとRxの接続）が間違っている
- 4) 電源電力が不足している

2. SYSTEM RSSI ①

■ 電波受信強度 (RSSI) の取得

項目	値など	説明	補足
機能分類	System		
機能名	RSSI	現在のRSSI値を取得する	
コマンド形式		\$YR¥n	
引数	-		
応答値	【正常時】	\$YR=OK rssi¥n	
	rssi	電波強度 (-51~-113) [dBm]	rssiは常にマイナス値
	【エラー時】	\$YR=NG errno¥n	
	errno	102 : コマンド形式に誤りがある 101 : 電波強度が取得できない	
前提条件	①	あらかじめ3G用のアンテナが正しく装着されていること	
補足事項	①	3GIMをONにした直後ではエラーとなる場合がある。そのため、確実にRSSIを取得するには、1秒程度の間隔を空けて3回程度のリトライを行うことを推奨する。	

RSSIとは、無線通信機器が受信する信号の強度を表す。(Received Signal Strength Indication, Received Signal Strength Indicator 別名：受信信号強度)

RSSI値は常にマイナスで、目安は下記の通りである：

- 113の場合には、アンテナが接続されていないとき
- 112~-90 の場合は、電波受信状態が悪い状況
- 89~-51 の場合は、電波受信状態が良い状況

2. SYSTEM RSSI ②

■ 事例：電波受信強度（RSSI）を取得サンプルプログラム

スケッチ名：system_rssi.ino

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
void setup() {
  Serial.begin(9600);
  Serial3G.begin(9600);
  Serial.println("begin SYSTEM RSSI");
  pinMode(7,OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YR");
  while (!Serial3G.available());
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}

void loop() {}
```

通信速度設定

コマンド送信

■ 実行モニタ画面（アンテナ正常接続）

```
begin SYSTEM RSSI
$YR=OK -86
end
```

応答受信

■ 実行モニタ画面（アンテナ未接続）

```
begin SYSTEM RSSI
$YR=OK -113
end
```

応答受信

RSSIの感度が悪い場合

- 1) 通信モジュールとアンテナのコネクタが正しく接続されていない
- 2) アンテナとケーブル・コネクタのネジ部が緩んでいる
- 3) 電波状態が悪い屋内の壁・天井・床などで閉ざされたところにある
- 4) SIMカードが正しく挿入されていないか、APN情報が正しく設定されていない

3. SYSTEM SERVICE ①

■SIMカードによる通信サービス状況を取得

項目	値など	説明	補足
機能分類	System		
機能名	SERVICE	現在利用できる通信サービスを取得する	
コマンド形式		\$YS¥n	
引数	-		
応答値	【正常時】	\$YS=OK serice¥n	
	service	0 : サービス利用不可 1 : パケット通信(PS)のみ利用可	
	【エラー時】	\$YS=NG errono¥n	SIMカードやアンテナが無い時
	error	131 : コマンドの形式に誤りがある 132 : 内部エラー (サービス種別を取得できない)	
前提条件	①	あらかじめSIMカードが装着されていること	SIMカードがないと常に結果としてNGが返る
補足事項	①	3GIM(Ver2.x)では音声通信の利用可否を取得できないため、音声通信SIMとしての値は返却しない。	
	②	3GIMをONにした直後ではエラーとなる場合、あるいはパケット通信が利用できるにも関わらず0を返す場合がある。	

3GIM V2.2で利用できる**SIMカード**は、NTTドコモおよびドコモFOMA回線を使ったMVNO提供のSIMに限ります。

3. SYSTEM SERVICE ②

スケッチ名 : system_service.ino

■ 事例 : SIMカードによる通信サービス状況を取得サンプルプログラム

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(9600);
  Serial3G.begin(9600);
  Serial.println("begin SYSTEM Service");
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YI");
  while(!Serial3G.available());
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}

void loop() {}
```

通信速度設定

コマンド送信

■ 実行モニタ画面（パケット通信のみの利用の場合）

```
begin SYSTEM Service
$YS=OK 1
end
```

応答受信

4. SYSTEM IMEI ①

■通信モジュールのID（固有）番号（IMEI）を取得

項目	値など	説明	補足
機能分類	System		
機能名	IMEI	IMEIを取得する	
コマンド形式		\$YI¥n	
引数	-		
応答値	【正常時】	\$YI=OK imei¥n	
	imei	15桁の数字（3GIMを一意に識別できる数字列）	
	【エラー時】	\$YI=NG errno¥n	
	errno	143：IMEIを取得できない 146：コマンドの形式に誤りがある	
前提条件			
補足事項			

IMEIは「国際移動体装置識別番号（端末識別番号）」を意味する英語"International Mobile Equipment Identifier"の略称



4. SYSTEM IMEI ②

スケッチ名 : system_imei.ino

■事例：通信モジュールのID（固有）番号（IMEI）を取得サンプルプログラム

■Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
void setup() {
  Serial.begin(9600);
  Serial3G.begin(9600);
  Serial.println("begin SYSTEM IMEI");
  pinMode(7,OUTPUT);
  digitalWrite(7,HIGH);
  delay(100);
  digitalWrite(7,LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YI");
  while (!Serial3G.available());
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}

void loop() {}
```

通信速度設定

コマンド送信

■実行モニタ画面（正常時）

```
begin SYSTEM IMEI
$YI=OK{359516050164625}
end
```

応答値

※このID番号の数値はサンプルです

■実行モニタ画面（エラー時）

```
begin SYSTEM IMEI
$=NG 10
end
```

※コマンドが正しく読み込めなかった場合

IMEI番号



5. SYSTEM LED ①

■ 3GIM上の緑LEDの点滅設定および状態取得

項目	値など	説明	補足
機能分類	System		
機能名	LED	緑LED (RUN) ピンの状態取得・設定を行う	
コマンド形式	状態取得	\$YL¥n	
	設定	\$YL status¥n	
引数	status	ONにするか(1の時)、OFFにするか(0の時)	1 : 点灯、0 : 消灯
応答値	【正常時】	\$YL=OK status¥n	
	status	本コマンド実行後のLED状態 (0:OFF/1:ON)	
	【エラー時】	\$YL=NG errno¥n	
	errno	191 : コマンド形式または引数statusがおかしい	
前提条件			
補足事項	①	本機能で扱うLEDは、1番ピン脇に配置されている緑色LEDである（基板上に「LED2」と記載）	



このLEDが点灯

5. SYSTEM LED ②

スケッチ名 : system_led.ino

■ 事例 : LED状態取得とLEDの点滅を10回繰り返すサンプルプログラム

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
void setup(){
  Serial.begin(9600);
  Serial3G.begin(9600);
  Serial.println("begin SYSTEM LED");
  pinMode(7,OUTPUT);
  digitalWrite(7,HIGH);
  delay(100);
  digitalWrite(7,LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YL");
  String rt = Serial3G.readStringUntil('\n');
  Serial.print(rt);
  for (int i=0; i<10; i++) {
    Serial3G.println("$YL 1");
    delay(500);
    Serial3G.println("$YL 0");
    delay(500);
  }
  Serial.print("$r$end");
}
void loop() {}
```

点滅を10回繰り返す

■ 実行モニタ画面（正常時）

```
begin SYSTEM LED
$YL=OK 0
end
```

6. SYSTEM BAUDRATE ①

■ 3GIMのUART（通信ポート）の通信速度（ボーレート）取得確認と設定

項目	値など	説明	補足
機能分類	System		
機能名	BAUDRATE	UARTの通信速度(ボーレート)の取得・設定を行う	
コマンド形式	取得 設定	\$YB¥n \$YB baudrate¥n	
引数	baudrate	設定するボーレート(9600/19200/38400/57600/115200)	出荷時は9600bps
応答値	【正常時】	\$YB=OK baudrate¥n	
	baudrate	本コマンド実行後のボーレート	
	【エラー時】	\$YB=NG errno¥n	
	errno	111 : コマンド形式または引数baudrateがおかしい	
前提条件	①	指定するボーレートで正しく動作することを確認しておくこと。	
補足事項	①	本コマンドで設定した新しいボーレートは直ちに反映される。	
	②	本コマンドで設定したボーレートは電源の再投入やリセットしても維持される。	

【注意事項】

- ① 現状のボーレートと、変更後のボーレートは、常に把握した上でこのコマンドを使うこと
(現在のボーレートが分からなくなる/通信できなくなると、一つ一つボーレートを試して探り当てる必要がある)
- ② Arduinoのソフトウェアシリアルを利用する場合は、ハードウェアシリアルを使った場合よりも低いボーレートでしか利用できない
(ソフトウェアシリアルの場合は38400bps以下での利用を推奨)

6. SYSTEM BAUDRATE ②

スケッチ名 : system_baudrate.ino

■ 事例 : 3GIMのUARTのボーレート取得確認サンプルプログラム

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
#define NOWBAUDRATE 9600
#define NEWBAUDRATE 38400
void setup() {
  Serial.begin(9600);
  Serial.println("begin SYSTEM Baudrate");
  Serial3G.begin(NOWBAUDRATE);
  pinMode(7,OUTPUT);
  digitalWrite(7,HIGH);
  delay(100);
  digitalWrite(7,LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YB " + String(NEWBAUDRATE));
  delay(20);
  Serial3G.begin(NEWBAUDRATE);
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}

void loop() {}
```

この通信速度にする

この待機時間重要

ボーレートを9600bpsから38400bpsに変更するサンプルです。

■ 実行モニタ画面（正常時）

```
begin SYSTEM Baudrate
$YB=OK 38400
end
```

ボーレート設定での注意点

- 1) ボーレート設定を変更すると、改めてシリアル通信速度も変更が必要となります。
- 2) ボーレート変更設定した場合には、その情報はメモ等で控えておいてください。

ボーレート変更によって起こる問題

- 1) 文字化けが起きることがあります。特にスクリーンモニタ画面に出す場合には、なるべく、通信速度をシリアルモニタ画面の通信速度と合わせるようにお願いいたします。
- 2) 処理の待機時間が関係してきますので、同じソフトウェアでも調整が必要となります。

7. SYSTEM RESET ①

■ 3GIMをリセットするコマンド

項目	値など	説明	補足
機能分類	System		
機能名	RESET	3GIMをリセットする	
コマンド形式	ソフトリセット	\$YE¥n	
	指定リセット	\$YE level¥n	
引数	level	リセットのレベル (0: ソフトリセット、1: ハードリセット)	levelの内容に拘らず常に再起動する
応答値	【正常時】	\$YE=OK level¥n	
	level	引数と同じ	
	【エラー時】	\$YE=OK errno¥n	
	errno	191: コマンドの形式に誤りがある	
補足事項	①	リセット後、復帰までには14秒程度の時間が掛かる。再起動するまで3GIMは利用できない。	
	②	実装上の理由で、ハードリセットはソフトリセットと同じ動作となっている。	

ハードウェアリセットは、#1ピンをHIGHにして電源をOFFすることでも実行できます。
電源がOFFとなった時は、3GIM上の橙LED(LED1)が消灯します。

7. SYSTEM RESET ②

スケッチ名 : system_reset.ino

■ 事例 : リセットのサンプルプログラム

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4,5);
void setup(){
  Serial.begin(38400);
  Serial3G.begin(38400);
  Serial.println("begin SYSTEM Reset");
  pinMode(7,OUTPUT);
  digitalWrite(7,HIGH);
  delay(100);
  digitalWrite(7,LOW);
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial3G.println("$YE");
  Serial.println(Serial3G.readStringUntil('\n'));
  while(Serial3G.readStringUntil('\n').indexOf("3GIM")<0);
  Serial.println("restart:");
  Serial.println("end");
}
void loop() {}
```

■ 実行モニタ画面（正常時）

```
SYSTEM Reset
$YE=OK
end
```


8. SYSTEM TIME ①

■ 通信モジュールの取得した時間取得

項目	値など	説明	補足
機能分類	System		
機能名	System Time	日時の取得	
コマンド形式	時刻取得	\$YT¥n	
	時刻取得(2)	\$YT 1¥n	時刻サーバから日時を取得し なおす
引数			
応答値	【正常時】	\$YT=OK datetime¥n	
	datetime	出力例として \$YT=OK 2016/05/14 15:52:23 などのように「年（4バイト） '/' 月（2バイト） '/' 日（2バイト） '」（スペース1バイト）時（2バイト） ':' 分（2バイト） ':' 秒（2バイト）」でリターン値が返ってくる。	日時はJST
	【エラー時】	\$YT=NG errno¥n	
	errno	121:コマンドの形式に誤りがある 122:内部エラー（日時の取得に失敗した）	errnoの後にエラー情報を出 力する場合あり
前提条件	①	アンテナ接続と正しいSIMカードの設定で正確な時刻が取得できる	
補足事項	①	本コマンドの最初の実行時にインターネット上の時刻サーバから正しい日時を取得して、HL8548-Gのリアルタイムクロック(RTC)に設定して保持する。2回目以降の実行では、RTCから時刻を読み出す。	初回の実行時はインターネットに接続するため、正しいプロファイル情報の設定、SIMカードおよび3Gアンテナの装着が必要となる。

ファームウェア起動後に「\$YT」を起動すると初回のみ5～10秒ほど掛って時間を取得する。
その後2回目以降は、瞬時に取得できる。

8. SYSTEM TIME ②

スケッチ名 : system_time.ino

■ 事例 : 日時取得表示のサンプルプログラム

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(38400);
  Serial3G.begin(38400);
  Serial.println("begin System Time");
  Serial3G.println("$YT");
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}
void loop() {}
```

【補足】取得した時間が間違っている場合

- 1) 正しいSIMカードが設定されているかを確認
 - 2) アンテナ接続が正しく接続されているかを確認
 - 3) アンテナ感度が良い環境かどうかを確認 (参照 : \$ YR)
 - 4) 正しい時間取得までに5~10 秒ほど掛る
- ※2回目以降は、瞬時に取得できる

※注意 : 正しいSIMカードとアンテナ接続がされてないと
時間取得ができません。

■ 実行モニタ画面 (正常時)

```
begin System Time
$YT=OK 2017/08/13 10:58:11
end
```

■ 時間取得関数 (例)

```
String datetime() {
  String dtime;
  uint32_t tim = millis();
  do {
    Serial3G.println("$YT");
    dtime = Serial3G.readStringUntil('\n');
    if (millis() - tim > 60000) return "error";
  } while (dtime.indexOf("20") < 0);
  return dtime.substring(7);
}
```

```
begin System Time
$YT=OK 2017/08/13 11:01:10
continue
2017/08/13 11:01:10
2017/08/13 11:01:15
2017/08/13 11:01:20
2017/08/13 11:01:25
2017/08/13 11:01:30
2017/08/13 11:01:35
2017/08/13 11:01:40
2017/08/13 11:01:45
```

9. AIRPLANE MODE ①

■ エアプレーンモード（機内モード：省エネモード）の取得・設定

項目	値など	説明	補足
機能分類	System		
機能名	Airplane mode	3GIMのエアプレーン（機内）モードを切り替える	
コマンド形式	モード取得	\$YP¥n	
	モード設定	\$YP mode¥n	
引数	mode	設定するモード（0: 通常モード、1:エアプレーンモード）	
応答値	【正常時】	\$YP=OK mode¥n	
	mode	設定後のモードを返す 0 : 通常モード 1 : エアプレーン（機内）モード	
	【エラー時】	\$YP=NG errno¥n	
	errno	181 : コマンド形式またはモードの値がおかしい 182 : 内部エラー（エアプレーンモードの変更ができない）	
前提条件			
補足事項	①	エアプレーンモードでは、 \$YB および \$YP 、 \$YE コマンド以外のコマンドは利用できない。	
	②	エアプレーンモード時の消費電流は実測値で約10mAである。	

補足：本エアプレーンモード時は、**3GIMの消費電流を数ミリアほどに抑えることができます。**
完全に消費電力をゼロにするには、3GIMの# 1 ピンをHIGHにすることで、電源をOFFにできます。

9. AIRPLANE MODE ②

■ 事例：エアプレーンモード値の取得サンプルプログラム

スケッチ名：airplane_mode.ino

■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(38400);
  Serial3G.begin(38400);
  Serial.println("begin AirPlane Mode");
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  while (Serial3G.readStringUntil('\n').indexOf("3GIM") < 0);
  Serial3G.println("$YP");
  Serial.println(Serial3G.readStringUntil('\n'));
  Serial.println("end");
}
void loop() {}
```

■ 実行モニタ画面（正常時）

```
begin AirPlane Mode
$YP=OK 0
end
```

10. AT COMMAND ①

■ ATコマンドモードへの切り替え

項目	値など	説明	補足
機能分類	System		
機能名	AT Pass Through	ATコマンドパススルーモードに入る	
コマンド形式	モード取得	\$YA [time]¥n	
引数	time	待機時間：単位100ミリ秒；（例：time=300 は 30秒間を意味する）	
応答値	【正常時】	\$YA=OK¥n	
前提条件	①	ATコマンドの機能を理解した上で使用のこと。	
補足事項	①	利用できるATコマンドの詳細は「AT Commands Interface Guide - AirPrime HL6 and HL8 Series」を参照のこと。	Sierra Wireless社のサイトで公開
	②	ATコマンドの使用に制限はないため、HL8548-Gの設定を任意に変更することができる。しかし、変更した設定等によってはgw3gファームウェアの動作に支障を来す場合があるので、十分の留意すること。	例えば、ATコマンドでプロファイルの設定等を変えてしまうと、3G通信ができなくなる可能性がある。
	③	ATコマンドパススルーモードから抜けるには、以下のいずれかの方法を使用する： ①PWR_ONピンを制御して、3GIMの電源を入れなおす。 ②「 AT+CPOF 」コマンドを実行して3GIMをリセットする。	①・②共に初期起動時に戻る。「Welcome to 3GIM(v*)」応答

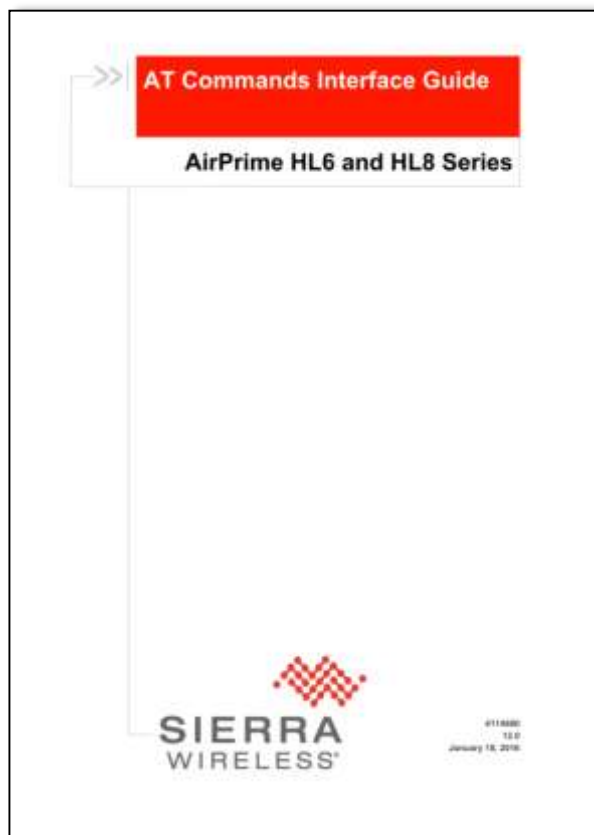
事例：アシストGPSを使う場合、以下のようなATコマンド設定を行います。
(以下の「\$YA 1」で100ミリ秒間だけATコマンドパススルーモード)

```
Serial3G.println("$YA 1");
delay(10);
Serial3G.println("at+wppp=2,4,¥\"username¥\",¥\"password¥\"");
```

※usernameとpasswordは、SIMカードのAPN情報のユーザ名とパスワード

10. AT COMMAND ②

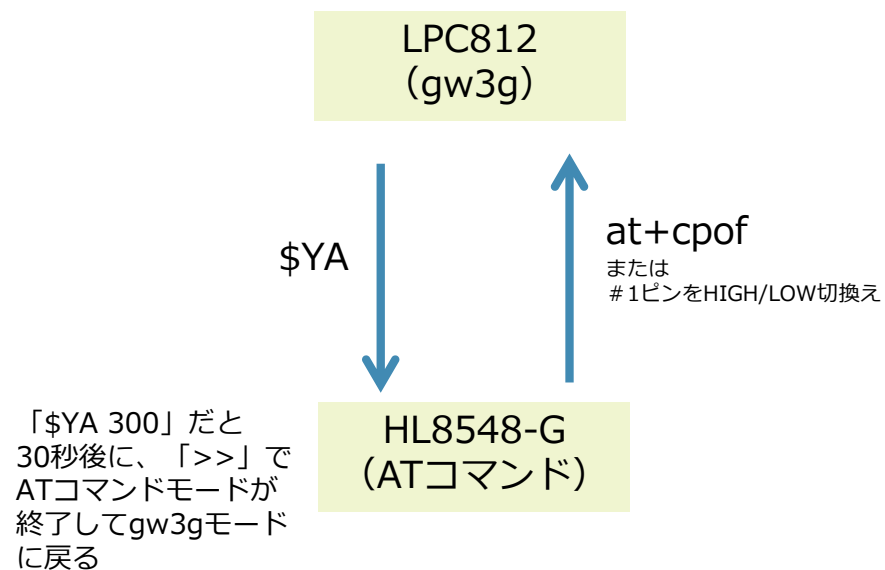
HL8548-GのATコマンドは、シエラワイヤレスサイトから以下の資料をダウンロードできます。
ドキュメント名 (AirPrime_HL6_and_HL8_Series_AT_Commands_Interface_Guide_Rev*_0.pdf)



【注意点】

ATコマンドでHL8548Gの特殊な設定を行った場合には、LPC812から制御不能となることがあり、3GIMが利用できなくなる場合があります。

ATコマンドで操作する場合には、十分な知識を得た上で、ご利用いただく事お勧めいたします。



2. GPS関連

GPS利用時は、GPS専用アンテナが必要です。
別売のGPSフレキアンテナか、あるいは
市販のアクティブGPSアンテナをご購入下さい。

1. LOCATION GPS ①

■ GPS(全地球測位システム) の取得コマンド (次頁につづく)

項目	値など	説明	補足
機能分類	GPS		
機能名	LOCATE	測位を行う。	
コマンド形式		\$LG method [option [repeatCount]]¥n	
引数	method	測位の方法（下記のいずれかを指定：任意文字/‘x’などでも可） MSBASED : GPSで測位、GPSが利用できない時は3Gネットワークを利用 MSASSISTED : 3Gネットワークを利用して測位（A-GPS） STANDALONE : GPS単体で測位	任意文字入力時はMSBASEDとなる（本文字列は3GIM V1.0用）
	option	測位のオプション 0 : 緯度・経度を表示（V1.0と同じ） 1 : 緯度・経度の他にUTC、品質、捕捉衛星数、高度を出力 2 : \$GPGGAセンテンスの生データを出力（ダブルクォート付） 3 : NMEAフォーマット（各NMEAセンテンス）出力※ 9 :ロケーションサービスを停止（SLEEP）させる	省略時は 0
	repeatCount	測位・出力の回数（option=0,1,2の場合）または行数（option=3の場合） 1～100までの回数を指定（出力間隔は測位状況によるが1秒間隔以上）	省略値は 1

次頁につづく

※NMEAフォーマットは、GPSの出力フォーマットで、主に以下の様なセンテンスとなっています。（以下参考）

センテンス	NMEAセンテンス内容(概要)	備考
\$GPGGA	UTC,緯度,N,経度,E,品質,使用衛星数,水平精度低下率,海拔高さ,M,ジオイド高さ,M, 差動基準地点ID,チェックサム	at+gpsnmea=1,,1
\$GPGSA	モード,特定タイプ,衛星番号,位置精度低下率,水平精度低下率,垂直精度低下率,チェックサム	at+gpsnmea=1,,2
\$GPRMC	UTC,A,緯度,N,経度,E,移動の速度,移動の真方位,日付,磁北と真北の間の角度の差,方向,モード,チェックサム	at+gpsnmea=1,,4

1. LOCATION GPS ②

■ GPS(全地球測位システム) の取得コマンド (つづき)

前頁よりつづく

項目	値など	説明	補足
応答値	【正常時】	\$LG=OK latitude longitude¥n	option=0の時
		\$LG=OK latitude longitude utc quality number height¥n	option=1の時
		\$LG=OK "gpgga"¥n	option=2の時
		\$LG=OK ¥nmea_sen¥n...	option=3の時
	latitude	緯度（北緯、9.99999形式、ただし桁数は場合により可変）	option=0の時
	longitude	経度（東経、9.99999形式、ただし桁数は場合により可変）	
	utc	世界標準時間（協定世界時） 日本は9時間プラスする必要がある	
	quality	位置特定品質。0：位置特定できない、1：GPS（標準測位サービス）モード、2：differencetial GPS（干渉測位方式）モードの何れか	option=1の時 上記含む
	number	捕捉衛星数	
	height	GPSアンテナの海拔高さ（m）	
	gpgga	NMEAで規定されている \$ GPGGA センテンスの生データ（ただし末尾の改行は削除）	option=2の時
前提条件	nmea_sen	各NMEAフォーマット 出力	option=3の時
	【エラー時】	\$LG=NG errno¥n	
	errno	501：引数指定エラー	
		508：GPS測位エラー（タイムアウトエラー：3分間）	
		509：BUSYエラー（すでに測位中）	
補足事項	①	GPSアンテナが正しく装着されてること	
	②	測位には、初回には数分以上の時間がかかる場合がある。	
	③	AGPSサーバとして、Googleのロケーションサーバを利用する。 repeatCountを2以上で指定した場合は、指定回数の出力が終わるまでは制御は戻らない（次のコマンドは受け付けない）	

※ 短時間でGPS取得するには、**Assisted GPS** をご利用ください。Assisted GPSについては、以下の資料をご参考ください
http://tabrain.jp/3GIM_V2.0/Part5%20Arduino%20and%203GIM.pdf

1. LOCATION GPS ③

■事例：GPS取得プログラム

スケッチ名：location_gps.ino

■Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
const unsigned long baudrate = 38400;

void setup() {
  Serial.begin(baudrate);
  Serial3G.begin(baudrate);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  Serial.println("begin Location GPS");
  while (Serial3G.readStringUntil('¥n').indexOf("3GIM") < 0);
}

void loop() {
  Serial.print(GPSget());
}

String GPSget() {
  Serial3G.println("$LG x 0");
  String rstr;
  do {
    rstr = Serial3G.readStringUntil('¥n');
  }
  while (rstr.indexOf("$LG") < 0);
  if (rstr.indexOf("OK") > 0 )
    return ("¥r¥n" + rstr.substring(7));
  else return ".";
}
```

■シリアルモニタ画面（正常時応答）

```
begin Location GPS
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
35.642001 139.604221
```

【補足説明：初回のGPS取得できない理由】

※ \$LG=NG 508 (T/O)が時間オーバで返ってくる時

- 1) 屋内などのGPS衛星がとらえられない
→ なるべく屋外などで電波状態が良い環境で実行
- 2) 数分でGPS取得できない
→ 初回は3分以上ほどかかる場合もあるので何度か実行
- 3) アンテナや電源の不備
 - ・ GPSアンテナが正しく接続されていない
 - ・ GPSアンテナの向きが悪い
 - ・ 外部電源供給が不足している
- 4) パソコンなどの近くでは電波障害でGPS取得しにくい
→ パソコン本体などから3GIMを離して実行する

※ この場合、3GアンテナやSIMカードが正しく接続されていると Assisted GPSによって短時間で位置情報が取得できます。

2. LOCATION GPS (Assisted GPS)

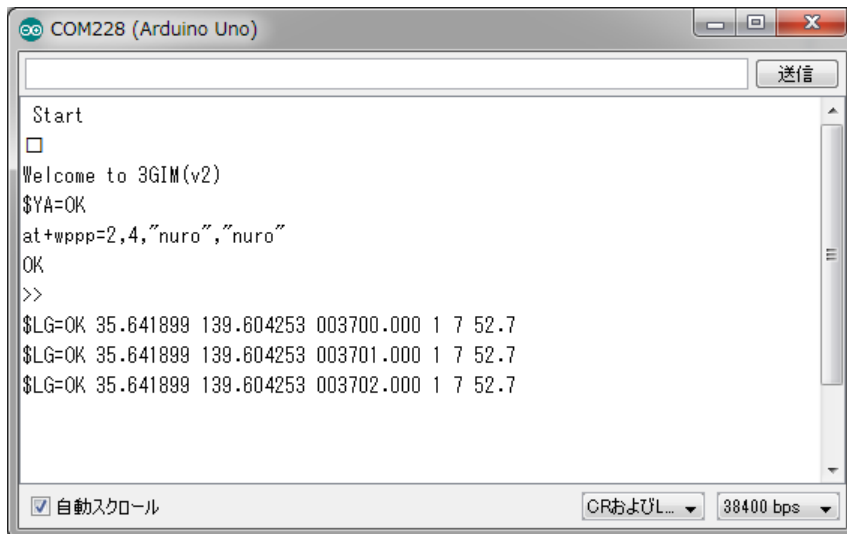
AGPS(Assisted GPS)機能は、GPS衛星の情報を3G通信を使ってSUPLサーバから受信し、いち早く位置情報を取得することを可能とする技術です。

3GIMでは、ATコマンドモードでのAssisted GSP機能を実行させるための以下の2行を実行しておくことで、切り替えることができます。

```
$YA 1  
at+wppp=2,4,"ユーザ名","パスワード"
```

ここで「\$YA 1」は、ATコマンドモードに100ミリ秒間だけ入ることになります。

次の「at+wppp=…」でのユーザ名とパスワードは、利用しているSIMカードのプロファイル情報となります。



スケッチ名 : location_agps.ino

■Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>  
SoftwareSerial Serial3G(4, 5);  
#define BAUDRATE 38400  
  
void setup() {  
  Serial.begin(BAUDRATE);  
  Serial3G.begin(BAUDRATE);  
  pinMode(7, OUTPUT);  
  digitalWrite(7, HIGH);  
  delay(100);  
  digitalWrite(7, LOW);  
  Serial.println("Start");  
  delay(14000);  
  Serial3G.println("$YA 1");  
  delay(5);  
  Serial3G.println("at+wppp=2,4,\"ユーザ名\",\"パスワード\");  
  delay(100);  
  Serial3G.println("$LG x 1 3");  
}  
  
void loop() {  
  if (Serial3G.available() > 0)  
    Serial.println(Serial3G.readStringUntil('\n'));  
  if (Serial.available() > 0)  
    Serial3G.println(Serial.readStringUntil('\n'));  
}
```

3GIM(V2.2)では、SUPLサーバとしてGoogle社が無償提供しているサーバを利用しています。

3. SET/GET PARAMETERS

■ GPS測位のタイムアウト時間の取得・設定

項目	値など	説明	補足
機能分類	GPS		
機能名	Locate	\$LGコマンドによる測位処理のタイムアウト時間を取得・設定する。	
コマンド形式	パラメータ取得	\$LX¥n	
	パラメータ設定	\$LX timeout¥n	
引数	timeout	設定するタイムアウト時間（10ミリ秒単位、0～540000の範囲）	デフォルトは180000
応答値	【正常時】	\$LX=OK¥n \$LX=OK timeout¥n	
	timeout	取得したタイムアウト時間（10ミリ秒単位）	
	【エラー時】	\$LX=NG errno¥n	
	errno	511：コマンド形式がおかしい 512：指定されたタイムアウト時間がおかしい	
前提条件			
補足事項	①	本コマンドで設定したタイムアウト時間は、3GIMの電源をOFFにした時やリセットした時に、デフォルトの時間（180000=3分間）に戻る。	

【補足説明】 アクティブGPSアンテナの使い方

スケッチ名 : location_active_gps.ino

3 GIM V2.2ではアクティブGPSアンテナが取り付けられることができるようになりました。このことで3～5mほどの長いGPSアンテナも3 GIMに取り付けることが可能となり、GPS受信感度を高めることができるようになります。

このアクティブGPSアンテナを使う場合、アンテナ・ケーブルに電源供給する必要があります、そのパラメータ設定をATコマンドの「at+gpsconf」を使って行います。

「at+gpsconf」コマンドの設定は以下の通りです。

at+gpsconf=<型>,<値>

この場合、<設定型>は、

1 : LNA (Low Noise Amplifier) 設定 (固定)

<値>は、

0 : 高ゲインによるLNA出力

1 : 低ゲインによるLNA 設定

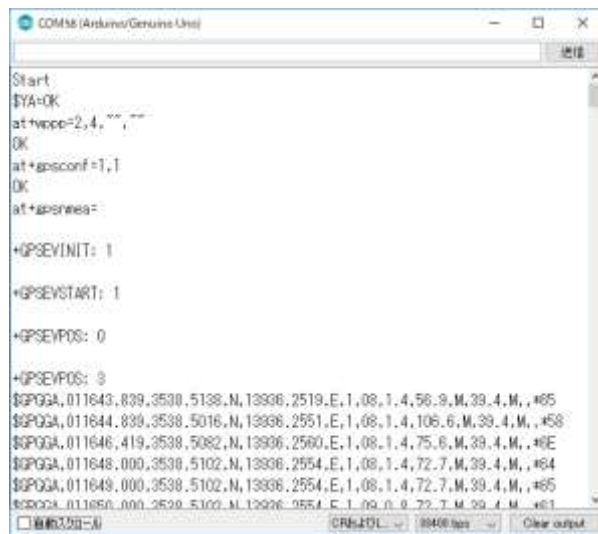
at+gpsconf=1,0 又は =1,1 のいずれかを設定してください。

(こちらは不揮発性メモリに保存)

また、NMEAの「GPGGA」のみを表示させる場合には、以下の通りにコマンドを設定してください。

at+gpsnmea=1,1,1

at+gpsstart=1



■ Arduinoでのサンプルスケッチ

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
#define BAUDRATE 38400

void setup() {
  Serial.begin(BAUDRATE);
  Serial3G.begin(BAUDRATE);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  Serial.println("Start");
  while (Serial3G.readStringUntil('\n').indexOf("3GIM") < 0);
  Serial3G.println("$YA");
  delay(5);
  Serial3G.println("at+wdpp=2,4,¥¥¥,¥¥¥");
  delay(10);
  Serial3G.println("at+gpsconf=1,1");
  delay(10);
  Serial3G.println("at+gpsnmea=1,1,1");
  delay(10);
  Serial3G.println("at+gpsstart=1");
  delay(10);
}

void loop() {
  String str=Serial3G.readStringUntil('\n');
  if(!str.indexOf("")) Serial.println(str);
}
```



3. Web関連

1. HTTP GET ①

■ HTTP GETの実行

項目	値など	説明	補足
機能分類	Web		
機能名	GET	HTTP/GETを指定されたURLへ送信して、レスポンスを取得する	ボディ部のみ取得できる
コマンド形式		\$WG url ["header"]¥n	カギ括弧 [] は、実際は不要
引数	url	GETリクエストを送信するURL（例えば、"https://www.arduino.cc/"等）	URLエンコードされていること 先頭に"http://"または"https://"を含むこと
	header	ヘッダ情報（例えば、"Authorization: Basic QWxhZGRpbGl2FtZQ=="等）	\$エンコードされていること。 Hostプロパティは省略可
応答値	【正常時】	\$WG=OK nbytes¥nresponse¥n	
	nbytes	レスポンス文字列のバイト数（末尾の'¥n'は含まず）	最大1023
	response	レスポンスの文字列	バイナリデータも取得可能
	【エラー時】	\$WG=NG errno¥n	
	errno	301：コマンド形式または引数指定エラー（urlの指定間違いも含む） 303～308：タイムアウトまたは内部エラー マイナス値：HTTPステータスコードの符号をマイナスにした値	タイムアウトは30秒設定 値の範囲は-400～-599
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しい事
補足事項	①	レスポンスにはヘッダ情報は含まれず、ボディ情報のみが含まれる。	
	②	レスポンスの文字コードは、urlで指定されたサーバに依存する。	
	③	HTTPのバージョンは「1.1」としてGETまたはPOSTする。	
	④	ヘッダにはUserAgentプロパティは含まれない。	
	⑤	本コマンドでは、レスポンスボディが大きい場合は、先頭の一部しか取得できない。ボディが大きい場合でもすべてを取得したい場合には、TCP機能を利用する。	HTTP/GETのレスポンスボディが大きい場合、本コマンドの実行には最大35秒程度の時間がかかる
	⑥	urlの長さとはheaderの長さを合わせて、最大1024バイトまでとする。ただし、urlに含まれるホスト名は最大96バイトまでとする。	

1. HTTP GET ②

■ 事例：ネット接続サンプルプログラム

■ Arduinoサンプルプログラム

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(38400);
  Serial3G.begin(38400);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  Serial.println("Start");
  while (Serial3G.readStringUntil('\n').indexOf("3GIM") < 0);
  Serial.println("begin HTTP GET");
  Serial3G.println("$WG http://tabrain.jp/demo/httpGET_test.txt");
  delay(1000);
  unsigned long tm = millis();
  while (millis() - tm < 35000) {
    while (Serial3G.available()) {
      char c = Serial3G.read();
      Serial.print(c);
    }
  }
  Serial.print("\r\nend");
}
void loop() {}
```

スケッチ名 : http_get.ino

■ シリアルモニタ画面（正常時応答）

```
begin HTTP GET
$WG=OK 44
Tabrain Web site
Complete access from 3GIM

end
```

■ www.tabrain.jp/demo/httpGET_test.txtのファイル内容

サンプルデータ

Tabrain Web site
Complete access from 3GIM

URLコードの変換が必要な文字（5文字）

文字		!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
コード	%20	%21	%22	%23	%24	%25	%26	%27	%28	%29	%2A	%2B	%2C	%2D	%2E	%2F	
文字	:	;	<	=	>	?	@	[¥]	^	_	`	{		}	~
コード	%3A	%3B	%3C	%3D	%3E	%3F	%40	%5B	%5C	%5D	%5E	%5F	%60	%7B	%7C	%7D	%7E

2. HTTP POST ①

■ HTTP POSTの実行

項目	値など	説明	補足
機能分類	Web		
機能名	POST	HTTP/POSTを指定されたURLへ送信して、レスポンスを取得する	
コマンド形式		\$WP url "body" ["header"]¥n	カギ括弧 [] は、実際は不要
引数	url	POSTリクエストを送信するURL	最大256バイト（\$エンコードされていること）
	body	POSTするボディ	最大1024バイト（"）
	header	ヘッダ情報	最大256バイト（"）、省略可
応答値	【正常時】	\$WP=OK nbytes¥nresponse¥n	
	nbytes	レスポンス文字列のバイト数（デコード前のサイズ）	最大1023バイト
	response	レスポンスの文字列（エンコードされた文字列）	バイナリデータも取得可能
	【エラー時】	\$WP=NG errno ..¥n	
	errno※	301：コマンド形式または引数指定エラー（urlの指定間違いも含む） 303～308：タイムアウトまたは内部エラー マイナス値：HTTPステータスコードの符号をマイナスにした値	値の範囲は-400～-599
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	全ての引数の長さの合計は改行'¥n'を含み1088バイト以下であること。	
補足事項	①	レスポンスにはヘッダ部は含まれず、ボディ部のみが含まれる。	
	②	レスポンスの文字コードは、urlで指定されたサーバの実装に依存する。	
	③～⑥	HTTP GETの補足事項を参照のこと	

※「errono」は、WiKiページなどで補足説明しています。

2. HTTP POST ②

■ 事例 : HTTP POSTによるツイート参照

スケッチ名 : twitter_sample.ino

本事例は、後述しています
「[3GIMでのツイッター連携使用例](#)」
を参考にしてください。

■ 本関数は、\$WG または \$WPを含んだコマンド文字列を引数として送る関数

```
//===== $WG & $WP command =====  
boolean _3G_WGP(String command) {  
  Serial3G.println(command);  
  String rstr;  
  unsigned long tim = millis(); // time set(ms)  
  do{  
    while(!Serial3G.available ());  
    rstr=Serial3G.readStringUntil('\n');  
  } while (rstr.indexOf("$W") != 0 && (millis() - tim) < LIMITTIME);  
  return (rstr.indexOf("$W") == 0);  
}  
//=====
```

ここで Serial3G は、シリアル通信ポート
LIMITTIMEは、制限時間 (14000msec) を設定のこと



4. TCP/IP関連

1. TCP/IP READ

■ コネクションからのデータ読み込み

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	READ	現在のコネクションからデータを読み出す	ノンブロッキングで動作する
コマンド形式		\$TR maxbytes¥n	
引数	maxbytes	読み出すデータの最大長 (バイト)	最大1024
応答値	【正常時】	\$TR=OK nbytes¥ndata¥n	
	nbytes	読み出したデータのバイト数 (≤maxbytes) 、このバイト数には末尾の¥nは含まない	指定バイト数以下の場合もある
	data	読み出したデータ	gw3g R2.0からバイナリデータも取扱可
	【エラー時】	\$TR=NG errno ..¥n	
	errno	631 : コマンドの指定、または引数に誤りがある	
		633 : READエラー	
		635 : コネクションがない (未接続)	
		636、637 : コネクションのステータスのエラー	
		639 : タイムアウトエラー	タイムアウト時間は60秒 (\$TXコマンドで変更可能)
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
補足事項	①	相手から受信したデータをそのまま加工せずに取得する	
	②	呼び出された時に3GIMに届いているデータを、最大msxbytes分まで読み出す。 常にブロッキングはせず、データがない時は nbytes=0 で直ちに 戻る。	常にノンブロッキングで動作する
	③	読み出す前にコネクションの状態チェックを行うため、コネクションが切断されている場合には直ちに制御が戻る。	

2. TCP/IP WRITE

■ コネクションへのデータ書き出し

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	WRITE	現在のコネクションヘデータを書き出す	
コマンド形式		\$TW "data"¥n	ダブルクォートを省略することもできるが推奨しない。
引数	data	書き出すデータ	最大1080バイトまで。ダブルクォートで囲む場合は\$エンコードされていること。
応答値	【正常時】	\$TW=OK nbytes¥n	
	nbytes	書き出したデータのバイト数（実際に相手に書き出したサイズ）	最大1024バイト
	【エラー時】	\$TW=NG errno ..¥n	
	errno	621：コマンドの指定、または引数に誤りがある	
		623：WRITEエラー	
625：コネクションがない（未接続）			
629：タイムアウトエラー		タイムアウト時間は60	
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
補足事項	①	dataとして指定できるデータは\$エスケープシーケンスにてエンコードされている必要がある。	バイナリデータを送りたい場合は、最低限 0x00(ヌル)、0x0a(LF)、0x22(")、0x24(\$)の4つのバイト値を\$でエスケープすればよい。
	②	dataとして指定できるデータは、エンコード前の生データのサイズが1024バイト以下であること。ただし、指定できるデータdataの長さは、コマンド文字列やダブルクォート等を含みコマンド行の最大長(改行を含み1088バイト)以下であること。	
	③	書き出す前にコネクションの状態チェックは行わない。そのため、相手方がコネクションを切断している場合は、タイムアウト時間が経過した後にエラーとなって制御が戻る。	性能を優先するために、コネクションの存在チェックを毎回行わない仕様としている。

3. TCP/IP CONNECT

■コネクション接続

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	CONNECT	TCP/IPコネクションを接続する	
コマンド形式		\$TC host_or_ip port¥n	
引数	host_or_ip	接続するホスト名またはIPアドレス	
	port	接続するポート番号	1～65535の範囲
応答値	【正常時】	\$TC=OK¥n	
	【エラー時】	\$TC=NG errno ..¥n	
	errno	601：引数がおかしい	
		603：すでに接続済み	
		604：コネクションエラー(ホストやポートが間違っている場合を含む)	タイムアウト時間は60秒（\$TXコマンドで変更可能）
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
補足事項	①	TCP/IPコネクションは一度に一つだけ使用できる。コネクションはWeb機能とは独立しているため、Web機能と同時に利用することができる。	例えば、TCP機能であるサーバとコネクションをつないだまま、Web機能を利用できる

4. TCP/IP DISCONNECT

■コネクション切断

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	DISCONNECT	現在のTCP/IPコネクションを切断する	
コマンド形式		\$TD¥n	
引数			
応答値	【正常時】	\$TD=OK¥n	
	【エラー時】	\$TD=NG errno ..¥n	
	errno	611 : コマンドの形式に誤りがある	
		614 : 内部エラー(Close)	
		615 : 接続されていない	
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
	③	read中あるいはwrite中ではないこと	
補足事項			

5. TCP/IP STATUS

■ コネクション状態の取得および状態設定

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	STATUS	現在のTCPコネクションの状態を取得する	
コマンド形式	取得	\$TS [option]¥n	
引数	option	1 : 付加情報を出力する	省略時は詳細情報は出力しない
応答値	【正常時】	\$TS=OK status¥n \$TS=OK status tcpnotif remainedBytes receivedBytes¥n	option省略時 option=1の時
	status	0 : CLOSED (接続なし) 1 : DISCONNECTING 2 : DISCONNECTED (接続待ち) 3 : CONNECTING 4 : CONNECTED (送受信待ち)	
	tcpnotif	下記の【TCP機能全般の留意点】を参照	オプション(1)が指定された時のみ出力される
	remainedBytes	未送信状態のデータのバイト数	同上
	receivedBytes	受信状態のデータのバイト数 (\$TRコマンドでの読み出し可能なバイト数)	同上
	【エラー時】	\$TS=NG errno ..¥n	
	errno	641 : 引数がおかしい 642 : ステータス取得エラー	
前提条件			
補足事項	①	本コマンド実行時にTCPコネクションの状態を取得して結果を返却するため、正確な最新情報が得られる。	

