高度ITを活用したビジネス創造プログラム
講師マニュアル

－IoT活用講座　Vol.2－

プログラム概要

高度IT 技術を活用したビジネス創造プログラムの概要

1.プログラム開発の目的と背景

■プログラムの目的
第4 次産業革命において必須であるIoT、AI やビッグデータに代表されるIT 系の技術を駆使し、新たな発想（サービス企画・デザイン思考）でビジネスを創造できる高度IT エンジニアを育成する。
■修得すべき能力とその理由
修得すべき能力を図で表すと下記のようなイメージになる。今回は一定のスキル（レベル2～3程
度）を修得しているエンジニアを受講者に想定しているので、学習対象範囲の各能力についての教育訓練プログラムを作成する。

（図1）

1. 分野横断的な発想力、分析力、企画力を備えたビジネス創造力
2. デザイン思考力
3. アジャイル開発力
4. AI
5. IoT
6. ビッグデータ
（統計解析・分散処理）
7. 仮想化
8. セキュリティ

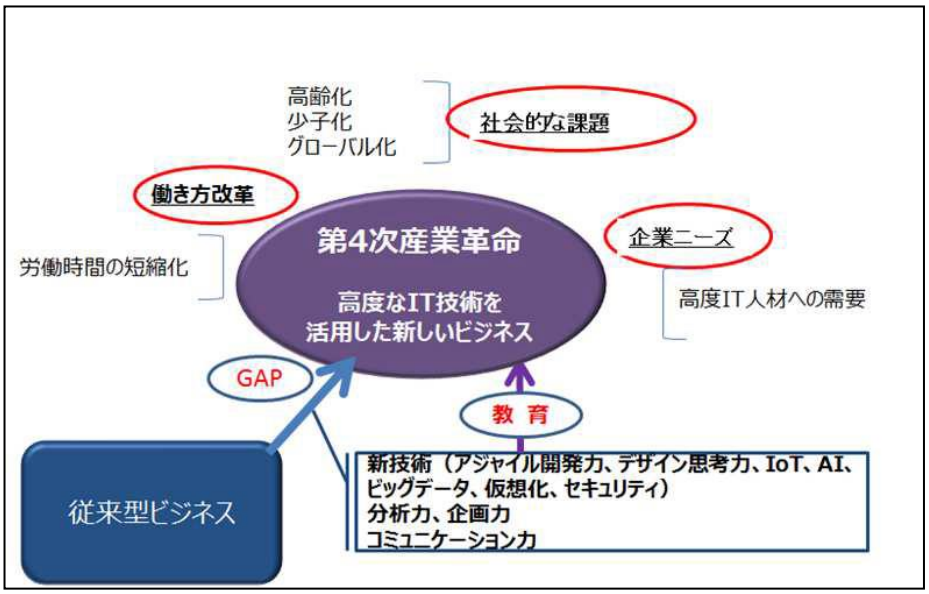
2.プログラム開発の背景

1)「第4 次産業革命」における必要性

現代社会において高齢化・少子化・グローバル化等の社会的な課題が山積する中で、それらの社会的な課題を解決することが求められている。それらの課題の解決には、新しい発想のビジネスが求められていて、それを実現するためには、高度なIT 技術の活用が不可欠である。企業も社会的な課題を解決するための新たなビジネスチャンスを求めてはいるが、現状はほとんどの技術者が従来型ビジネスに対応した人材であり、高度なIT 技術をもった人材へのニーズがある。

また、「働き方改革」という労働時間の短縮化という中で、労働時間が削減されても経済成長を促すには、単位時間あたりの労働生産性の向上が欠かせない。労働生産性の向上には、より高度なIT 技術の修得が不可欠である。

このように、「社会的必要性」「企業ニーズ」「働き方改革」という3 つの要因で上記に挙げた8つの能力が必要である。上述の内容を表したのが図2 である。

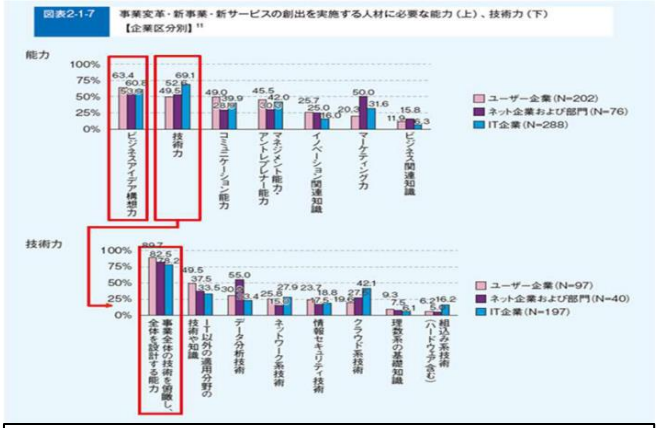


(図2)

2.プログラム開発の背景

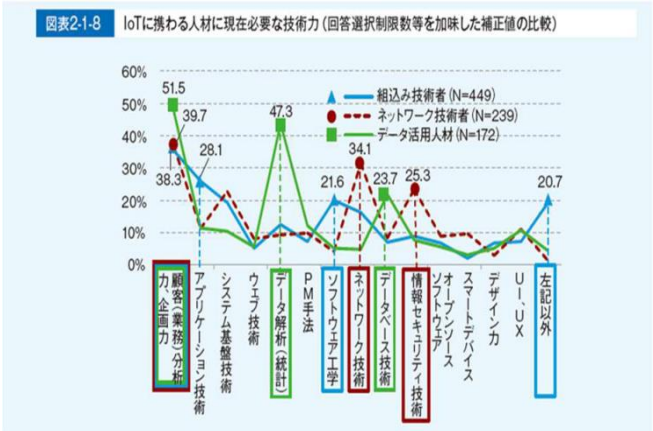
2)8 つの能力の根拠

図2 のように、企業が新しいビジネスを創造するためには、従来型ビジネスとのGAP を埋めるための能力が必要であり、IPA の「IT 人材白書2016」(資料1)によると「ビジネスアイデア構想力」「技術力」が求められる