6. TCP/IP GETSOCKNAME

■ コネクション状態のIPアドレスとポート番号取得

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	Get Sockname	自分のIPアドレスを取得する	ポート番号は取得できない
コマンド形式	取得	\$TN¥n	
引数			
応答値	【正常時】	\$TN=OK ipAddr ¥n	
	ipAddr	自分のIPアドレス(IP v4のみサポート)	
	【エラー時】	\$TN=NG errno ..¥n	
	errno	663 : コマンドの形式に誤りがある 662 : 接続していない 661 : IPアドレス取得エラー	
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
補足事項			

7. TCP/IP TUNNEL WRITE

■現在のコネクションへのデータの直接書き出し

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	TUNNEL WRITE	現在のコネクションへダイレクトにデータを書き出す	サイズの大きいバイナリデータをサーバへ送信する手段として最適である。
コマンド形式		\$TT nbytes¥ndata	dataの後に改行は不要である
引数	nbytes	書き出すデータのバイト数	最大32000バイト
	data	書き出すデータ	\$エンコードは不要、バイナリデータもそのままOK
返却値	【正常時】	\$TT=OK nbytes¥n	
	nbytes	書き出したデータのバイト数	
	【エラー時】	\$TT=NG errno ..¥n	
	errno	621 : コマンドの指定、または引数に誤りがある 623 : WRITEエラー 625 : コネクションがない（未接続）	
前提条件	①	パケット通信サービスが利用できる状態であること。	プロファイル設定が正しいこと
	②	TCP/IPコネクションが確立されていること	
補足事項	①	dataとして指定できるデータの内容はバイナリデータでもよい。\$エンコードの必要はなく、ヌルデータ(0x00)を含めてそのまま書き出すことができる。	
	②	最初に指定したデータサイズ(バイト数)分を必ず書き出す必要がある。書き出すデータのサイズが最初に指定したサイズに満たない場合は、タイムアウト後に半角スペースが自動で補填される。	
	③	書き出す前にコネクションの状態チェックは行わない。そのため、相手方がコネクションを切断している場合は、タイムアウト時間が経過した後にエラーとなって制御に戻る。	性能を優先するために、コネクションの存在チェックを毎回行わない仕様としている。
	④	ボーレートを115200bpsに設定している場合に限り、概ね1024バイトを送信するたびに10ミリ秒程度のディレイ時間を設けることを推奨する。	

8. SET/GET PARAMETERS

■ TCP/IP機能のパラメータの設定・取得

項目	値など	説明	補足
機能分類	TCP/IP		
機能名	Control	TCP/IP関連コマンドでのタイムアウト時間を取得・設定する。	
コマンド形式	パラメータ取得	\$TX¥n	
	パラメータ設定	\$TX timeout¥n	
引数	timeout	設定するタイムアウト時間（10ミリ秒単位、0～6000の範囲）	デフォルトは6000
応答値	【正常時】	\$TX=OK¥n \$TX=OK timeout¥n	
	timeout	取得したタイムアウト時間（10ミリ秒単位）	
	【エラー時】	\$TX=NG errno¥n	
	errno	671：コマンド形式がおかしい 672：指定されたタイムアウト時間がおかしい	
前提条件			
補足事項	①	本コマンドで設定したタイムアウト時間は、3GIMの電源をOFFにした時やリセットした時に、デフォルトの時間（6000=60秒間）に戻る。	
	②	極端に短いタイムアウト時間に設定することは推奨しない。TCP/IPネットワークの世界では、ネットワーク上でデータをやり取りする関係で、本質的にエラーを検出するためには一定の時間（通常は数秒から30秒程度で、ネットワークや3Gの電波状態に大きく依存する）が掛かる。この時間よりも短いタイムアウトを設定した場合、例えば\$TCコマンドからエラーが返ってきた後に、正常に接続できた状態となったりする等の挙動があり得る。従って、設定するタイムアウト時間は、いろいろと試してみて具体的な値を決めるといった手順を踏むことを推奨する。	

9. TCP/IP 利用サンプルプログラム ①

■ 事例 : TCP/IP関連一覧のコマンドを使ったサンプルプログラム

スケッチ名 : tcp_ip.ino

■ Arduinoサンプルプログラム

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(38400);
  Serial3G.begin(38400);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  Serial.println("Ready...");
  while (Serial3G.readStringUntil('\n').indexOf("3GIM") < 0);
  Serial.println("begin TCP/IP sample");
  delay(100);
  Serial3G.println("$TC www.tabrain.jp 80");
  Serial3G.println("$TW ¥"GET / HTTP/1.1$r$n¥");
  Serial3G.println("$TW ¥"HOST: www.tabrain.jp$r$n$r$n¥");
  tcpip("$TR 500");
  Serial3G.println("$TD");
  Serial.print("¥r¥nend");
}

void loop() {}
```

```
String tcpip(String cmd ) {
  Serial.print(cmd + " -> ");
  Serial3G.println(cmd);
  String str;
  do{
    str = Serial3G.readStringUntil('\n');
    if(str.length()>0 )Serial.println(str);
  }
  while( str.indexOf(cmd.substring(0,3))<0);
  int n = str.substring(6).toInt();
  if(cmd.indexOf("TR")>0 && n>0) {
    for(int i=0; i<n ; i++) {
      while(!Serial3G.available());
      Serial.write(Serial3G.read());
    }
  }
}
```

10. TCP/IP 利用サンプルプログラム ②

■ 事例：TCP/IP関連一覧のコマンドを使ったサンプルプログラムの出力結果

■ シリアルモニタ画面（正常時応答）

```
Ready...
begin TCP/IP sample
$TR 500 -> $TC=OK
$TW=OK 16
$TW=OK 24
$TR=OK 251
HTTP/1.1 200 OK
Date: Sun, 13 Aug 2017 03:37:08 GMT
Server: Apache
Last-Modified: Sat, 20 May 2017 00:44:24 GMT
Accept-Ranges: bytes
Content-Length: 66
Content-Type: text/html

<meta http-equiv="refresh"
content="1;URL=http://tabrain.jp/new/">
end
```

読み込みバッファの出力結果
(ここでは251バイト出力)

11. TCP/IP 機能の留意点

【TCP機能全般の留意点】		
1)	\$TR/\$TW/\$TT/\$TN/\$TSでコマンド形式エラー以外のエラーが発生した場合は、一旦、\$TDでコネクションを切断して、\$TCからやり直すこと。	
2)	\$TSコマンドで取得できるtcpsnotifの値の意味は下記の通り：	
		<ul style="list-style-type: none">0 Network error1 No more sockets available; max. number already reached2 Memory problem3 DNS error4 TCP disconnection by the server or remote client5 TCP connection error6 Generic error7 Fail to accept client request's8 Data sending is OK but KTCPSND was waiting more or less characters9 Bad session ID10 Session is already running11 All sessions are used
3)	\$TR/\$TCコマンドのタイムアウト時間は、デフォルトで60秒である。この時間は、\$TXコマンドによって変更することができる。 また同様に、\$TW/\$TTコマンドのエラー発生時のタイムアウト時間は、デフォルトで60秒である。この時間も、\$TXコマンドで変更することができる。（このタイムアウト時間は\$TR/\$TCのタイムアウト時間と共通であるため、片方だけを変更することはできない）	

5. Profile関連

1. PROFILE SET/GET①

■SIMカードのプロファイル情報の設定・取得

項目	値など	説明	補足
機能分類	PROFILE		
機能名	SET	デフォルトのプロファイル情報を設定・取得する	
コマンド形式	設定	\$PS "apn" "user" "password"¥n	※
	取得	\$PS¥n	
引数	apn	APN情報（例えば：iijmio.jp）	
	user	ユーザ名（例えば：mio@iij）	
	password	認証用パスワード（例えば：iij）	
応答値	【正常時】	\$PS=OK¥n	設定時
		\$PS=OK "apn","user","password"¥n	取得時
	【エラー時】 errno	\$PS=NG errno¥n	
		211：コマンドの形式に誤りがある 212：内部エラーまたはSIMカード・アンテナなし	
前提条件	①	本コマンドの実行にあたっては、有効なSIMカード（利用可能なSIMカード）と3Gアンテナを装着しておく必要がある。	
補足事項	①	apnの長さ、userの長さ、passwordの長さの合計は、最大51文字まで	
	②	デフォルトプロファイルは、一つだけ保持することができる。本機能で設定したプロファイルは、電源を切っても3GIM内に保持される。	
	③	出荷時のデフォルト状態では、iijmioのプロファイルが設定されている。	

※：例えば「**\$PS iijmio.jp mio@iij iij¥n**」と設定する（引数をダブルクォートで囲む必要はないが、囲んでもよい）
ユーザ名やパスワードが無いSIMカードもあるが、その場合には適当な文字列を指定すること（引数を省略することはできない）。

補足：SIMカードのプロファイル情報は、内部メモリに保存されますので、電源を切っても保存されたままとなります。

1. PROFILE SET/GET②

■ SIMカードのプロファイル情報の設定

スケッチ名 : profile_set.ino

■ Arduinoサンプルプログラム

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
void setup() {
  Serial.begin(38400);
  Serial3G.begin(38400);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  Serial.println("begin Profile Set");
  while (Serial3G.readStringUntil('\n').indexOf("3GIM") < 0);
  String str;
  Serial3G.println("$PS iijmio.jp mio@iij iij");
  do { str = Serial3G.readStringUntil('\n');
  } while (str.indexOf("$PS") < 0);
  Serial.println(str);
  Serial3G.println("$PS");
  do { str = Serial3G.readStringUntil('\n');
  } while (str.indexOf("$PS") < 0);
  Serial.println(str);
  Serial.println("end");
}
void loop() {}
```

■ シリアルモニタ画面（正常終了の場合）

```
begin Profile Set
$PS=OK
$PS=OK "iijmio.jp","mio@iij","iij"
end
```

■ プロファイル設定サンプル（他MVNO製品も同様に設定）

SIMメーカー製品名	設定方法
DOCOMO mopera	\$PS mopera.net
iijmio	\$PS iijmio.jp mio@iij iij
iijmobile	\$PS iijmobile.jp mobile@iij iij
SONET	\$PS so-net.jp nuro nuro
SORACOM	\$PS soracom.io "" ""
EXCITE	\$PS vmobile.jp bb@excite.co.jp excite
HI-HO	\$PS vmobile.jp lte@hi-ho hi-ho
BMOBILE	\$PS bmobile.ne.jp bmobile@u300 bmobile
DTI	\$PS dream.jp user@dream.jp dti
MMT	\$PS mmtcom.jp mmt@mmt mmt

もくじ

1. シリアルモニタ画面との操作
2. 起動とコマンド操作



第3章 コマンド・サンプルスケッチ

1. シリアルモニタ画面との操作

本サンプルプログラムでは、シリアルモニタ画面でのキー入力とモニタ出力を行うことで、UART \$ コマンドの動きを確認することができる。

【サンプルスケッチ】

```
#define BAUDLATE 9600UL
```

```
void setup() {  
  Serial.begin(BAUDLATE);  
  Serial1.begin(BAUDLATE);  
  pinMode(7,OUTPUT);  
  digitalWrite(7,HIGH);  
  delay(500);  
  digitalWrite(7,LOW);  
  Serial.println("Ready.");  
}
```

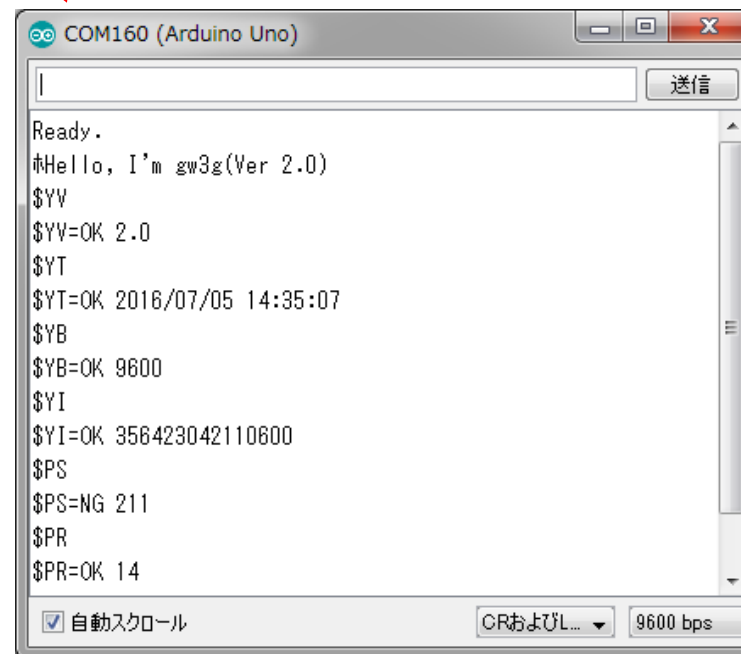
```
void loop() {  
  if (Serial1.available() > 0) {  
    char c = Serial1.read();  
    Serial.print(c);  
  }
```

3 GIM
からの読み込み

```
  if (Serial.available() > 0) {  
    char c = Serial.read();  
    Serial.print(c); // Echo back  
    Serial1.print(c);  
  }  
}
```

3 GIM
への書き込み

monitor_3GIM_G101.ino



2. 起動とコマンド操作

\$コマンドを起動し、インターネット接続を行うまでのスケッチを紹介

【サンプルスケッチ】

```
#define _3GIM "3GIM"
#define baudrate 9600UL

void setup() {
  Serial.begin(baudrate);
  Serial1.begin(baudrate);
  Serial.println("Ready");
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH);
  delay(100);
  digitalWrite(7, LOW);
  unsigned long tim=millis();
  String str;
  do{ str=Serial1.readStringUntil('\n');
    if(millis()-tim>15000) break;
  }while(str.indexOf(_3GIM)<0);
  if(str.indexOf(_3GIM)<0) {
    Serial.println("connect error"); while(1);
  }
  Serial.println("start");
  Serial1.println("$WG http://tabrain.jp");
  do{ while(!Serial1.available());
    str=Serial1.readStringUntil('\n');
  }while(str.indexOf("$WG")<0);
  int n=str.substring(7).toInt();
  for(int i=0; i<n; i++) {
    while(!Serial1.available());
    Serial.write(Serial1.read());
  }
}

void loop() {}
```

3 GIM電源Off/On

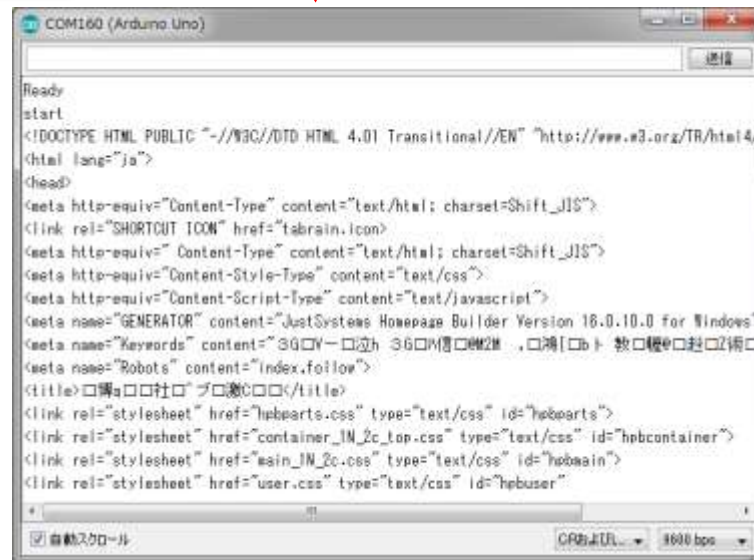
a3gs.start() に相当

a3gs.begin() に相当

初期応答確認

MGコマンド処理

a3gs.httpGET() に相当



第4章 ライブラリ概説

ライブラリa3gim、a3gim2について紹介します。

1. a3gimライブラリとは

1. ライブラリを利用するハードウェア

- ▶ Arduinoから3GIMを簡単に利用できるようにするために、ライブラリ**a3gim**が提供されています。
 - ▶ **a3gim**ライブラリは、Arduino UNO/Pro等でソフトウェアシリアルを使って3GIMを利用する際に使用します。
 - ▶ **a3gim2**ライブラリは、Arduino Mega/Due/M0 Pro/Leonardo/101/Zero等でハードウェアシリアル3GIMを利用する際に使用します。提供される機能は、a3gimと同等です。
 - ▶ 以下では、a3gim/a3gim2を総称してa3gimと呼びます。
 - ▶ 3 GIM V2.2では、a3gim および a3gim2ともに バージョン R4.3を使用してください。
- ▶ このライブラリを利用することで、関数の呼び出しという形で3GIMの各機能を利用することができます。
- ▶ 本ライブラリ群は、タブレイン製の「IoTABシールド」または「3GIMシールド」を利用し、Arduino UNO、Zero(M0) Pro、Genuino101、ArduinoMEGAなどの上で稼働できます。
- ▶ このa3gimはCPUアーキテクチャに依存しないライブラリであるため、Arduino互換機のMCUがAVRでもARMでも利用可能となっています。



IoTABシールドV3.0



3 GIMシールドV2.0

2. ライブラリ概要

- ▶ Arduinoから3GIMを簡単に利用できるようにするために、ライブラリ**a3gim**が提供されています。
 - ▶ **a3gim**ライブラリは、Arduino UNO/Pro等でソフトウェアシリアルで3GIMを利用する際に使います。
 - ▶ **a3gim2**ライブラリは、Arduino Mega/Due/Leonardo/101/Zero等でハードウェアシリアル3GIMを利用する際に使います。提供される機能は、a3gimと同等です。
 - ▶ 以下では、a3gim/a3gim2を総称してa3gimと呼びます。
- ▶ このライブラリを利用することで、関数の呼び出しという形で3GIMの各機能を利用することができます。



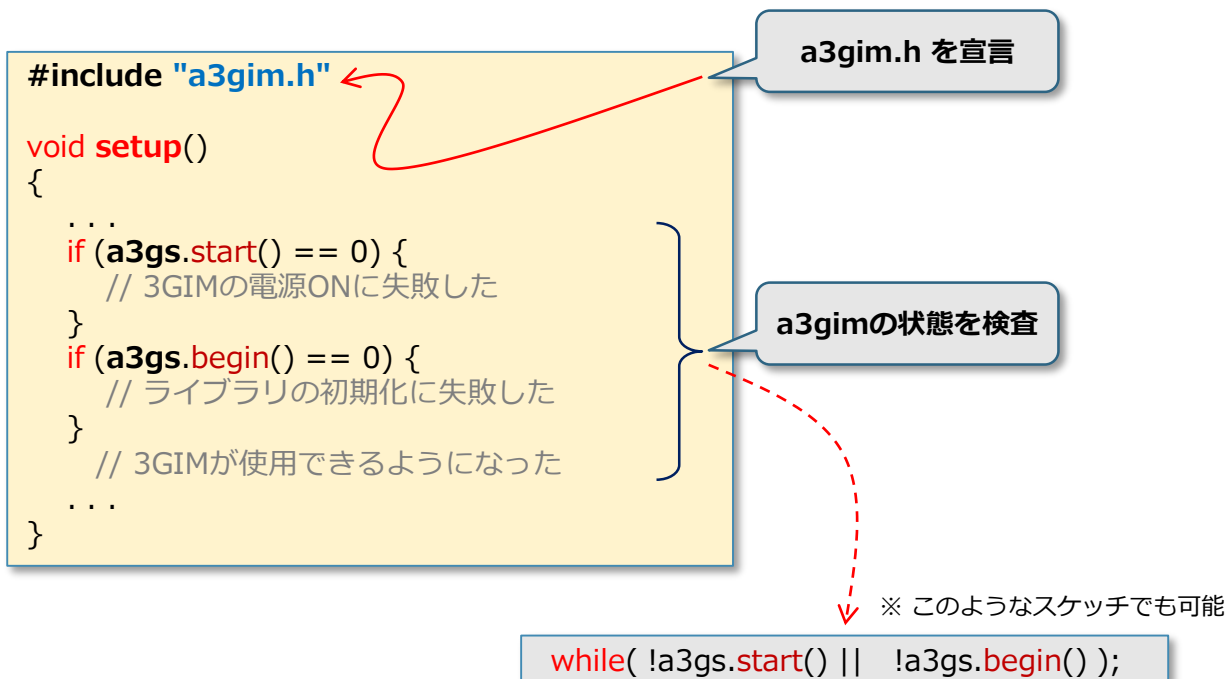
http://tabrain.jp/3GIM_V2.0/3GIM%20V2.0R01manual.pdf

http://tabrain.jp/3GIM_V2.0/a3gimR4.0manual.pdf

3. Arduino UNO 用ライブラリ概要

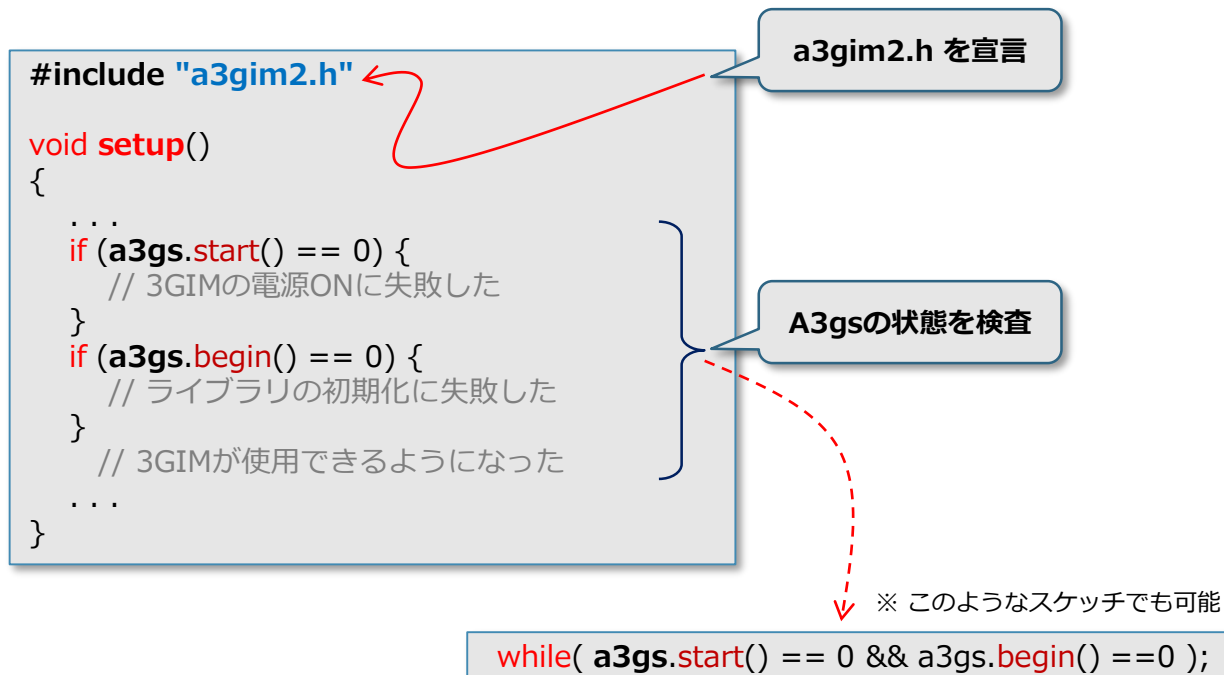
▶ a3gimライブラリを使用する方法

- ▶ ライブラリは、スケッチの中で以下の順序でコントロール用のメソッド（関数）を呼び出すことにより利用できます。



4. Genuino 101/Arduino MEGAライブラリ概要

- ▶ ライブラリa3gim2（Genuino101/ Arduino MEGAなど）を使用する方法
 - ▶ ライブラリは、スケッチの中で以下の順序でコントロール用のメソッド（関数）を呼び出すことにより、利用できるようになる：



補足説明：「a3gs」は、『a3gim2.h』の中で「Serial1」として定義してあります。
すなわちGenuino101のハードウェアシリアル通信のSerial1が割り当ててあります。

2. a3gimライブラリ概説

1. a3gimとは

a3gimとは、Arduinoやその互換ボード上で、3GIM V2.2を簡単に利用できるようにしたライブラリです。

これらのライブラリ群は、前述したさまざまな「\$ コマンド」の利用・設定を簡単な関数呼び出しで利用できるようにしたライブラリで、3GIMの詳細を知ることなく簡単に利用することができます。

このライブラリは、次ページ以降に記載する通り、ネット上からダウンロードし、Arduino IDE上にコピーして利用できるように環境を設定します。

※ a3gim2ライブラリも併せて提供しています。a3gimライブラリは主にUNO用となっており、3GIMとの通信には SoftwareSerialライブラリを使用します。一方、a3gim2ライブラリは、ハードウェアシリアル(Serial/Serial1/Serial2..)を使用する仕様となっており、MegaやDue、ZeroやM0、101等のハードウェアシリアルを複数持つArduinoで使用することを想定しています。両方のライブラリは、常に同時に同じバージョンで公開していく予定です。

注意：3 GIM V2.2では、a3gim のバージョンは、R4.3 となります。

a3gim Version	3 GIM V2.2のライブラリが利用できるボード	内部電圧
a3gim R4.3	Arduino UNO R3	5V系
a3gim2 R4.3	Arduino Mega,Leonardo,Zero Pro, Genuino101ほか	3.3V系又は5V系

2. ライブラリa3gim.zipのダウンロード

1) このZIPファイルを、以下のサイトから**ダウンロード**してください

<https://github.com/openwireless/3gim>

2) このzipファイルを、以下のいずれかのフォルダーに解凍してください。

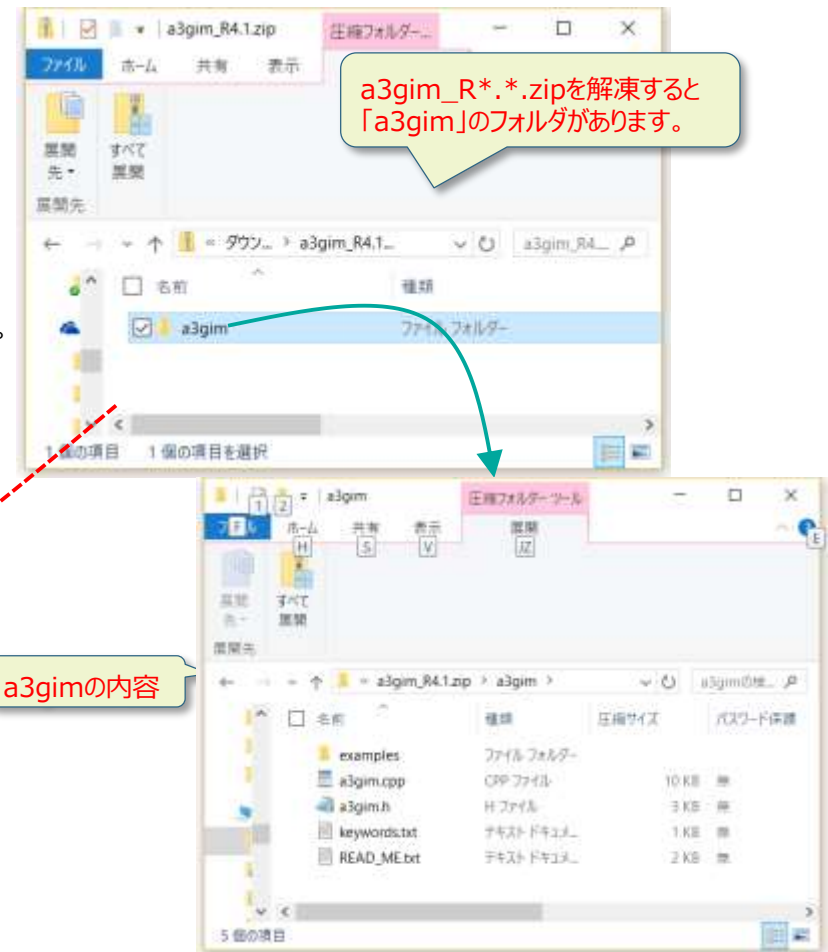
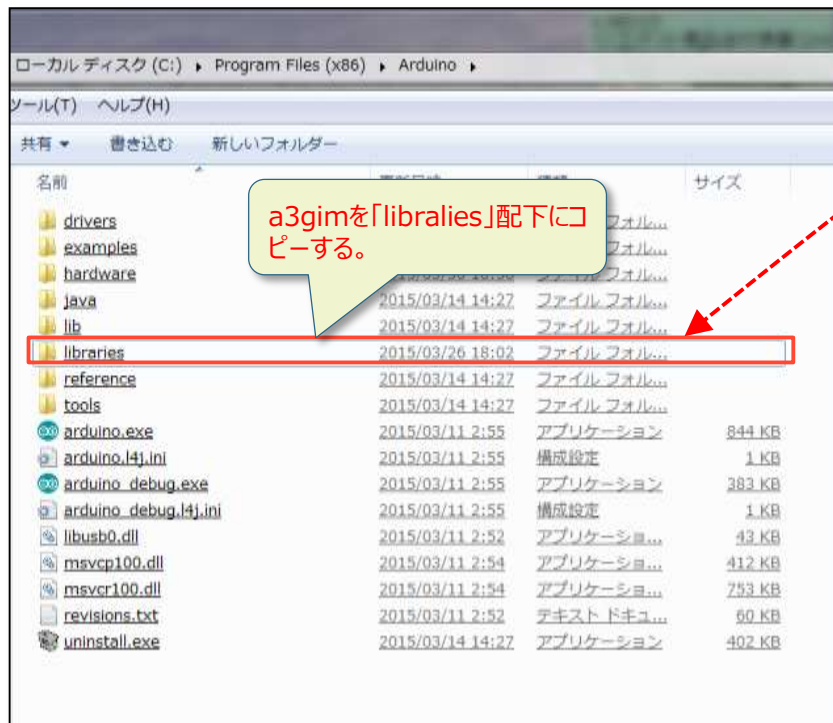
① Arduino IDE環境下の「`..¥libraries`」配下

② スケッチブックの保存場所（環境設定）の「`..¥libraries`」配下

コピーした後、Arduino IDEを起動すると、

メニュー「ファイル」⇒「スケッチの例」に、「a3gim」が表示されます。

「a3gim2」ライブラリのインストールも、必要に応じて同様に行ってください。



ダウンロードURL：

下記のページから最新版をダウンロードしてください。

<https://github.com/openwireless/3gim>

3. a3gimライブラリを利用する方法

▶ 概要

- ▶ 提供するa3gimライブラリ機能は、以前の3GIMの提供ライブラリとほぼ同等です。
- ▶ そのため、Arduinoと3GIMとの接続を工夫することで、3GIM用の下記のライブラリを使用することができます。
＜本バージョンR4.3から「3GIM用ライブラリ」と呼びます＞

- ▶ a3gim UNO/Pro用（SoftwareSerialを使用）：3GIM専用に改訂
- ▶ a3gim2 Mega/M0/Zero/101用（Serial1を使用）：3GIM専用に改訂

▶ 互いのライブラリの違い

- ▶ ヘッダファイル(デフォルトのボーレートの違い)
 - ▶ a3gim.hのシンボル a3gsBAUDRATE の定義は、「9600」となっています。
 - ▶ a3gim2.hのシンボル a3gsBAUDRATEの定義は、「9600」となっています。

【注意】3GIMの出荷時は、
9600bpsとしています。

▶ 3GIMとArduinoとの接続方法（例）

- ▶ UNOの場合
 - #6をGND、#4を5V、#3をD4、#2をD5、#1を開放(何も接続しない：常時電源ON)、に接続する
- ▶ Mega/Dueの場合（ハードウェアシリアル通信利用）
 - #6をGND、#4を5V、#3をRX3、#2をTX3、#1を開放(何も接続しない：常時電源ON)、に接続する
- ▶ Leonardoの場合（ハードウェアシリアル通信利用）
 - #6をGND、#4を5V、#3をRX1、#2をTX1、#1を開放(何も接続しない：常時電源ON)、に接続する
- ▶ Zero (M0) Pro/Genuino101の場合（ソフトウェアシリアル通信利用）
 - #6をGND、#4、#5をV3.3に、#2をD0、#3をD1、#1を開発（何も接続しない：常時電源ON）、に接続する。

【補足】ハードウェアシリアル
利用のため高速設定可能

【補足】3GIMのボーレートを
57600bpsまで高速設定可能
※環境が良ければ115200pbsも可能

4. A3gim R4.3 ライブラリー一覧表

分類	メソッド名	機能概要	補足
コントロール (Control)	begin※	ライブラリの初期化	
	end※	ライブラリの終了	
	restart※	3GIMのリセット	
	start※	3GIMの電源ON	
	shutdown※	3GIMの電源OFF	
	getServices	現在使用できる通信サービス	
	getIMEI	IMEI-IDの取得	
	setBaudrate	UARTの通信速度の設定	初期は9600bps
	setLED	LED1のON/OFF	
	setAirplaneMode	エアプレーンモードのON/OFF	
インターネット関係 (Web)	httpGET※	GETメソッドの要求	http/https利用可能
	httpPOST	POSTメソッドの要求	同上
	tweet※	Twitterへの投稿	
通信機能その他	getRSSI	電波強度の取得	
	getTime	現在時刻の取得	日付・時刻形式
	getTime2	現在時刻の取得	通算秒形式
	getVersion	3GIMのバージョンの取得	
	getResult	3GIMからの応答値受信	
	sendCommand	3GIMへのコマンド送信	
	sendData	3GIMへのデータ送信	
	enterAT	ATコマンドパススルーモード	*仕様変更
	discardUntil	3 GIMからの文字の受信	
現在位置取得 (GPS)	getLocation	現在位置の取得	緯度経度情報
	getLocation2	現在位置の取得 2	緯度経度他情報
	setLocationParams	GPS機能関連設定	*追加
その他ライブラリ	connectTCP※	TCPコネクションを接続	
	disconnectTCP※	TCPコネクションを切断	
	getStatusTCP	TCPコネクション最新状況取得	
	setTCPParams	TCPパラメータ設定	*追加
	writeBegin	シリアル通信で直接書き込み	
	read※	データの読み込み	2つのバージョン有
	write※	データの書き出し	3つのバージョン有
プロファイル	setDefaultProfile	デフォルトプロファイルを設定	※仕様変更
	getDefaultProfile	デフォルトプロファイルを取得	

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

* 追加 : V2.2で追加となったもの

* 仕様変更 : V2.2で以前のものから仕様変更されたもの

5. ライブラリ機能一覧 ライブラリが提供する関数(1/3)

分類	メソッド名※ ¹	機能概要	補足
コントロール (Control)	begin ※	ライブラリの初期化	
	end ※	ライブラリの終了	
	restart ※	3GIMのリセット	
	start ※	3GIMの電源ON	
	shutdown ※	3GIMの電源OFF	
	getServices	現在使用できる通信サービス	
	getIMEI	IMEI-IDの取得	
	setBaudrate	UARTの通信速度の設定	初期は9600bps
	setLED	LED1のON/OFF	
	setAirplaneMode	エアプレーンモードのON/OFF	

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数です。

5. ライブラリ機能一覧 ライブラリが提供する関数(2/3)

分類	メソッド名	機能概要	補足
Web機能	httpGET ※	GETメソッドの要求	http/httpsを利用可
	httpPOST	POSTメソッドの要求	同上
	tweet ※	Twitterへの投稿	*
現在位置取得 (GSP)	getLocation	現在位置の取得	緯度経度情報
	getLocation2	現在位置の取得 2	緯度経度他情報
	setLocationParams	GPS機能関連設定	※追加
通信機能その他	getRSSI	電波強度の取得	
	getTime	現在時刻の取得	日付・時刻形式
	getTime2	現在時刻の取得	通算秒形式
	getVersion	3GIMのバージョンの取得	
	getResult	3GIMからの応答値受信	
	sendCommand	3GIMへのコマンド送信	
	sendData	3GIMへのデータ送信	
	enterAT	ATコマンドパススルーモード	※仕様変更
	discardUntil	3 GIMからの文字の受信	

- ※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数
- * 無償サービス「<http://arduino-tweet.appspot.com/>」を利用（要登録）
- ※ 追加 : V2.2で追加となったもの
- ※ 仕様変更 : V2.2で以前のものから仕様変更されたもの

5. ライブラリ機能一覧 ライブラリが提供する関数(3/3)

分類	メソッド名	機能概要	補足
TCP/IP機能	connectTCP ※	TCPコネクションを接続	
	disconnectTCP ※	TCPコネクションを切断	
	getStatusTCP	TCPコネクション最新状況取得	
	setTCPParams	TCPパラメータ設定	※追加
	writeBegin	シリアル通信で直接書込み	
	read ※	データの読み込み	2つのバリエーション有
	write ※	データの書出し	3つのバリエーション有
プロファイル	setDefaultProfile	デフォルトプロファイルを設定	※仕様変更
	getDefaultProfile	デフォルトプロファイルを取得	

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

※ **追加** : V2.2で追加となったもの

※ **仕様変更** : V2.2で以前のものから仕様変更されたもの

6. ライブラリ定数一覧 ライブラリが定義している主な定数

- ▶ 各ライブラリを利用する上で、留意すべき定数（主に最大値の定義）を下表に示す。これらは、ヘッダファイル“a3gim.h”で定義されています。

分類	定数名	意味	設定	補足
Web	a3gimMAX_URL_LENGTH	URLの最大バイト数	256	※1
	a3gimMAX_HEADER_LENGTH	POSTのヘッダの最大バイト数	512	※1
	a3gimMAX_BODY_LENGTH	POSTのボディの最大バイト数	1024	※1
	a3gimMAX_RESULT_LENGTH	GET/POSTのレスポンスの取得可能な最大バイト数	192	※1
	a3gimMAX_TWEET_LENGTH	ツイートメッセージの最大バイト数	60	※1
TCP/IP	a3gimMAX_HOST_LENGTH	ホスト名の最大バイト数	96	※1
	a3gimMAX_DATA_LENGTH	一度に読み書きできるデータの最大バイト数	1024	※2

※1 これらの定数は、ATmega328*/32U*（Unoなど）を利用したArduinoではSRAMのサイズが小さいことからかなり制約が厳しい。ATmega2560/1280（Megaなど）またはADKを利用することで、これらの最大値を大きくすることができる。

※2 大きなデータを読み書きする場合は、複数回に分けてread/writeを実行する。

3. コントロール関連関数

コントロール関連の関数

▶ 提供関数ライブラリ

コントロール (Control)	begin ※	ライブラリの初期化	
	end ※	ライブラリの終了	
	restart ※	3GIMのリセット	
	start ※	3GIMの電源ON	
	shutdown ※	3GIMの電源OFF	
	getServices	現在使用できる通信サービス	
	getIMEI	IMEI-IDの取得	
	setBaudrate	UARTの通信速度の設定	初期は9600bps
	setLED	LED1のON/OFF	
	setAirplaneMode	エアプレーンモードのON/OFF	

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

▶ 概要

- ▶ a3gimライブラリの初期化・終了、ライブラリの状態の取得、3GIMシールドのリセット、電源ON/OFF、IMEIの取得、LED1のON/OFF、UARTの通信速度の設定を行う

▶ 留意点

- ▶ 電源ONには、15秒程度の時間が掛かる。リセットには、5秒程の時間が掛かる。
- ▶ 電源OFFには、1秒ほどの時間が掛かる
- ▶ **setBaudrate**による通信速度の変更には、十分留意すること

1. コントロール関連の関数 `begin`※

● バリエーション1

<code>int begin(char* pin)</code>	
機能概要	ライブラリを初期化する
引数	pin: 未使用(指定は不要)
戻り値	0: 正常に初期化を実行できた時
	1: エラーが発生した時(ライブラリは使用不可)
	2: IEM上のgw3gアプリのバージョンが古い(ライブラリは使用不可)
補足	<p>3GIMの電源がONの状態、本ライブラリの使用に先立って一度だけ本関数を呼び出す必要がある。</p> <p>初期化に失敗した場合は、<code>end()</code>関数を呼び出して、時間をおいて何度かリトライすることで成功する場合がある。</p> <p>終了関数<code>end()</code>を呼び出した後は、再度、本関数を呼び出すことができる。</p> <p>本関数の中では、デフォルトの通信速度で標準ライブラリ「SoftwareSerial」を初期化(<code>begin</code>)する。</p>

【使い方の例】

```
if (a3gs.begin() == 0)
  Serial.println("Succeeded.");
```

1. コントロール関連の関数 `begin`※

● バリエーション2

<code>int begin(char* pin, uint32_t baudrate)</code>	
機能概要	ライブラリを初期化する
引数	pin: 未使用(指定は不要)
	baudrate: 設定する通信速度 (1200/2400/4800/9600/19200/38400/57600/115200)
戻り値	0: 正常に初期化を実行できた時
	1: エラーが発生した時(ライブラリは使用不可)
	2: IEM上のgw3gアプリのバージョンが古い(ライブラリは使用不可)
補足	<p>3GIMの電源がONの状態で、本ライブラリの使用に先立って一度だけ本関数を呼び出す必要がある。</p> <p>初期化に失敗した場合は、<code>end()</code>関数を呼び出して、時間をおいて何度かリトライすることで成功する場合がある。</p> <p>終了関数<code>end()</code>を呼び出した後は、再度、本関数を呼び出すことができる。</p> <p>本関数の中では、指定された通信速度で標準ライブラリ「SoftwareSerial」を初期化(<code>begin</code>)する。</p> <p>通信速度の変更は、<code>setBaudrate()</code>関数を使って事前に行っておく必要がある。</p>

【使い方の例】

```
if (a3gs.begin(0, 9600) == 0)
  Serial.println("Succeeded.");
```

2. コントロール関連の関数 end※

int end(void)		
機能概要	ライブラリの使用を終了する	
引数	なし	
戻り値	0: 正常に終了処理を実行できた時	
	0以外: エラーが発生した時	
補足	本関数の中では、標準ライブラリであるSoftwareSerialを終了(end)する。	

【使い方の例】
・次頁参照

3. コントロール関連の関数 restart※

int restart(char* pin)		
機能概要	3GIMを再起動(リセット)する	
引数	pin: 未使用(指定は不要)	
戻り値	0: 正常にリセットを実行できた時	
	0以外: エラーが発生した時(リセットできない時)	
補足	IEM全体をリセットする。 通常、本関数を呼び出してから10秒程度でIEMはリセット処理を開始し、40秒程度で利用可能な状態となる。 本関数によるリセット後に再度ライブラリを利用する場合は、一旦、終了関数end()を呼び出した後に、初期化関数begin()を呼び出すこと。	

【使い方の例】

```
if (a3gs.restart() == 0) {  
  Serial.println("Restarting..");  
  a3gs.end();  
  if (a3gs.begin() == 0)  
    Serial.println("I'm OK.");  
}  
else  
  Serial.println("Restart Failed.");
```

4. コントロール関連の関数 start※

int start(char* pin)	
機能概要	3GIMの電源をONにする
引数	pin: 未使用(指定は不要)
戻り値	0: 正常に電源ONを実行できた時
	0以外: エラーが発生した時(電源ONできない時)
補足	本関数を呼び出しには、40秒程度掛かる(本関数の呼び出しが完了した時点で、3GIMが利用可能な状態となっている)。その後、初期化関数begin()を呼び出すことで本ライブラリを利用することができる。

【使い方の例】

```
if (a3gs.start() == 0 && a3gs.begin()) {  
    Serial.println("Succeeded.");  
    // 成功処理  
}  
else  
    Serial.println("Restart Failed.");
```

5. コントロール関連の関数 shutdown※

int shutdown(void)	
機能概要	3GIMの電源をOFFにする
引数	なし
戻り値	0: 正常に電源OFFを実行できた時
	0以外: エラーが発生した時(電源OFFできない時)
補足	本関数を呼び出しには、15秒程度掛かる 本関数を呼び出した後は、再度、電源ON関数start()を呼び出すことで3GIMを利用することができる。

【使い方の例】

```
a3gs.end();  
a3gs.shutdown();
```

6. コントロール関連の関数 getIMEI

int getIMEI(char* imei)	
機能概要	3GIMに装着されているIEMのIMEIを取得する
引数	imei : 取得したIMEI(サイズは a3gimIMEI_SIZE バイト)
戻り値	0: 正常に取得できた時
	0以外: エラーが発生した時(取得できない時)
補足	IMEIとは3G通信モジュール(IEM)の識別IDである(電話番号とは無関係) 引数imeiが指す結果格納場所のスペース(a3gimIMEI_SIZEバイト=16桁)は、あらかじめ呼び出し側で確保しておくこと。

【使い方の例】

```
char imei[a3gimIMEI_SIZE];  
if (a3gs.begin() == 0) {  
  a3gs.getIEI(imei);  
  Serial.println(imei);  
}
```

【出力結果例】

354563020267950

IMEI番号

このIMEIの中に、IEMモジュールに記載されたIDが含まれています。



7. コントロール関連の関数 setBaudrate

int setBaudrate(int baudrate)	
機能概要	Arduinoと3GIMを仲介するUARTの通信速度を設定する
引数	baudrate : 設定する通信速度(9600/19200/38400/57600/115200のいずれか)
戻り値	0: 正常に変更できた時
	0以外: エラーが発生した時
補足	<p>本関数の利用には十分留意すること。不適切な値を設定した場合は、3GIMと通信できなくなる。</p> <p>工場出荷時の通信速度は、安定動作が可能な 9600(bps) となっている。</p> <p>通信速度をデフォルトの設定値よりも高くするには、ハードウェアシリアルの利用を推奨する。</p> <p>本関数による通信速度の変更は、直ちに有効となる。</p> <p>デフォルトの通信速度と異なる通信速度を設定した場合は、次の初期化の際には通信速度を指定してbegin()を呼び出すこと(詳細はbegin()の項を参照)</p>

【使い方の例】

```
if (a3gs.setBaudrate(9600) == 0) {  
  Serial.println("Baudrate was changed.");  
  Serial.println("Please reset me now.");  
}
```

注意：設定した通信速度をスケッチ内で呼び出す必要があります。
a3gim.h のスケッチ内の「a3gimBAUDRATE」の設定となります。
間違った場合には、サンプルスケッチの「check_baudrate.ino」
で確認してみてください。

8. コントロール関連の関数 setLED

int setLED(boolean sw)	
機能概要	3GIMに搭載されているLED1を制御する
引数	sw : ONにする時はTRUE、OFFにする時はFALSEを指定する
戻り値	0: 正常に設定できた時
	0以外: エラーが発生した時
補足	LED1(緑色のLED)が3GIMのどこの位置に配置されているか等は、「取扱説明書」を参照のこと。

【使い方の例】

```
if (aFlag) {  
  a3gs.setLED(TRUE);  
  led1_status = TRUE;  
  Serial.println("LED1 is turned on.");  
}
```



このLEDが点灯

9. コントロール関連の関数 setAirplaneMode

int setAirplaneMode(boolean sw)	
機能概要	3GIMのエアプレーン(機内)モードを制御する
引数	sw : ONにする時はTRUE(=1)、OFFにする時はFALSE(=0)を指定する
戻り値	0: 正常に設定できた時
	0以外: エラーが発生した時
補足	エアプレーンモードがONの時は、3G通信は実行できないが、SMSの受信のみ可能である(ただし、受信したSMSの読み出しやSMSの送信はできない) エアプレーンモードをONにすることで、消費電力を大幅に節約することができる。 リセットまたは電源のOFF/ONで、デフォルトの設定(OFF)に戻る。

【使い方の例】

```
if (aFlag) {  
  a3gs.setAirplaneMode(true);  
  airplaneMode = true;  
  Serial.println("Airplane mode on");  
}
```

4. Web関連関数

5. Web関連の関数

▶ 提供関数ライブラリ

分類	メソッド名	機能概要	補足
Web機能	httpGET※	GETメソッドの要求	http/httpsを利用可
	httpPOST	POSTメソッドの要求	
	tweet※	Twitterへの投稿	*

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

* 無償サービス「<http://arduino-tweet.appspot.com/>」を利用（要Twitterの登録）

▶ 概要

- ▶ http/httpsを簡単に利用できる。
- ▶ GET/POSTメソッドを利用できる。
- ▶ Web機能の関数はすべて同期処理である。そのためレスポンスが取得できるまで、あるいは通信がタイムアウト(30秒程度)するまで呼び出し元には制御は戻らない。
- ▶ tweetはサードパーティのフリーサービスを利用することで使用できる（ユーザ登録が必要、利用条件はそのサービスに従う）。
 - ▶ 詳細は <http://arduino-tweet.appspot.com/> （タブレインとは直接関係のないサービスです）

▶ 留意点

- ▶ 使用するSIMカードで、3Gパケット通信が利用できること
- ▶ 通信料金（http/https通信の利用は、通常、定額プランの範囲内）に留意すること
- ▶ 日本語の取り扱いにはご注意ください。リクエストを送る相手サーバにより、日本語の文字コードが決まりますが、Arduinoでは日本語の処理を簡単に記述することができません。英語のみを取り扱うことを推奨します。

1. Web関連の関数 httpGET ※

int httpGET (const char* server, uint16_t port, const char* path, char* result, int resultlength, boolean ssled=false, const char* header=NULL)

機能概要	指定したサーバ、ポート、パスに対して、httpまたはhttps/GETリクエストを発行して、そのレスポンスを返却する	
引数	server : サーバのドメイン名またはIPアドレス	
	port : サーバのポート番号 (通常は80を指定)	
	path : URLのパス	
	result : [OUT] レスポンスの格納先 (スペースは呼び出し側で確保)	
	resultlength : resultのサイズを指定	
	ssled : httpsを利用する場合はtrue、httpを利用する場合はfalseを指定 (省略可能で、省略時はfalseと同じ)	
	header : 特殊なヘッダの指定 (最大a3gimMAX_HEADER_LENGTHバイト)	
戻り値	0 : 正常にGETできた時	
	0以外 : GETできなかった時	
補足	本関数の実行時間は、状況によって最大30秒程度掛かる。 サーバからのレスポンスのサイズが大きい場合は、resultlength以降のデータは破棄される。 resultは、'¥0'文字で終端される。文字コードは、接続先のサーバに依存する。 また、レスポンスにはヘッダは含まれない (ボディ部分のみ)	

1. Web関連の関数 httpGET ※

▶ 補足

- ▶ 特にヘッダの指定がない場合は、リクエストのヘッダは下記の通りである：

Host: **server:port**

- ▶ ヘッダを指定する場合は、下記のように指定する。2つ以上のヘッダを指定する場合は、それらの間を「`rn`」で区切り、末尾も「`rn`」で終わる（終端の改行は省略可能）。

```
char *header = "Authorization: Basic QWxhZGRpbjpvGVuIHNIc2FtZQ==$r$nX-Myheder: XYZ$r$n"
```

- ▶ path引数にはクエリー文字列を含めることができるが、事前にURLエンコードを施しておく必要がある。また、httpGET()関数では'\$'文字を特殊扱いするため、'\$'文字を文字列に含める時は、httpPOST()で解説しているようなエスケープシーケンスに従うこと。

2. Web関連の関数 httpPOST

int httpPOST (const char* server, uint16_t port, const char* path, const char* header, const char* body, char* result, int* resultlength, boolean ssled=false)		
機能概要	指定したサーバ、ポート、パスに対して、httpまたはhttps/POSTリクエストを発行して、そのレスポンスを返却する	
引数	server	サーバのドメイン名またはIPアドレス
	port	サーバのポート番号(通常は80を指定)
	path	URLのパス
	header	HTTPのヘッダ文字列(最大a3gimMAX_HEADER_LENGTHバイト)
	body	HTTPのボディ文字列(最大a3gimMAX_BODY_LENGTHバイト)
	result	[OUT] レスポンスの格納先(スペースは呼び出し側で確保)
	resultlength	[IN/OUT] resultのサイズを指定、呼び出し結果のサイズが返却
戻り値	ssled	httpsを利用する場合はtrue、httpを利用する場合はfalseを指定(省略可能で、省略時はfalseと同じ)
	0	正常にPOSTできた時
	0以外	POSTできなかった時

次ページ
へ続く

2. Web関連の関数 httpPOST

前ページ
から続く

```
int httpPOST (const char* server, uint16_t port, const char* path, const char* header, const char* body, char* result, int* resultlength)
```

補足

引数headerでは、HostとContent-Lengthの指定は不要である。
引数bodyでは、最後の空行は指定不要である。
本関数の実行時間は、状況によって最大30秒程度掛かる。
サーバからのレスポンスのサイズが大きい場合は、resultlength以降のデータは破棄される。
resultは'¥0'文字で終端される。文字コードは、接続先のサーバに依存する。
レスポンスには、ヘッダは含まれない(ボディ部分のみ)
引数headerおよびbodyでは、下記の'\$'文字を使ったエスケープシーケンスをサポートする。
直接制御文字を指定することはできないので注意すること:
\$t: TAB(0x09)
\$r: CR(0x0d)
\$n: NL(0x0a)
\$: "そのもの"
\$\$: "\$そのもの"
\$xhh または \$Xhh: 16進数hh(スケッチでの文字列における"0xhh"と同義)
引数headerやbodyでは、バイナリデータをそのまま取り扱うことはできない。また、レスポンスは文字列として返却するため、'¥0'文字を含むことはできない。バイナリデータを透過的に扱った通信を行いたい場合は、TCPIP関数を利用する。

3. Web関連の関数 tweet ※

int tweet (const char* token, const char* msg)	
機能概要	Twitterへ投稿する
引数	token : アクセスに必要なトークン(認証情報)
	msg : 投稿するメッセージ(最大a3gimMAX_TWEET_LENGTHバイト)
戻り値	0: 正常に投稿できた時
	0以外: 投稿できなかった時
補足	<p>tweetは、下記のフリーサービスを利用することで使用できる。ユーザ登録が必要で、利用条件はこのサービスに従う。 詳細は http://arduino-tweet.appspot.com/ を参照のこと。</p> <p>【注意】上記サービスの制限により、同一メッセージを連続して投稿することはできない。また、一定時間内に投稿できるメッセージ数に制限がある(2018/12時点の制限では、1分間に1回の投稿まで)。この制限を守らない場合は、正しく引数を指定している場合でも本関数はエラーを返却する。</p>

5. 位置情報取得（GPS）関連関数

現在位置取得（GPS）関連の関数

▶ 提供関数ライブラリ

分類	メソッド名	機能概要	補足
現在位置取得（GPS）機能	getLocation	現在位置の取得	内蔵GPSを使用
	getLocation2	現在位置の取得 2	
	setLocationParams	GPS関連機能設定	アシストGPSおよび アクティブGPSアンテナ

▶ 概要

- ▶ HL8548-G内蔵GPSや3Gネットワークを利用して位置を測位
- ▶ 引数の指定により、下記のいずれかの測位方法を選択できる（3 GIM V2.0以降は、区別なしで、任意文字対応）
 - ▶ **MPBASED**
 - GPSを利用して現在位置を測位する。GPSが利用できない場合は、3Gネットワークを利用する。
 - ▶ **MPASSISTED**
 - 3Gネットワーク上のロケーションサーバを利用して現在位置を測位する。
 - ▶ **MPSTANDALONE**
 - GPSのみを利用して現在位置を測位する。

▶ 留意点

- ▶ 通信サービス（例えば、IIJmio等）によっては、3Gネットワーク上のロケーションサーバ（Googleサーバ利用）を利用することができない。その場合は、GPS単独の測位のみが利用できる。
- ▶ 測位方法としてアシストGPSを利用する場合は、通信料金（通常、定額プランの範囲内だが、SIMカードの通信サービスによる）が発生する
- ▶ 屋内や都心等のように、上空にある衛星の電波がGPSアンテナで補足できない場所では、正しく測位できない場合がある。

1. 現在位置取得（GPS）関連の関数 getLocation

int getLocation(int method, char* latitude, char* longitude)	
機能概要	現在位置を取得する
引数	method : 測位方法 (a3gimMPBASED / a3gimMPASSISTED / a3gimMPSTANDALONE のいずれか) を指定 (現在任意文字対応)
	latitude : [OUT] 緯度(北緯) dd.dddddd形式、ただし桁数は場合により可変
	longitude : [OUT] 経度(東経) ddd.dddddd形式、同上
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時 (latitude, longitudeの値は不定)
補足	<p>本関数の実行には、数十秒～3分程度の時間が掛かる。 AGPSサーバは、3GIM(V2.2)ではGoogleのサーバを利用する(今後変更となる可能性がある)</p> <p>本関数は、同期処理である。そのため、測位処理が完了するまで、あるいは測位が失敗するまで呼び出し元には制御は戻らない。</p>

2. 現在位置取得（GPS）関連の関数 getLocation2

int getLocation2(char* latitude, char* longitude, char *height, char *utc, int *quality, int *number)	
機能概要	現在位置を取得する
引数	latitude : [OUT] 緯度(北緯:度) dd.ddddddd形式、ただし桁数は場合により可変
	longitude : [OUT] 経度(東経:度) ddd.ddddddd 同上
	height : [OUT] アンテナの海拔高さ(単位:m)
	utc : [OUT] 協定世界時での時刻(hhmmss)
	quality : [OUT] 位置特定品質。0 = 位置特定できない、1 = SPS(標準測位サービス)モード、2 = differential GPS(干渉測位方式)モード
	number : [OUT] 使用GPS衛星数
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時(latitude, longitudeの値は不定)
補足	<p>本関数の実行には、数十秒～3分程度の時間が掛かる。</p> <p>AGPSサーバは、3GIM V2.2ではGoogleのサーバを利用する(今後変更となる可能性がある)</p> <p>本関数は、同期処理である。そのため、測位処理が完了するまで、あるいは測位が失敗するまで呼び出し元には制御は戻らない。</p>

3. GPSの初期設定関数 setLocationParams

int setLocationParams(int timeout, boolean useAGPS, boolean useActiveAntenna)	
機能概要	AGPS(アシストGPS)やアクティブGPSアンテナの設定など関数
引数	timeout:GPS取得の制限時間(10ミリ秒)【省略値:180000(3分間)】
	useAGPS:AGPS(アシストGPS)の設定、ONにする時はTRUE(=1)、OFFにする時はFALSE(=0)を指定する
	useActiveAntenna : アクティブGPSアンテナの設定、ONにする時はTRUE(=1)、OFFにする時はFALSE(=0)を指定する
戻り値	0:正常に設定できた時
	0以外:設定できなかった時
補足	本関数の実行では、SIMカードの設定やアクティブGPSアンテナの接続が必要。

6. サービス他関連関数

サービス他関連関数

▶ 提供関数ライブラリ

分類	メソッド名	機能概要	補足
通信その他機能	getServices	利用可能サービスの取得	
	getRSSI	電波強度の取得	
	getTime	現在時刻の取得	日付・時刻形式
	getTime2	現在時刻の取得	通算秒形式
	getVersion	3GIMのバージョンの取得	
	getResult	3GIMからの応答値受信	
	sendCommand	3GIMへのコマンド送信	
	sendData	3GIMへのデータ送信	
	enterAT	ATコマンドパススルーモード	
	discardUntil	3 GIMからの文字の受信	

▶ 留意点

- ▶ 時刻は、使用している3Gネットワークを介してインターネット上のサーバから取得する（タイムゾーンや時刻精度は利用するネットワークに依存する）

1. サービス他関連関数 `getServices`

int getServices(int& status)	
機能概要	現在利用できるネットワークサービスを取得する
引数	status : [OUT] 利用できるネットワークサービス(下記のいずれか) a3gimSRV_NO(=0) : サービス利用不可 a3gimSRV_PS(=1) : パケット通信サービスのみ a3gimSRV_CS(=2) : 音声通信(+SMS)サービスのみ a3gimSRV_BOTH(=3) : パケット通信、音声通信(SMS)いずれも可
戻り値	0 : 正常に取得できた時
	0以外 : 取得できなかった時(statusの値は不定)
補足	3GIM(V2.2)では、a3gimSRV_CSとa3gimSRV_BOTHは返却しない。つまり、SMSが利用できるかどうかは判断できない。

【使い方事例】

```
int status;  
if (a3gim.getServices(status) == 0)  
    Serial.println(status);
```

SIMカードによる提供サービスの違いを取得

2. サービス他関連関数 getRSSI

int getRSSI(int& rssi)	
機能概要	3Gの電波強度(RSSI)を取得する
引数	rssi : [OUT] 取得した電波強度(単位はdBm) <範囲:-1 ~ -113>
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時(rssiの値は不定)
補足	電波強度は必ずマイナス値が返却される。0に近いほど電波強度は強い。

【電波受信レベルの測定サンプル】

実際に測定した実績では、3Gアンテナを付けずに測定した場合は「-113dBm」を計測、また電波状態の良いところでは「-68dBm」程度を計測できている。

【使い方事例】

```
int rssi;  
if (a3gim.getRSSI(rssi) == 0)  
    Serial.println(rssi);
```

3. サービス他関連関数 getTIME

int getTime(char* date, char* time)	
機能概要	現在の日付・時刻を取得する
引数	date : [OUT] 取得した日付 ("YYYY-MM-DD"形式)
	time : [OUT] 取得した時刻 ("HH:MM:SS"形式)
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時 (date/timeの値は不定)
補足	日付・時刻は使用している3Gネットワークを介して日本のサーバから取得するため、精度およびタイムゾーンはネットワークに依存する。(日本国内で利用する場合は、タイムゾーンは日本(JST)となる) 時刻は24h制(固定)である。ただし、自動調整出力となるため、変更は不可。

【使い方事例】

【利用例】

```
if (a3gim.getTime(date, time) == 0) {  
    Serial.print(date);  
    Serial.print(" ");  
    Serial.println(time);  
}
```



【出力結果例】

2012/12/01 12:10:43

【注意：時刻自動調整機能について】

長く電源を入れていないと、時刻が一旦1980年1月5日16:00:00に戻る。
電源を入れて動かしている間に、一時的に現在時刻より16時間遅れで表示され、さらに、日本時間で表示されるようになる。

4. サービス他関連関数 getTIME2

int getTime2(uint32_t& seconds)	
機能概要	現在の時刻(1970/1/1からの通算秒:「UNIX時間」とも言う)を取得する
引数	seconds : [OUT] 取得した通算秒
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時(date/timeの値は不定)
補足	時刻を秒単位で取得できるため、時刻の比較処理等が簡単となる。 日付・時刻の精度およびタイムゾーンについては、getTime()を参照。 秒への換算処理では、閏(うるう)年も考慮している。 2038年※問題が発生する可能性がある。

※ **2038年問題**は、ISOの通算秒の定義に1970年1月1日からとしていて、C言語の標準で32ビット符号付intを採用している場合、2038年1月19日3時14分7秒(UTC、以下同様)を過ぎると、この値がオーバーフローし、負と扱われるため、コンピュータが誤動作する可能性があるとする問題。
【詳細はウィキペディア参照】

5. サービス他関連関数 getVersion

int getVersion(char* version)	
機能概要	3GIM ファームウェア gw3gアプリのバージョンを取得する(最新版V3.3)
引数	version : [OUT] 取得したバージョン("9.9"形式)
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時(versionの値は不定)
補足	begin()の処理の中で3GIMのバージョンと本ライブラリの整合性をチェックしているため、通常、本関数を利用する必要はない。

6. 3 GIMからの応答値受信関数 getResult

int getResult(char *buf, int *len, uint32_t timeout)	
機能概要	3GIM からの応答値受信
引数	buf : 結果を返すバッファ
	len : バッファサイズ
	timeout : タイムアウト時間
戻り値	0 : 正常に取得できた時
	1 : 取得できなかった時 (タイムアウト)
補足	

7. 3GIMへのコマンド送信 sendCommand

void sendCommand(const char* cmd)

機能概要	3GIM へのコマンド送信
引数	cmd: コマンド送信(「\$コマンド」などを送信)
戻り値	戻り値なし
補足	

8. 3 GIMへのデータ送信関数 sendData

void sendData(const char* data)	
機能概要	3GIM へのデータ送信
引数	data: 配列文字
戻り値	戻り値なし
補足	

9. ATコマンド実行 enterAT

int enterAT(unit32_t duration)	
機能概要	ATコマンドパススルーモードに切り替え
引数	duration : ATコマンドパススルーモードから戻るまでの時間(単位:0.1秒) 例: 300=30秒 600=1分 =0の場合には、ATコマンドパススルーモードに切り替わったまま
戻り値	0: ATコマンドモードに切り替わった場合
	1: 引数の指定がおかしい時など(モードは切り替わらない)
補足	<p>ATコマンドパススルーモードでは、HL8548-Gに対して直接ATコマンドを送信して、そのままの結果を取得することができる。</p> <p>ATコマンドの使用に制限はないため、HL8548-Gの設定を任意に変更することができる。しかし、変更した設定等によっては、gw3gファームウェアの動作に支障をきたす場合があるので、十分留意すること。</p> <p>利用できるATコマンドの詳細は、HL8548-Gの開発元であるSierra Wireless社のサイトで公開されている「AT Commands Interface Guide - AirPrime HL6 and HL8 Series」を参照のこと。</p> <p>※ATコマンドそのものに関しては、Tabrainでは技術的なサポートは致しかねますので、ご了承ください。</p>

10. 3GIMからの文字の読み捨て関数 discardUntil

void discardUntil(const char match)

機能概要	3GIM からの文字の読み捨て関数
引数	match: 読み捨てるまでの文字(この文字と一致したら処理終了)
戻り値	戻り値なし
補足	

7. TCP/IP関連関数

TCP/IP関連の関数

▶ 提供関数ライブラリ

分類	メソッド名	機能概要	補足
TCP/IP	connectTCP※	TCPコネクションを接続	
	disconnectTCP※	TCPコネクションを切断	
	getStatusTCP	TCPコネクション最新状況取得	
	setTCPParams	TCPパラメータ設定	
	writeBegin	シリアル通信で直接書込み	
	read※	データの読み込み	3バリエーション有
	write※	データの書出し	3バリエーション有

【注意事項】

※ Arduino GSM/GPRSシールド用ライブラリと互換性がある関数

- TCP/IP v4のみサポートする。
- 一度に一つのコネクションだけを利用できる。（Web機能とは独立して使用できる）
- 本機能で提供する関数では、すべて同期的に処理する。そのため、サーバから結果が得られるまで、エラーが発生するまで、あるいは通信がタイムアウトするまで呼び出し元には制御が戻らない。
- 接続や通信では、タイムアウト時間としてデフォルトで60秒が設定されている
- readやwriteでエラーが発生した時は、一旦disconnectTCPを呼び出した後、connectTCPによる接続からやり直す必要がある。
- TCP/IP機能を利用するためには、利用するSIMカードでパケット通信が利用できる必要がある。
また、契約プランによっては通信料金が高額になる場合があるため注意すること
- 利用するSIMカードによっては、使用できるポート番号に制限(80/443番のみ等)がある場合がある。
- 1バイト単位でのread/writeは、実行効率が悪く、かつ実行速度が遅い。そのため、できるだけ複数バイト単位でread/write関数を呼び出して処理することが望ましい。
- 一部のread/write関数で、バイナリデータを透過的に取り扱うことができる。（Ver2.0以降）
- read関数はノンブロッキング（読み出すべきデータがない場合は、待たずに直ちに呼び出し元へリターンする）で動作する。

1. TCP/IP機能の関数 connectTCP

int connectTCP(const char* server, int port)	
機能概要	指定したサーバ、ポート番号へ接続して、TCPコネクションを確立する
引数	server : 接続するサーバのホスト名またはIPアドレス port : 接続するポート番号
戻り値	0 : 正常に接続できた時
	0以外 : エラーが発生した時(戻り値はエラー番号を表す)
補足	本機能の処理には、状況によって最大30秒程度掛かる。 serverには、IPv4アドレス("x.x.x.x"形式)またはホスト名を指定することができる。

【使い方の例】

```
char *svr = "arduino.cc";
if (a3gim.connectTCP(svr, 80) == 0) {
    // GET Request for google site
    a3gim.write("GET / HTTP/1.0\n");
    a3gim.write("HOST:");
    a3gim.write(svr);
    a3gim.write("\n\n");
    // Get response..
}
else {
    Serial.println("Error: can't connect");
}
```

2. TCP/IP機能の関数 disconnectTCP

int disconnectTCP(void)	
機能概要	接続しているTCPコネクションを切断する
引数	なし
戻り値	0 : 正常に切断できた時
	0以外 : エラーが発生した時(戻り値はエラー番号を表す)
補足	本機能の処理には、状況によって1～数秒程度掛かる。 本ライブラリの終了関数endでは、TCPコネクションは自動的に切断しない。そのため、必要に応じて必ず本関数を呼び出してTCPコネクションを明示的に切断すること。

3. TCP/IP機能の関数 getStatusTCP

int getStatusTCP(int *status,int *tcpnotif, logn *remainedBytes, long *receivedBytes)	
機能概要	接続しているTCPコネクションの最新状況を取得する
引数	status: TCPコネクションのステータス(別表①参照) tcpnotif: TCPコネクションで最後に発生したエラーの内容(別表②参照) remainedBytes: 相手に送信できていないデータのサイズ(バイト数) receivedBytes: 受信したがまだread()していないデータのサイズ(バイト数)
戻り値	次頁参照
補足	

3. TCP/IP機能の関数 getStatusTCP

表① statusの意味

status	意味
0	未接続
1	接続済み
2	接続失敗
3	クローズした
4	接続中
5	アイドルタイム*2のカウント開始
6	アイドルタイム*2のカウント取り消し

*2 TCPコネクションが切断された時からアイドルタイムのカウントが開始される。アイドルタイムが30秒となった時に、3Gネットワークのセッションが解放される。アイドルタイム中にread()やwrite()を行うと、アイドルタイムはいったんリセットされる。

【エラー発生時の対処方法】

何らかのエラーが発生した場合は、通常、TCPコネクションを切断して、再度接続からやり直すことを推奨する。

表② tcpnotifの意味

tcpnotif	意味
0	ネットワークエラー
1	ソケットエラー
2	メモリ問題
3	DNS問題
4	TCPが相手から切断された
5	TCPコネクションエラー
6	一般的なエラー
7	クライアントからのリクエスト受けエラー*1
8	AT+KTCPSNDで文字待ちが発生*1
9	セッションIDがおかしい*1
10	セッションは使用中である
11	すべてのセッションは使用中である*1

*1 通常は発生しない内部エラー

4. TCP/IP機能の関数 setTCPParams

int setTCPParams(int timeout1, int timeout2)	
機能概要	TCPコネクションのパラメータ設定
引数	timeout1 : connectTCP関数のタイムアウト時間(設定: 10m秒)
	timeout2 : write()/read()関数のタイムアウト時間(設定: 10ミリ秒)
戻り値	0 : 正常処理
	1: パラメータの間違い
	2: 内部処理エラー発生
補足	引数は、ともに0より大きく6000(1分)より短いこと

5. TCP/IP機能の関数 writeBegin

int writeBegin(size_t sz)	
機能概要	指定したサイズszのデータをTCPコネクションに対して書き込む準備をする
引数	sz : データのサイズ(バイト数)、最大 a3GsMAX_TUNNEL_DATA_LENGTH
戻り値	0 : 正常に準備ができた時
	-1 : エラーが発生した時(引数szがおかしい)
補足	<p>本機能の処理には、状況によって数秒程度掛かる。 バイナリデータを相手へ書き込むための最速の関数である。\$エスケープが不要であり、一度の書き込めるサイズも大きいため、write()関数よりも高速である。 本関数を呼び出した後は、指定したサイズ分のデータ(バイナリデータもそのままOK)を、シリアルa3gsに対して直接書き込む。正確にszバイト分のデータを書き込む必要がある。 szに満たない場合は、30秒のタイムアウト後に、szバイトに満たない不足分のデータとして、3GIMが自動的に0x00バイトを充当する。 通信速度を115,200bpsに設定している場合は、フロー制御を行っていないため1024バイトのデータ送るたびに5ミリ秒以上のディレイを挿入する必要がある。</p>

【使い方の例】

```
char *svr = "someserver.aaa";
uint8_t data[256];
if (a3gim.connectTCP(svr, 80) == 0) {
    a3gim.writeBegin(sizeof(data));
    for (int i = 0; i < sizeof(data); i++)
        a3gs.write(data[i]);
    int len = sizeof(buf) - 1;
    char buf[20];
    if (a3gs.getResult(buf, &len, 60000) != 0)
        // Error handling ..
}
```

左記の使い方にあるように、writeBegin()を呼んだ後は、シリアルa3gsに対して直接write()やprint()を使って所定のサイズ分のデータを書き込む。
書き込んだ後は、getResult()関数を呼び出して、書き込みの成否をチェックする。

6. TCP/IP機能の関数 read

● バリエーション1 (バイナリデータの読み出し可)

int read(void)	
機能概要	現在のTCPコネクションから1バイトのデータを読み出す
引数※	なし
戻り値	0~0xFF: 読み出した1バイトのデータ
	-1: エラーが発生した時(コネクションがcloseされた、エラーが発生した、タイムアウトした)
	-2: データが読み出せなかった時(データがない時)
補足	本関数は常にノンブロッキングで動作する。そのため、読み出せるデータがない時は直ちに呼び出し元へリターンする。 本関数は、コネクションがcloseされた時やエラーが発生した時は、直ちに呼び出し元へリターンする。

【使い方の例】

```
char *svr = "arduino.cc";
if (a3gim.connectTCP(svr, 80) == 0) {
    // GET Request for google site
    a3gim.write("GET / HTTP/1.0\n");
    a3gim.write("HOST:");
    a3gim.write(svr);
    a3gim.write("\n\n");
    handleResponse();
}
else
    Serial.println("Error: can't connect");
```

```
void handleResponse(void) {
    int c;
    while ((c = a3gim.read()) > 0) {
        // 読み出した文字cを処理する
    }
}
```


6. TCP/IP機能の関数 read

● バリエーション2 (テキストデータの読み出しのみ)

int read(char* result, int resultlength)	
機能概要	現在のTCPコネクションから最大resultlengthバイトのデータを読み出す
引数	result : [OUT] 読み出したデータを格納するバッファアドレス resultlength : 呼び出し側で確保したバッファのサイズ(バイト数)
戻り値	1 ~ (resultlength-1) : 正常に読み出した時(読み出したバイト数を返す)
	0 : データが読み出せなかった時
	0未満 : エラーが発生した時(コネクションがcloseされた、エラーが発生した)
補足	本関数は常にノンブロッキングで動作する。そのため、読み出せるデータがない時は直ちに呼び出し元へリターンする。 本関数は、コネクションがcloseされた時やエラーが発生した時は、直ちに呼び出し元へリターンする。 resultで返却するデータは、ヌル文字('¥0')で終端させる。

【使い方の例】

```
void handleResponse(void) {  
    char res[a3gimMAX_RESULT_LENGTH+1];  
    int nbytes;  
    while(a3gim.read(res,a3gimMAX_RESULT_LENGTH+1) > 0) {  
        Serial.print(res);  
    }  
}
```

6. TCP/IP機能の関数 read

● バリエーション3 (バイナリデータの読み出し可)

int read(uint8_t* buffer, size_t sz)	
機能概要	現在のTCPコネクションから最大szバイトのデータを読み出す
引数	buffer : [OUT] 読み出したデータを格納するバッファアドレス sz : 呼び出し側で確保したバッファbufferのサイズ(バイト数)
戻り値	1 ~ sz : 正常に読み出した時(読み出したバイト数を返す)
	0 : データが読み出せなかった時
	0未満 : エラーが発生した時(コネクションがcloseされた、エラーが発生した)
補足	<p>本関数は常にノンブロッキングで動作する。そのため、読み出せるデータがない時は直ちに呼び出し元へリターンする。</p> <p>本関数は、コネクションがcloseされた時やエラーが発生した時は、直ちに呼び出し元へリターンする。</p> <p>機能はバリエーション2と同じであるが、バイナリデータを扱う場合は本関数を使用すること。バリエーション2と異なり、読み出したデータをヌル文字('¥0')で終端することはない。</p>

7. TCP/IP機能の関数 write

- バリエーション1 (バイナリデータの書き出し可能)

int write(uint8_t c)	
機能概要	現在のTCPコネクションへ1バイトのデータを書き出す
引数	c: 書き出すデータ
戻り値	1: 正常に書き出した時
	0未満: エラーが発生した時(コネクションがcloseされた、エラーが発生した、タイムアウトした)
補足	<p>本関数の処理には、状況によって最大30秒程度掛かる データcとして、制御文字、ヌル文字、あるいは特殊文字(ダブルクォート、\$)等もそのまま指定することができる。</p> <p>本関数は同期処理であるため、コネクションへデータを書き出すまで、コネクションがcloseされるまで、エラーが発生するまで、あるいはタイムアウトするまで呼び出し元に制御は戻らない。</p>

7. TCP/IP機能の関数 write

● バリエーション2 (テキストデータの書き出しのみ)

int write(const char* str)	
機能概要	現在のTCPコネクションへ文字列データを書き出す
引数	str : 書き出す文字列データ('¥0'で終端する文字配列)
戻り値	l 以上 : 正常に書き出した時(書き出したバイト数を返す)
	0未満 : エラーが発生した時(コネクションがcloseされた、エラーが発生した、タイムアウトした)
補足	<p>本関数の処理には、状況によって最大30秒程度掛かる。</p> <p>本関数は同期処理であるため、コネクションヘデータを書き出すまで、コネクションがcloseされるまで、エラーが発生するまで、あるいはタイムアウトするまで呼び出し元に制御は戻らない。</p> <p>バリエーション1・3と異なり、バイナリデータを取り扱うためのエスケープ処理を実行しないため、これらに比べて高速に実行できる。</p>

【使い方の例】

```
char *svr = "arduino.cc";
if (a3gim.connectTCP(svr, 80) == 0) {
    // GET Request for google site
    a3gim.write("GET / HTTP/1.0$n");
    a3gim.write("HOST:");
    a3gim.write(svr);
    a3gim.write("$n$n");
    // Get response..
}
```

7. TCP/IP機能の関数 write

● バリエーション3 (バイナリデータの書き出し可能)

int write(const uint8_t* buffer, size_t sz)	
機能概要	現在のTCPコネクションへ指定したバイト数のデータを書き出す
引数	buffer : 書き出すデータ(バイト配列) sz : データのサイズ(バイト数:型 size_t)
戻り値	1以上 : 正常に書き出した時(書き出したバイト数を返す)
	0未満 : エラーが発生した時(コネクションがcloseされた、エラーが発生した、タイムアウトした)
補足	バリエーション2ではデータの中にヌル文字('¥0')を取り扱うことができないが、本バリエーションではデータをエスケープ処理するため、データの中に制御文字、ヌル文字、あるいは特殊文字(ダブルクォート、\$)をそのまま含めることができる。 本関数の処理には、状況によって最大30秒程度掛かる。 本関数は同期処理であるため、コネクションへデータを書き出すまで、コネクションがcloseされるまで、エラーが発生するまで、あるいはタイムアウトするまで呼び出し元に制御は戻らない。

【使い方の例】

```
char *svr = "192.168.1.1";
uint8_t binaryData[] = { 0x0, 0x1, 0x2, 0x3, ..};
if (a3gim.connectTCP(svr, 8080) == 0) {
    a3gim.write(binaryData, sizeof(binaryData));
}
```

8. プロファイル関連関数

プロファイル関連の関数

▶ 提供関数ライブラリ

分類	メソッド名	機能概要	補足
プロファイル	setDefaultProfile	デフォルトプロファイルを設定	
	getDefaultProfile	デフォルトプロファイルを取得	

【注意事項】 プロファイルは、通信サービス事業者が提供するSIMカードを利用するための設定情報である。
詳細は、setDefaultProfile 関数の説明を参照。

1. プロファイルの関数 setDefaultProfile

int setDefaultProfile(const char *apn, const char *user, const char *password)	
機能概要	デフォルトのプロファイルを設定する
引数	apn: SIMカードのAPN情報 user: SIMカードのユーザ名 password: SIMカードのパスワード
戻り値	0: 正常に設定できた時
	0以外: 設定できなかった時(引数の指定が間違っている等)
補足	設定したデフォルトのプロファイル番号は、内臓マイコンのフラッシュROMに記録されるため、電源をOFFにしても維持される(再度、本ライブラリにて設定するまで有効である) 3GIMV2.2の出荷時の設定プロファイル情報は、下記の通りである: 出荷時デフォルト: IIJ様の iijmio サービス(iijmio.jp/mio@iij/iij)

2. プロファイルの関数 getDefaultProfile

int getDefaultProfile(char *apn, char *user, char *password)	
機能概要	デフォルトのプロファイル番号を取得する
引数	apn: SIMカードのAPN情報 user: SIMカードのユーザ名 password: SIMカードのパスワード
戻り値	0: 正常に取得できた時
	0以外: 取得できなかった時
補足	

【使い方の例】

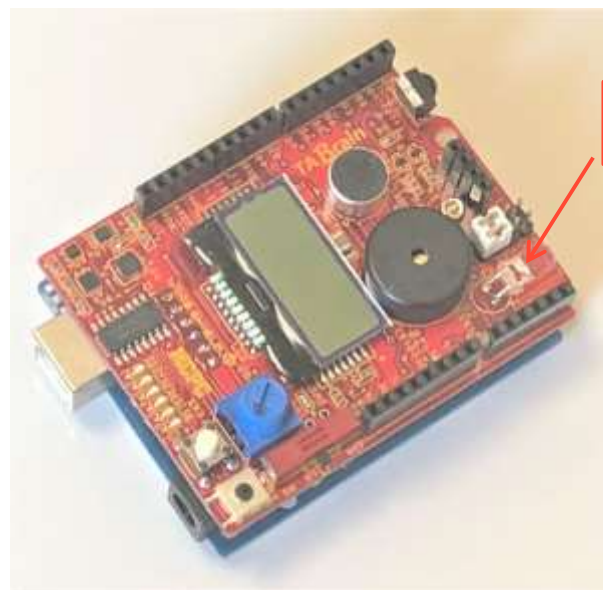
```
int pfNum;
if (a3gim.getDefaultProfile(&pfNum) == 0) {
    Serial.print("Default Profile No is ");
    switch (pfNum == 1)
    case 1 :
        Serial.println("docomo mopera");
    case 2 :
        Serial.println("IJJmio");
    case 3 :
        Serial.println("IJJmobile");
```

```
case 11 :
    Serial.println("bmobile");
case 15 :
    Serial.println("DTI ServerMan");
default :
    Serial.println("unknown profile");
}
```

第5章 ライブラリ・サンプル・スケッチ

1. 3 GIMシールド・IoTABシールド設定方法

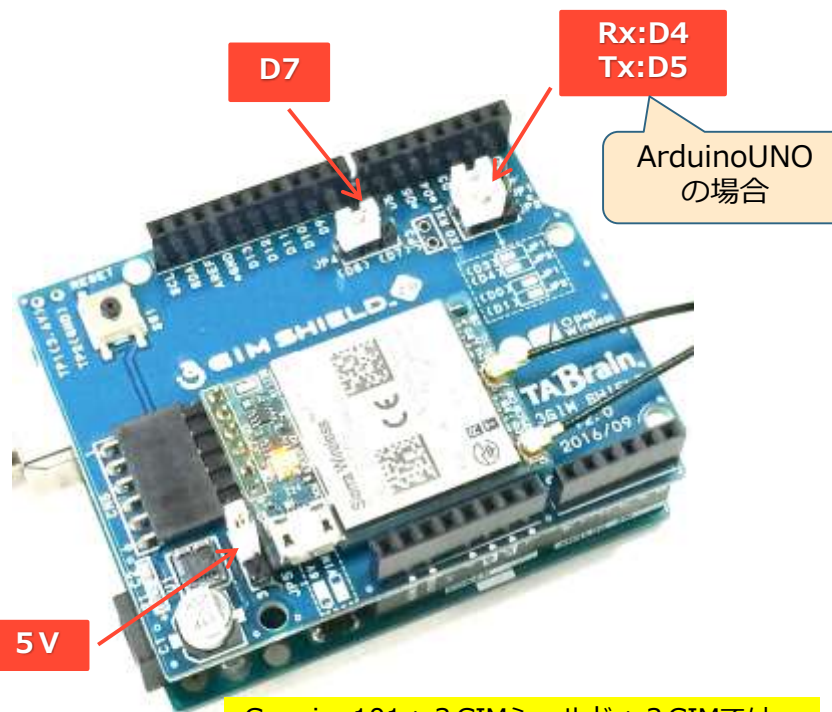
a3gim、a3gim2のサンプルスケッチをArduino IDEで読み込み、ArduinoUNOやGenuino101ほか互換機にコンパイルし書き込んで、実行してみてください。ここでは、**ArduinoUNO+IoTABシールド+ 3 GIM** または **ArduinoUNO+ 3 GIMシールド+ 3 GIM** を使った設定を紹介しておきます。



ArduinoUNO
の場合

Rx:D4
Tx:D5

Genuino101 + IoTABシールド + 3 GIMでは、
① JP1とJP2は、RxとTxのジャンパピンをD0とD1とに接続



Rx:D4
Tx:D5

ArduinoUNO
の場合

D7

5V

Genuino101 + 3 GIMシールド + 3 GIMでは、
① JP 1 とJP2のRxとTxのジャンパピンをD4とD5に接続
② JP 4 はD7に接続
③ JP5は 5 Vに接続

2. a3gimサンプルスケッチ一覧

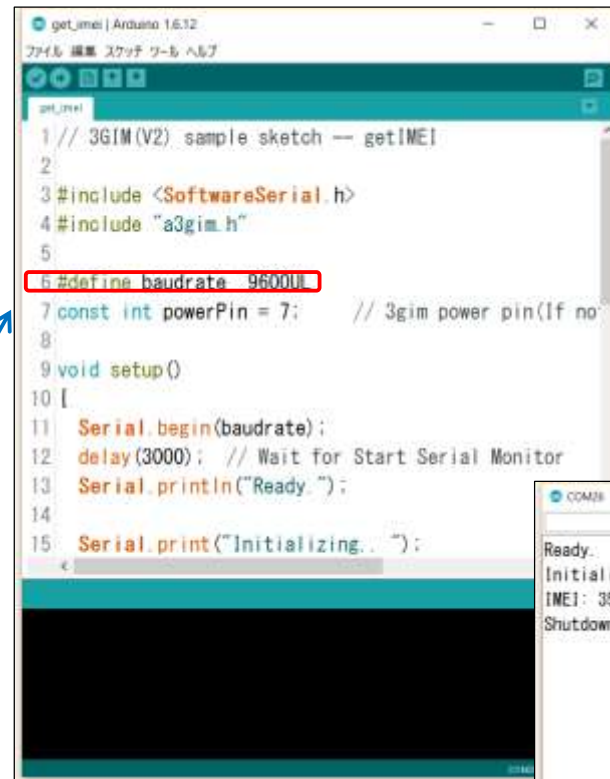
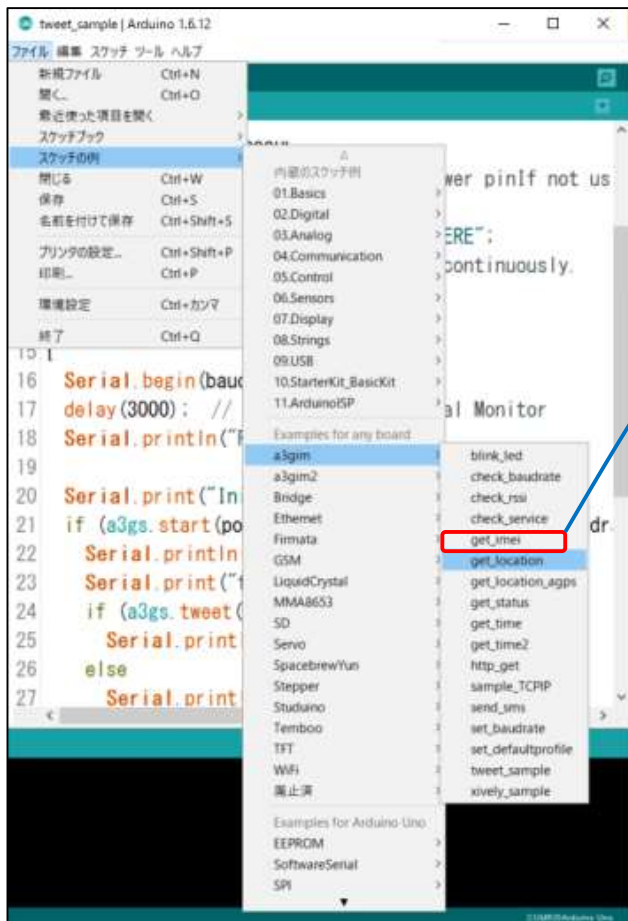


a3gimのサンプルスケッチを動かしてみましょう。
あらかじめ設定したサンプルスケッチは
Arduino IDEのファイル>スケッチの例に表示されます。

- blink_led
- check_baudrate
- check_rssi
- check_service
- get_imei
- get_location**
- get_location_agps
- get_status
- get_time
- get_time2
- http_get
- sample_TCPIP
- send_sms
- set_baudrate
- set_defaultprofile
- tweet_sample
- xively_sample

3. a3gimサンプルスキッチの読込・書込・実行

- ▶ a3gimのサンプルスキッチをArduinoIDEで読み込み、コンパイルし**ArduinoUNO+**（IoTABまたは3GIM）シールド+3GIMに書込みます。そのあと実行してみてください。
- ▶ 下記はサンプルスキッチ「get_imei.ino」を読込・書込・実行を行った例です。



4. a3gimサンプルスケッチの実行結果例

a3gim_examples

■ check_baudrate.ino (通信ボーレート確認)

```
Ready.  
Initializing..  
Try baudrate: 9600  
Recognize succeeded.  
Current baudrate is 9600 bps.
```

■ check_service.ino (SIMカードのサービス情報)

```
Ready.  
Initializing.. Succeeded.  
Packet Service Only.  
Shutdown..
```

■ check_rssi.ino (電波強度取得)

```
Ready.  
Initializing.. Succeeded.  
RSSI = -81 dBm  
Shutdown..
```

■ get_location.ino (GPS:位置情報・緯度・経度取得)

```
Ready.  
Initializing.. Succeeded. It maybe takes several minutes.  
OK: 35.641996, 139.604198  
Shutdown..
```

■ get_status.ino (機能状態取得)

```
Ready.  
Initializing.. Succeeded.  
Status is IDLE  
Shutdown...
```

■ get_time.ino (現日時取得)

```
Ready.  
Initializing.. Succeeded.  
2016/10/22 12:47:58  
Shutdown..
```

■ get_time2.ino (現日時取得 2)

```
Ready.  
Initializing.. Succeeded.  
1477140604 Sec.  
Shutdown..
```

■ http_get.ino (httpgetによるサーバデータ取得確認) ※8行目のサーバを tabrain.jp に変更

```
Ready.  
Initializing.. Succeeded.  
httpGET() requesting.. OK!  
[<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">  
<html lang="ja">  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=Shift]  
Shutdown..
```

■ sample_tcpip.ino (tcpipテスト確認)

```
Ready.  
Initializing.. Succeeded.  
www.arduino.org : Page title is "Arduino - Open Source Products for Electronic  
Projects"  
Shutdown..  
Initializing..
```

■ tweet_sample.ino (ツイート送信確認)

※10行目のtokenを設定し、11行目に文章挿入
(11行目の文章で空白はエラーとなるため「%20」に変更する)

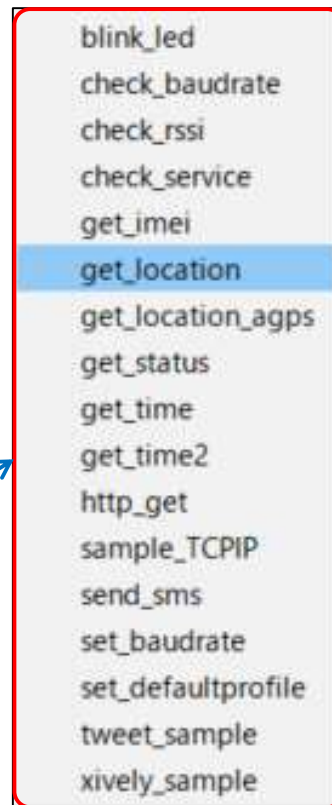
```
Ready.  
Initializing.. Succeeded.  
tweet() requesting.. OK!  
Shutdown..
```



5. a3gim2サンプルスケッチ一覧



a3gim2のサンプルスケッチを動かしてみましょう。
あらかじめ設定したサンプルスケッチは、
Arduino IDEのファイル>スケッチの例に表示されます

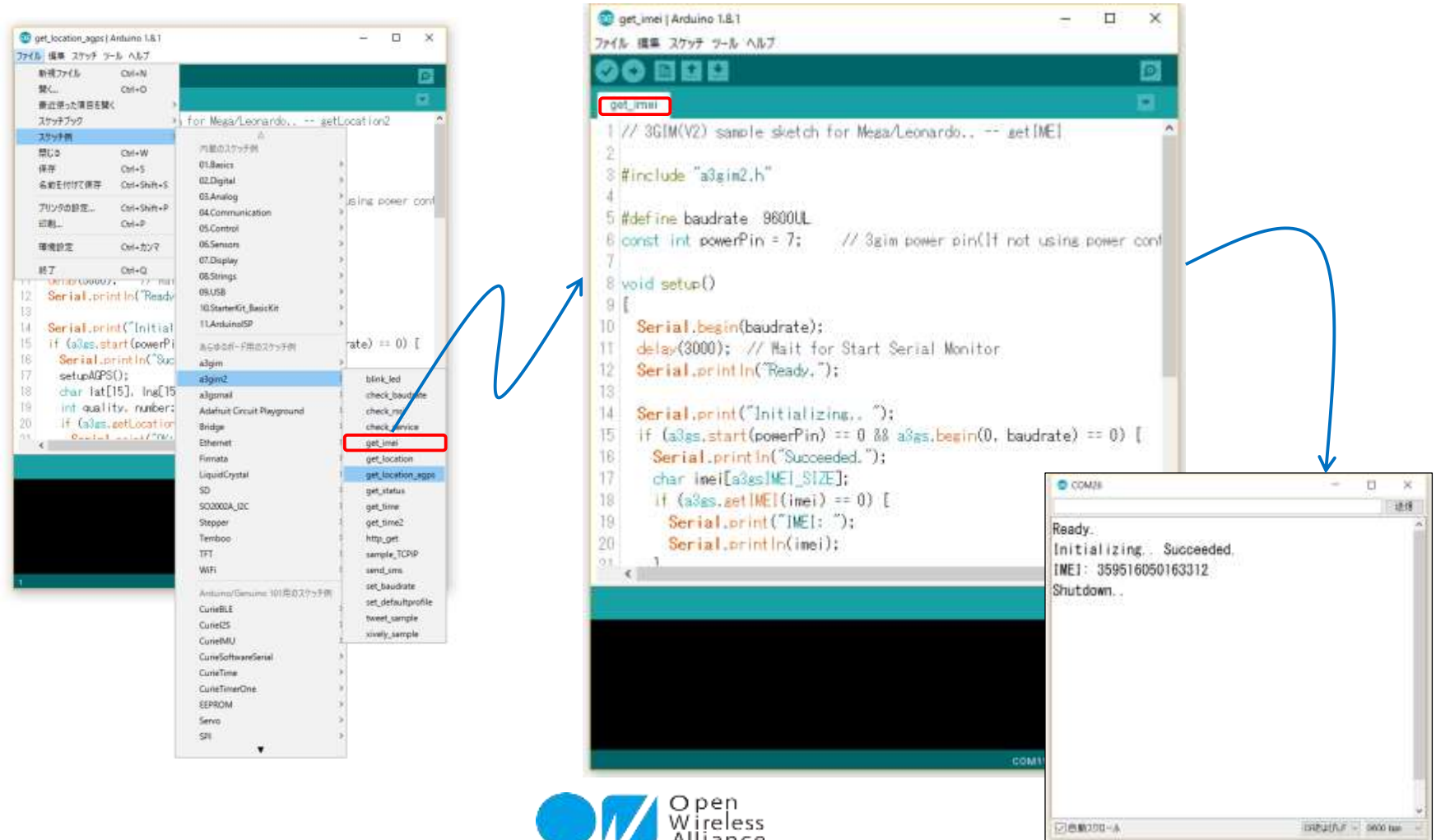


A list of sample sketches for the a3gim2 library, displayed in a scrollable list. The list includes: blink_led, check_baudrate, check_rssi, check_service, get_imei, get_location (highlighted with a blue bar), get_location_agps, get_status, get_time, get_time2, http_get, sample_TCPIP, send_sms, set_baudrate, set_defaultprofile, tweet_sample, and xively_sample. A blue arrow points from the 'a3gim2' folder in the IDE screenshot to this list.

ライブラリ群内のスケッチは以下のフォルダから参照
a3gim_examples

6. a3gim2サンプルスケッチの読込・書込・実行

- ▶ a3gim2のサンプルスケッチをArduinoIDEに読み込み、コンパイルし、**Genuino101+**（IoTABまたは3 GIM）シールド + 3 GIMに書込みます。そのあと実行してみてください。
- ▶ 下記はサンプルスケッチ「get_imei.ino」を読込・書込・実行を行った例です。



7. サンプルスケッチ (blink_led.ino)

本サンプルスケッチは、3 GIM上の緑LEDを点滅させるスケッチです。
特にインターネットとの通信は行っていないので、3 GIM立上げ後すぐにLEDの点滅を行います。



このLEDが点滅

- 宣言(a3gim2.hの読込)
 - `#include <a3gim2.h>`
- 電源ONと起動準備
 - `a3gs.start()` : 電源On
 - `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - `a3gs.end()`
 - `a3gs.shutdown()`
- LED点灯と消灯関数
 - `a3gs.settled(sw)`

```
// 3GIM(V2) sample sketch for Mega/Leonardo.. -- setLED  
  
#include <a3gim2.h>  
  
#define INTERVAL 1000 // Blink interval  
#define baudrate 9600UL  
const int powerPin = 7; // 3gim power pinIf not using power control, 0 is set.
```

a3gim2ライブラリの宣言

```
void setup()  
{  
  Serial.begin(baudrate);  
  delay(3000); // Wait for Start Serial Monitor  
  Serial.println("Ready.");  
  
  _retry:  
  Serial.print("Initializing..");  
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0)  
  {  
    Serial.println("Succeeded.");  
  }  
  else {  
    Serial.println("Failed.");  
    Serial.println("Shutdown..");  
    a3gs.end();  
    a3gs.shutdown();  
    delay(30000);  
    goto _retry; // Repeat  
  }  
  Serial.println("Blinking..");  
}  
  
void loop()  
{  
  a3gs.setLED(true);  
  delay(INTERVAL);  
  a3gs.setLED(false);  
  delay(INTERVAL);  
}
```

3 GIM電源ONと起動

7. サンプルスケッチ (check_baudrate.ino)

本サンプルスケッチは、3 GIMの通信速度が分からなくなった場合に起動するサンプルとなります。
9600から115200までの7種類の設定のいずれかに設定されている3 GIMの通信速度を調査します。

- 宣言 (a3gim2.hの読込)
 - ・ #include <a3gim2.h>
- 電源ONと起動準備
 - ・ a3gs.start() : 電源On
 - ・ a3gs.begin(0,baudrate) : 通信設定の確認
- 終了と電源OFF
 - ・ a3gs.end()
 - ・ a3gs.shutdown()

```
// 3GIM(V2) sample sketchfor Mega/Leonardo.. -- check current baudrate
#include "a3gim2.h"
const int powerPin = 7; // 3gim power pin(If not using power control, 0 is
set.)
long baudrates[7] = { 9600, 19200, 38400, 57600, 115200 };
//-- 2400bps and 4800bps are not supported in 3GIM(V2)
```

a3gim2ライブラリの宣言

通信速度を調べるために、
電源をOn/Offしながら
通信速度を上げながら
通信できるかを検査する。
一致した段階で処理終了し、
現在の通信速度を表示する。

実行例 (シリアルモニタ画面)

```
Ready.
Initializing..
Try baudrate: 9600
Recognize succeeded.
Current baudrate is 9600 bps.
```

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Ready.");

  Serial.println("Initializing..");
  for (int i = 0; i < 7; i++) {
    if (a3gs.start(powerPin) == 0) {
      Serial.print("Try baudrate: ");
      Serial.println(baudrates[i]);
      if (a3gs.begin(0, baudrates[i]) == 0) {
        Serial.println("Recognize succeeded.");
        Serial.print("Current baudrate is ");
        Serial.print(baudrates[i]);
        Serial.println(" bps.");
        return;
      }
      Serial.println("Failed.");
      a3gs.end();
      a3gs.shutdown();
    }
  }
  Serial.println("Can't recognize baudrate.");
}

void loop() {}
```

3 GIM電源ON

3 GIM起動

7. サンプルスケッチ (check_rssi.ino)

本サンプルスケッチは、3 GIMの電波強度（RSSI）を表示させるスケッチです。

- 宣言(a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- 電波強度関数
 - ・ `a3gs.getRSSI(rssi)` ;

```
// 3GIM(V2) sample skech for Mega/Leonardo.. -- getRSSI
```

```
#include "a3gim2.h"
```

a3gim2ライブラリの宣言

```
#define baudrate 9600UL  
const int powerPin = 7; // 3gim power pin(If not using power control, 0 is  
set.)
```

実行例（シリアルモニタ画面）

```
Ready.  
Initializing.. Succeeded.  
RSSI = -81 dBm  
Shutdown..
```

```
void setup()  
{  
  Serial.begin(baudrate);  
  delay(3000); // Wait for Start Serial Monitor  
  Serial.println("Ready.");  
  
  Serial.print("Initializing..");  
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0){  
    Serial.println("Succeeded.");  
    int rssi;  
    if (a3gs.getRSSI(rssi) == 0) {  
      Serial.print("RSSI = ");  
      Serial.print(rssi);  
      Serial.println(" dBm");  
    }  
  }  
  else  
    Serial.println("Failed.");  
  
  Serial.println("Shutdown..");  
  a3gs.end();  
  a3gs.shutdown();  
}  
  
void loop() {}
```

3 GIM電源ON/起動

電波強度の出力

7. サンプルスケッチ (check_service.ino)

本サンプルスケッチは、SIMカードによるサービス内容を表示するスケッチです。

- 宣言(a3gim2.hの読込)
 - ・ #include <a3gim2.h>
- 電源ONと起動準備
 - ・ a3gs.start() : 電源On
 - ・ a3gs.begin() : 通信設定の確認
- 終了と電源OFF
 - ・ a3gs.end()
 - ・ a3gs.shutdown()
- SIMサービス取得関数
 - ・ a3gs.getServices(status);

// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getServices

#include "a3gim2.h"

a3gim2ライブラリの宣言

#define baudrate 9600UL
const int powerPin = 7;

実行例 (シリアルモニタ画面)

Ready.
Initializing.. Succeeded.
Packet Service Only.
Shutdown..

```
void setup(){
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");
  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    int status;
    if (a3gs.getServices(status) == 0) {
      switch (status) {
        case a3gsSRV_NO :
          Serial.println("No Service.");
          break;
        case a3gsSRV_PS :
          Serial.println("Packet Service Only.");
          break;
        case a3gsSRV_CS : // Never returned at R4.0
          Serial.println("Voice Service Only."); // also SMS
          break;
        case a3gsSRV_BOTH : // Never returned at R4.0
          Serial.println("Packet And Voice Services.");
          break;
        default :
          break;
      }
    }
    else
      Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
  }
}

void loop() {}
```

3 GIM電源ON/起動

SIMサービス取り出し

7. サンプルスケッチ (check_imei.ino)

本サンプルスケッチは、3GIMの通信モジュールの固有番号IMEIを取り出し表示します。

- 宣言(a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- IMEI取得関数
 - ・ `a3gs.getIMEI(imei);`

```
// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getIMEI  
  
#include "a3gim2.h"  
  
#define baudrate 9600UL  
const int powerPin = 7;
```

a3gim2ライブラリの宣言

実行例 (シリアルモニタ画面)

```
Ready.  
Initializing.. Succeeded.  
IMEI: 354563020267950  
Shutdown..
```

```
void setup()  
{  
  Serial.begin(baudrate);  
  delay(3000); // Wait for Start Serial Monitor  
  Serial.println("Ready.");  
  
  Serial.print("Initializing.. ");  
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {  
    Serial.println("Succeeded.");  
    char imei[a3gsIMEI_SIZE];  
    if (a3gs.getIMEI(imei) == 0) {  
      Serial.print("IMEI: ");  
      Serial.println(imei);  
    }  
  } else  
  {  
    Serial.println("Failed.");  
    Serial.println("Shutdown..");  
    a3gs.end();  
    a3gs.shutdown();  
  }  
}  
  
void loop() {}
```

3 GIM電源ON/起動

IMEIの取り出し

7. サンプルスケッチ (get_location.ino)

本サンプルスケッチは、位置情報取得（GPS）のサンプルです。

- 宣言(a3gim2.hの読み込み)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- 緯度・経度の位置情報取得関数
 - ・ `a3gs.getLocation(x,lat,lng);`

```
// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getLocation
#include "a3gim2.h"
#define baudrate 9600UL
const int powerPin = 7;
```

a3gim2ライブラリの宣言

実行例（シリアルモニタ画面）

```
Ready.
Initializing.. Succeeded. It maybe takes several minutes.
OK: 35.641996, 139.604198
Shutdown..
```

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for start serial monitor
  Serial.println("Ready.");

  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded. It maybe takes several minutes.");
    char lat[15], lng[15];
    if (a3gs.getLocation(a3gsMPBASED, lat, lng) == 0) {
      Serial.print("OK: ");
      Serial.print(lat);
      Serial.print(", ");
      Serial.println(lng);
    }
    else
      Serial.println("Sorry, I don't know this location.");
  }
  else
    Serial.println("Failed.");
  Serial.println("Shutdown..");
  a3gs.end();
  a3gs.shutdown();
}

void loop() {}
```

3 GIM電源ON/起動

SIMサービス取り出し

7. サンプルスケッチ (get_location_agps.ino) ①

本サンプルスケッチは、Assited-GPSによる位置情報取得（GPS）のサンプルです。

- 宣言(a3gim2.hの読込)
 - ・ #include <a3gim2.h>
- 電源ONと起動準備
 - ・ a3gs.start() : 電源On
 - ・ a3gs.begin() : 通信設定の確認
- 終了と電源OFF
 - ・ a3gs.end()
 - ・ a3gs.shutdown()
- 緯度・経度の位置情報取得関数 2
 - ・ a3gs.getLocation2(lat,lng,height, utc,&quality,&number);

// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getLocation

#include "a3gim2.h"

a3gim2ライブラリの宣言

#define baudrate 9600UL
const int powerPin = 7;

※接続されたSIMカードによって、Googleのサーバにアクセスするため、プロファイルを設定する処理がATコマンドで必要となります。
(次ページ参照)

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for start serial monitor
  Serial.println("Ready.");
  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    setupAGPS();
    char lat[15], lng[15], utc[7], height[8];
    int quality, number;
    if (a3gs.getLocation2(lat, lng, height, utc, &quality, &number) == 0) {
      Serial.print("OK: ");
      Serial.print(lat);
      Serial.print(",");
      Serial.print(lng);
      Serial.print(",");
      Serial.print(height);
      Serial.print(",");
      Serial.print(utc);
      Serial.print(",");
      Serial.print(quality);
      Serial.print(",");
      Serial.println(number);
    }
    else
      Serial.println("Sorry, I don't know this location.");
  }
  else
    Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
}

void loop() {}
```

3 GIM電源ON/起動

位置情報取得

7. サンプルスケッチ (get_location_agps.ino) ②

以下のサンプルスケッチは、ATコマンドモードに一旦入り、SIMカードのプロファイルの設定を行う処理となります。

この処理によって、この後のGPSは、Assisted GPS機能が働き、GPSの位置情報や移動情報は、3Gの通信基地局から入手します。これによって、短時間でGPSの位置情報が取得できるようになります。

■ Assited GPS設定関数スケッチ

```
// setup AGPS function
void setupAGPS()
{
  char apn[20], user[20], password[20];
  if (a3gs.getDefaultProfile(apn, user, password) == 0) {
    char atwppp[50];
    sprintf(atwppp, "at+wppp=2,4,\"%s\",\"%s\",%s", user, password);
    Serial.println(atwppp);
    a3gs.enterAT(2);
    a3gSerial.println(atwppp);
    delay(200);
    Serial.println("Assisted GPS set OK");
  }
  else
    Serial.println("NG: getDefaultProfile(), can't use AGPS..");
}
```

実行例（シリアルモニタ画面）

```
Ready.
Initializing.. Succeeded. It maybe takes several minutes.
OK: 35.641996, 139.604198
Shutdown..
```


7. サンプルスケッチ (get_status.ino)

本サンプルスケッチは、3 G通信状態を取得するスケッチです。

- 宣言(a3gim2.hの読込)
 - `#include <a3gim2.h>`
- 電源ONと起動準備
 - `a3gs.start()` : 電源On
 - `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - `a3gs.end()`
 - `a3gs.shutdown()`
- 通信状態取得関数
 - `a3gs.getStatus()`;

// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getStatus

`#include "a3gim2.h"`

a3gim2ライブラリの宣言

`#define baudrate 9600UL`
`const int powerPin = 7;`

実行例 (シリアルモニタ画面)

Ready.
Initializing.. Succeeded.
Status is IDLE
Shutdown...

```
void setup(){
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");
  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0){
    Serial.println("Succeeded.");
    int status;
    status = a3gs.getStatus();
    Serial.print("Status is ");
    switch (status) {
      case A3GS::ERROR :
        Serial.println("ERROR");
        break;
      case A3GS::IDLE :
        Serial.println("IDLE");
        break;
      case A3GS::READY :
        Serial.println("READY");
        break;
      case A3GS::TCPCONNECTEDCLIENT :
        Serial.println("TCPCONNECTEDCLIENT");
        break;
      default :
        Serial.println("Unknown");
        break;
    }
  } else {
    Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
  }
}

void loop() {}
```

3 GIM電源ON/起動

通信状態取得

7. サンプルスケッチ (get_time.ino)

本サンプルスケッチは、日時取得のスケッチです。

- 宣言(a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- 日時取得関数
 - ・ `a3gs.getTime(date,time);`

// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getTime

`#include "a3gim2.h"` a3gim2ライブラリの宣言

`#define baudrate 9600UL`
`const int powerPin = 7;`

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");

  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    char date[a3gsDATE_SIZE], time[a3gsTIME_SIZE];
    if (a3gs.getTime(date, time) == 0) {
      Serial.print(date);
      Serial.print(" ");
      Serial.println(time);
    }
  } else {
    Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
  }
}

void loop() {}
```

3 GIM電源ON/起動

時刻取得

実行例 (シリアルモニタ画面)

```
Ready.
Initializing.. Succeeded.
2016/10/22 12:47:58
Shutdown..
```

7. サンプルスケッチ (get_time2.ino)

本サンプルスケッチは、日時取得（累積秒数）のスケッチです。

- 宣言(a3gim2.hの読込)
 - `#include <a3gim2.h>`
- 電源ONと起動準備
 - `a3gs.start()` : 電源On
 - `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - `a3gs.end()`
 - `a3gs.shutdown()`
- 日時取得（累積秒数）関数
 - `a3gs.getTime2(seconds)`;

// 3GIM(V2) sample sketch for Mega/Leonardo.. – getTime2

`#include "a3gim2.h"` a3gim2ライブラリの宣言

`#define baudrate 9600UL`
`const int powerPin = 7;`

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");

  Serial.print("Initializing..");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    uint32_t seconds;
    if (a3gs.getTime2(seconds) == 0) {
      Serial.print(seconds);
      Serial.println(" Sec.");
      Serial.println(time);
    }
    else
      Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
  }
}

void loop() {}
```

3 GIM電源ON/起動

日時取得（累積秒）

実行例（シリアルモニタ画面）

```
Ready.
Initializing.. Succeeded.
1477140604 Sec.
Shutdown..
```

7. サンプルスケッチ (http_get.ino)

本サンプルスケッチは、httpGETによってインターネット接続するサンプルです。

- 宣言 (a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- httpGET(インターネット接続) 関数
 - ・ `a3gs.httpGET(server,port,path,res,len);`

```
// 3GIM(V2) sample sketch for Mega/Leonardo.. -- httpGET
#include "a3gim2.h"
#define baudrate 9600UL
const int powerPin = 7;
const char *server = "www.arduino.org";
const char *path = "";
int port = 80;

char res[a3gsMAX_RESULT_LENGTH+1];
int len;
```

a3gim2ライブラリの宣言

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");

  Serial.print("Initializing..");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    Serial.print("httpGET() requesting.. ");
    len = sizeof(res);
    if (a3gs.httpGET(server, port, path, res, len) == 0) {
      Serial.println("OK!");
      Serial.print("[");
      Serial.print(res);
      Serial.println("]");
    }
    else {
      Serial.print("Can't get HTTP response from ");
      Serial.println(server);
    }
  }
  else
  {
    Serial.println("Failed.");
    Serial.println("Shutdown..");
    a3gs.end();
    a3gs.shutdown();
  }
}

void loop() {}
```

3 GIM電源ON/起動

httpGET

実行例 (シリアルモニタ画面)

```
Ready.
Initializing.. Succeeded.
httpGET() requesting.. OK!
[<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=Shift]
Shutdown..
```

7. サンプルスケッチ (check_service.ino)

本サンプルスケッチは、SIMカードによるサービス内容を表示するスケッチです。

- 宣言(a3gim2.hの読み込み)
 - #include <a3gim2.h>
- 電源ONと起動準備
 - a3gs.start() : 電源On
 - a3gs.begin() : 通信設定の確認
- 終了と電源OFF
 - a3gs.end()
 - a3gs.shutdown()
- SIMサービス取り出し
 - a3gs.getServices(status);

// 3GIM(V2) sample sketch for Mega/Leonardo.. -- getServices

#include "a3gim2.h" a3gim2ライブラリの宣言

#define baudrate 9600UL
const int powerPin = 7;

実行例 (シリアルモニタ画面)

```
Ready.  
Initializing.. Succeeded.  
Packet Service Only.  
Shutdown..
```

```
void setup(){  
  Serial.begin(baudrate);  
  delay(3000); // Wait for Start Serial Monitor  
  Serial.println("Ready.");  
  Serial.print("Initializing.. ");  
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {  
    Serial.println("Succeeded.");  
    int status;  
    if (a3gs.getServices(status) == 0) {  
      switch (status) {  
        case a3gsSRV_NO :  
          Serial.println("No Service.");  
          break;  
        case a3gsSRV_PS :  
          Serial.println("Packet Service Only.");  
          break;  
        case a3gsSRV_CS : // Never returned at R4.0  
          Serial.println("Voice Service Only."); // also SMS  
          break;  
        case a3gsSRV_BOTH : // Never returned at R4.0  
          Serial.println("Packet And Voice Services.");  
          break;  
        default :  
          break;  
      }  
    }  
    else  
      Serial.println("Failed.");  
    Serial.println("Shutdown..");  
    a3gs.end();  
    a3gs.shutdown();  
  }  
}  
  
void loop() {}
```

3 GIM電源ON/起動

SIMサービス取り出し

7. サンプルスケッチ (sample_tcpip.ino) ①

本サンプルスケッチは、TCP/IP機能を使ってhttp通信を行う事例です。
この例では、Arduinoの公式ページからtitleタグの中身を抜き出して、シリアルモニタへ出力します。

- 宣言 (a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- TCP/IP関連関数
 - ・ `a3gs.connectTCP(server, port)` ;
 - ・ `a3gs.write(data)` ;
 - ・ `a3gs.read(res, sizeof(res)-1)` ;

```
// 3GIM(V2) sample sketch for Mega/Leonardo..
// -- connectTCP/disconnectTCP/read/write
// A title is extracted from a homepage.

#include "a3gim2.h"

#define baudrate    9600UL
#define MAX_RETRY  3

const int powerPin = 7;
const char *server = "www.arduino.org"; // URL to extract a title
const int port = 80;
char res[200];

boolean getTitle(char *res);
```

a3gim2ライブラリの宣言

```
boolean getTitle(char *p)
{
    char *title;

    while (*p != '\0') {
        if (*p++ != '<')
            continue; // skip not tag
        if (strncmp((const char *)p, "title>", 6))
            continue; // skip not title tag
        // title tag found
        p += 6;
        title = p;
        while (*p != '\0' && *p != '<')
            p++;
        *p = '\0';
        Serial.print(server);
        Serial.print(" : Page title is ");
        Serial.print(title);
        Serial.println("");
        return true;
    }
    return false;
}
```

次頁へ続く

7. サンプルスケッチ (sample_tcpip.ino) ②

前頁の続き

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");
  Boolean sw=false;
  _redo:
  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    // Connect to server
    int nCount = 0;
    while (nCount++ < MAX_RETRY && a3gs.connectTCP(server, port) != 0)
    {
      Serial.println("connectTCP() can't connect, retry..");
      delay(100);
    }
    if (nCount > MAX_RETRY) {
      Serial.println("connectTCP() abort");
      goto _end;
    }
    // Send GET request
    a3gs.write("GET / HTTP/1.0\r\n");
    a3gs.write("HOST:");
    a3gs.write(server);
    a3gs.write("\r\n\r\n");
  }
```

```
// Receive response
do {
  int nbytes;
  if ((nbytes = a3gs.read(res, sizeof(res)-1)) < 0) {
    Serial.println("read() failed");
    break;
  }
} while (! (sw=getTitle(res)));
}else
  Serial.println("Failed.");

Serial.println("Shutdown..");
a3gs.end();
a3gs.shutdown();

delay(15000);
if(!sw) goto _redo; // Repeat

_end:
Serial.println("end of TCP/IP");
}

void loop() {}
```

実行例 (シリアルモニタ画面)

```
Ready.
Initializing.. Succeeded.
www.arduino.org : Page title is "Arduino - Open Source Products for Electronic Projects"
Shutdown..
```

7. サンプルスケッチ (set_baudrate.ino)

本サンプルスケッチは、3 GIMとのUARTでの通信速度を変更設定するサンプルです。

- 宣言 (a3gim2.hの読込)
 - `#include <a3gim2.h>`
- 電源ONと起動準備
 - `a3gs.start()` : 電源On
 - `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - `a3gs.end()`
 - `a3gs.shutdown()`
- 通信速度設定関数
 - `a3gs.setBaudrate(NEW_BAUDRATE)`

```
#include "a3gim2.h"

#define NEW_BAUDRATE 38400
#define CURRENT_BAUDRATE 9600
const int powerPin = 7;
```

実行例 (シリアルモニタ画面)

```
Ready.
Initializing.. Succeeded.
Baudrate was changed as 38400 bps.
Shutdown..
```

```
void setup()
{
  Serial.begin(CURRENT_BAUDRATE);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");

  Serial.print("Initializing..");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, CURRENT_BAUDRATE) == 0) {
    // @ if (a3gs.start() == 0 && a3gs.begin(0, NEW_BAUDRATE) == 0) {
    Serial.println("Succeeded.");
    if (a3gs.setBaudrate(NEW_BAUDRATE) == 0) {
      Serial.print("Baudrate was changed as ");
      Serial.print(NEW_BAUDRATE);
      Serial.println(" bps.");
    }
  }
  else
    Serial.println("Failed, baudrate was not changed.");

  Serial.println("Shutdown..");
  a3gs.end();
  a3gs.shutdown();
}

void loop(){}
```

3 GIM電源ON/起動

通信速度変更

7. サンプルスケッチ (tweet_sample.ino)

本サンプルスケッチは、ツイートするスケッチ例です。

- 宣言 (a3gim2.hの読込)
 - ・ `#include <a3gim2.h>`
- 電源ONと起動準備
 - ・ `a3gs.start()` : 電源On
 - ・ `a3gs.begin()` : 通信設定の確認
- 終了と電源OFF
 - ・ `a3gs.end()`
 - ・ `a3gs.shutdown()`
- ツイート関数
 - ・ `a3gs.tweet(token, message)`

```
#include "a3gim2.h"

#define baudrate          9600UL
const int powerPin = 7;    // 3gim power
pinIf not using power control, 0 is set.
const char *token = "YOUR_TOKEN_HERE";
const char *message =
"TWEET_MESSAGE_HERE";
//-- Note: can't tweet same message
continuously.
```

トークン設定

```
void setup()
{
  Serial.begin(baudrate);
  delay(3000); // Wait for Start Serial Monitor
  Serial.println("Ready.");

  Serial.print("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin(0, baudrate) == 0) {
    Serial.println("Succeeded.");
    Serial.print("tweet() requesting.. ");
    if (a3gs.tweet(token, message) == 0)
      Serial.println("OK!");
    else
      Serial.println("Can't tweet.");
  }
  else
    Serial.println("Failed, baudrate was not changed.");

  Serial.println("Shutdown..");
  a3gs.end();
  a3gs.shutdown();
}

void loop(){}

3 GIM電源ON/起動
```

実行例 (シリアルモニタ画面)

```
Ready.
Initializing.. Succeeded.
tweet() requesting.. OK!
Shutdown..
```



注意：
トークンについては、別途説明

【補足資料】 注意点

- ▶ 本製品で利用している3G通信モジュール（HL8548-G、以下3Gモジュールと呼ぶ）は、付属している3Gアンテナとの組合せで、日本の技適（技術基準適合証明※1）を取得をしています。よって、日本以外の海外での利用や、アンテナの取り換えやケーブルの取り外し等を行った使い方は、電波法違法利用となりますので、絶対行わないでください。
- ▶ 3GアンテナおよびGPSアンテナ、それにそれぞれのケーブルとコネクタは小さく、壊れやすいため、取扱いには、十分注意してください。特に、頻繁な取り外し・取り付けは行わないようお願い致します。（GPSアンテナ関係は別売オプションとなります）
- ▶ Arduinoと3GIMを接続して、電源をONあるいはリセットを掛けた場合、利用できるようになるまで15秒程度の時間が掛かります。
- ▶ 3Gモジュールは瞬間的に消費電力が高くなる場合があります、なるべく外部電源(ACアダプタ)をご利用頂くことをお勧めいたします。
 - ▶ ご利用されるパソコンの特性により、Arduino側へのUSB接続からの電力供給だけでは、3GIMが利用できない場合がありますのでご注意ください。動作が不安定となる場合は、外部電源(ACアダプタ)の利用をお勧めします。

※1 技術基準適合証明とは、特定無線設備（総務省令「電波法施行規則」で定める小規模な無線局に使用するための無線設備）が電波法令の技術基準に適合していることを証明（電波法第38条の2）することである。（Wikipediaより）

【ブレイク】 3GIM V2.2で使えるSIMカード

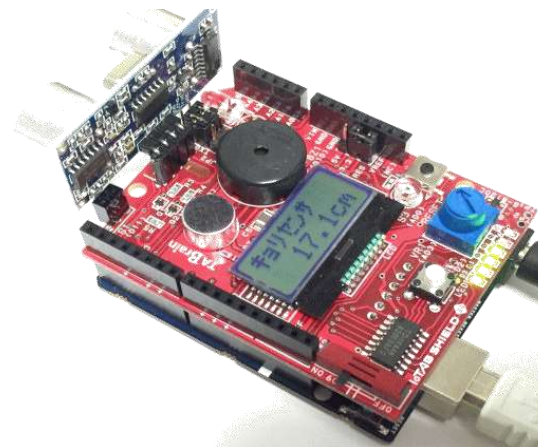
**オープン ワイヤレス アライアンスでは、
安価なSIMカードも利用可能に**

現時点 NTTドコモの回線を利用するMVNO様のSIMカードは、そのほとんどをご利用いただくことができます。

(参照：「\$PS」コマンド)



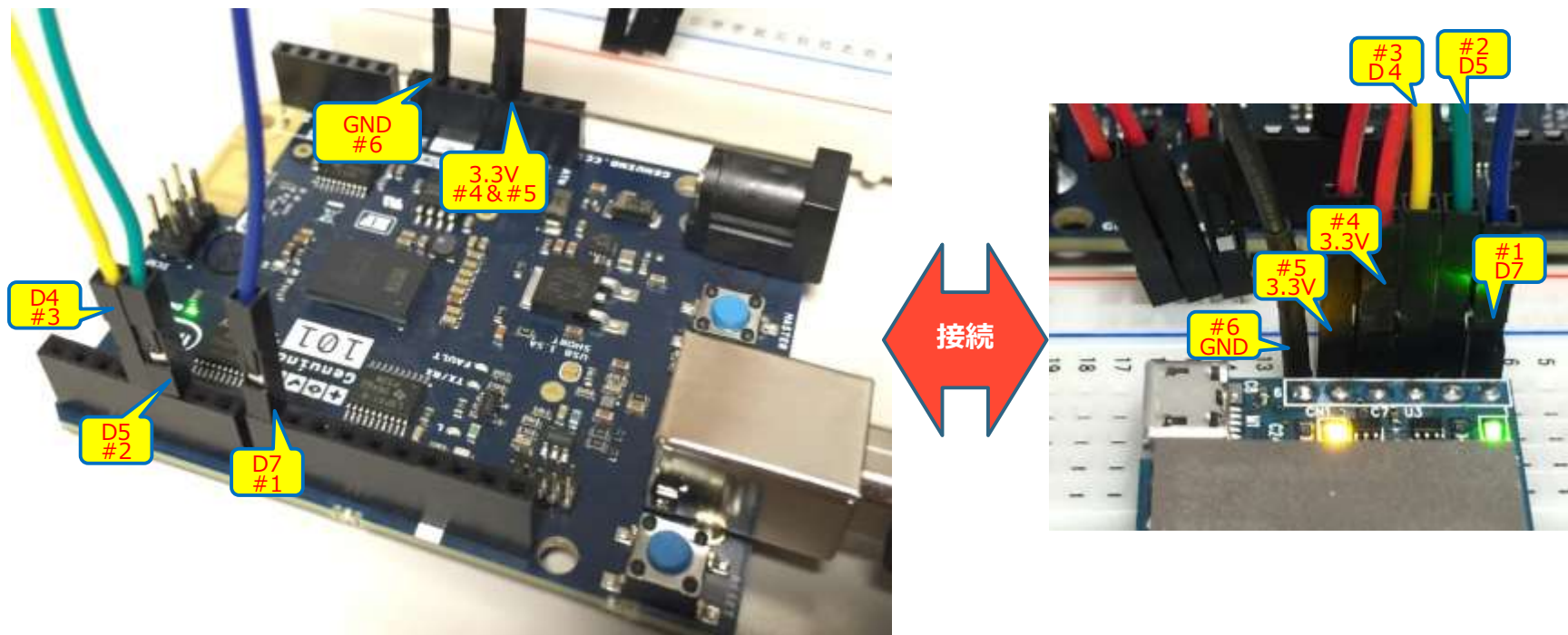
もくじ	
第1章	シリアルモニタによる操作
第2章	メール送信とツイッター送信
第3章	室内環境モニタリング
第4章	HTML+PHPによる遠隔操作
第5章	HTML+PHPによる遠隔操作の準備
第6章	HTML+PHPによる遠隔操作（回答例）



第Ⅲ編 IoT試作演習

第1章 シリアルモニタによる操作

1. Genuino/Arduino 101 との接続例



※Genuino 101の3.3V出力が 1 Aあるとのことです3GIMと接続してみました。

【注意】 電源（電流）不足になると 3 GIMが立ち上がらなかったり、インターネット接続ができないことがあります。
電源不足の場合には外部電源を取る必要があります。3GIMシールドやIoTABシールドなどをご利用ください。

2. Arduinoシリアルモニタ画面操作スケッチ

- ▶ Arduino UNO+3GIMシールド+3GIM V2.2を接続し、シリアルモニタ画面上でコマンド入力して、その結果を見てみることにしましょう。
- ▶ Arduinoのスケッチは以下のとおりです。

■ シリアルモニタ画面での3GIM入出力プログラム

(**monitor3gim.ino** : * 3 GIMシールド+Arduinoで利用可能)

```
#include <SoftwareSerial.h>
SoftwareSerial Serial3G(4, 5);
const unsigned long baudrate = 38400;

void setup() {
  Serial.begin(baudrate);
  Serial3G.begin(baudrate);
  pinMode(7, OUTPUT);
  digitalWrite(7, HIGH); delay(5);
  digitalWrite(7, LOW); //3GIMシールド電源ON
  Serial.println("Ready.");
}

void loop() {
  if (Serial3G.available() > 0) {
    char c = Serial3G.read();
    Serial.print(c);
  }

  if (Serial.available() > 0) {
    char c = Serial.read();
    Serial.print(c); // Echo back
    Serial3G.print(c);
  }
}
```

本サンプルスケッチは、シリアルモニタ画面で、コマンドをキー入力することで、応答（レスポンス）を表示確認できるもので、マニュアル操作でのコマンド/レスポンスが即座に見ることができます。

スケッチ名 : **monitor3gim.ino**
monitor_3GIM_G101.ino

■ シリアルモニタ画面での操作例

Ready.

Welcome to 3GIM(v2.2)

\$YV

\$YV=OK 3.3

\$YI

\$YI=OK 359516050532664

\$YR

\$YR=OK -77

\$YB

\$YB=OK 38400

\$LG x 1

\$LG=OK 35.641889 139.604123 040056.756 1 7 83.4

\$YT

\$YT=OK 2017/08/13 13:01:11

\$WG http://tabrain.jp

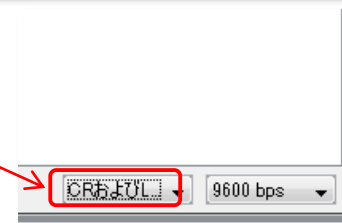
\$WG=OK 66

<meta http-equiv="refresh"

content="1;URL=http://tabrain.jp/new/">

赤字 : 入力
黒字 : 出力

補足 : Arduino IDEのシリアルモニタ画面の改行モードは「CRおよびLF」に設定のこと



3. Arduino上での簡単な利用例

■ ソフトウェア（前頁の [monitor3gim.ino](#)）をビルドし、Arduino UNO や Genuino/Arduino 101に書き込んで動かしてみてください。

■ 必要な部品：

- ① Arduino UNO もしくはGenuino/Arduino 101
（IDEは、「arduino.cc」からダウンロードしてください）
- ② 3 GIMシールドまたはIoTABシールド
もしくは、ジャンパワイヤおよびブレッドボード
- ③ マイクロSIMカード

■ 稼動テスト

- ① Arduino IDEを起動すると、シリアルモニタ画面上にメッセージ「Welcome to 3GIM(v2.2)」が表示されます。
- ② 3GIMの各\$コマンドを入力して、応答を確認してみてください。

第2章 メール送信とツイッター送信

1. メール送信

PHPが使用可能なFTPサーバーを用意し、sendmail.php、3gform.html、3gmakefile.phpの3ファイルをアップロードします。
(sendmailプログラム.zipを解凍してご利用ください。内容については第6章で説明しています。)

まず、インターネットブラウザのコマンドラインにて、以下を実行してみましょう。

http://**所有サーバーURL**/sendmail.php?email=**メールアドレス**&temp=**メール内容**

動作を確認したら、次は3 GIMで実現してみましょう。

■ サンプルスケッチ (sendmail.ino)

```
#include <a3gs.h>
char *server = "所有サーバーURL";
char *path = "/sendmail.php?email=メールアドレス&temp=メール内容";
const int port= a3gsDEFAULT_PORT;

void setup() {
  int len = 500;
  char res[501];
  Serial.begin(9600);
  Serial.println("start");
  if(!(a3gs.start()==0 && a3gs.begin()==0))
  { Serial.println(" 3G error"); while(1); }
  Serial.println("succeeded");
  if( a3gs.httpGET(server,port,path,res,len)==0 ) {
    Serial.println(res);
  } else { Serial.println("error");}
  Serial.println("end ");
}

void loop() { }
```



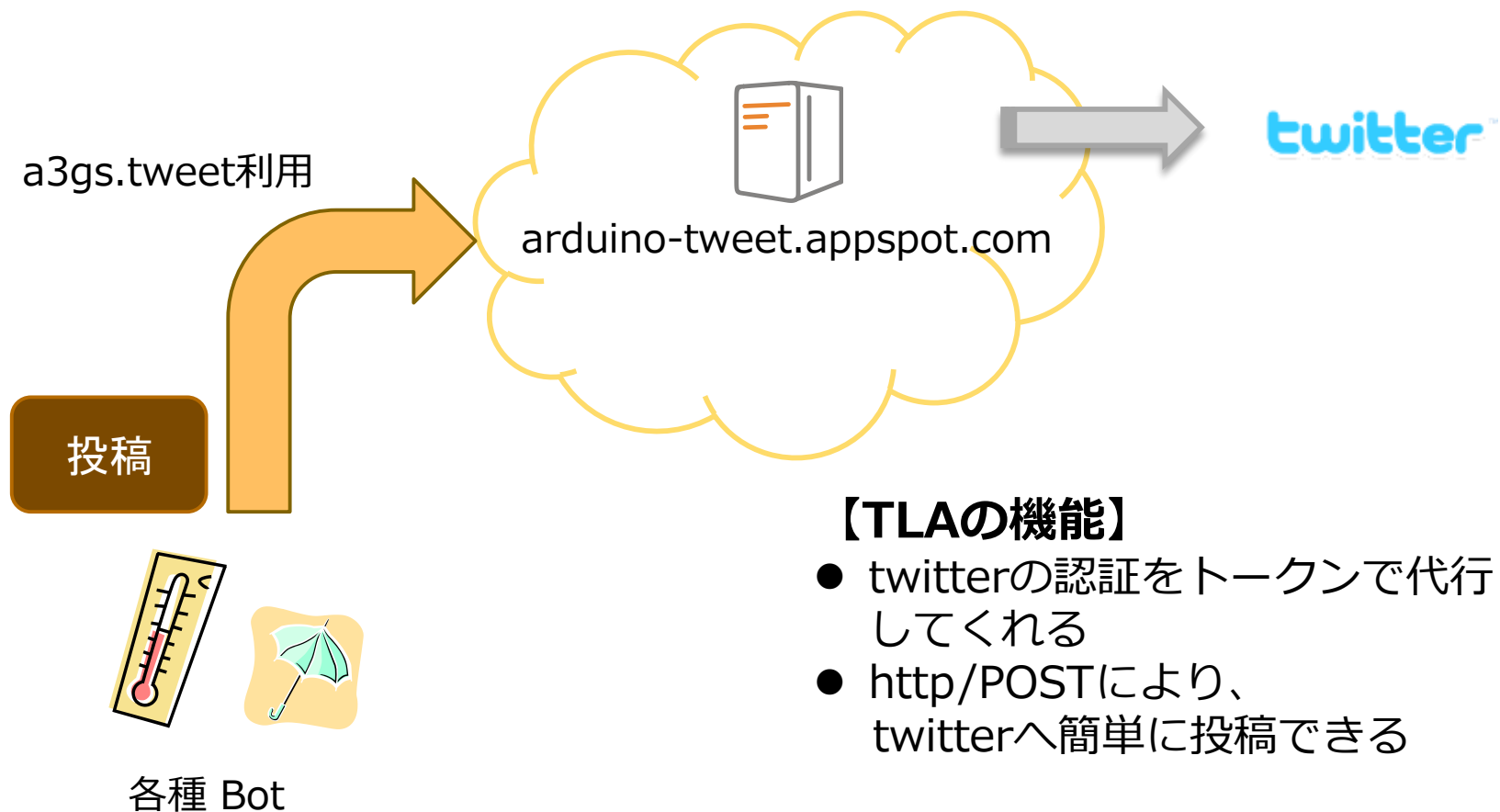
メールが飛んできます

```
3GIM sensor E-mail
一昨日 14:49

3GIM send ALARM
DATE = 2016-06-22  TIME = 14:49:39
cont : TEMP= 18.56 C
```

メール内容例

2. TLA(Tweet Library for Arduino)の利用イメージ



3. TLAの利用手順 ①

- ①ブラウザで、下記のサイトにアクセスする
<http://arduino-tweet.appspot.com/>



3. TLAの利用手順 ②

②ツイッターのアカウント情報を入力する

Authorize Arduino to use your account?

この連携アプリを認証すると、次の動作が許可されます。

- タイムラインのツイートを見る。
- フォローしている人を見る。新しくフォローする。
- プロフィールを更新する。
- ツイートする。

ユーザー名、またはメールアドレス
パスワード

☐ 保存する。パスワードを忘れた場合はこちら

連携アプリを認証 キャンセル

この連携アプリを認証しても、次の動作は許可されません。

- ダイレクトメッセージを見る。
- Twitterのパスワードを見る。

既定のアプリ連携からいつでも連携アプリの許可を取り消すことができます。
連携アプリを認証することでTwitterのサービス利用規約に同意したことになります。また、いくつかの連携アプリの利用情報はTwitterにも共有されます。詳細についてはプライバシーポリシーをご覧ください。

Arduino
開発者: NewCat
arduino-tweet.appspot.com/
Twitter Library for Arduino: post a tweet easily using Arduino

新規登録

3. TLAの利用手順 ③

③トークンを記録しておく



4. TLAの利用する上での注意点

▶ Notice

- ▶ The library uses this site as a proxy server for OAuth stuff. Your tweet may not be applied during maintenance of this site.
- ▶ **Please avoid sending more than 1 request per** overload the server.
- ▶ Twitter seems to reject repeated tweets with the same content (returns error 403).

メンテナンス中は、利用できない
(常時利用は不可)

1分間に1回に限定

▶ Reference

- ▶ See [Arduino: Playground](#)

同じツイートはエラーになる
(エラーコード403)

▶ License

- ▶ Tweet Library for Arduino is distributed under MIT license.(GPL v2)

参照サイト
<http://www.arduino.cc/playground/Code/TwitterLibrary>

ライセンスの所在
(無断での公開は禁止)

5. ツイッター送信

インターネットブラウザのコマンドラインにて、以下を実行してみましょう。

http://arduino-tweet.appspot.com/update?token=YOUR_TOKEN&status=MESSAGE

これを 3 GIM で実現してみましょう。

■ サンプルスケッチ (tweet.ino)

```
#include "a3gs.h"

const char *token = "YOUR_TOKEN";
char message[] = "MESSAGE";

void setup()
{
  Serial.begin(9600);
  Serial.println("Ready.");
  Serial.print("Initializing a3gs.. ");
  if (a3gs.start()==0 && a3gs.begin()==0)
    Serial.println("Succesdeded.");
  else{ Serial.println("Failed."); while (1);} // STOP HERE

  if(a3gs.tweet(token,message)==0)
    Serial.println("tweet OK");
  else
    Serial.println("tweet NG");
}

void loop(){}
}
```



ツイートされます。

※トークンは、予めTwitterのIDを登録したのち、arduino-tweet.appspot.comから取得してください。

第3章 室内環境モニタリング

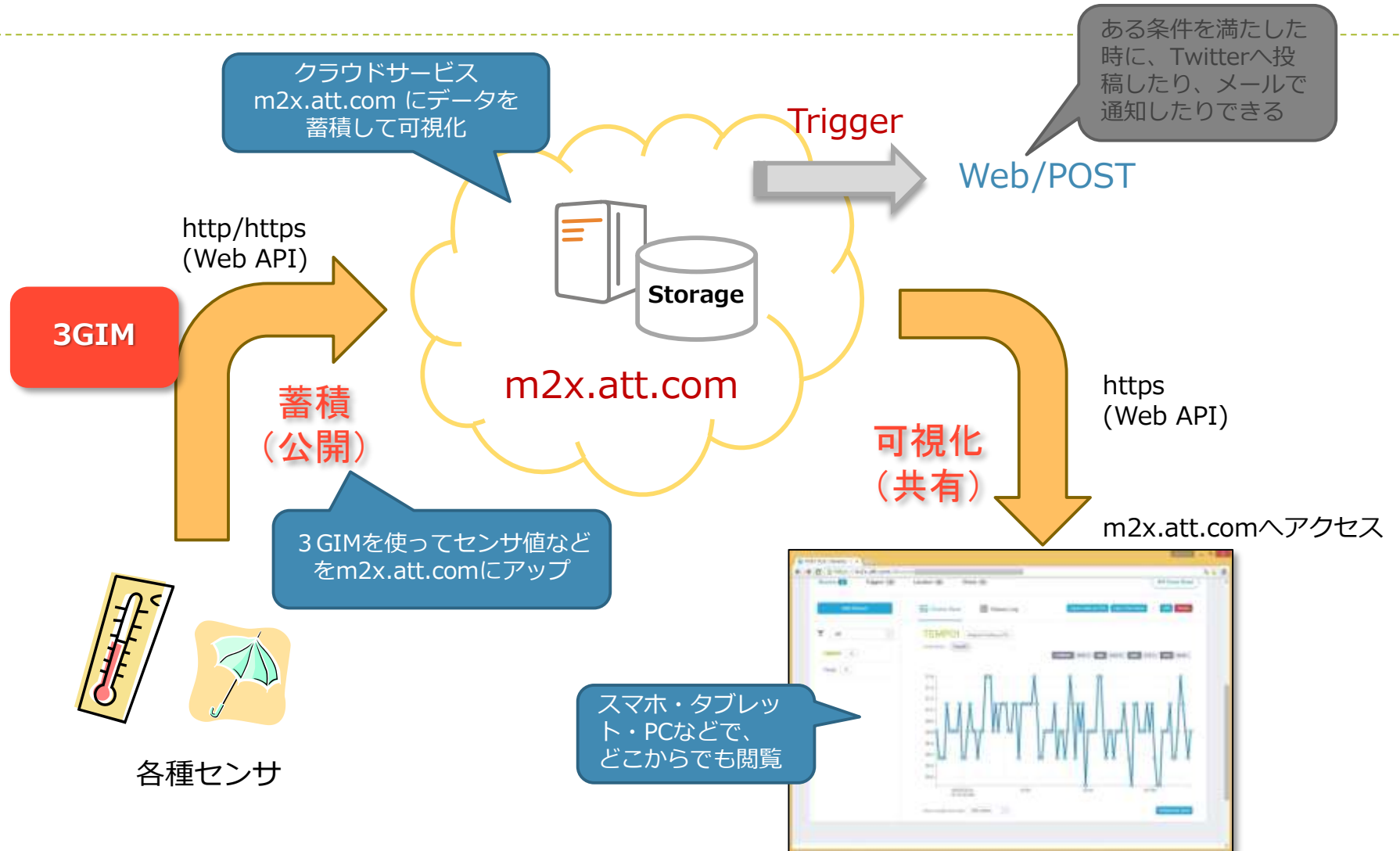
3Gを使った演習。
温度、湿度を定期的に計測し、クラウドサービスにアップして、
ブラウザで可視化するデモ。ある条件を満たしたときにツイー
トすることもできる。

クラウドサービスには、
今回M2Xを利用

1. 室内環境モニタリング（概要）

- ▶ IoTABシールド上の温度センサ、照度センサで温度と照度を計測して、3G経由でクラウドサービスM2Xへ一定間隔でアップロードする
- ▶ アップロードした温度と湿度は、M2Xにログインしてアクセスすると、折れ線グラフとしてみる事ができる
- ▶ 以下の手順で、演習を実施する：
 1. M2Xにユーザ登録をする
 2. M2X上にデバイスとストリームを登録する
 - ・ IDとAPI KEYを取得する
 3. スケッチを作り、Arduinoへ書き込む
 - ・ 取得したIDとAPI KEYを使用する
- 1. 実際に動かしてみて、温度と湿度をグラフで可視化できることを確認する

2. 室内環境モニタリングの全体イメージ



3. M2X (AT&T IoTサービス) の利用手順

M2X (AT&T IoTサービス) は、2013年から米国通信事業最大手のAT&Tが、M2MおよびIoTビジネスに向けたサービスを開始したものです。

ArduinoやRaspberryPi、Mbedなど、多くのオープンソースハードウェアで利用できる環境を提供したものとなっています。

フリーで使え、センサデータの蓄積・グラフ表示、データのダウンロードなどができるようになっています。

ただ、時間設定が、世界標準のみで行なっていて、日本時間での表示が現時点できないのが難点となっています。

ただ、登録の簡単さは

まだ、英語版しかありませんが、グラフ表示やデータのアップやダウンロードだけの利用と考えると、問題なく簡単に使うことができます。

ここでは、本 3 GIMとセンサを使い、この m2x.att.com にデータをアップしていくサンプルをご紹介します。

詳細な規約等は、M2X関連の公開情報等をご参照ください。

① ユーザの登録

② device の追加

③ stream の追加

④ x-m2x-keyの確認

4. M2X (AT&T IoTサービス) のID登録

The image shows a screenshot of the AT&T M2X website with several annotations in red boxes and arrows:

- Top Navigation:** The top navigation bar includes "AT&T IoT Platform", "Services", "Solutions", "Solution Providers", and "Contact". A "LOGIN" button is located in the top right corner.
- Main Header:** The header features the "M2X" logo, "FEATURES", and "SHOWCASE" links.
- URL Annotation:** A red box contains the URL <https://m2x.att.com/> with the text "にアクセス" (Access here).
- Hero Section:** The main heading reads "Build Intelligent, Enterprise Grade Connected Device Solutions". Below it, a paragraph states: "AT&T M2X provides time-series data storage, device management, message brokering, event triggering, alarming, geo-fencing, and data visualization for your Industrial Internet of Things (IIoT) products and services."
- Call to Action:** A green button labeled "GET STARTED FOR FREE" is highlighted with a red box. A red arrow points from this button to a red box containing the text "ID登録へ" (Go to ID registration).
- Login Form:** The login form is titled "Login with your AT&T IoT Platform Account". It includes fields for "Username / Email" and "Password", a "Forgot your username or password?" link, and a "LOGIN" button.
- Alternative Login:** Below the login form, there is a section "Or login using:" with buttons for "GITHUB" and "AT&T DEVELOPER".
- Registration Link:** A red box highlights the link "Create an account here." with the text "ID新規登録" (New ID registration).
- Activation Email:** A red box highlights the "Confirm your email and activate your account" button in an email template, with an arrow pointing to a red box containing the text: "必要情報を入力・送信後、設定アドレスにメールが届く。有効化 (activate) ボタンを押下して登録完了。" (After entering and sending necessary information, an email will arrive at the designated address. Press the activation (activate) button to complete registration.)

Page-Footer: The footer includes the "Open Wireless Alliance" logo and text.

5. デバイス (Device) の作成登録

The image shows the M2X 'Create Device' form with several Japanese annotations in red speech bubbles:

- デバイス画面へ** (To Device screen): Points to the 'Devices' tab in the top navigation bar.
- 新しいデバイス登録画面へ** (To new device registration screen): Points to the 'Create New' button and the 'Device' option in the dropdown menu.
- デバイス名登録** (Device name registration): Points to the 'Device Name' input field, which contains the example text 'e.g. Geiger Counter'.
- 非公開：個人利用** (Non-public: personal use): Points to the 'Private Device' radio button option.
- 公開：共有利用** (Public: shared use): Points to the 'Public Device' radio button option.

The 'Create Device' form itself includes the following fields and options:

- Device Name**: Input field with example text 'e.g. Geiger Counter'.
- Device Description (optional)**: Text area with placeholder 'Describe your Device...'.
- Device Serial**: Input field with example text 'eg. 1234abc'. Below it, a note states: 'An alphanumeric identifier for this device. Serial numbers are non-unique at the account level, while Devices contained in a Distribution must have a unique Device Serial.'
- Tags**: Input field with placeholder 'Add Tags to your Device'. Below it, a note states: 'Add multiple tags by separating each tag with a comma'.
- Visibility**: Two radio button options:
 - Private Device**: Selected by default. Description: 'You use API keys to choose if and how you share data from a Device.'
 - Public Device**: Description: 'You agree to make this device publicly available under the CC0 1.0 Universal license.'
- Buttons**: 'Cancel' and 'Create' buttons at the bottom right.

※ deviceIDは、英数文字のキーワードとして自動的に設定されます。

6. ストリーム（StreamID）の作成登録

The image shows two views of the 'Add a Stream' interface. The left view is a sidebar menu with 'Streams 1' and 'Triggers 0'. A red box highlights the 'Add Stream' button, with a callout saying '新しいStream 登録画面へ' (To the new Stream registration screen). The right view is the 'Add a Stream' form itself, with various fields and callouts:

- Stream ID**: A text input field with the example 'e.g. temperature, garage-humidity'. A callout points to it saying '<StreamID>'. Below the field, it says 'Stream IDs can only contain letters, numbers, underscores, and dashes — no spaces or special characters are allowed.'
- Display Name**: A text input field with the example 'e.g. garage humidity'. A callout points to it saying 'Stream表示名' (Stream display name).
- Stream Type**: Two radio buttons, 'Numeric' (selected) and 'Non-Numeric'. A callout points to them saying '数値・非数値の選択' (Selection of numeric/non-numeric).
- Unit & Symbol (optional)**: Two text input fields. The 'UNIT' field has the example 'e.g. Ohm, Watt'. The 'SYMBOL' field has the example 'e.g. W, a, Ω'. A callout points to the symbol field saying '単位設定' (Unit setting).

At the bottom of the form are 'Cancel' and 'Save' buttons.

※ StreamIDは、入力した名前が設定されます

7. デバイスIDとストリームIDの登録

The screenshot displays the AT&T M2X Developer portal interface. The browser address bar shows `https://m2x.att.com/dev/`. The main content area shows the details for a device named "OfficeTemp&Light".

Annotations on the screenshot include:

- デバイス名** (Device Name): Points to the device name "OfficeTemp&Light".
- デバイス作成画面でデバイス作成** (Create device on device creation screen): Points to the "Edit" and "Delete" buttons.
- <deviceID>**: Points to the "DEVICE ID" field.
- <x-m2x-key>**: Points to the "PRIMARY API KEY" field.
- Stream表示名** (Stream display name): Points to the "Stream ID" field "Temp".
- デバイス作成登録名称** (Device creation registration name): Points to the "OfficeTemp" part of the stream name.

The interface also shows tabs for "Overview", "API Keys", "API Request Log", and "Trigger Log". The "Overview" tab is active, displaying "Streams 1", "Triggers 0", "Location 0", and "Charts 1". The "Add Stream" button is visible. The "Charts View" and "Values Log" sections are also present.

8. 無料での利用枠・制限について

無料の利用枠・制限

DEVELOPER <small>FREE</small>		PROFESSIONAL <small>PRO</small>			CUSTOM
FREE (limit of 10 devices)		\$99 annually (unlimited paid devices, 20 free High Tier) plus Premium AT&T Developer Account Charges begin after the first 1,000 data points are written per device			
USAGE TIER	FREE	LOW	MEDIUM	HIGH	
Data Points Written	Up to 100,000 (per device/month)	1 to 100,000 (per device/month)	100,000 to 500,000 (per device/month)	500,000 to 1 Million (per device/month)	Interested in a custom pricing plan? Get in touch with our sales team.
Data Points Read	Unlimited*	Unlimited*	Unlimited*	Unlimited*	
Device Cost	\$0 (per device/month)	\$0.50 (per device/month)	\$2.50 (per device/month)	\$5.00 (per device/month)	

* See the M2X API documentation for detailed information on rate limiting.

9. M2Xへのデータアップロードの方法

M2Xへのセンサ値のアップロードは、httpPOST()関数を使って行う。
下記の表に、httpPOST()関数の引数で指定する内容を下記に説明する：

server:

http://api-m2x.att.com/

path:

v2/devices/<deviceID>/updates/

header:

"X-M2X-KEY:<x-m2x-key>\$r\$nContent-Type:application/json\$r\$n"

body:

"{"values\$": {"<streamID>\$": [{"timestamp\$": "<date-time>\$", "value\$": "<val>\$"}]}}"

※ 変数の説明

<deviceID>	: デバイスID
<streamID>	: ストリームID
<x-m2x-key>	: M2Xキー
<val>	: アップロードするセンサの値
<date-time>	: 日時（文字列） 例 “2015-09-20T23:55:36\$+09:00”（\$+は特殊文字）

※ 日時は、日本時間を登録（ただしM2Xでの表示は、世界標準時となる）

10. サンプルスケッチ ①

スケッチ (M2X_3GIMshield_G101_Library_Restart_sample.ino)

```
/*
 *Indoor Environment Monitoring
 */

#include "a3gim2.h"

const char *server = "api-m2x.att.com";
const char *path = "v2/devices/<DEVICE_ID>/updates/";
const char *header="X-M2X-KEY:<PRIMARY_API_KEY>
$r$nContent-Type:application/json$r$n";
const int port = a3gsDEFAULT_PORT;
const char *bodyTemplate = "{ \"$\"values$\" : { \"$\"temp$\" :
[{ \"$\"timestamp$\" : $\"%s$\", \"$\"value$\" : $\"%d.%d$\"}],
$\"light$\" : [{ \"$\"timestamp$\" : $\"%s$\", \"$\"value$\" :
$\"%d$\"}] } }";
const int powerPin = 7; // 3gim power pin
const int tempPin = 1, illPin = 0; // Sensors pin number
char res[100], body[200];
int len;

void setup() {
  delay(3000); // Wait for Start Serial Monitor
  Serial.begin(9600);
  Serial.println(">Ready.");

  Serial.println("Initializing.. ");
  if (a3gs.start(powerPin) == 0 && a3gs.begin() == 0)
    Serial.println("Succeeded.");
  else {
    Serial.println("Failed.");
    while (1); // STOP
  }
}
```

M2Xから取得した
IDとKEYを
正確に置き換える

データをアップロード
する時のJSON形
式のテンプレート

3GIMを初期化して
利用できるようにする

```
void loop() {
  char datetime[30];
  Serial.print(">httpPOST requesting: ");
  if (getDateAndTime(datetime) != 0) {
    Serial.println("getDateAndTime() Failed.");
    return;
  }
  int temp = getTemperature();
  int ill = getIlluminance();

  sprintf(body, bodyTemplate, datetime, (temp/10), (temp%10),
datetime, ill);
  Serial.println(body);
  len = sizeof(res);

  int stat;
  if ((stat = a3gs.httpPOST(server, port, path, header, body, res,
&len)) == 0) {
    Serial.println("Succeeded.");
    Serial.print(">Response=");
    Serial.print(res);
    Serial.println("]");
  }
  else {
    Serial.print("Failed: ");
    Serial.println(stat);
  }

  delay(60000); // take an interval
}
```

現在の日付・時刻、温度、
照度を取得して、
JSON形式に整形する

httpPOST()関数
でデータをアップする

概ね60秒間隔で
データをアップする

10. サンプルスケッチ ②

日付と時刻を読み出して、所定の形式へ変換する

```
int getDateAndTime(char *datetime) {  
  char date[a3gsDATE_SIZE], time[a3gsTIME_SIZE];  
  if (a3gs.getTime(date, time) != 0) {  
    Serial.println("getTime() Failed.");  
    return 1; // error  
  }  
  date[4] = '-';  
  date[7] = '-';  
  sprintf(datetime, "%sT%s+09:00", date, time);  
  return 0; // ok  
}
```

```
int getTemperature(void) { // get temperature(x10 C)  
  float mV = analogRead(tempPin) * 3.23;  
  return (((mV - 1712.5) / (-8.20)) * 10.0);  
}
```

温度センサから温度を取得して、10倍にした整数を返す
(単位は℃)

```
int getIlluminance(void) { // get illuminance(x1 lux)  
  float mV = analogRead(illPin) * 3.23;  
  return (mV / 2.9);  
}
```

照度センサから照度を取得して返す
(単位はlux)

【正しく実行されている時のシリアルモニタの表示例】

```
> Ready.  
Initializing.. Succeeded.  
> httpPOST requesting: {"values":  
  {"temp": [{"timestamp": "2016-05-14T17:24:31+09:00", "value": "27.2"}],  
  "ill": [{"timestamp": "2016-05-14T17:24:31+09:00", "value":  
    "235"}]} }  
Succeeded.  
> Response=[{"status": "accepted"}]
```

Arduinoのスケッチの中で使用する **sprintf()**関数では、浮動小数点が使えない場合があります。また、8ビットマイコンを使用しているArduino UNOやMegaでは実数演算が遅いため、左記のgetTemperature()関数のように、あえて値を10倍にして整数として取り扱う方法が有効です。

11. M2Xにデータをアップロードした例

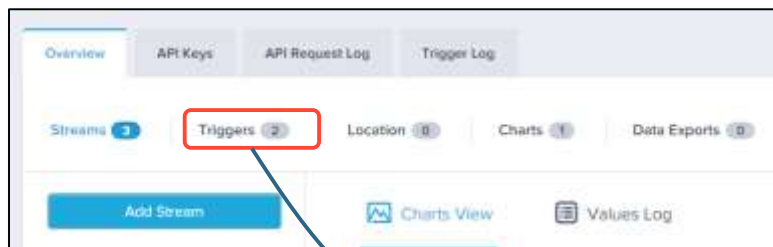


12. M2Xからトリガーでツイートする方法

M2Xにアップしているセンサの値をトリガーにして、他のサービス（例えばTwitterなど）と連携させることができる。

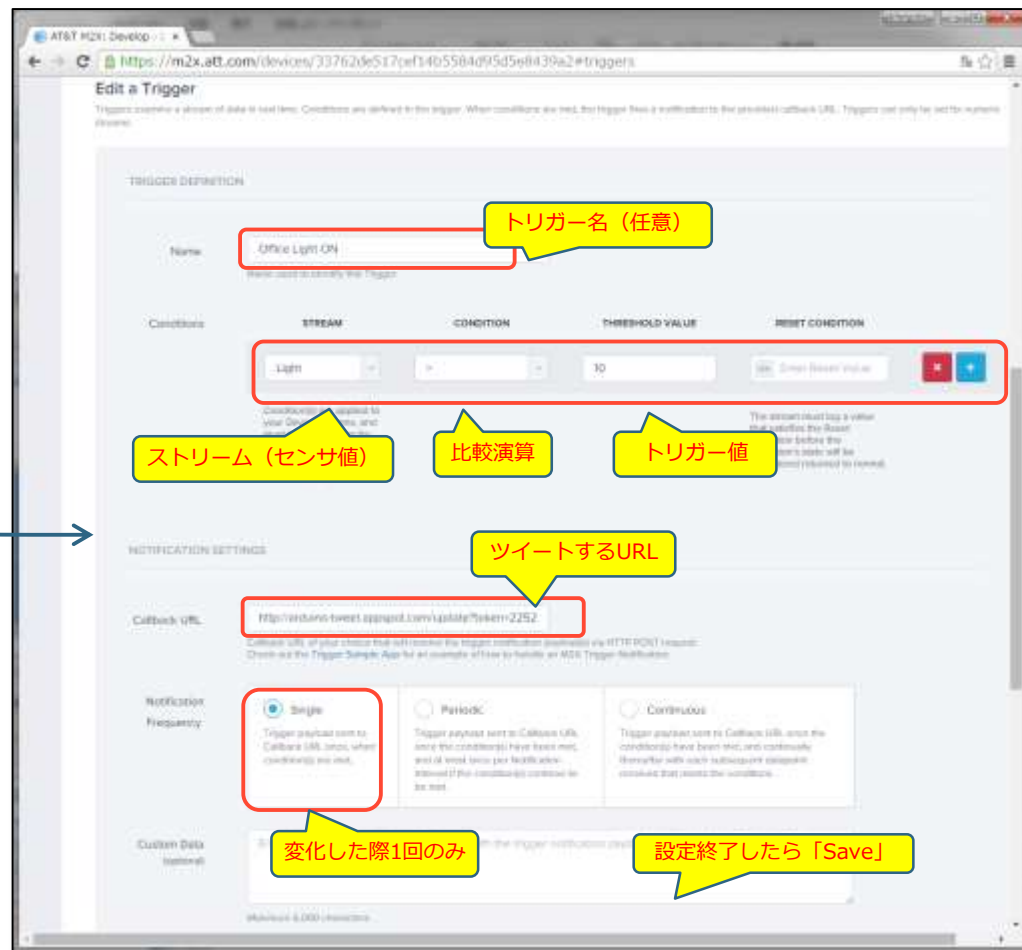
トリガーの設定

トリガー設定の選択



トリガーは、M2Xに送られてきたセンサ値をチェックして、指定されている条件を満たした時にアクションを起こす機能です。

この例の場合、センサ値がある閾値より大きいかどうかで、指定されたURLをHTTP/POSTします。
ここでは、センサ値(温度)がある閾値を超えていたら、ツイートする例です。



13. M2Xからトリガーでツイートする方法 twitterの代理認証サービス「StewGate U」

- twitterの認証OAuthを代行してくれるWebサービス
<http://stewgate-u.appspot.com/>

TLA以外の
認証代理サ
ービスの紹介

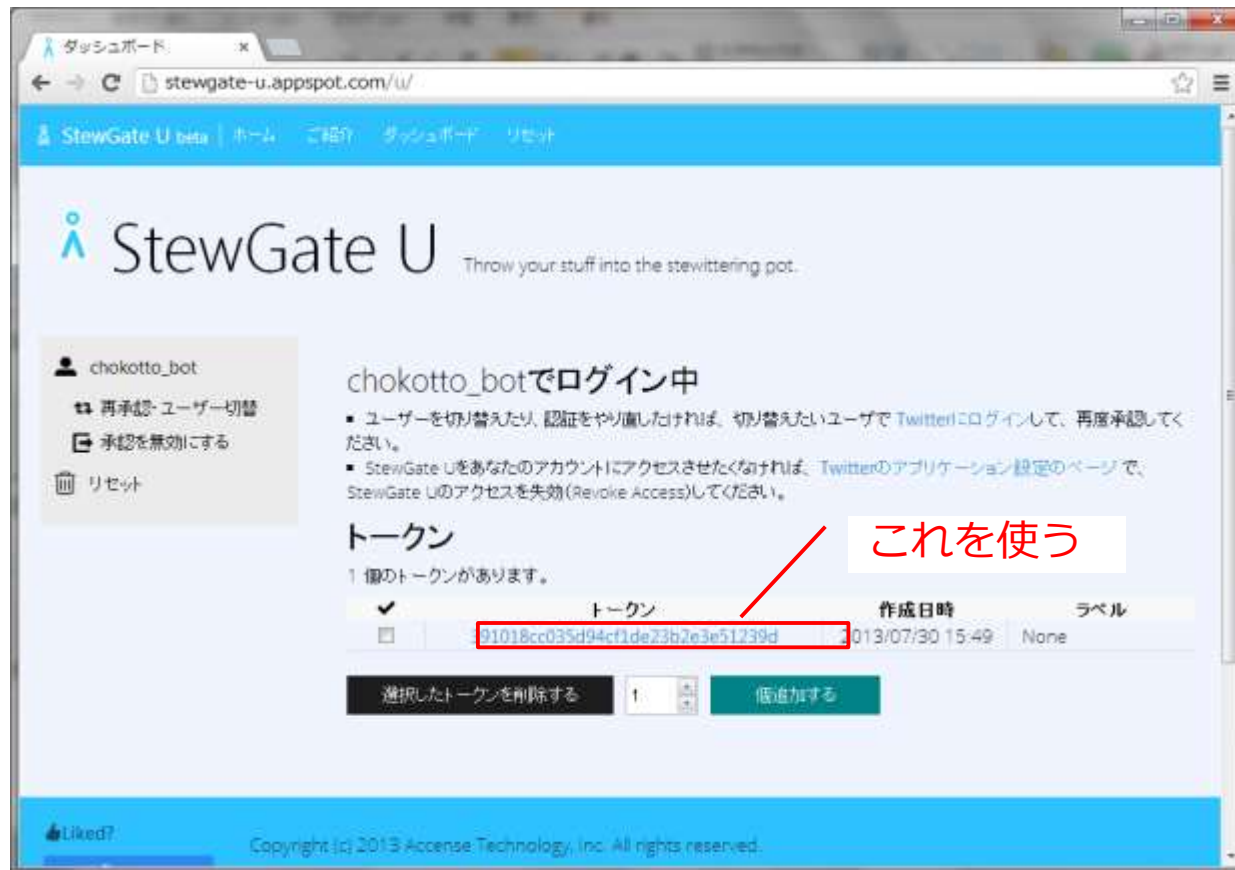


これをクリックして、
Twitterのログイン画面へ遷移し、
ログインする。

13. M2Xからトリガーでツイートする方法

twitterの代理認証サービス「StewGate U」

- twitterにログインしてStewGate Uに権限を許可することで、トークンを取得、これを使って投稿する



13. M2Xからトリガーでツイートする方法②

▶ トリガーとして設定するURL

- ▶ 「StewGate U」というサービスを使ってツイートする例
- ▶ あらかじめ登録・入手したトークンtokenを使って、下記の内容をURLとして指定する

`http://stewgate-u.appspot.com/api/post/?_t=<token>&msg="<文字列>"`

- ▶ 文字列はURLエンコードしておく必要があります。URLエンコードとは、「URLやクエリ文字列などの一部としては使用できない記号や特殊な文字を、使用できる文字の特殊な組み合わせによって表記する変換規則」で、下記に例を示す：

- ✓ 半角スペース → %20
- ✓ 半角\$ → %24
- ✓ 半角< → %3c 等

第4章 HTML+PHPによる遠隔操作

遠隔操作で 3 GIM上の入出操作 + メール送信

1. LEDを点滅させる

* ARDUINO+ 3 GIM上にあるLEDを遠隔操作で
On/Off させる

課題：D2-D7番ピンのLEDを遠隔からOn/Offする



2. ブザー（スピーカ）を鳴らす

* ARDUINO+ 3 GIM上にあるスピーカを遠隔操作で 5 秒ほど鳴らす

課題：D 8 番ピンとGNDに接続したスピーカを5秒ほど鳴らす



3. 温度センサの値をメールで受信する

* ARDUINO+ 3 GIM上にある温度センサ値を自分のメールアドレスに送る

課題：指定したメールアドレスに「温度センサーの値」を返す



4. 総合メニューによる遠隔操作

* ARDUINO+ 3 GIM上にある温度センサ値を自分のメールアドレスに送る



5. プログラミング

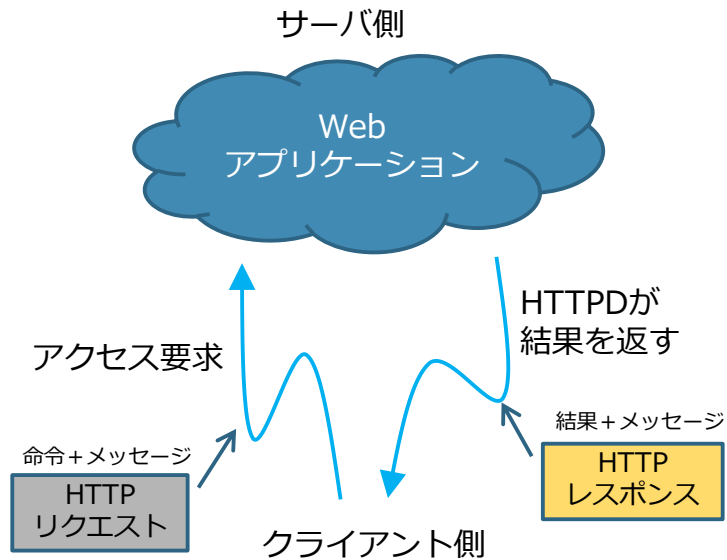


第5章 HTML+PHPによる遠隔操作の準備

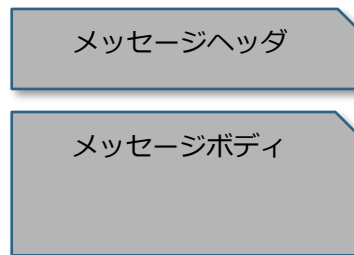
遠隔操作で 3 GIM上の入出操作 + メール送信

1. HTTPの基礎（1）

▶ リクエストとレスポンス



▶ メッセージの構造



▶ HTTPリクエスト

命令（例）メソッド（GET または POST）
GET /index.html HTTP/1.1

リクエスト ヘッダ

リクエスト ボディ

GETメソッドによるデータ送信（パラメータを渡す方法）

`http://www.***.co.jp/index.html?para1=123¶2=345`

※ ここでは、変数「para1」に123を、「para2」に345を値として渡す

POSTメソッドによるデータ送信（パラメータを渡す方法）

HTMLフォームで、「method」属性にPOSTを与えて実現

`<FORM action="http://www.tabrain.jp/form.php" method="POST">`

`***`

`***`

`</FORM>`

▶ HTTPレスポンス

レスポンスコード
HTTP/1.1 200 OK

レスポンス ヘッダ

レスポンス ボディ

1. HTTPの基礎 (2)

POSTによるデータ送信とレスポンス

①

```
<FORM action="http://www.tabrain.jp/form.php" method="POST">
  名前: <INPUT type="text" name="name" /><BR>
  年齢: <INPUT type="text" name="age" /><BR>
  <INPUT type="submit" name="buttonName" value="送信する" />
</FORM>
```

サーバ側に
あっても同じ

このファイル名を test.html として
IEなどのブラウザで実行すると

②

名前と年齢を入力し
「送信する」ボタン
を押す

サーバ側

サーバ側のプログラムが起動（この場合、
<http://www.tabrain.jp/form.php>）

③

```
<HTML>
<BODY>
<H3> こんにちは<?=$_POST["name"] ?> さん </H3>
<H3> 年齢: <?=$_POST["age"] ?> 歳 </H3>
</BODY>
</HTML>
```

④

クライアント側

サーバ側



③ サーバ側
プログラム起動

① Webアプリ
ケーション起動

④ クラウド側の
プログラム起動

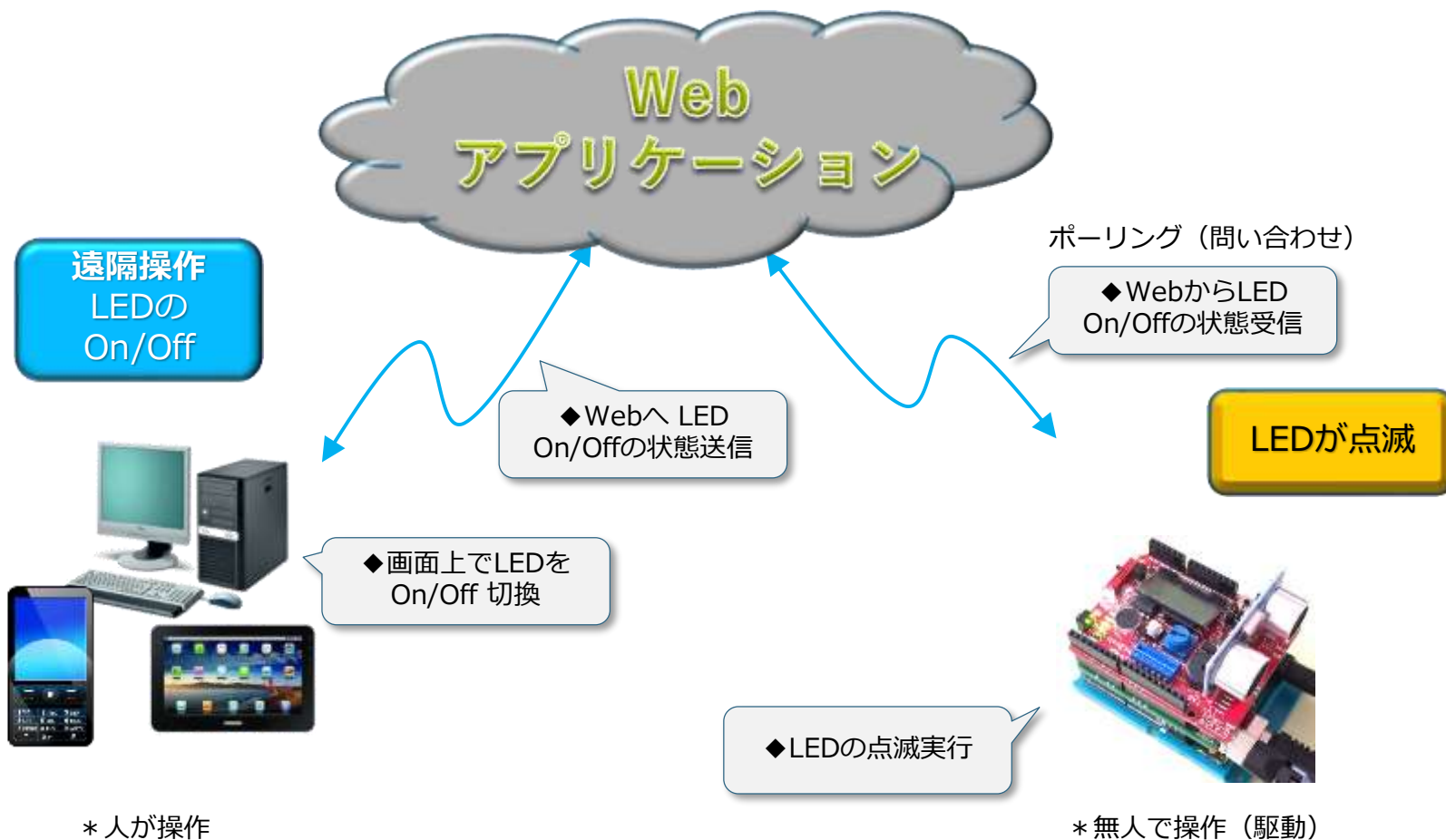
② メニュー起動



クライアント側

2. 3 GIM上のLED配線と課題解決

* 端末（PC、タブレット、スマホなど）から、遠隔地にある3 GIM上のLEDを点滅させる

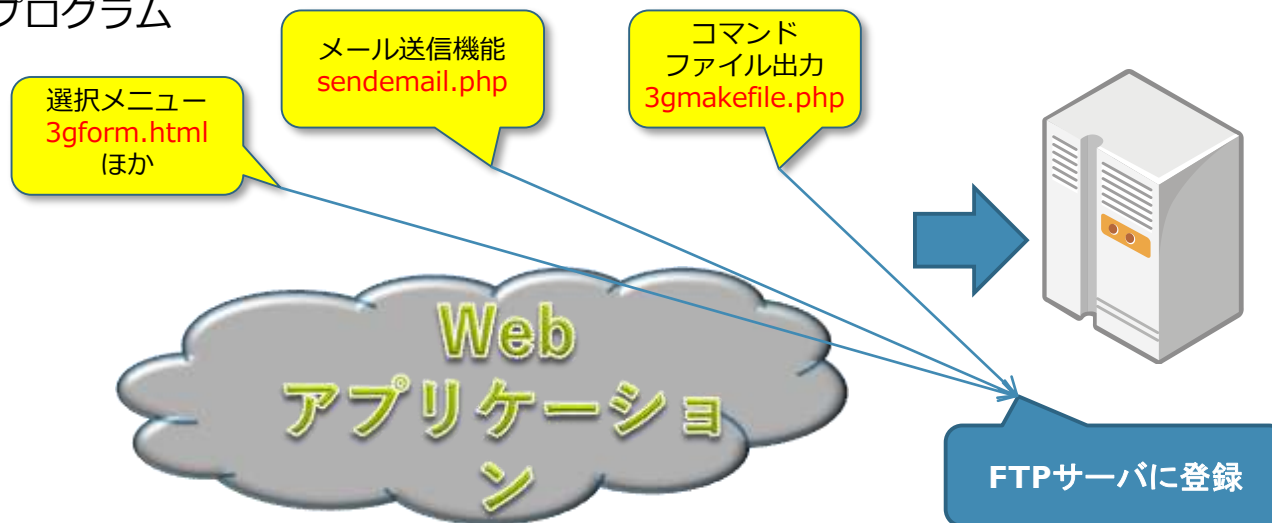


第6章 HTML+PHPによる遠隔操作（回答例）

遠隔操作で 3 GIM上の入出操作 + メール送信

1. 各種関連するプログラム

■ サーバ側プログラム



■ IoTデバイス側プログラム



デバイス側
remote3g.ino
ほか

2. アップロードする3ファイル

sendmailプログラム.zip

■メール送信機能 (sendmail.php)

温度センサ値をメールにて宛先に送信するプログラム

```
<HTML>
<HEAD><TITLE> 3GIM send TEMP sensor e-mail </TITLE><HEAD>
<BODY>

<H3> 3GIM TEMP sensor get </H3>
<?php
if(mail($_GET["email"], // to
    'Hi 3GIM sensor E-mail' ,// タイトル
    ' 3GIM ALARM ' . "¥r¥n" . 'DATE = ' . date('Y-m-d') .
    ' TIME = ' . date('H:i:s') . "¥r¥n" .
    'TEMP = ' . $_GET["temp"] ,//本文
    'From: 3GIM<temp@tabrain.jp>' . "¥n" .
    'X-Mailer: PHP/' . phpVersion()))
{ echo '<B>SUCCESS TO SEND</B><BR>';}
else
{ echo '<B>faile to mb_send_mail</B><BR>'; }
?>

</BODY>
```

■選択メニュー (3gform.html)

遠隔制御のためのメニュー操作画面

```
<form action = "3gmakefile.php" method = "post">
<p> 3GIM menu<br>
<select name="cmd" id="cmd">
<option value="LED on">LED on</option>
<option value="LED off">LED off</option>
<option value="Send E-mail">Send E-mail</option>
<option value="Speaker Alarm">Speaker Alarm</option>
</select></p>
<input type="submit" value="command send">
</form>
```

■制御ファイル出力 (3gmakefile.php)

制御メニュー操作画面よりコマンドファイルの作成

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<?php
$x=$_POST["cmd"] . "¥n";
$fn1="temp" . ".xxx";
$fp=fopen($fn1,'w');
fputs($fp, $x);
fclose($fp);
$fn2 = '3gsa' . '.xxx';

copy($fn1, $fn2);
echo 'command = [' . $x . ']';

?>
<br>
<input type="submit" name="buttonName" value="back" onClick="history.back()" />
```

3. PHPによるメール送信

▶ sendmail.php

```
<HTML>
<HEAD><TITLE> 3G send e-mail </TITLE><HEAD>
<BODY>

<H3> 3G shield temp get </H3>
<?php
    if(mail($_GET["email"], // to
        'Hi 3G shield', // タイトル
        '3Gshield ALARM' . "¥r¥n" . 'DATE = ' . date("Y-m-d") . 'TIME = ' . date("H:i:s") . "¥r¥n" . 'TEMP = ' . $_GET["temp"], // 本文
        'From: info@tabrain.jp' . "¥n" .
        'X-Mailer: PHP/' . phpVersion()))
        echo '<B>SUCCESS TO SEND</B><BR>';
    else
        echo '<B>faile to mb_send_mail</B><BR>';
?>

</BODY>
```

メールアドレス
の引数設定

メール本文



画面表示

4. リモート制御プログラム remote3g.ino①

```
// remote 3G program

#include "a3gim2.h"

const char *server = "所有サーバーURL"; // 登録されているFTPサーバーURL
const char *pathcommand = "/temp.xxx";
char *pathemail = "/sendmail.php?email=SEND_EMAIL_ADDRESS&temp=%d.%d";

int port = 80;
#define TMP_PIN A1
#define SPK_PIN 9
#define LED_PIN 3

char path2[100];

int getTemp(void){
    int tmp = analogRead(TMP_PIN) * 4.88;
    return (mV - 600);
}

void spkAlarm() {
    for(int i=0; i<3; i++)
    { tone(SPK_PIN,250,500); delay(1000);}
}
```

メール送信
アドレス

温度センサ
取得関数

アラーム
関数

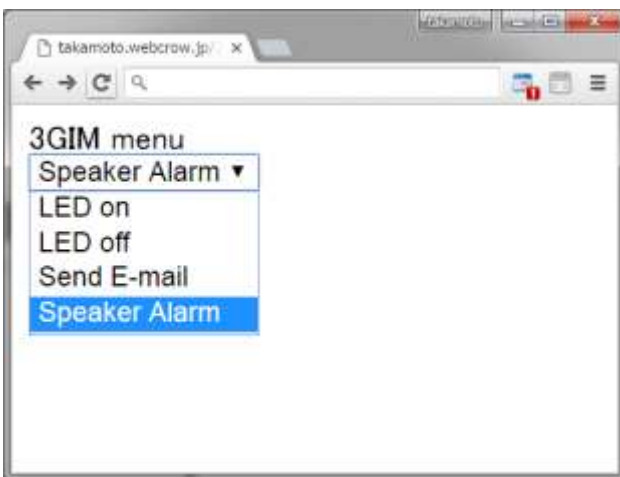
4. リモート制御プログラム remote3g.ino②

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Ready");
  pinMode(LED_PIN,OUTPUT);
  while(!(a3gs.start() == 0 && a3gs.begin()==0)) {
    Serial.println("3G connect error");
  }
  Serial.println("Start..");
}
```

loop関数にて、コマンドを読み込み、
それぞれのコマンド毎に処理を実行。

```
void loop()
{
  static boolean swa=true, swm=true;
  if(swa==false || swm==false) a3gs.setLED(false);
  char res[15];
  const int len = 15;
  Serial.print("httpGET:");
  if (a3gs.httpGET(server, port, pathcommand, res, len) == 0) {
    Serial.println(res);
    if(strncmp(res,"LED on",6)==0 )    {
      swa=true; swm= true;
      digitalWrite(LED_PIN,HIGH);
    }
    else if(strncmp(res,"LED off",7)==0) {
      swa=true; swm= true;
      digitalWrite(LED_PIN,LOW);
    }
    else if(strncmp(res,"Send E-mail",11)==0) {
      if ( swm ) {
        int temp = getTemp();
        char path2[100];
        sprintf(path2, pathemail, temp/10, temp%10);
        if (a3gs.httpGET(server, port, path2, res, len) == 0)
          Serial.println( res );
        delay(5000);
      }
      swm = false;
    }
    else if(strncmp(res,"Speaker Alarm",13)==0) {
      if (swa) { spkAlarm(); } ;
      swa = false;
    }
    delay(100);
  }
  delay(3000);
}
```

4. リモート制御プログラム remote3g.ino③



LED on と LED off は、即座に対応
Send_E-mail と Speaker Alarm は、
初回一回のみ実行（連続しない）

もくじ

- 第1章 3G関連応用事例紹介
- 第2章 3GIMのAssisted GPSについて
- 第3章 3GIMの可能性



3GIMシールド + 3GIM + Genuino I/O

第V編 IoT展開事例

第1章

3G関連応用事例紹介

1. 監視・見守り系システム（ユーザ事例）

□ 今後、独居高齢者の増加にともなう見守りシステムのニーズ増大

- ① 遠方にいる親族などにも状況を逐次知らせるシステムが必要
- ② クラウドサービスによる「いつでも・どこでも」情報把握が可能
- ③ 設置および運用が簡単であること
- ④ 運用コスト（SIMカードや機器レンタルなど）が安いこと

□ アライアンス企業の開発事例紹介

- ① アライアンスメンバー（株式会社ハローシステム様）による開発事例
- ② 親機（3 GIM利用）と複数の子機を使ったシステム
 - ・ 各部屋などでの動き・温度・明るさなどが分かる
- ③ 独居高齢者の状況（動きや明るさなど）をスマホで確認可能
- ④ 独居高齢者からのアラーム発信（メール送信）も可能
- ⑤ 設置が簡単（電源入れるのみ）→ あとはクラウド確認のみ

□ 発展・展開について

- ① 温度センサと湿度センサによる「熱中症」などのアラームを発信（警告）
- ② 見守り側との双方向での連絡も充実化（音と文字情報なども追加可能）
- ③ ペット（猫や犬など）の見守りシステムにも可能に



* 株式会社ハローシステム様の開発事例

2. スマートグリッド関連(IEEE1888)

□消費電力の見える化およびエネルギー削減へのニーズ増大

- ① スマートグリッドによる消費電力の意識が高まる
 - ・具体的な見える化によって、対応策や節電を取ることが可能に
- ② 既存のメーカシステムとの差別化必要
 - ・データの蓄積だけでなく、データ分析・解析が自由に行えること
 - ・必要に応じて、ユーザが加工できること（現状、販売システムは難しい）
 - ・安価なSIMカードおよびクラウドシステムで利用できること
- ③ 3 GIM活用のメリット
 - ・LANだとセキュリティ問題と敷設・維持に課題が多い。3 GIM版は、課題解決し、簡単・迅速に設置可能。

□東京大学とIIJ、およびオープン ワイヤレス アライアンスで昨年10月にプレスリリース

- ① 「世界初：IEEE1888対応の組込み 3 G通信モジュールを開発/M2Mクラウドサービスとの接続に成功」プレスリリースに発表
- ② 今後スマートグリッド（BEMSやHEMS）で標準化として利用
- ③ IEEE1888採用で標準的なクラウドのデータ相互運用が可能に

□今後の発展系

- ① データ標準化により、相互運用が可能となり、応用展開が容易に
- ② M2Mの課題解決のひとつに
 - ・クラウド関連での多くの分析・解析ツールとの連携が容易に



電力見える化システム（東大・フタバ企画）

3. スマートアグリ関連の開発（オーダ品）

□農業分野でのIT活用が活発化

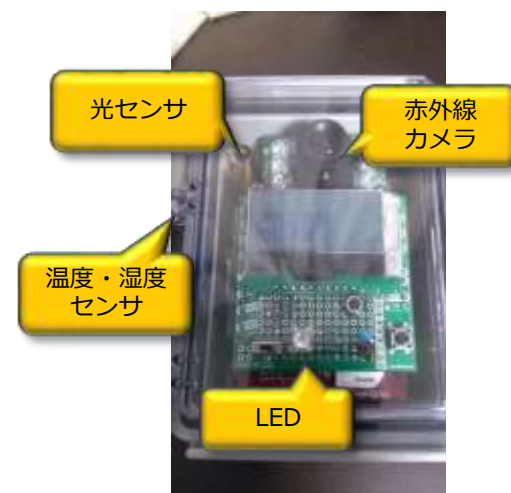
- ① スマートアグリにより、自動制御による食糧生産が現実
・全自動による管理下での農業が実現 ⇒ しかしとても高価で、一般農家では利用できない
・はたして、日本の農業に全自動化が必要か？（現状は、採算が合わない）
- ② 日本の農家に高価なスマートアグリの製品（IT化）は無駄
・すでにスマートアグリに挑戦した企業が倒産する事態も
・ビジネスモデルがまだ、農業IT化では、苦戦中
- ③ 現実にやれるIT化により農家が助かることとは何か
・離れた農地やビニールハウスなどで起きている現象を知りたい
・温度・湿度・土壌状態などから、農作物を荒らす泥棒や野生動物など

□農家として本当に必要なシステムとは何か？

- ① 農家の高齢化により人手不足などが深刻化
- ② 農産物の被害も深刻化（泥棒や野生動物の被害）
- ③ 人手を補うもので、安価で手軽で、簡単に使えるシステムがまずはIT化

□遠隔による見守り・監視システムがまずは必要か？

- ① 温度・湿度・照度・二酸化炭素・土壌などの状況を遠隔地に知らせる（常時観測）
・場合によっては、メールにてアラーム送信
- ② 異常事態でのカメラ撮影や環境変化を遠隔地に知らせる（臨時観測・監視）
・盗難・野生動物被害などの防止、画像伝送による物象の転送



プラント見守りシステム
（拓殖大・構造計画）

4. ICT百葉箱（IEEE1888利用）の開発

□インドにおけるDISANETプロジェクト (JICA/JST)

- ・南インドの都市ハイデラバードに気象センサ20台を高密度に設置し、データ収集を行い、これからの気象防災に役立てる
- ・M2Mゲートウェイによるデータ収集
- ・設置された気象センサからの情報はM2Mゲートウェイを通してIEEE1888形式に変換され、IEEE1888通信プロトコルにより、インド気象局内のサーバに転送。

□気象観測項目

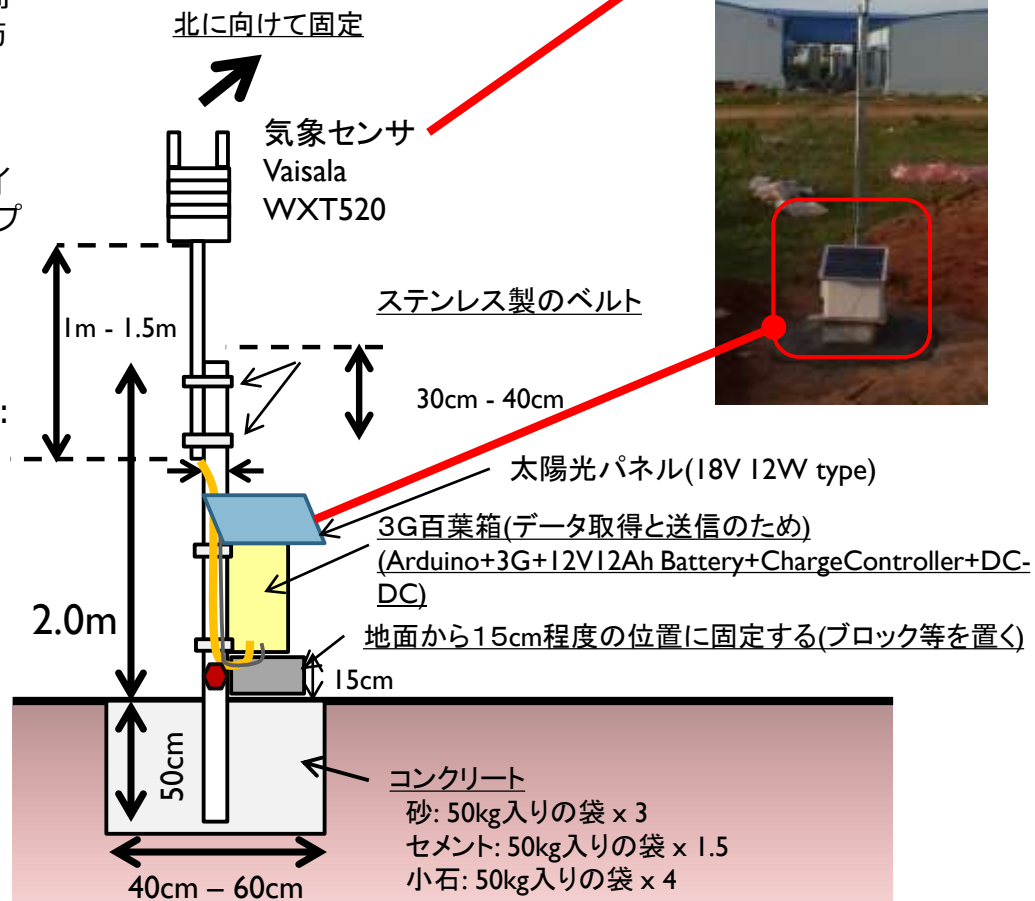
Temperature：気温/Humidity：湿度/Pressure：気圧/RainFall：雨量/DayRainFall：積算雨量/WindDir：風向/WindSpeed：風速

□デジタル百葉箱の特徴

- ・太陽光発電と蓄電による自律システム
- ・IEEE1888対応のM2Mゲートウェイの使用
- ・3G通信によるデータ収集

□ M2Mゲートウェイの役割

- ・RS232Cシリアル通信によるセンサからのデータ収集
- ・収集されたデータのIEEE1888フォーマットへの変換
- ・3Gを使ったIEEE1888通信によるデータ転送



(東大・フタバ企画)

5. 子ども見守りシステム（オーダ品）

小学生の登下校の見守りシステム

学校の校門に親機を設定

子どもがタグ（BLE）を付けて校門前を通過すると親機が感知し、3Gでサーバに送信。サーバ側では、保護者へ通過のメールを送信。

■ 開発プロセス

試作1 → 試作2 → 試作3 で開発進める

■ 課題

- ・ 登下校時の子供集団のトラフィック
- ・ 堅牢なシステム構築

試作を2回繰り返し、3回目のボードで実運用に入る。ただ、試作1、試作2で開発したボードも実運用で利用中

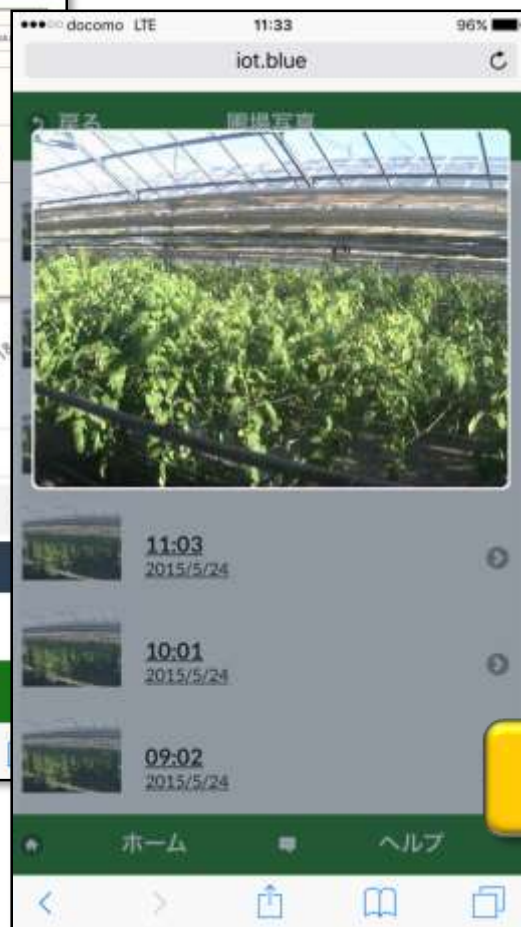


6. 農業用モニタリング（オーダ品）



遠隔での農業用ビニールハウス向け環境モニタリング開発

課題は、センサの設置場所、電源の確保、計測間隔など



光センサ

赤外線
カメラ

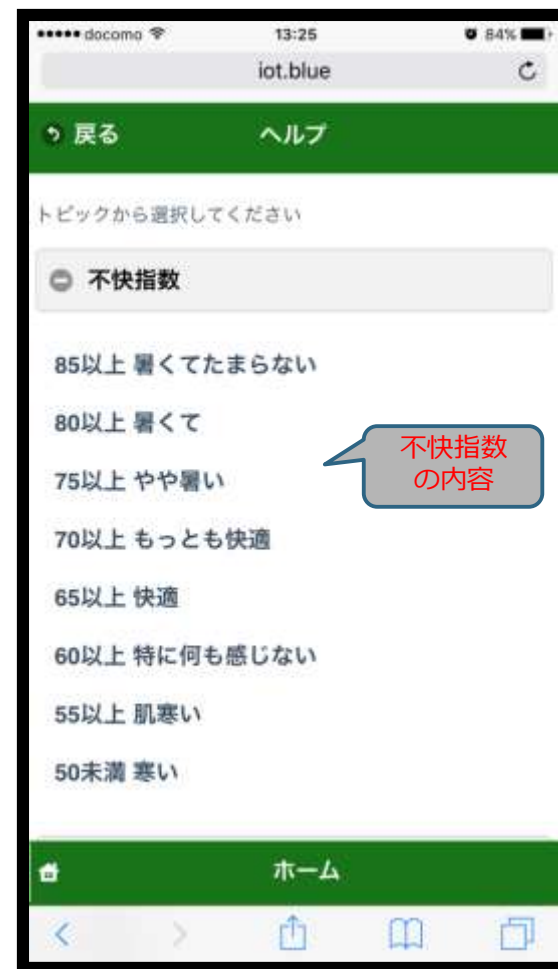
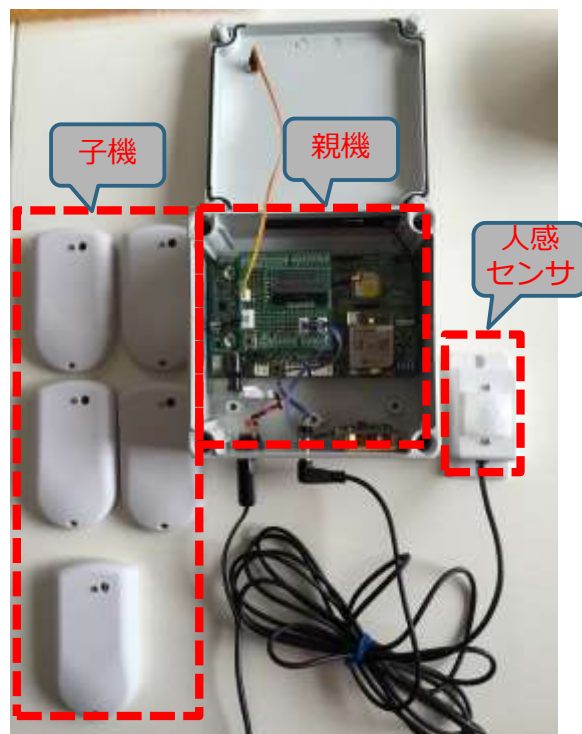
LED

温度・湿度
センサ

7. 会議室環境モニタリング（オーダ品）

某メガバンクの会議室の利用状況把握
及び環境モニタリング

僅か1週間で、親機・子機5セット開発
クラウドも同じ1週間後サービス開始

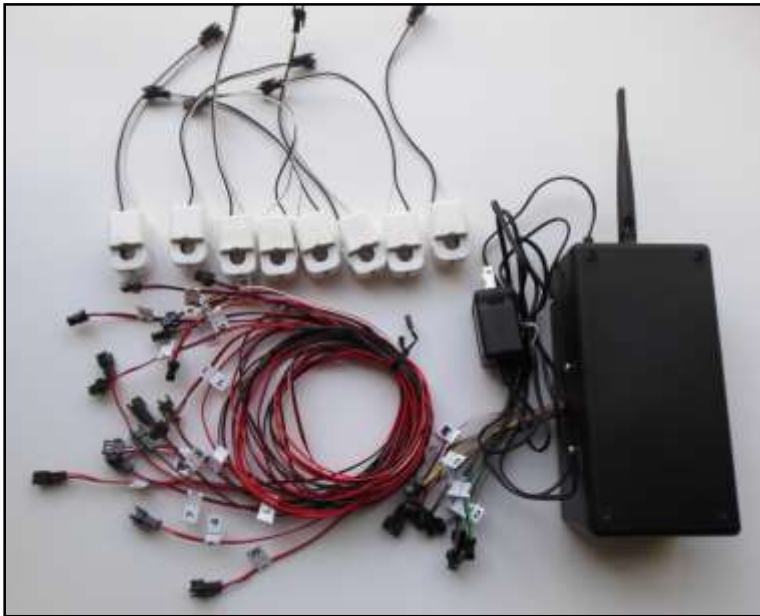


不快指数値の内容

8. 太陽光発電量モニタリング（ユーザ事例）

太陽光発電量モニタリング

2015年6月5日 3Gシールド購入後、僅か1週間ほどでクラウドにデータがアップされインターネット上で配信



2018年の今年は50セット近く利用

現在は、

Arduino Mega + 3 GIMシールド + 3 GIM 利用
ほとんど安定した稼働で、開発案件は全国へ展開中

太陽光発電遠隔監視システム おひさまモニター

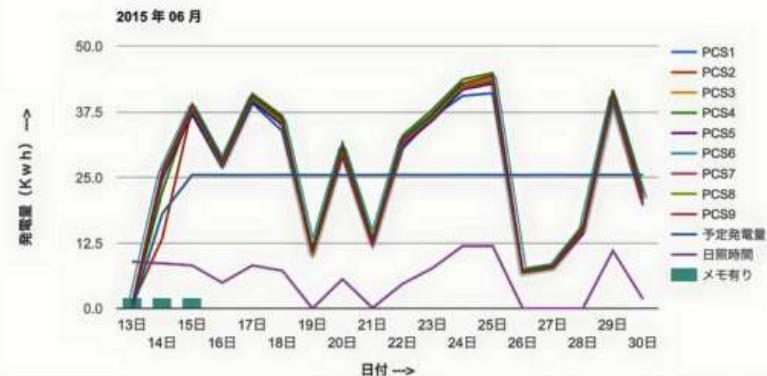
▲トップページ ▲システムの特長 ▲稼働サンプル ▲設置と費用 ▲お問い合わせ

新発売

ハイスペックな太陽光発電遠隔監視システム

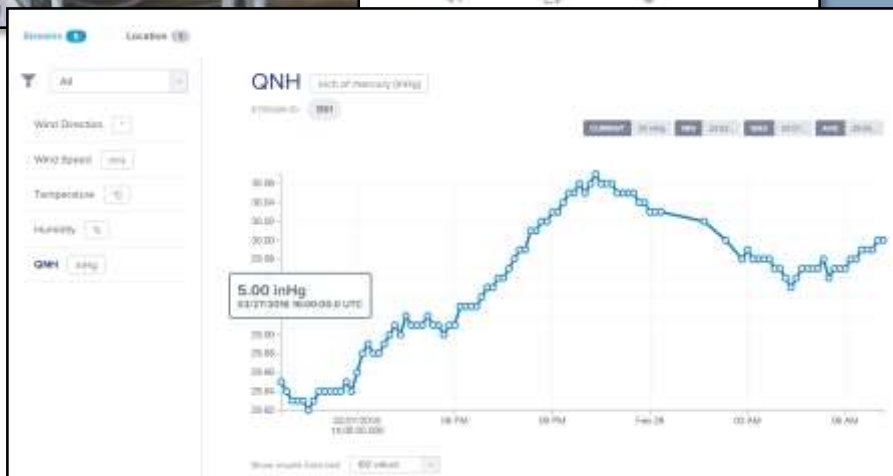
太陽光発電遠隔監視システムおひさまモニター

2015年06月 発電量 (Kwh) 合計発電量 4192.13kwh 予定発電量 3829.32kwh



9. 気象観測モニタリング1（ユーザ事例）

ほとんど手作りでの気象観測データをフリーのクラウドやツイッターにアップ



10. 気象観測モニタリング2（ユーザ事例）

某気象関連企業からの依頼での開発製品
今後、設置場所を増やすとのこと

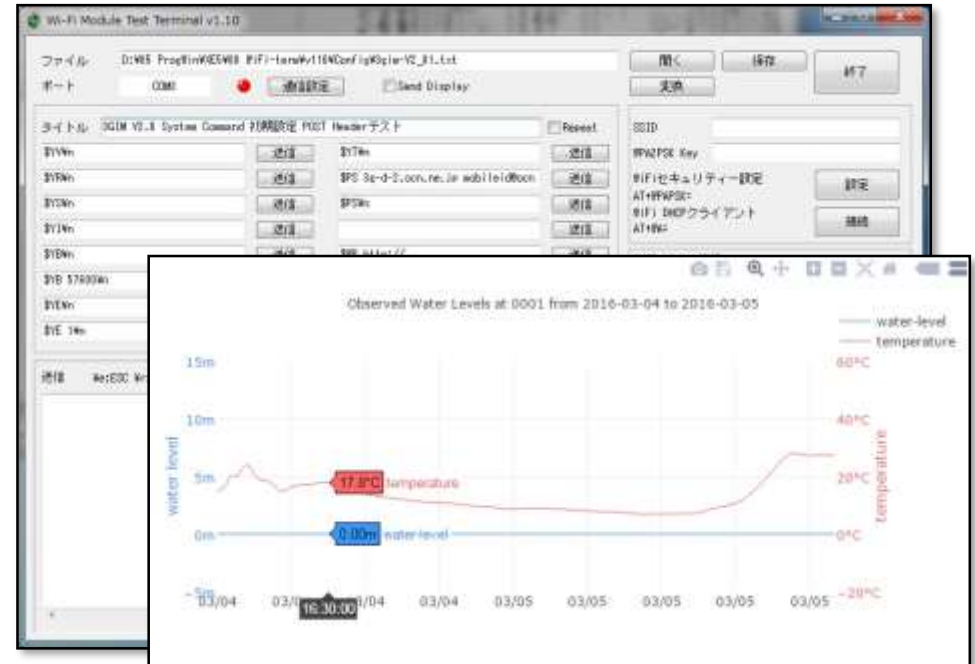
アマチュア無線でテレメータ開発してきたが、
3 GIMを使いはじめたら簡単に開発できることを確認し、すべて3 GIM開発に切替えた。

今年も継続して大量に製造中とのこと



1 1. 水位観測モニタリング（ユーザ事例）

河川水位計モニタリング機器



2015年夏WiFiで試作していたが、プログラム量の問題、ルータ設置の問題から、即時に3 GIMに切り替え
2016年4月から本格的に3 GIMを利用して8セットを現場配置しスタート



1 2. 遠隔保守監視システム（オーダ品）

汎用的な遠隔保守監視システム

- 1) 屋外機器（太陽光発電駆動）の監視
 - ・機器の信号を取得
 - ・バッテリーの容量監視
 - ・低バッテリー時のセンサ値保存（SDメモリ）
- 2) 低バッテリー時の非常時対応
 - ・非常時バッテリー時の駆動
- 3) 信号線と電源接続だけで即起動
 - ・簡単な現場設置対応

利用目的

- ・遠隔地の屋外にある設備機器の監視モニタリング
- ・これまで定期的な人的保守サポートが不要に
（大幅な人件費のコストダウン化）

主な顧客：

- ・某大手でのNTT関連野外機器の保守メンテ用として
- ・JR関連の線路切り替え信号機の保守メンテ用として
- ・電力会社所有の風力発電機振動機器の無線化として
- ・防塵建屋に取り付ける機器の保守メンテ用として



1 3. 工場用IoT汎用ボード（オーダ品）

汎用的なIoTゲートウェイボード

- 1) 屋内および屋外機器の各種監視
 - ・機器の信号を取得
 - ・LAN・3G対応
 - ・RS485、I2C、SPI、UART対応
 - ・SDメモリ
 - ・WDT機能（ハードウェアリセット）
 - ・Arduino互換機
 - ・3GIMライブラリ利用
- 2) 豊富なI/F機能
 - ・各種機器対応可能
- 3) 堅牢なシステム
 - ・システムの安定的な動きを展開
 - ・ハードウェアリセット対応
 - ・非通信時のSDメモリー保管

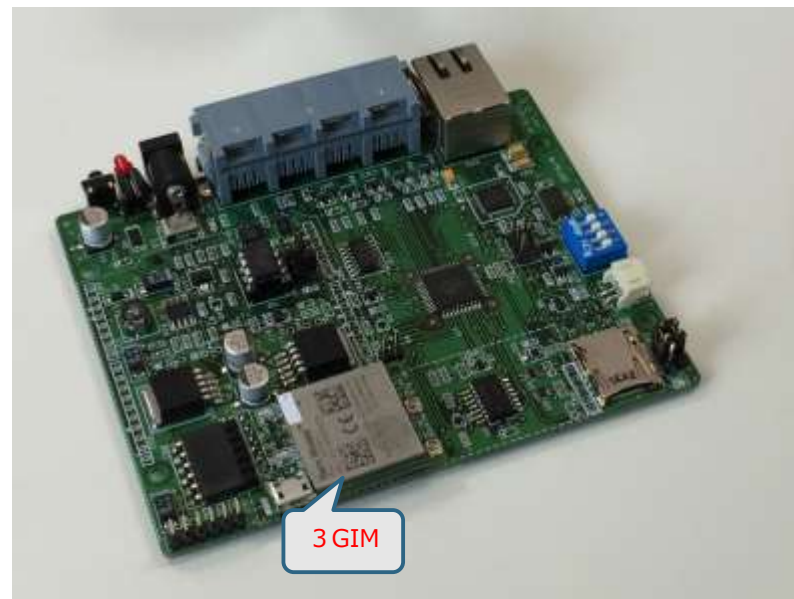
利用目的

- ・屋内・屋外での各種機器の保守サポート対応向け機器
- ・安定的で堅牢なシステム利用現場向け機器

主な顧客：

- ・工場（製造現場）向け対応
- ・移動体（車載器）向け対応
- ・遠隔監視・モニタリング向け対応

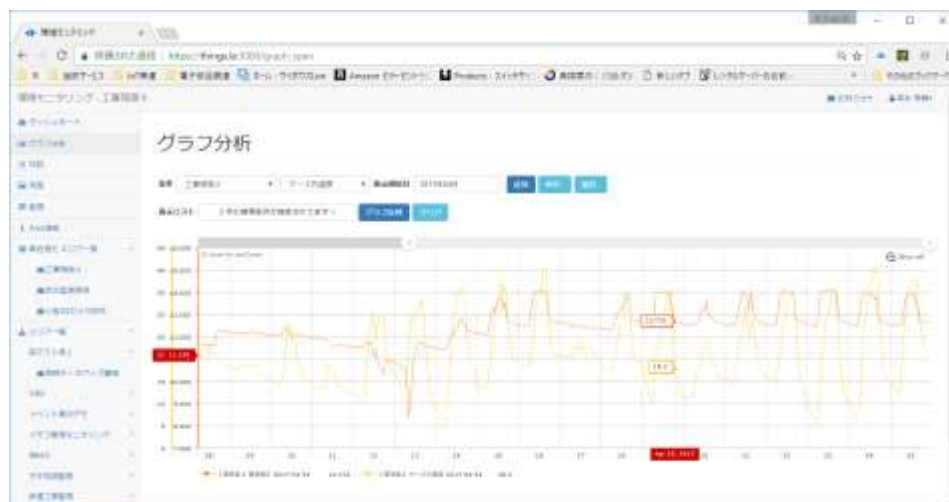
※専用ケースにて対応



1 4. 横取り失念装置遠隔監視システム（オーダ品）

超省エネタイプの屋外バッテリーの遠隔監視システム

- 1) 鉄道線路切り替え工事中のリミッタスイッチ監視
- 2) 装置
 - ・ 3 GIM + IoT-COMB(Arduino互換機) 対応
 - ・ IoT-COMB (ARM利用)
 - ・ 温度センサ
 - ・ 電源電圧測定
 - ・ リミッタスイッチ情報取得
- 3) 既存装置に敷設
 - ・ ケーブル接続のみで電源ONの状態に
- 4) 検査モードと運用モード
 - ・ 現場設置前の検査モード (1分間隔) と
 - ・ 現場設置での運用モード (1時間間隔を区別)



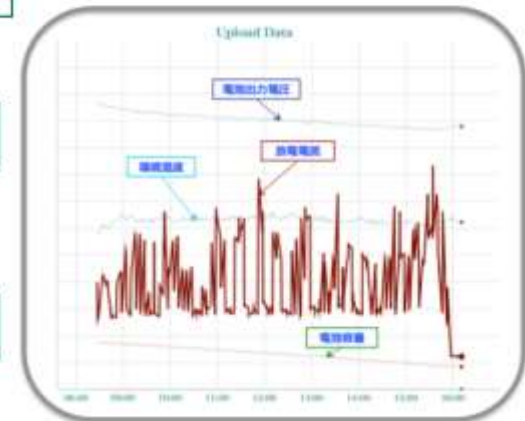
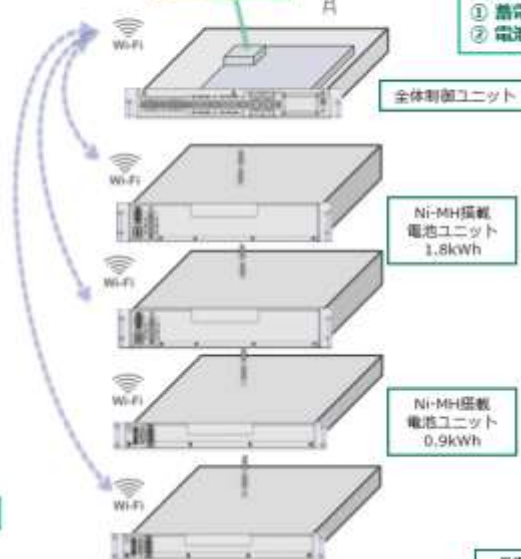
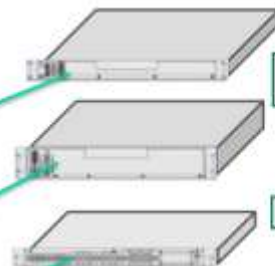
1 5. 蓄電監視モニタリング装置（ユーザ事例）

検診車のバッテリー（NI-MH）遠隔モニタリング装置

- ・遠隔でのモニタリングをすることで
バッテリー追尾車を
- ・機器の信号を取得
- ・LAN・3G対応

特徴

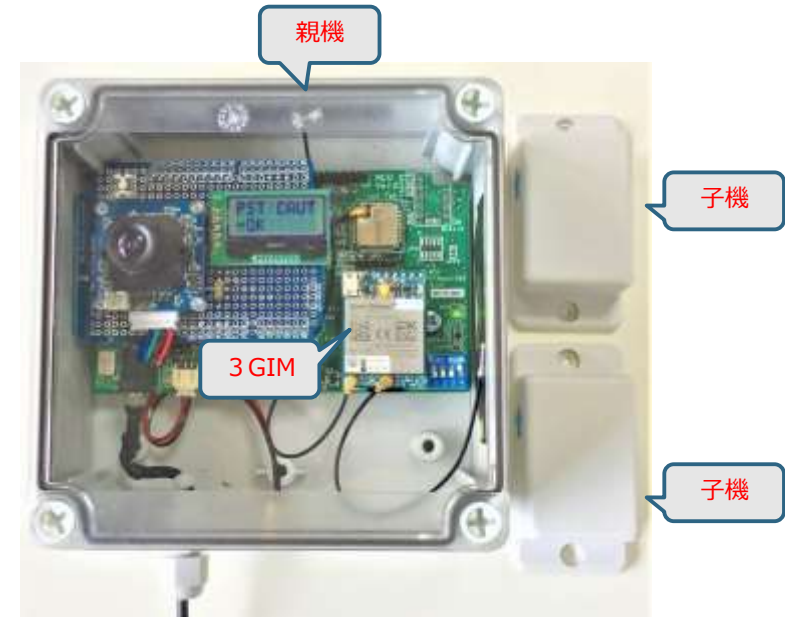
- ・電池容量及び設置場所に柔軟な対応が可能
- ・蓄電装置稼働状況をリアルタイムで管理可能



16. 防災カメラ監視システム試作（オーダ品）

防災用監視カメラ試作

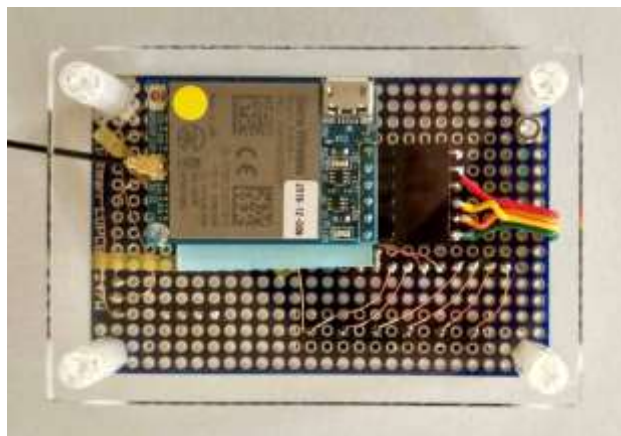
- 1) バッテリーのみで稼働する可搬性のあるシステム
- 2) 仕様
 - ・ 3 GIM + TabrainoV1.1対応
 - ・ TWE-Lite（親機と子機との関係）
 - ・ 温度センサ・外部電源電圧測定
 - ・ 子機：加速度センサ・電源電圧
- 3) 子機の加速度センサ利用
 - ・ 微振動の抽出
 - ・ 傾斜角度を抽出
- 4) 重要仕様：長期安定での防災モニタリング
- 5) 課題は、消費電力とバッテリーとの関係



17. ひび割れ監視システム試作（オーダ品）

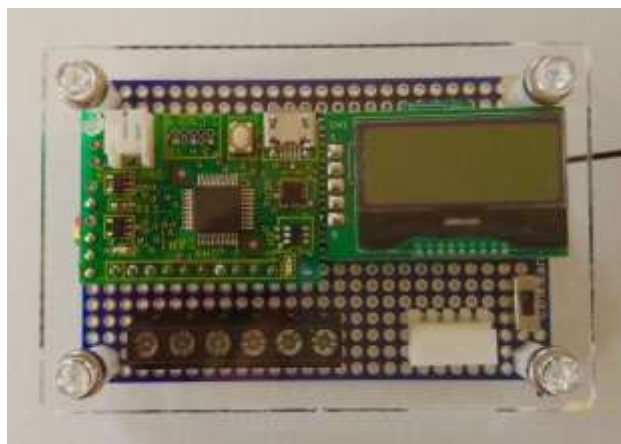
高速道路柱脚・橋梁コンクリート部のひび割れ監視

- 1) バッテリーのみで稼働する可搬性のあるシステム
- 2) 仕様
 - ・ 3 GIM + Tabraino 2 対応
 - ・ 歪ゲージ + 距離センサ + 加速度センサ
 - ・ 温度センサ・外部電源電圧測定
- 3) 加速度センサ
 - ・ 大きな振動時を捉えてメール送信
 - ・ 常時観測が重要（ひび割れ進行を観測）
- 4) 重要仕様：バッテリーのみで長期間での稼働



※長期間での省エネ対応は

- 1) 常時消費電力を抑える（0.5mA程度）
- 2) ソフト制御
 - ① スリープ処理
 - ② ウェイクアップ処理
 - ③ 割込み処理
 - ④ 消費電力が少ないセンサ取得術
 - ⑤ 効率的な 3 G でのデータアップ処理



1 8．車両専用移動体IoTボード（オーダ品）

移動体用IoTボード

- 1) バス・トラック等のGPS監視モニタリング
 - ・ 自動車内さまざまな装置の信号を取得
 - ・ 3G/LTE（4GIM）対応
 - ・ I2C、SPI、UART対応
 - ・ SDメモリ
 - ・ WDT機能（ハードウェアリセット）
 - ・ Arduino互換機
 - ・ GPSモジュール搭載
- 2) 豊富なI/F機能
 - ・ 各種機器対応可能
- 3) 堅牢なシステム
 - ・ システムの安定的な動きを展開
 - ・ ハードウェアリセット対応
 - ・ 非通信時のSDメモリー保管

利用目的

- ・ バス・車両等の各種機器の対応向け機器
- ・ 安定的で堅牢なシステム利用現場向け機器

主な顧客：

- ・ バス・車両等に搭載対応
- ・ その他移動体向け対応
- ・ 遠隔監視・モニタリング向け対応

※専用ケースにて対応



量産化前に試作対応は3回実施。

19. ひび割れ振動監視モニタリング（オーダ品）

首都高速道路柱脚ひび割れ監視モニタリング装置

1) IoTボード

- ・ 3 GIM/ 4 GIM敷設コネクタ付き
- ・ MCU : SAMD21G18A
- ・ 加速度センサ/温度センサ/距離センサ付き
- ・ ひび割れセンサ付き
- ・ 3 GIMライブラリ利用可能
- ・ 開発期間（ハードウェア1.5ヶ月間）
- ・ ソフトウェアは事前の試作品のものを活用
- ・ 長期間稼働するための超省エネタイプ

2) 利用目的

利用目的：バッテリーで数年間稼働のこと

2018年11月21-22日 テクノハイウェイ展で展示



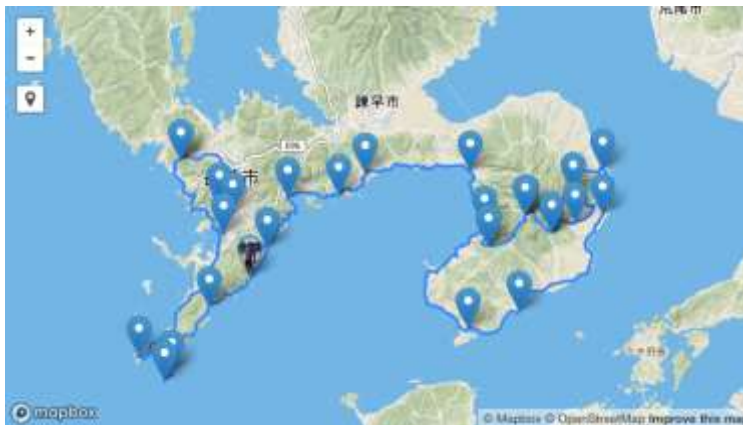
20. ランナー追跡モニタリング（ユーザ事例）

長崎での273キロのマラソンランナーに着けて
追跡モニタリングに利用

- ・デバイス重さは、わずか50 g 前後から120 g 程度
 - ・バッテリーは3日間継続して動くことが求められた
- 利用デバイスは

3 GIM+IoTABボード+リチウムイオン電池

（作成者・ランナー：長崎市の菅崎氏）



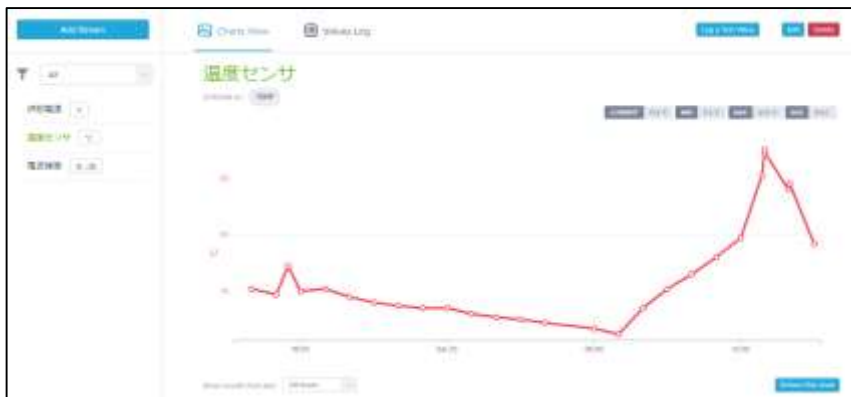
2 1. イノシシ罠の感知システム（自社試作）

最近では、害獣駆除のためにさまざまな対応策がとられています。イノシシも農作物を荒らし、被害も甚大なものとなってきていて、その駆除・対策も多く行われています。

本件では、イノシシの罠を畑や山林に設置し、罠にかかるとそれをメールで知らせるIoTデバイスを作成しました。

バッテリーのみで長期間稼働し続けるもので、そのバッテリー電圧も常時監視できるものとしています。

1時間に1回程度、バッテリー電圧および温度をクラウドにアップし、罠にイノシシが掛かるとメールが関係者に飛んでくるものとしています。ここでは、加速度センサによる振動で、メールを送信します。



上記写真の8000mAhのリチウムイオン電池で約4ヶ月ほど稼働し続けます

2 2. IoTAB & IoTAB Proボード（自社試作）

3 GIM/ 4 GIM対応用IoT利用ボード

1) IoTABボード

- ・ 3 GIM/ 4 GIM敷設コネクタ付き
- ・ MCU : SAMD21G18A
- ・ 加速度センサ/温度センサ
- ・ Arduino互換機
- ・ 3 GIMライブラリ利用可能
- ・ アナログIN 4 (+ 2) ピン
- ・ GPIO 2 (+ 6) ピン
- ・ I2C/UART*3

利用目的：超省エネ・超小型開発向け

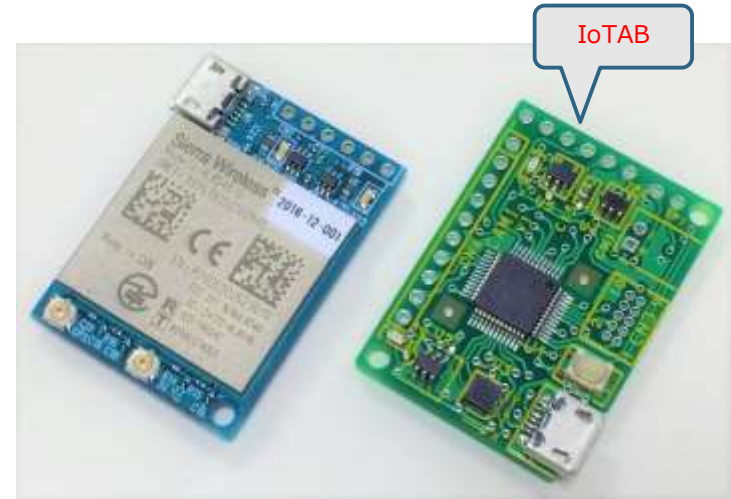
利用事例：マラソン走者追跡・動物（サル）追跡

2) IoTAB Proボード

- ・ 3 GIM/ 4 GIM敷設コネクタ付き
- ・ MCU : SAMD21G18A
- ・ 加速度センサ/温度センサ
- ・ Arduino互換機
- ・ 3 GIMライブラリ利用可能
- ・ アナログIN 4 ピン
- ・ GPIO 10ピン
- ・ I2C/UART*3/SPI
- ・ システムの安定的な動きを展開
- ・ ハードウェアリセット対応
- ・ 非通信時のSDメモリー保管

利用目的：汎用IoT省エネ開発ボード

利用事例：高速道路柱脚ひび割れ監視



2 3. IoTAB利用小型IoTカメラ試作（自社品）

IoTABボード + 3 GIMを使ったIoT小型カメラ試作

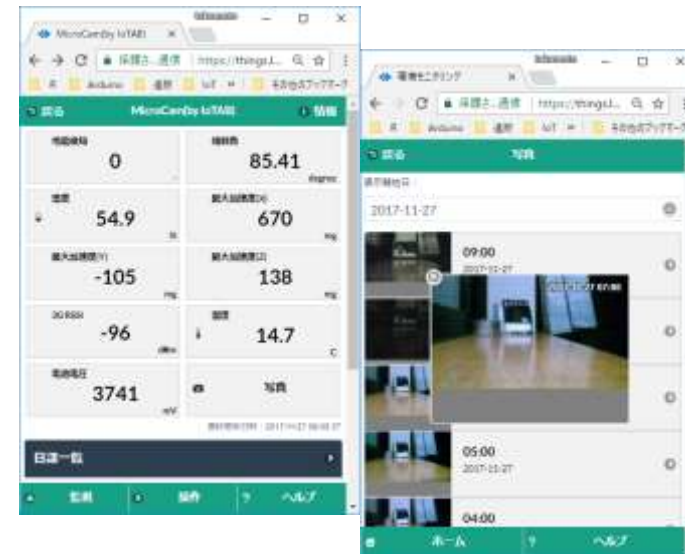
筐体（ケース）サイズは、わずか5cm×5cm×2.5cm

バッテリー充電して設置するだけ、簡単に利用できる。
300mAh程度のバッテリーで、30分のセンサ値取得、
1時間ごとのカメラ画像アップで、2日間ほど連続使用可能。
また加速度センサによる割込みで起動する。

今後の改良点

- 外部電源利用対応（オプション）
- 遠隔制御可能

利用目的：一時的な防犯・防災・見守りなど



24. 河川水位監視デバイス開発（オーダ品）

国土交通省では、最近の多くの水害に対し、上流河川でも水位監視のニーズが高まり、そのデバイス設置が急がれています。
今回開発した水位監視デバイスは、すでに新潟県、神奈川県、高知県、熊本県などでの試験運用として、Arduino+ 3 GIM/ 4 GIM + IoTAB ボードをベースに開発したものです。
ここでは、安価で、耐久性があり、長期（5年）に渡って稼働することが要求されています。ここでは、太陽光発電パネルとバッテリーによって長期安定稼働する仕組みをとっています。

現在（2018年6月）では、カメラ付きのデバイスも開発し、超音波距離センサ（5～10m）を使って水面までの距離を計測し、クラウドにデータをアップし続けています。
本システムでの水位監視間隔は、水位が低い場合には1時間ごとに、危険水位に達すると5分ごとにクラウドにデータをアップします。
（危険水位距離や計測時間間隔は設置後クラウドで調整可能）

【機能】

クラウドにアップするデータは、水位（水面までの距離）、電源電圧（バッテリー容量監視）、温度、電波強度などで、雨天が1ヶ月継続しても継続して稼働する仕組みとなっています。

【特長】

- ・現場危険水位（河川水面までの高さ調整）がクラウド調整可能
- ・監視調整（危険水位監視間隔調整）がクラウド対応可能
＜デバイスのID管理によって設置場所と紐づけ＞
- ・Arduino互換機+ 3 GIM/ 4 GIMで製造コストは安価
- ・超省エネのArduino互換機IoTABボードを搭載
- ・雨天時の長期対応可能（1ヶ月ほど稼働）※

※電波強度なども影響し、1ヶ月以下になることもありえる。



太陽光発電パネル
(5W)



水位センサは、MaxBotix社の
MB7093 や MB7386利用
(5m～10mまで監視可能)



河川水位監視デバイス
防水対応 IP65



カメラ対応版



現場設置事例

25. 量産化農業用IoT監視デバイス開発（オーダ品）

本デバイスは、RaspberryPi Zeroをベースに開発した農業用監視デバイス製品です。量産化対応として、ビニールハウス内の温度、湿度、照度、二酸化炭素濃度を観測し、そのほか子機のバッテリーや電波強度をモニタリングします。これによって安価で利用できるデバイスとなり、広くご利用いただけるまでになりました。ここでも安定して稼働することが重要ですが、親機の電源を入れるだけで、クラウドにデータがアップされることから、使い勝手も超簡単となっています。

温湿度や照度については、子機として複数台増やすことができ、約50mほどの距離でも親子間での無線が利用できるようになっています。

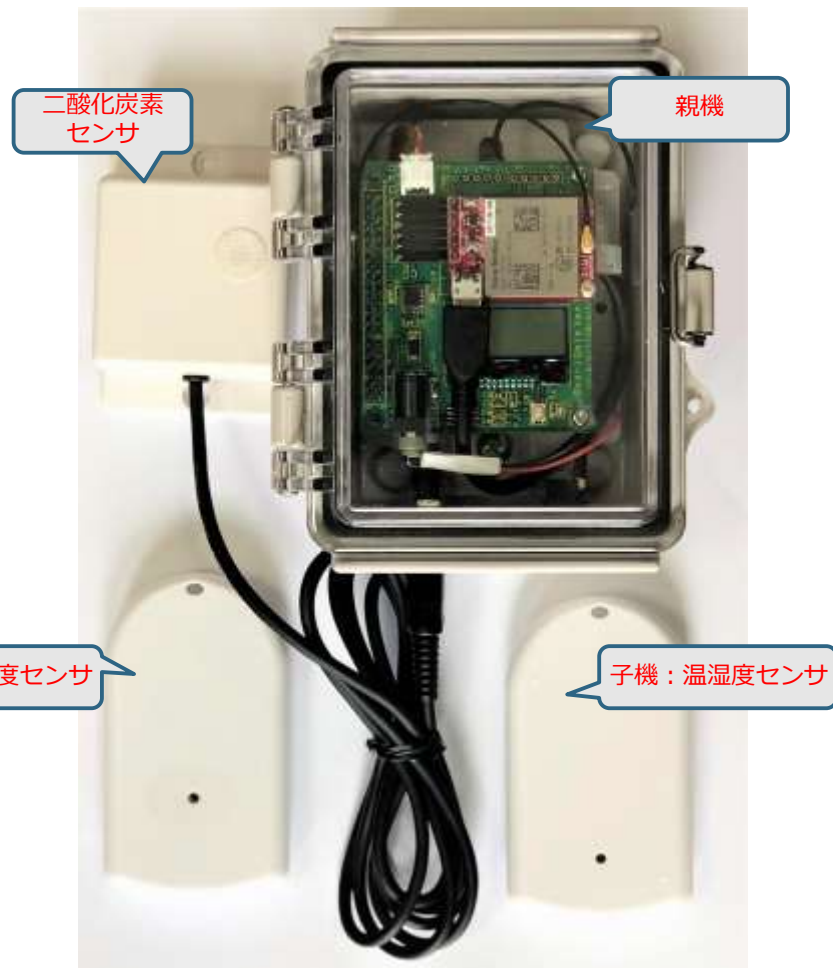
こちらは某国立大学の農学部による研究開発で開発したのですが、すでに（前頁での）試作も多いことから、量産用として開発したものとなっています。

【機能】

- ・農業用専用として、二酸化炭素濃度および温湿度・照度のデータをクラウドにアップして監視可能となる
- ・監視は、5分から10分間隔で設定（調整）が可能
- ・閾値をクラウドに設定し、メール送信が可能
- ・子機側は、数10台まで増やすことが可能
- ・親機と子機間の通信距離は50mほどまで可能
- ・電源ON/OFFだけで利用可能

【特長】

- ・安価なデバイス提供
- ・超簡単な利用（電源On/Offで利用開始・終了が切り替え）
- ・子機側のバッテリー切れの監視が可能



農業用IoT監視デバイス

26. 汎用IoT遠隔監視デバイス開発（自社製品）

本デバイスは、保全用（保守メンテナンス）用として試作したIoTABマザーボード上に、3 GIM（または4 GIM）、それにIoTABボードを搭載し、リチウムイオンバッテリーで長期間に渡って、遠隔の監視を行うものとなります。
すでに、橋梁における圧電素子での遠隔監視などで試験テストとして利用してきました。今後においては、遠隔監視のデバイスとして広く利用できることで、試作や量産化で対応していく汎用デバイスとして考えています。

本デバイスでは、搭載したIoTABボード上の加速度センサや温度センサのほか、圧電素子やひずみゲージ、そのほかカメラ、超音波距離センサなども取り付けることができるようになっています。

【機能】

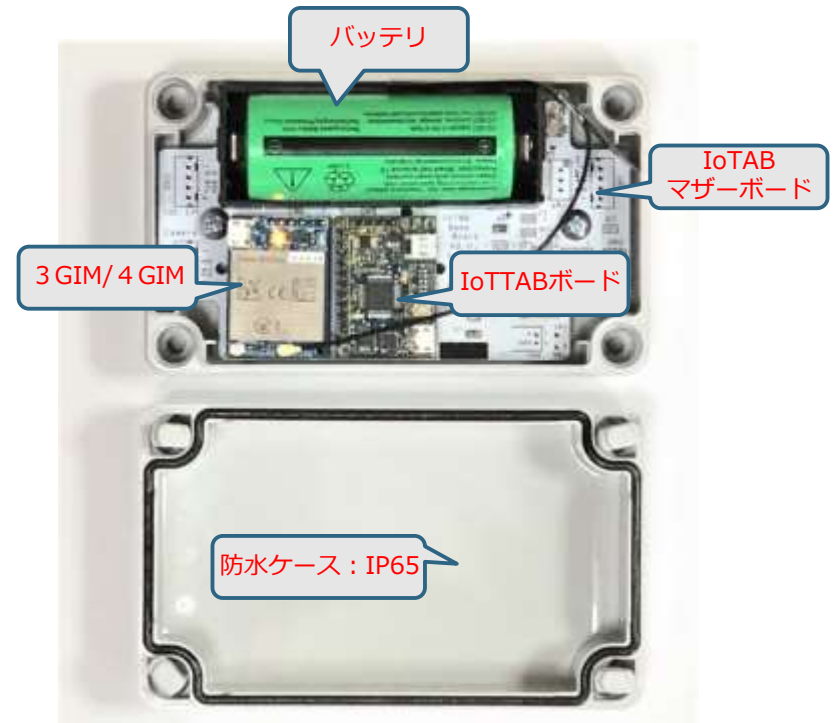
- ・遠隔監視による保全関連として、さまざまなセンサ値を取得し、クラウドにアップしたり、メールで知らせたりすることができます。
- ・取付け可能なセンサは、アナログ、デジタル、I2C、UART、SPIなどの通信によるもので、さまざまなものが取付け可能となります。

【応用分野】

- ・遠隔での監視デバイスとして
- ・保全対応（傾斜・振動・衝撃など）が必要な監視
- ・太陽光森林などで利用する遠隔監視向け
- ・利用例：イノシシ罾の監視、橋梁ひずみ監視、橋梁振動監視、地盤傾斜監視等

【特長】

- ・安価なデバイス提供（ハードウェア：5万円～7万円、ソフトウェアは別途）
- ・超簡単な利用（電源On/Offで利用開始・終了が切り替え）
- ・子機側のバッテリー切れの監視・振動（震度5など）での割込み機能
- ・常時監視とセンサ割込み監視
- ・バッテリー単独利用（別途太陽光パネル蓄電追加も可能）
- ・バッテリー利用時間は、数カ月間となりますが、通信頻度や電波状態などで前後
- ・オプションとして、カメラや各種センサも追加利用可能



汎用遠隔監視IoTデバイス

27. 自転車用盗難防止ロガー試作（ユーザ事例）

本デバイス試作は、自転車の盗難防止のためのもので、3 GIMのGPS機能と3 G通信機能を使ったシステムとなります。

このデバイスは、自転車以外のバイクや自動車、さらには幼児・子供、老人などにつけても利用価値があるものと思われます。

主な機能：

- ・ **走行経路のロギング**
→履歴はスマホアプリから閲覧
- ・ **駐輪中の盗難防止**
→内蔵センサにより振動を検知、大音量ブザーによる警告と同時にスマホに即座にお知らせ
- ・ **安全用リアライト**
→デバイスの動作確認と兼用。走行中の安全確保

特徴：

- ・ 充電池内蔵で半日以上使用可、スマホの充電器で充電可能
- ・ 小型軽量（4cm×8cm×2cm）自転車のサドル下に収納
- ・ ロガー・盗難防止・リアライトという自転車に必要な三要素をひとつにまとめたデバイス
- ・ スマホアプリからログ間隔など各種項目を設定可能

今後の改良点：

- ・ リアライトの光量増加（視認性向上）
- ・ スマホアプリの開発

（試作・情報提供は石田貴行様）

<今後量産・販売も計画中とのことです>



デバイス内部

筐体（ケース）は、4cm×8cm×2cmとコンパクトです。

ツーリング中の走行履歴

現時点での位置や走行履歴がとれます。



装置の収納位置

自転車のサドル下に固定しています。

28. 害獣罾監視IoTデバイス開発（オーダ品）

本デバイスは、害獣罾（檻罾・括り罾など）に取り付けるもので、メールによって位置情報およびバッテリー情報を送信する仕組みとなっている。

主な特徴：

- ・ 大きさ4.2cm×3.2cm×8.5cmサイズの小型化
- ・ リチウムイオン電池 26650搭載
- ・ 3 GIM+IoTABボード+IoTABマザーボード対応
- ・ 塩ビ管ケース対応（防水対応）

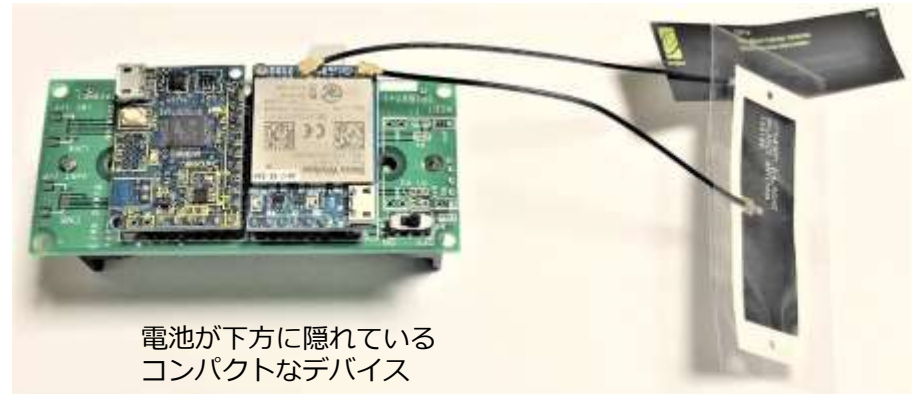
特徴：

- ・ バッテリーで3ヶ月以上稼働
- ・ 1日1回バッテリー情報をメール送信
- ・ 罾（IoTABボード上の加速度センサ）が反応した時点で、GPS（位置情報）をメール送信
- ・ バッテリー減少が分かるメール送信

ユーザ発注の背景：

・ 既存デバイスが存在したが、価格が高く、機能は罾にかかったときのみメール送信するだけで、位置情報はわからない状態。

今回の提案デバイスが、既存製品の1/2以下となることから、いきなり量産化対応



電池が下方に隠れている
コンパクトなデバイス



筒状の両側に
ネジ式のフタ
を取付けた
オーダによる
塩ビ管ケース

29. 積雪量観測IoTデバイス試作品（研究用）

本デバイス試作は、積雪量を観測するデバイスの試作品で、レーザー距離センサを使つての実験装置となります。

主な機能：

- ・ **積雪量の遠隔監視**
→ 走道路上の積雪量を定時間隔で観測し、積雪量が変わったときにクラウドにそのデータをアップするIoTデバイス
- ・ **高精度な積雪量測定**
→ レーザー距離センサを搭載したことで、1 mm単位の誤差範囲での計測が可能
- ・ **バッテリー単独の自律型IoTデバイス**
→ リチウムイオン電池26650（3.7V5000mAh）搭載し、コンパクトなケースで、取り付け簡単なフレームも準備

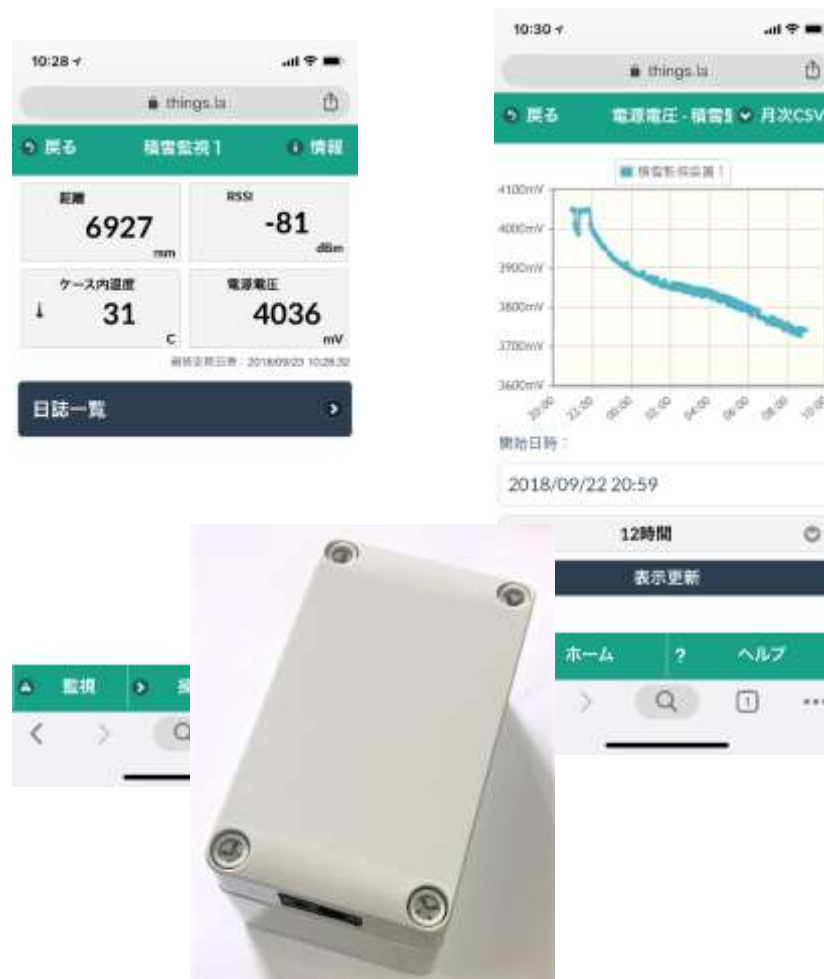
特徴：

- ・ レーザー距離センサによる測定は、数10mまで距離を観測することができ、精度も1 mm単位と高精度であることから、定点観測などでも利用が可能
 - ・ 小型軽量ボックス（8 cm× 13 cm× 8 cm）に搭載したが、さらに小型化が可能
- ※ 別途、超音波距離センサでの試作も行い、同時に積雪量の観測実験を実施

今後の改良点：

- ・ バッテリー交換の容易化（乾電池交換式など）
- ・ 取り付け容易な金具の準備
- ・ 実験評価での改良・改善対応予定

* 2018年10月に実験予定



30. その他 特殊IoTデバイスの試作

① 鳥獣追跡IoTデバイス

猿や鹿に3GIMを取付けGPS取得によって、
鳥獣の追跡を行うデバイス開発を行う。
(顧客事例)

③ GPS機能を使ったデバイス

移動体監視・動態監視のデバイス開発が多発
・徘徊老人向けデバイス開発
・害獣調査デバイス開発
・観光者調査向けデバイス開発
などなど

→ 2018年春から「みちびき」対応可能に

② 野鳥に取り付けた気象観測デバイス



大型野鳥に取り付けた観測装置
超小型化（軽量化）を目指し、バッテリーのみ
で長期間稼働し続けること
特にGPS機能を使ったデバイスが特徴

④ 機器保全対応デバイス

工学な機器製造メーカーが顧客向け保守対応と
して、機器にIoTデバイスを組み込み
保全対応を遠隔で行うケース増大
→ 消耗品の計画的な対応
→ 故障発生などの早期対応
→ 総合的なメンテナンス対応
など多くのメリットが存在

3 1. その他試作・プロトタイプ開発ほか

□人や動物の動きをキャッチしデータとして収集・分析

- ① 親機と子機との関係でシステム開発（課題は子機の長寿命化）
→ 親機にゲートウェイ機能と子機受信器、子機は発信機能のみ
- ② クラウド連携において動きをモニタリング分析
→ 子機固有IDの動きをモニタリング
- ③ 関係者にはメールによって動きを知らせる

□特殊環境下の維持装置モニタリングシステム試作

- ① 助成金によって特殊環境下の維持管理をモニタリングするシステムの試作
- ② 複数のセンサ値をクラウドにアップし、その状況を把握できる環境下に
- ③ 異常な状態をいち早く知らせるシステムとして開発中（端末系はスマホや携帯のメール）

□オープンソースハードウェア関連での試作・プロトタイプ開発支援

- ① 既存システムに組み込む高価な機材を安価なシステムに切り替えるために試作・プロトタイプ開発
・通信モジュールを用いてモニタリング機能に対応
- ② 工場内機器の保守モニタリングや設定制御などを遠隔操作によって自動化
・自社内LANとは別系統でセキュリティ対応に無関係

3 2. 大学・高専での開発事例

- ▶ 信州大学（電磁波解析と磁界発電の研究）
 - ▶ 東京大学（腐食センサーによる橋梁保全研究）
 - ▶ 徳島大学（橋梁の保全研究）
 - ▶ 東京海洋大学（近海小型船避難緊急発信装置）
 - ▶ 沼津工業高専（農業用エリアモニタリング研究）
 - ▶ 千葉大学（農業用ビニールハウスモニタリング）
 - ▶ 東海大学（農業用モニタリング研究）
 - ▶ 東海大学（EVカー蓄電モニタリング）
 - ▶ 拓殖大学（EnOceanと3GIMの連携）
 - ▶ 和歌山県立海南高校（缶サットに搭載）
- その他多くの大学・高専で、3GIMを使った研究活動実施



電磁解析
（信州大学・田代研究室）

3 3. 3 GIM関連の共通技術提供



34. 第1回アイデアコンテスト(2013年度)

<http://3gsa.org/information.html> にて公開

- ・最優秀賞 3GSコミュニティバスお知らせシステム (資料) 飯島幸太氏
- ・準優秀賞 クラウドコレクタ (資料) 山本三七男氏
- ・努力賞 愛車の記録 (資料) 小林康晃氏
- ・その他応募作 相互見守り(コミュニケーション)システム

- ・最優秀賞 Himawari3搭載 3G火山ガスモニタリングシステム 拓殖大学 瀬谷鮎太氏
- ・優秀賞 ペット管理システム (資料) 東京都立小石川中等教育学校 小島和将君
- ・準優秀賞 Arduinoを使ったFOXテーリング (資料) 東京都立総合工科高校
高橋君・土屋君・平林氏
- ・努力賞 山の幸、獲ったどー (資料) 拓殖大学 南川俊氏ほか
- ・特別賞 わいっち目 (資料) 拓殖大学 井上龍氏ほか
- ・その他応募作
自動車防犯装置 (資料) 東京都立総合工科高校 土屋君・高橋君・平林氏
水田あんばい (資料) 拓殖大学 金山祐大氏ほか
アマモ場の生育環境観測システム (資料) 広島商船高専 芝田研究室

3 5. 第2回アイデアコンテスト(2014年度)

<http://3gsa.org/information.html> にて公開

平成26年11月16日開催されました第2回 3 GIM・アイデア・コンテストの結果報告となります。以下のFacebookなどで掲載しています。

- ・最優秀賞 「快適マネージャー」
東京都立小石川中等教育学校 小川広水君（中学一年生） [資料/ビデオ](#)
- ・優秀賞 「ポチっとじょうろ」
東京都立小石川中等教育学校 中野龍太君・中本一輝君・小林俊介君（中学二年生） [資料/ビデオ](#)
- ・特別賞「Rubyを用いたマイコンプログラムの遠隔書き換えシステム」 チーム海南 山本三七男氏
他和歌山県立海南高校（瀧本君、若勇君、和田君、筈谷君、岸田先生）/ルアリダワークス [資料/ビデオ](#)
- ・特別賞 「Dustino(ダスティーン) ～ゴミ箱管理システム」
九州工業大学 備後博生君、城戸翔兵君、張思嘉君 [資料/ビデオ](#)
- ・アイデア賞「冷蔵庫を使ったお年寄り見守りシステム」
東京都立小石川中等教育学校 金子知洋君（高校二年生） [資料/ビデオ](#)
- ・アイデア賞「熱中症予防散システム」
九州工業大学 待野翔太君・友永健太君 [資料/ビデオ](#)
- ・技術賞「3 GIMを使用したGPS+GLONASS vs GPSの位置精度比較」
チームmochi 望月康平氏 [資料/ビデオ](#)
- ・技術賞「自動散水システム」
九州工業大学 中山一平君、永山雄一君、Avinash Dev Nagumanthri君 [資料/ビデオ](#)
- ・技術賞「心拍数測定システム」
東京都立小石川中等教育学校 佐藤和哉君（高校二年生） [資料/ビデオ](#)

このほかにも入選からもれたのもありましたが、どれも素晴らしい作品でした。

第2章

3 GIMのAssisted GPSについて

1. 衛星測位システム（GNSS）利用について

3 G通信モジュールのGPS機能利用

▼衛星測位システムGNSS

GPS・GLONASS（ロシア衛星）の利用可

▼Assisted GPSを利用することで
屋内でも短時間に誤差範囲の少ない位置情報取得

▼Google Mapで位置情報確認

▼移動体（自動車・近海船・動物・人間など）の
リアルタイムの追跡が簡単に可能

ここでの事例は、
1) 10秒ごとにGPS測位し
2) 1分ごとに3 G通信で、
サーバに時刻と位置情報を
アップ

応用展開は

- 1) 子どもや老人などの追跡
(既存製品あるが開発に制限あり)
 - 2) 近海漁船の位置管理
 - 3) 集配トラックの位置情報管理
 - 4) ゴルフ場カートの位置管理
- などなど多く利用可能に・・・

本件は、RaspberryPiでは難題



GoogleMap利用（アプリ開発）

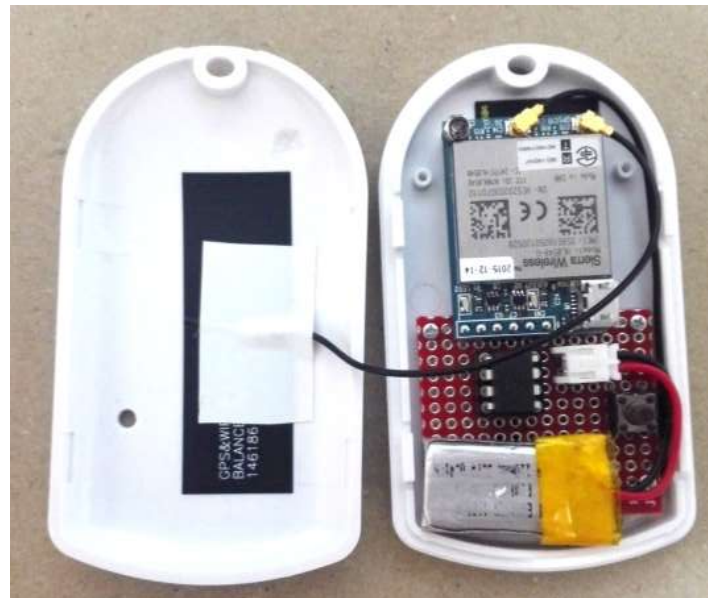
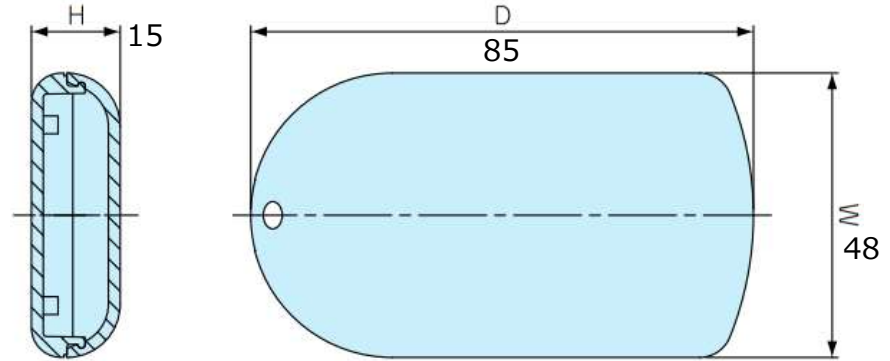
使ったものは
Arduino + 3 GIM シールド +
3 GIM + 5V バッテリー
(1500Wh で 9 時間 利用)

2. GPS機能を使った携帯センサデバイス

小型で軽量のGPS付センサデバイスを試作
僅か8.5cm×4.8cm×1.5cmのケースで収納

機能：ボタンを押すことで光センサを取得し、
3G通信でサーバに時間と位置、センサ値を
アップ

- ・エアブレーンモードを使って省エネモード
- ・3GとGPSともにフレキアンテナ採用
- ・Arduino互換チップ採用



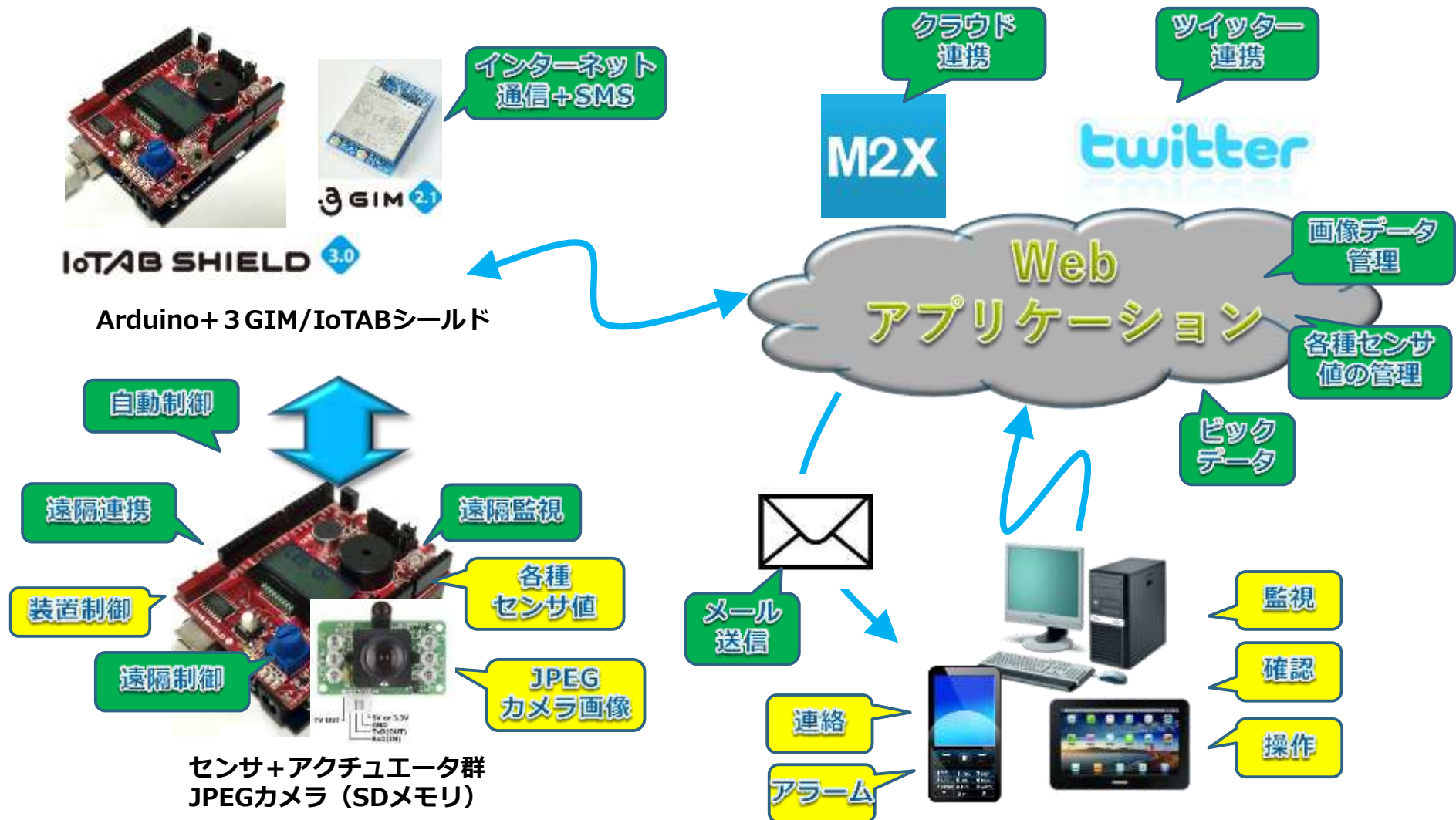
もくじ

1. 全体機能概要
2. 遠隔アクチュエータ起動
3. 遠隔監視（センサ値のメール受信）
4. ツイッター連携
5. クラウド連携（M2X・xively.com利用）
6. 画像データのサーバアップ（httpPOST利用）
7. 3G通信技術との連携
8. Arduino用3GIM普及
9. 他の無線機器との比較



第3章 3GIMの可能性

1. 全体機能概要



2. 遠隔アクチュエータ起動

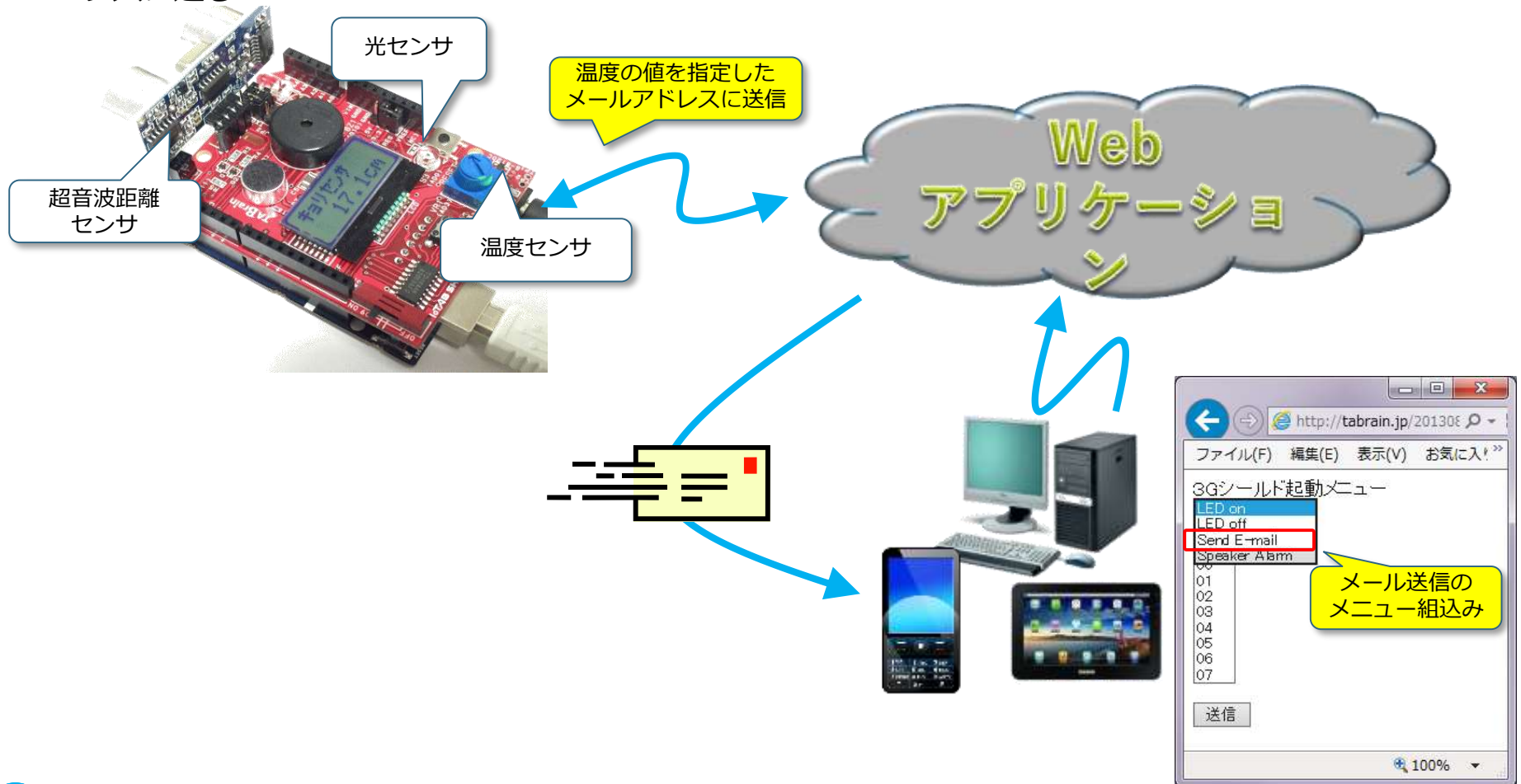
* ARDUINO+ 3 GIM+ IoTABシールドを使って、
IoTABシールド上にあるスピーカを遠隔操作で5秒ほど鳴らす

課題：IoTABシールド上の**D9番**ピンとGNDに接続したスピーカを5秒ほど鳴らす



3. 遠隔監視（センサ値のメール受信）

* ARDUINO+ 3 GIM+ IoTABシールドを使って、
IoTABシールド上の温度センサ値を自分のメールアドレスに送る



4. ツイッター連携①

- ▶ 3 GIMで取得したデータを連続してツイート可能。もしくは異常データが出た場合にツイートで知らせる。
- ▶ その逆操作も可能。3 GIMがツイートを見張り、何かあれば、3 GIMが感知して、挙動を起こすことも可能。

(簡単に双方向の連携が可能で、無料で利用可能)



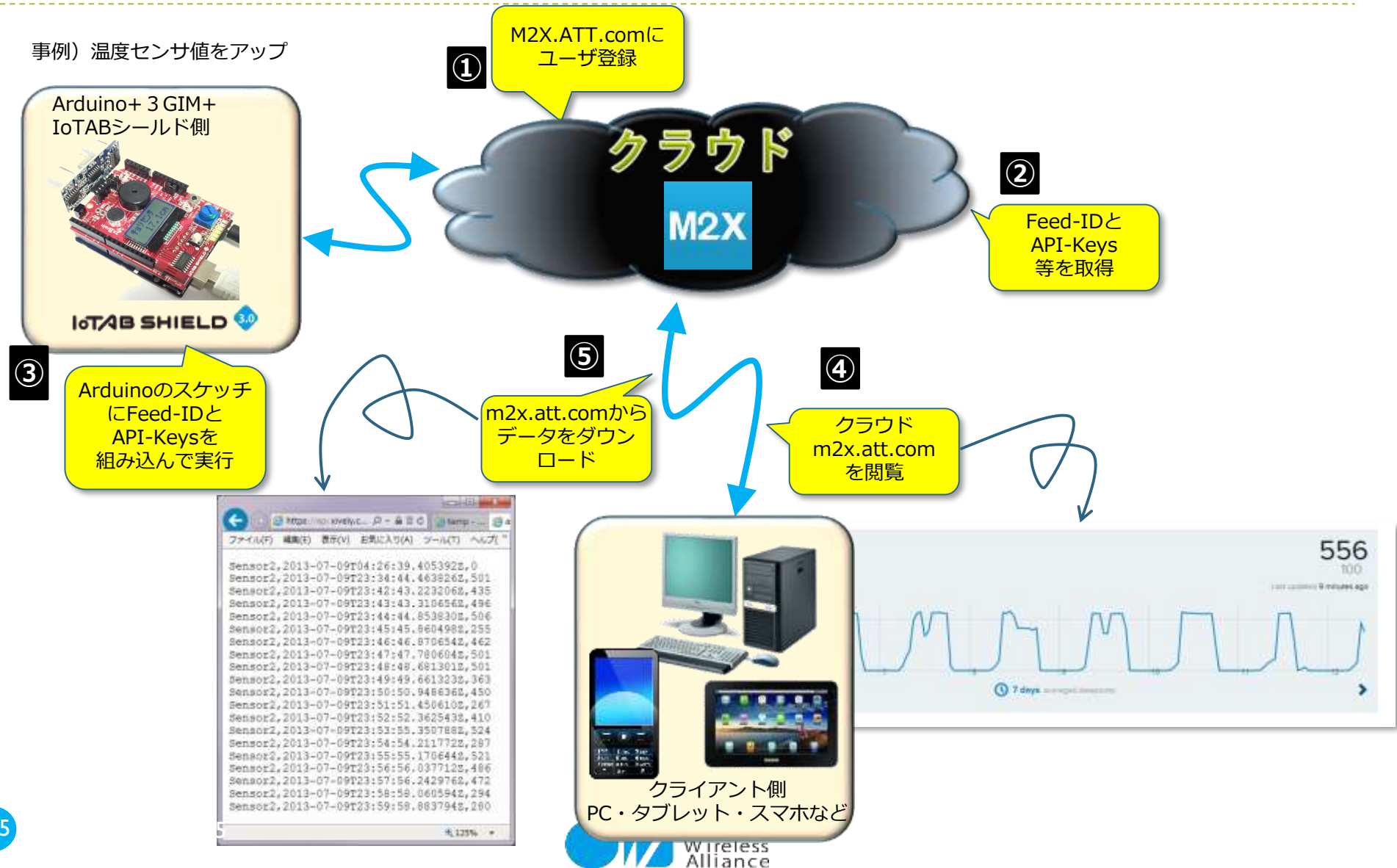
4. ツイッター連携②

- ▶ 環境モニタリングの構築事例紹介：環境モニタリングデータは、クラウドサーバ（COSM.COMなど）にアップ
- ▶ 異常なセンサ値が取れたときに、ツイッターに知らせる。（関係者一同にツイートされる）



5. クラウド連携（M2X利用）①

事例) 温度センサ値をアップ



5. クラウド連携（M2X利用）②

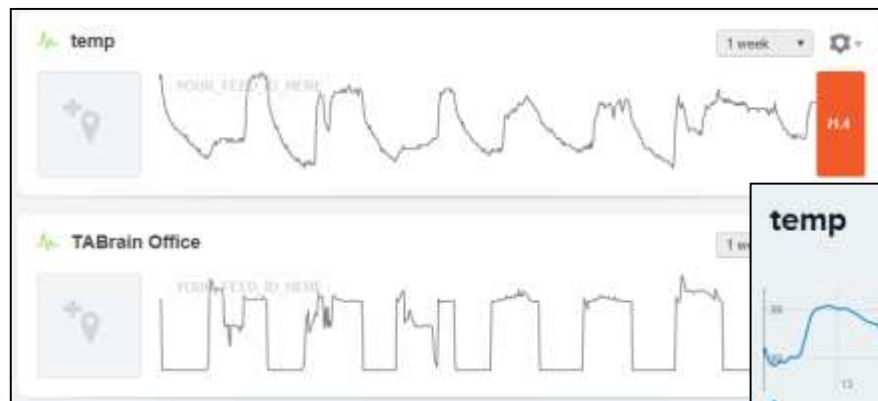
- ▶ 3 GIMを使って、簡単にクラウドへデータ連携が可能（教育で紹介）国内・海外でも多く利用中。
- ▶ Arduino上のセンサデータをアップすることが容易。
- ▶ すでに、大震災後の放射能測定データをアップしている事例も存在。
- ▶ 誰もが、無償で簡単に利用ができることで人気。



5. クラウド連携（xively.com利用）③

▶ オフィスの監視システム

- ▶ 温度センサと光センサをArduino + 3 GIM上で構築し、データをxively.comにアップ。毎時4回（15分毎）
くただし、光センサによる変化が大きくなった場合には、都度データ送信
- ▶ 開発時間は3時間ほどで、2012年10月から稼動中

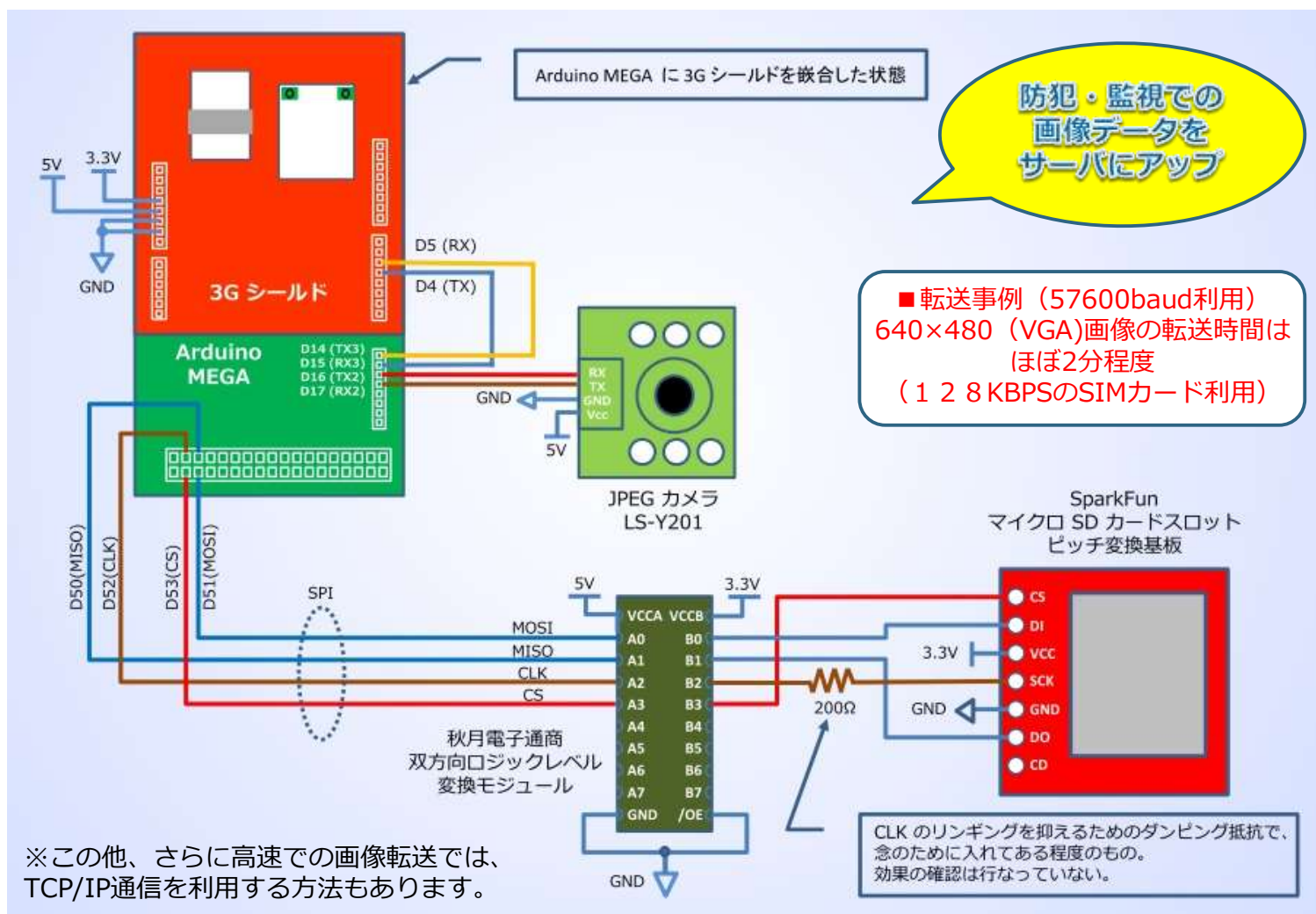


今年春にxively.comに変更されてもアップ中

2012年10月18日から
xively.comにアップ



6. 画像データのサーバアップ (httpPOST利用)



7. 3G通信技術との連携

- ▶ オープン ワイヤレス アライアンスではオープンソースハードウェア上の3G通信技術を提供
- ▶ 誰もが簡単に短時間で通信技術が利用できる
- ▶ ゲートウェイとして簡単に利用可能
- ▶ センサ技術との連携でネット接続が容易
- ▶ 月額掛るSIMカードが、月500円よりも安価に使える環境下に（M2Mシステム開発の促進に）



8. Arduino用 3 GIM普及

オープン ワイヤレス アライアンス設立（NPO法人化中） <2012年1月>

3 GIMの普及の目的

- ① 最先端で高度な技術をArduino上で簡単に学べる
- ② あらゆるものをネットにつなぐアイデアの場を提供
- ③ センサ技術との連携で将来に役立つデータを収集・分析



9. 他の無線機器との比較

多量生産でないと
融通が利かない
メーカーへ技術流出

M2Mでの展開でのメリット

比較項目	3 GIM	モバイル・ルータ機器	スマホ
通信エリア	○ 広域な3G通信網のみ (将来はLTEも視野に)	○ 3G+LTE / WiMAX	○ 3G+LTE / WiMAX
通信速度	△ (3G利用SIMカードに依存)	○ (高速だと通信費大)	○ (高速だと費用大)
通信費用	○ 安価なMVNOのSIMが利用可能	× 通信費はキャリア依存	△ スマホのキャリア依存 (SIMフリー であれば安価なSIM利用可能)
消費電力	◎ 省電力化の技術対策が容易	△ 機器依存、省電力化が難しい	× 省電力化は難しい
開発環境 (試作)	○ 取得しやすい開発環境	× 他に機器利用が必要	△ Androidなどの環境利用できるが 技術的レベルは高い (M2Mではスマホの画面は不要)
機器量産	◎ 安価に抑えることが可能 (アライアンス対応)	△ メーカー依存 (多量生産でないと融通不可)	△ メーカー依存
M2Mでの利用評価	◎ 短時間に高機能の試作品実現可能 (量産も容易) 中小企業でも導入開発が可能	△ 他の機器との連携で価値が出るもの (機器開発・ソフト開発などはメー カに依存)	× M2Mでのスマホ利用は不向き

もくじ

- 第1章 ツイッターID登録
- 第2章 FTPサーバID登録
- 第3章 M2XクラウドID登録
- 第4章 IoTシステムのワイヤレス技術
- 第5章 IoTセキュリティ対応とは
- 第6章 添付資料



Tabraino(農業・防災他用)

第VI編 補足資料

第1章 ツイッターID登録

ツイッターID登録

<https://twitter.com/signup> にアクセス



次へ

アカウントを作成

名前

呼び名
(ID名になる)

0/50

電話番号

かわりにメールアドレスを登録する

メールアドレス
または電話番号

第2章 FTPサーバID登録

1. フリーサーバについて

PHPが使えるフリーサーバを利用して下記の作業を試してみましょう。

- 1) フリーサーバにユーザ登録
- 2) PHPが使えるFTPサービス（サーバ）を選択する
- 3) FTPを利用して後述のプログラム3種類をアップロード

ここでは、無料レンタルサーバ「スターサーバーフリー」を例として手順をご案内します。

2. フリーサーバ「スターサーバーフリー」の登録

- ▶ フリーサーバ：スターサーバーフリーに新規登録

<https://www.star.ne.jp/free/>



3. フリーサーバ「スターサーバフリー」の登録

Netowl 新規会員登録

ネットオウルの新規会員登録は **かんたん 3ステップ**

1. 認証IDの取得・確認
2. 登録情報の入力・確認
3. サービスを選択しよう!

① 認証IDを取得して、登録フォームへ移動しよう!

認証IDを取得

当社の「ネットオウル」サービスにご登録いただくメールアドレスを入力し、「認証IDを取得」を押してください。ご入力されたメールアドレスに認証IDが送信されます。

メールアドレスを入力!

確認メールより認証IDを確認

このウインドウを開く前に、ご登録メールアドレスの受信をおこなってください。確認メールより「認証ID」をご確認ください。

◆登録メールアドレスに関する情報

【登録メールアドレス】 : sample@example.com

【認証ID】 : sample1example1sample1example1sa

認証IDを入力し、登録フォームへ移動!

「認証ID」を下記に入力し「登録フォームへ移動」を押してください。

認証IDを入力!

②会員登録

Netowl スターサーバ新規申込フォーム

[スターサーバ管理](#) [⇒ サービスサイトへ](#)

SMS認証

スターサーバフリーのお申し込みには、SMS認証が必要です。
SMS認証とは、携帯電話のSMS(ショートメッセージサービス)を利用した認証方法です。
スターサーバフリーお申し込みには、SMS認証を完了する必要があります。
以下のボタンより、画面の案内に沿ってお手続きください。

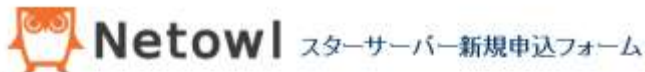
スターサーバフリーについて

SMS認証が完了した後、以下のプランをお申し込みいただくことが可能です。

フリー	フリー 容量増加	フリー PHP+MySQL
容量 2GB 初期費用・月額 無料	容量 4GB 初期費用・月額 無料	容量 2GB 初期費用・月額 無料
独自ドメイン 1個	独自ドメイン 1個	独自ドメイン 1個
PHP —	PHP —	PHP ○ ※一部制限あり
MySQL —	MySQL —	MySQL 1個

③SMS認証後、サーバ種類を選択 (フリーPHP+MySQL)

4. フリーサーバ新規申込



③ スターサーバー管理 → サービスサイトへ

新規お申し込み

サーバーの新規取得お申し込みができます。

プラン	フリー PHP+MySQL
サーバーID/ドメイン	starfree.jp
月額	無料
ディスク容量	2GB
マルチドメイン	1個
メールアドレス	なし
PHP・MySQL	利用可能
独自SSL	不可
広告表示	あり

④ サーバー名を入力

規約

第1章 【総則】

第1条 定義等

本規約はネットオウル株式会社（以下、「当社」）が提供するホスティング・PHPアプリケーション提供サービス「スターサーバー」のサーバープラン「エコノミー」「ライト」「スタンダード」「プレミアム」「ビジネス」「フリー」「フリー 容量増加」「フリー PHP+MySQL」、ドメイン取得・管理サービス「スタードメイン」の利用者に提供するサーバー「スタードメイン無料サーバー」「旧スタードメイン契約特典サーバー」「旧スタードメイン契約特典 PHP+MySQLサーバー」（以下、「本サービス」）の利用に対して適用されます。

第2条 規約の適用及び変更

1. 利用者は新規利用、および利用継続中において、本規約に同意されているものとみなす。

☐ 利用規約に同意する

キャンセル 確認画面

③ スターサーバー管理 → サービスサイトへ

新規お申し込み

サーバーの新規取得お申し込みができます。

月額250円(税込)～高機能プランが利用可能！

プラン	フリー PHP+MySQL	StarServer ライトプラン
サーバーID		
初期ドメイン		
月額	無料	250円～
ディスク容量	2GB	50GB 25倍！
マルチドメイン	1個	50個 複数サイトを運営できる！
メールアドレス	なし	500個 メールアドレスが作成できる！
PHP・MySQL	利用可能	利用可能
独自SSL	不可	可能 無料・無制限！
広告表示	あり	使いやすさアップ！

⑤ フリープランを選択

選択したプランで申し込む

有料プランを2週間無料で試す

Netowl スターサーバー新規申込フォーム

③ スターサーバー管理 → サービスサイトへ

新規お申し込み

サーバーの新規取得が完了しました。ログインするをクリックしてください。
設定完了後、WEBサイト（）が表示されるまで 時間経過かからる場合がございます。

※独自ドメインをご利用の場合は、和ログインするからログイン（独自ドメイン）から通知 いただけます。

ログインする

メンバー管理ツールに戻る

5. サーバー管理ツールからFTPアップロード

⑥無料プラン管理

Netowl サーバー管理ツール

こんにちは、あるでい さん

3 ステータスサーバー管理 → サービスサイトへ

サーバーアカウント一覧

現在管理中のサーバーアカウントは以下の通りです。

無料プラン (フリー/フリー 容量増加/フリー PHP+MySQL/フリー WP/フリー WP プレミアム)

PHP+MySQL/フリー WPはご利用期限の更新を行ってください。

ご利用期限の2か月前から可能です。

ご利用期限までに更新しない場合は該当のサーバーアカウントを凍結します。

サーバーID	プラン	契約単位	状態	ご利用期限	ログイン
artex	フリー PHP+MySQL	契約中	稼働中	2018/02/28 無料更新	サーバー管理ツール 契約メニュー

⑦サーバー管理ツール

StarServer Free サーバー管理ツール

プラン名: フリー PHP+MySQL ユーザー名: ユーザー名

各種情報

- アカウント情報
- サーバー情報
- ホームページ
- FTPアカウント管理
- データベース設定
- パスワード管理
- サイト設定
- アクセスログ管理
- 更新インストール
- WordPressセキュリティ設定
- おが基本設定
- おがバージョン更新

対象ドメイン選択

FTPアカウントの設定を行ったドメインを選択してください。

ドメイン名	FTPアカウント
artex.starfree.jp	0 個

⑧FTPアカウント設定

アドレス

1つ上のフォルダへ ルートフォルダへ 新規作成 アップロード

ディレクトリツリー

全て閉く 全て開く

名前	操作	サイズ	更新日時
index.html		3 KB	18/11/28 12:20:05
default_page.php		34 KB	18/11/28 12:20:05
.htaccess		1 KB	18/11/28 12:20:08

⑨WebFTPにログイン

12. アップロードする3ファイル

■メール送信機能 (sendmail.php)

温度センサ値をメールにて宛先に送信するプログラム

```
<HTML>
<HEAD><TITLE> 3GIM send TEMP sensor e-mail </TITLE><HEAD>
<BODY>

<H3> 3GIM TEMP sensor get </H3>
<?php
if(mail($_GET["email"], // to
    'Hi 3GIM sensor E-mail' ,// タイトル
    ' 3GIM ALARM ' . "¥r¥n" . 'DATE = ' . date('Y-m-d') .
    ' TIME = ' . date('H:i:s') . "¥r¥n" .
    'TEMP = ' . $_GET["temp"] ,//本文
    'From: 3GIM<temp@tabrain.jp>' . "¥n" .
    'X-Mailer: PHP/' . phpVersion()))
{ echo '<B>SUCCESS TO SEND</B><BR>';}
else
{ echo '<B>faile to mb_send_mail</B><BR>'; }
?>

</BODY>
```

■選択メニュー (3gform.html)

遠隔制御のためのメニュー操作画面

```
<form action = "3gmakefile.php" method = "post">
<p> 3GIM menu<br>
<select name="cmd" id="cmd">
<option value="LED on">LED on</option>
<option value="LED off">LED off</option>
<option value="Send E-mail">Send E-mail</option>
<option value="Speaker Alarm">Speaker Alarm</option>
</select></p>
<input type="submit" value="command send">
</form>
```

■制御ファイル出力 (3gmakefile.php)

制御メニュー操作画面よりコマンドファイルの作成

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<?php
$x=$_POST["cmd"] . "¥n";
$fn1="temp" . ".xxx";
$fp=fopen($fn1,'w');
fputs($fp, $x);
fclose($fp);
$fn2 = '3gsa' . '.xxx';

copy($fn1, $fn2);
echo 'command = [' . $x . ']';

?>
<br>
<input type="submit" name="buttonName" value="back" onClick="history.back()" />
```

第3章 M2XクラウドID登録

4. M2X (AT&T IoTサービス) のID登録

The image shows the M2X AT&T IoT Platform website. The main navigation bar includes 'AT&T IoT Platform', 'Services', 'Solutions', 'Solution Providers', and 'Contact'. A 'LOGIN' button is in the top right. The main content area features the M2X logo, 'FEATURES', and 'SHOWCASE' tabs. A large blue banner reads 'Build Intelligent, Enterprise Grade Connected Device Solutions' and lists services: 'time-series data storage, device management, message brokering, event triggering, alarming, geo-fencing, and data visualization'. A green button 'GET STARTED FOR FREE' is highlighted with a red box and a callout 'ID登録へ' (Go to ID registration). A red callout points to the URL 'https://m2x.att.com/' with the text 'にアクセス' (Access).

The login page is shown on the right. It has a header 'Login with your AT&T IoT Platform Account'. There are input fields for 'Username / Email' and 'Password', a 'LOGIN' button, and a link 'Forgot your username or password?'. Below these are 'Or login using:' options for 'GITHUB' and 'AT&T DEVELOPER'. A red callout 'ID新規登録' (New ID registration) points to the 'Create an account here.' link. A red callout at the bottom right says '必要情報を入力・送信後、設定アドレスにメールが届く。有効化 (activate) ボタンを押下して登録完了。' (After entering required information and sending, an email will arrive at the designated address. Press the activate button to complete registration).

A confirmation email template is shown at the bottom left. It says 'Congratulations on your new AT&T IoT Platform account. To activate your AT&T IoT Platform Account please click the following activation link:'. There is a blue button with an envelope icon and the text 'Confirm your email and activate your account'. Below it, it says 'If you're having trouble confirming your email, have questions, or need support, please visit our Support Center.'

At the bottom center is the 'Open Wireless Alliance' logo.

1. ワイヤレス技術について
2. ワイヤレスによる通信網
3. キャリア・セルラー系（3GやLTE）の今後
4. 非セルラー系の動き
5. LoRaとSigFoxの日本での動き
6. LoRaWANサービスイメージ（SoftBank）
7. SORACOMのLoRa戦略
8. 今後の通信網の利用にむけて

第4章 IoTシステムのワイヤレス技術

1. ワイヤレス技術について

- ・ワイヤレス技術の利用
 - ・赤外線・超音波・デジタル無線などが存在
 - 1) 赤外線：リモコンで今でも利用（指向性、情報漏えいに強い、省エネなど）
 - 2) 超音波：ソナー・ワイヤレス充電などに利用
 - 3) アマチュア無線：電波法に基づき許可されたアマチュア局を設けた無線
 - 4) 特定小電力トランシーバ・デジタル簡易無線など
 - 5) 携帯電話無線：WAN
 - 6) 船舶無線・航空無線・列車無線・車両無線など
 - などなど
- ・ワイヤレスつまりコードレスだから設置が簡単
- ・課題は、通信距離・通信速度・通信容量・消費電力・アンテナ・許認可など
- ・IoTでは、PAN・WANセルラー系・非セルラー系、

2. ワイヤレスによる通信網

	LTE	3G	WiMAX	WiFi	Bluetooth	ZigBee	EnOcean	Wi-SUN
IEEE規格	Long Term Evolution	ITU-2000	802.16	802.11a/b/g/n	802.15.1	802.15.4		IEEE802.15.4g
周波数帯	1.25/5/10/20MHz	800MHz/2GHz	2.5GHz	2.4GHz/5GHz	2.4GHz	866MHz/915MHz/2.4GHz	315MHz (日本)	920MHz (日本)
通信距離	最大10Km前後	最大10Km前後	最大50km	最大100m程度	最大100m程度	100m以上 (2.4GHz)	ビル内30m 最大300m	約1Kmほどまで
通信速度	最大85Mbps	最大7Mbps	最大70Mbps	最大300Mbps	最大3Mbps	最大250kbps	125kbps	~200kbps
接続ノード数				32	7	65535		
電池寿命	機器依存	機器依存	機器依存	数時間	数日 (BLEは数年)	数年	長寿長電池なし	特定小電力無線
ネットワーク構成	スター型	スター型	スター型	スター型	スター型	スター型・メッシュ型など	スター型	スター型・メッシュ型など
主な用途	広域エリア・スマートフォン対応	広域エリア・携帯電話	ほぼ広域エリア・モバイル通信	狭いエリア・無線LAN	携帯電話・コンピュータ付属・他	セキュリティ機器	バッテリーレス照明スイッチ他	ガス・水道・電気のメータ系端末
備考	今後IoT向けのCat1やCatMなどが登場	4Gでは、通信速度が下り100Mbps、上り50Mbps実現	速度と伝達距離が次世代技術と現行技術の中位にあり、今後のWiMAX 2に期待		機器やロボット制御などで利用 BLEは、商用・モニタリング・監視・見守りなど	ロボット・などで利用	物理的なスイッチング動作を電気エネルギーに変換 (建物内で利用)	東京電力でメータ監視に利用開始・マルチホップなど特徴



無線の魅力： ケーブル（電源・電波）なしでの通信 ⇒ 可搬性や移動が可能



3. キャリア・セルラー系（3GやLTE）の今後

3Gの今後

- ・古い設備をどこまで維持し続けるか？

auは2020年で終了予告

NTTドコモは、MVNOで、3G回線含めて安価に提供

→ 低価格のSIMカードが増大 → IoT向けに利用増大
(2020年以降も利用し続けるのでは)

5Gは2020年ごろか？

IoT向けの携帯電話（LTE）規格

<http://businessnetwork.jp/Detail/tabid/65/artid/4274/Default.aspx>

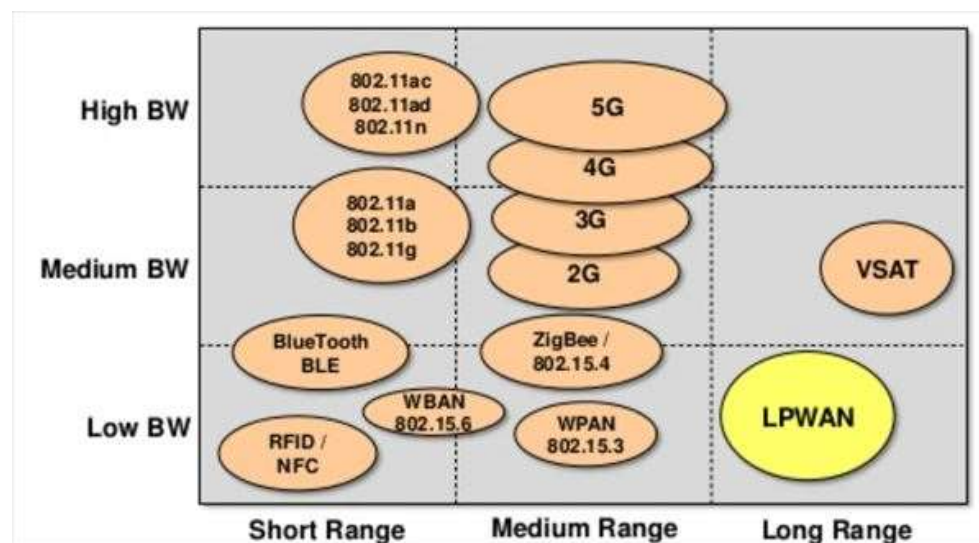
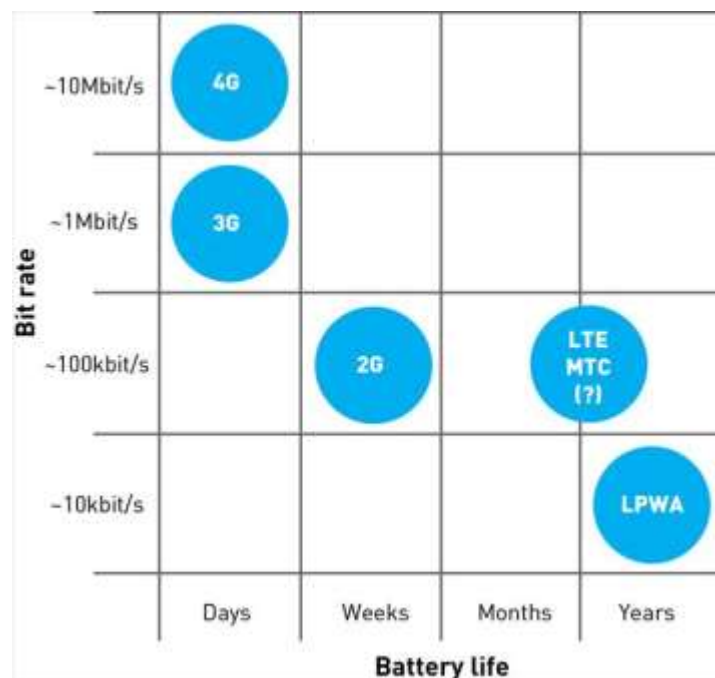
	端末の動作帯域幅 (最大)	MIMO	最大通信速度 下り/上り	モジュールの価格 (導入時期)
LTE Cat.4 (参考)	20MHz	2 × 2	150/50Mbps	40ドル以上
LTE Cat.0 (R12)	20MHz	なし	1/1Mbps	10 ～ 15ドル (2016年)
LTE Cat-M (R13)	1.4MHz	なし	1/1Mbps	5 ～ 10ドル (2017年)
NB-IoT (R13)	200kHz	なし	未定 (100kbps程度)	2 ～ 4ドル (2017年)

エリクソンの資料などから作成、端末価格は同社の予測

4. 非セルラー系の動き

▶ 長距離・省エネ通信モジュール（LPWAN）

LoRaとSigFoxが欧州から日本にも浸透中



(LPWAN Technologies for Internet of Things (IoT) and M2M Scenarios <http://www.slideshare.net/PeterREgli/lpwan> より)

(<http://www.analysismason.com/About-Us/News/Insight/For-IoT-CSPs-may-need-multiple-networks-each-optimised-for-a-different-use-case/> より)

5. LoRaとSigFoxの日本での動き

低価格化と省エネの動きで
欧州から2つの通信モジュールが届く

- SigFox
通信距離が50Kmまで
通信スピードが遅く、送信のみ
通信容量も少ない
→ メータ系などに利用
- LoRa
通信距離が15Kmまで
通信スピードもそこそこ遅く
送受信
→ センサネットワークに利用

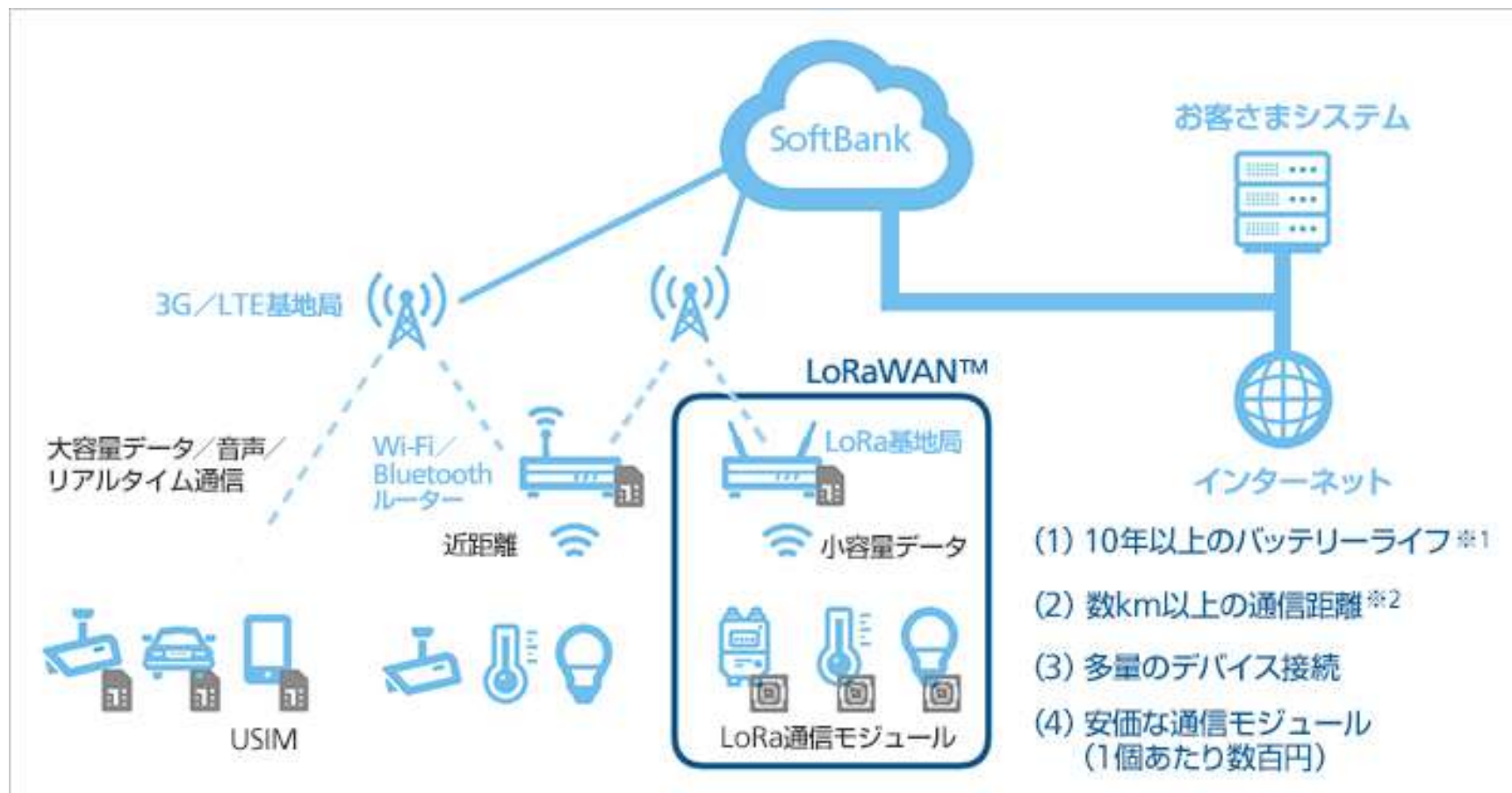
IoT向けの主な通信方式と陣営

	通信速度	提供状況
	主な企業	
SIGFOX (シグフォックス)	0.1kbps	提供中
	シグフォックス、テレフォニカ、ドイツテレコム、SKテレコム、NTTドコモ	
LoRa (ローラ)	10kbps	提供中
	セムテック、IBM、シスコシステムズ、仏ブイグ・テレコム、蘭KPN	
NB-IoT	100kbps	2017年にも開始
	エリクソン、華為技術(ファーウェイ)、インテル、ボーダフォン、中国移動(チャイナ・モバイル)、KDDI	
第5世代(5G) 携帯通信	10Mbps程度	2020年前後

(2016/3/1付 日本経済新聞 より)

6. LoRawANサービスイメージ (SoftBank)

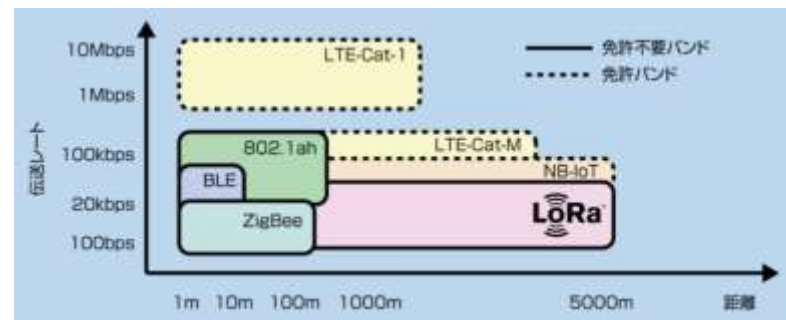
SoftBankによるLoRaWANビジネス(2016年9月12日発表から)



<http://k-tai.watch.impress.co.jp/docs/news/1019604.html>

7. SORACOMのLoRa戦略

SORACOMが発表したLoRa戦略
ライセンスなしで利用できる920MHz帯を利用



	Wi-SUN	ZigBee	LoRa	Sigfox	HaLow
通信距離	1 km	100 m	5 ~ 15 km	3 ~ 10 km 30 ~ 50 km	1 km
伝送速度	50k~400 kbps	20k~250 kbps	980~30k bps	100 bps	~150kbps
省電力			送信時18mA 待機時0.001mA		
推進団体等	Wi-SUN Alliance	ZigBee Alliance	LoRa Alliance IBM, Semtech	Sigfox他	WiFi Alliance
特徴・課題	日本国内のみ	通信距離が短い	オープン	仕様が クローズド	2018年以降

<https://iotnews.jp/archives/20770>

8. 今後の通信網の利用にむけて

▶ データ通信の価値

- ▶ キャリア通信網では音声通信とデータ通信の2種類利用
- ▶ 音声通信は有限個数での通信。しかしデータ通信は待ち状況はあるが、ほとんどデータ送受信は可能。

端末側で取れたデータをキャリア通信でやり取り可能。

※ 地震などの災害時には、音声通話がフルとなり、繋がり難くなる。

▶ データ通信での音声処理

- ▶ Skypeなどの処理をデータ通信を通じて電話を掛ける
- ▶ Wi-Fiの通信可能なところでのスマートフォンは、無償Skypeでコンピュータや他のスマートフォンに繋ぐことが可能。

▶ データ通信の利用価値

- ▶ 動画や画像以外のテキストデータだと、瞬時に簡単にデータ転送可能
- ▶ 何ができるか ⇒ 違う場所（エリア）での継続的・断片的なデータ転送が可能

▶ IoTデバイスとスマートデバイスとの連携も強化

▶ IoTデバイス（異なる通信機器同士）間の連携強化

1. 情報セキュリティとは
2. 認識が必要な「脅威」と「脆弱性」
3. セキュリティ対策技術
4. IoTセキュリティの動き
5. IoTシステムでのセキュリティ技術の課題



第5章 IoTセキュリティ対応とは

1. 情報セキュリティとは

▶ 情報セキュリティの要件（CIA）

- ▶ 機密性（Confidentiality）：権限範囲内でのアクセス制限、暗号化・認証・アクセス制御
- ▶ 完全性（Integrity）：情報資産の破壊・改ざん対応、ハッシュ関数・デジタル署名
- ▶ 可用性（Availability）：継続的なアクセス対応、二重化

▶ リスク対処

- ▶ リスクの回避：中止
- ▶ リスクの保有：最小限で受け入れ
- ▶ リスクの移転：他への乗り換え
- ▶ リスクの低減：少ない方策選択

2. 認識が必要な「脅威」と「脆弱性」

脅威：情報資産を脅かすもの

脆弱性：システム設計上の欠陥

1) ネットワークスキャンとパスワードクラック

- ・ネットワークスキャン：サイバー攻撃での攻撃対象を狙ったネットワーク情報の収集
(ホストやネットワーク機器の製品名・バージョン、IPアドレス、稼働中のサービス等を特定)
→ ファイアウォールのフィルタリングルールにより、特定のサービスのみ接続を許可
- ・パスワードクラック：パスワードを盗み、ホストへ侵入すること
→ アカウントロック機能の設定、ワンタイムパスワードや生体認証の導入

2) バッファオーバーフロー

- ・プログラミング上のバッファオーバーフロー（BOF）がプログラムの脆弱性につながる
→ セキュリティパッチの適用やバッファオーバーフローを起こさないプログラミング対応

3) マルウェア

- ・コンピュータウィルスやワーム、トロイの木馬、ボット、スパイウェアなどの総称
- ・コンピュータの感染などによって、外部からの遠隔操作が行われ、データの破壊や改ざん、他のコンピュータへの感染
→ ウィルス対策ソフトの導入

3. セキュリティ対策技術

・情報セキュリティ対策

技術	対策
認証	パスワード認証、ICチップ認証、生体認証など
暗号化	共通鍵と公開鍵、デジタル署名
耐タンパー性	物理的なデバイスのアクセスや盗難後のデータアクセス対策
アクセス制御	アクセスできる範囲を限定
侵入検知	攻撃パターンをデータベース化し、信号路を監視、攻撃をリアルタイムで検知

・IoTセキュリティ対策

- ・サーバ・クラウド側（一般の情報セキュリティ対策で対応）
- ・IoTゲートウェイ・IoTデバイス・IoTノード

対象	対策
3G・LTE	SIMカード盗難対策、IMEI対応（個別ID）対応
公衆無線	暗号化・PW定期的変更（プログラミングが大変）
ローカル	グループごとのID/PWの個別対応（ただし設定が大変）

4. IoTセキュリティの動き

- ▶ 情報セキュリティの国際標準の動き
 - ▶ IoTの適用分野の家電・自動車・制御システム等幅広い分野で、セーフティ（機能安全）に関する国際標準制定の整備が進んでいる
 - ▶ 国際標準化の動きのあるセーフティとセキュリティ
 - 原子力、プロセス産業、自動車、医療機器、家庭用電気機器、産業機械
 - IEC61508（電気・電子・プログラマブル電子）
 - ISO/IEC 27001（情報セキュリティマネジメントシステム）
- ▶ IPAによる「つながる世界の開発指針」
 - ▶ IoT製品開発において、企業としての方針策定・リスク分析とそれに基づいた設計方法、製品導入後の保守・運用法を17項目でまとめた
<https://www.ipa.go.jp/files/000060387.pdf>（第2版）
- ▶ 総務省による「IoTセキュリティガイドライン」
 - ▶ 長期に利用されるIoTシステムに対するもので特に安全な生活を守るための方策
 - ▶ 特にコネクテッドカーやスマートハウス、医療分野にフォーカス
http://www.soumu.go.jp/main_content/000428393.pdf

5. IoTシステムでのセキュリティ技術の課題

▶ IoTデバイス・ゲートウェイ関連

▶ 個別IoTノードやデバイス間のローカル通信

- ▶ グループینگ対応
- ▶ 親子の関係性の対応
- ▶ ルータ（中間機）のセキュリティ対策

▶ ローカル無線機は個別対応での知識が必要

- ▶ ZigBee（Xbee）での送信元と送信先の個別アドレス設定（1つ1つの設定が大変）
- ▶ TWE-Lite：通信状態では暗号化通信が行われる
ただし、個別でのID/PWを設定していないと同製品で筒抜け
- ▶ WiFi：PWの設定が必須だが、自動変更の場合にはプログラミングが大変

▶ 3G通信・LTE通信のデバイスでは

- ▶ SIMカードとキャリア（MVNO）側サーバとの関係でセキュリティ対応はほとんど不要（プログラミング上は、公衆LAN：Wi-Fiよりもとても簡単）

▶ IoTサーバ・クラウド関連

▶ 従来の情報セキュリティ対策が通用

特に安全を脅かすデータに対してセキュリティを高める必要性が発生

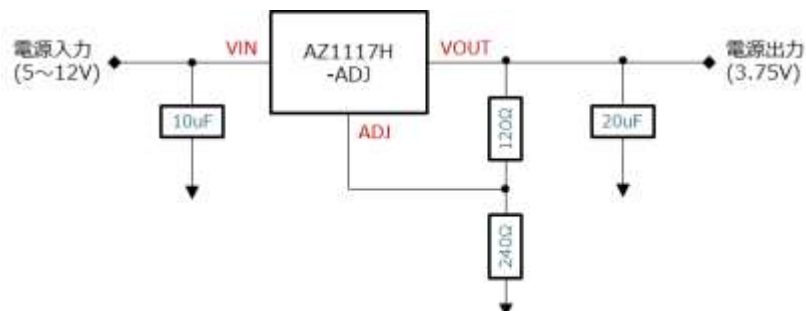
まとめ

1. 3Gシールド（3GIM）が簡単なのはATコマンドを使わず、タイミングを配慮することがなく、インターネット接続・通信できること
2. 3Gシールド（3GIM）のインターネット接続・通信は、2種類あり
 - ① UARTの\$コマンドで制御する場合（Arduino以外でも利用可能）
 - ② Arduino用のライブラリで制御する場合
3. メール送信やツイッター送信の基本を知る事で、ネット活用が簡単になる
4. ネット上の情報・知識を活用することが重要
5. IoTとは、モノのインターネットと呼ばれ、インターネット接続・通信を経て、スマートデバイスとの連携や、自動連携で意味を成す。
6. 今後、センサ・ネットワークと呼ばれるセンサ値を蓄積し（ビックデータ）、その内容を分析し、活用する人工知能（AI）化で、価値を出すことが夢見られている。
7. 現場・現場の対応によってはその価値を見出すことができない。（現場力が重要）

第6章 添付資料

【補足資料 1】 5Vから3.7Vを作り出す回路例

- ◆ 5～12Vの電源(ACアダプタ等)から3GIMが必要とする3.7V電源を作り出す回路の例を以下に示す：



【図】 5Vから3.7Vを出力する電源回路例

- ◆ 必要な部品は下記の通り：

No	分類	パーツ	数量	実売価格(円)	補足・販売店
1	3端子レギュレータ	AZ1117H-ADJ	1個	30	秋月電子にて10個単位で販売
2	抵抗	1/4W抵抗(240Ω)	1個	10	秋月電子・千石電商等で販売
3	抵抗	1/4W抵抗(120Ω)	1個	10	秋月電子・千石電商等で販売
4	積層セラミックコンデンサ	25V 10μF	1個	80	秋月電子・千石電商等で販売
5	タンタルコンデンサ(または積層セラミックコンデンサ)	10V 22μF	1個	42	千石電商等で販売

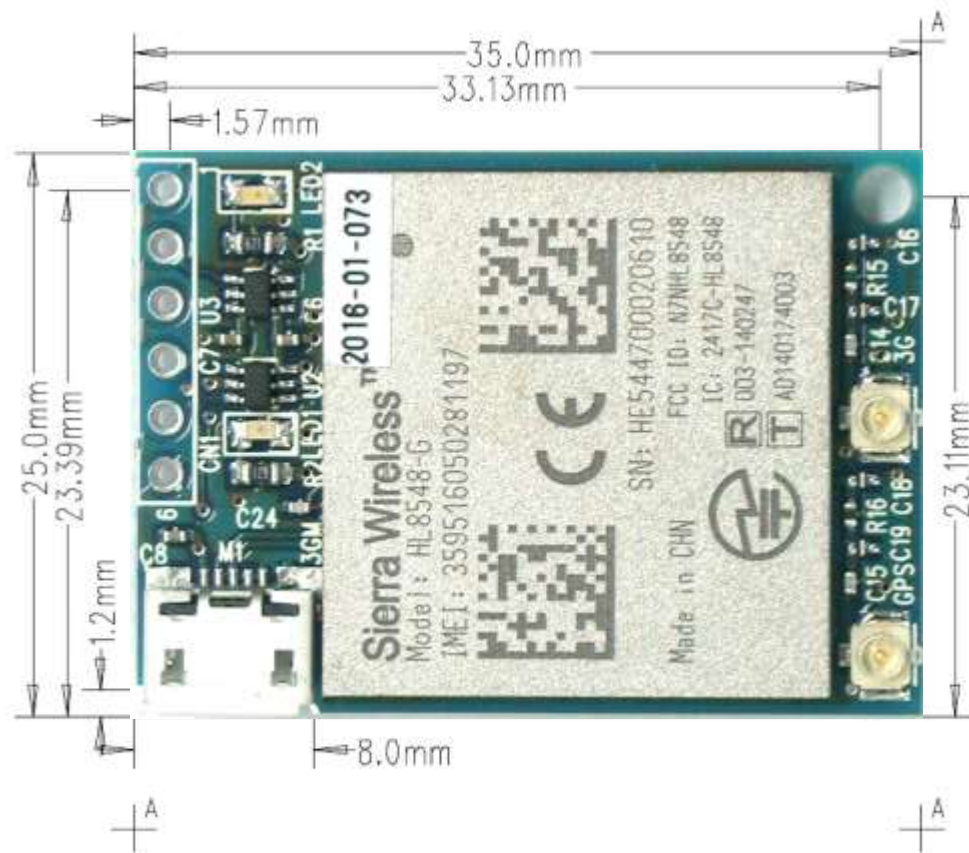
【補足資料 2】 トラブルシューティング

#	課題	現象	問題点・対応策	補足
1	配線・接続	・ UART (Tx:送信、Rx:受信)、電源およびGNDが正しく理解できていない	・ 3 GIMコネクタ部の#1～#6までを正しく理解して上で配線・接続のこと #1 (電源On/Off : 任意)、#2 (RX)、#3 (Tx)、#4 (1.8～5V 電源)、#5 (3.3～4.2V電源)、#6 (GND)	・ #5(VCC)で外部電源を利用する場合には、3.7Vリチウムイオン電池を推奨
2	応答 (レスポンス)	・ コマンドを送っても、返信がない ・ 正しい応答でない	・ 通信モジュールとマイコンボードとの通信、またはマイコンボードとPCとの通信において以下の原因が考えられる ① 3GIMの配線が正しくできていない (配線・接続確認) ② 電源供給に問題がある (電源電圧の確認) ③ UART通信速度の設定が間違っている (確認設定) ④ 初期電源後の待ち時間を考慮不足 (15秒以上待機) ⑤ プログラムに間違いがあることで再確認 ⑥ Arduino IDE シリアルモニタ画面の改行コード変更	・ 応答が正しく表示されない場合の原因は、配線ミスや配線での接触不良が考えられる ・ ②の電源供給で、VCCの3.3～4.2Vを間違えるケースが多発 ・ ⑥の場合、改行選択メニューで「CRおよびLF」を選択のこと
3	エラー頻発	・ #=NGが多発 ・ 立ち上げタイミングの問題 ・ 電源供給 (電流が小さい) 問題	・ 配線・接続が正しくできていること ・ 適正なSIMカードの挿入されていること ・ 正しく電源供給できていること	・ RSSI (電波強度測定) やSIMカードのサービス確認 ・ \$YRや\$YSコマンドで確認
4	電波強度測定の取得	・ 電波強度が取得できない	・ 正しいSIMカードとアンテナ接続によって正しく設定される ・ 正しい電波強度を取得するにはしばらく時間が掛る	・ 同 上
5	SMS送受信	・ SMSの送受信ができない ・ SMSの応答が無い	・ SIMカードが、SMS対応になっていない (切替え必要) ・ SMSサーバとのやり取りでの不備 (何度か読み込み必要)	・ 同 上
6	GPS取得	・ GPS取得ができない ・ GPS取得に時間が掛る	・ GPSアンテナが正しく接続されていること ・ GPS電波状態が良い所 (屋外・PCから離す) で実施のこと ・ 初期立ち上げでは数分から10分ほど掛る場合がある ・ 電源供給が正しくできていること	・ 一度GPS取得でき、電源が入った状態だと、次からは即取得可能
7	ネット接続	・ Webコマンド群やTCP/IPコマンド群が正しく応答しない	・ 3 Gアンテナを正しく接続する ・ 正しいSIMカードが挿入されていない ・ SIMカードの接続不良 (再度再挿入などを実施) ・ 電源供給が正しくできていること	正しいSIMカードとは、\$ PSコマンドを使ってプロファイル設定されたSIMカードであること。Wikiページで情報公開

基本的なことはWikiサイト (<https://3gim.wiki/doku.php>) や、書籍資料等にて掲載していますので、そちらをご覧ください。

\$ コマンドのエラーコード一覧表は「http://tabrain.jp/3GIM_V2.0/3GIM_ERR_LIST.pdf」でご確認いただけます。

【補足資料 3】 3 GIM V2.2 外形寸法



【補足資料 4】 3 GIM サポートサイト

3 GIMに関する技術情報・活用情報が数多く掲載されています。

<https://3gim.wiki/doku.php>

The screenshot shows the 3GIM Wiki website. At the top, there is a header with the 3GIM logo, the text '3GIM Wiki for IoT Innovation', a search bar, and a 'ログイン' (Login) button. Below the header, there is a breadcrumb trail: '現在位置 / トレース / inquiry / start'. The main content area features a 'What's New' section with a list of updates. On the right side, there is a '目次' (Table of Contents) sidebar with links to 'What's New', '4GIM(V1)', '3GIM(V2.2)', '3GIM HAT(V1)', 'IoTAB(V1)', and 'お問合せ'.

本サイトは、特定非営利活動法人（NPO法人）オープンワイヤレスアライアンスが企画・提供する3GIM(スリージム: 3G IoT Module)や4GIM¹⁾の技術情報や活用情報などを広く提供するとともに、利用者向けの技術サポートを提供することを目的としています。

What's New

- **4GIM用のArduinoライブラリを修正&公開**(2018.11.27)
 - [4GIMライブラリ](#)
- **LTE(docomo Xi)を利用できる4GIMを11月中旬から販売開始**(2018.11.12)
 - LTEサービスを利用する3GIM互換の通信モジュール 4GIM(フォージム)の販売を [スイッチサイエンス](#) 様開始します。
 - 4GIM(V1)に関する技術情報は、[4GIM\(4G IoT Module\) Ver1](#) について をご参照ください。
- **IoTAB(V1)を近日中に販売開始予定**(2018.11.12)
 - 3GIM/4GIMと同じサイズの小型マイコンボードIoTAB(V1)をスイッチサイエンス様で販売する予定です。
 - この小型マイコンボードはArduino Zeroと互換ですので、Arduino IDEで開発を行うことができます。
 - IoTAB(V1)に関する技術情報は、[IoTAB \(Ver1\)](#)について をご参照ください。

【補足資料5】 3 GIM関連商品のご紹介

3 GIMをArduino UNO上やRaspberryPi (B+、2 B、3 B、Zero)でも簡単に稼働させることができる3GIMシールドや3 GIM HAT、それに14種類ものセンサ類を持つTABシールドや10種類のIoTABシールドV3.0を使うことで、誰もが、簡単に、短時間で「IoTデバイス」のモノづくりできるプロトタイピング開発環境が揃います。

(センサ値などをメールで送信し、ツイッター連携、クラウド連携が、容易に学べます)

3 GIMシールドは、5V系 Arduino上のUNOやMEGA上で簡単に3 GIMを利用することができます。

3 GIMシールドは、電源電圧3.7Vを3 GIMに安定供給し、安心して3 GIMを稼働させることができます。



3 GIM HAT 1.0

3 GIM HATは、Raspberry Pi の多くの種類で稼働する3 GIM専用の拡張ボードで、USBケーブルによるインターネット接続と、UART接続による\$コマンド通信ができるものです。

Raspberry Piを野外のゲートウェイとして利用もでき、アナログセンサ対応ボードとしても利用できます。



TAB SHIELD 1.1

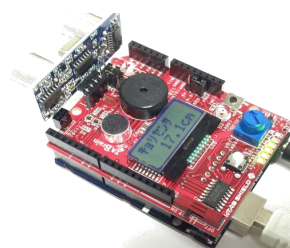
TABシールドは、多くのセンサ類およびLCD・LED・スピーカなどの14種類もの電子部品を搭載した試作用キットとなります。

Arduino上で、センサ類やLCD・LED等を使った処理を検討する場合、半田付けやジャンパーケーブルの接続なしに、プログラミングだけで動かすことができます。



IoTAB SHIELD 3.0

IoTABシールドV3.0は、これまでのセンサ拡張ボードTABシールドに、3 GIMが取り付けられる仕様となりました。このことで、IoTデバイスの試作に取り掛かることができ、そのままセンサネットワークとしても利用できるようになります。



まとめ

1. 3 GIMが簡単なのはATコマンドを使わず、タイミングを配慮することがなく、インターネット接続・通信できること
2. 3 GIMのインターネット接続・通信は、3種類あり
 - ① 超簡単：Arduino用のライブラリで制御する場合
 - ② 簡単：UARTの\$コマンドで制御する場合（Arduino以外でも利用可能）
 - ③ 難解：ATコマンドで直接制御する場合
3. メール送信やツイッター送信の基本を知る事で、ネット活用が簡単になる
4. ネット上の情報・知識を活用することが重要
5. IoTとは、モノのインターネットと呼ばれ、インターネット接続・通信を経て、スマートデバイスとの連携や、自動連携で意味を成す。
6. 今後、センサ・ネットワークと呼ばれるセンサ値を蓄積し（ビッグデータ）、その内容を分析し、活用する人工知能（AI）化で、価値を出すことが夢見られている。
7. 最終的には、多くの現場・現場の対応によってその知恵を見出す。（現場力・実践力が重要）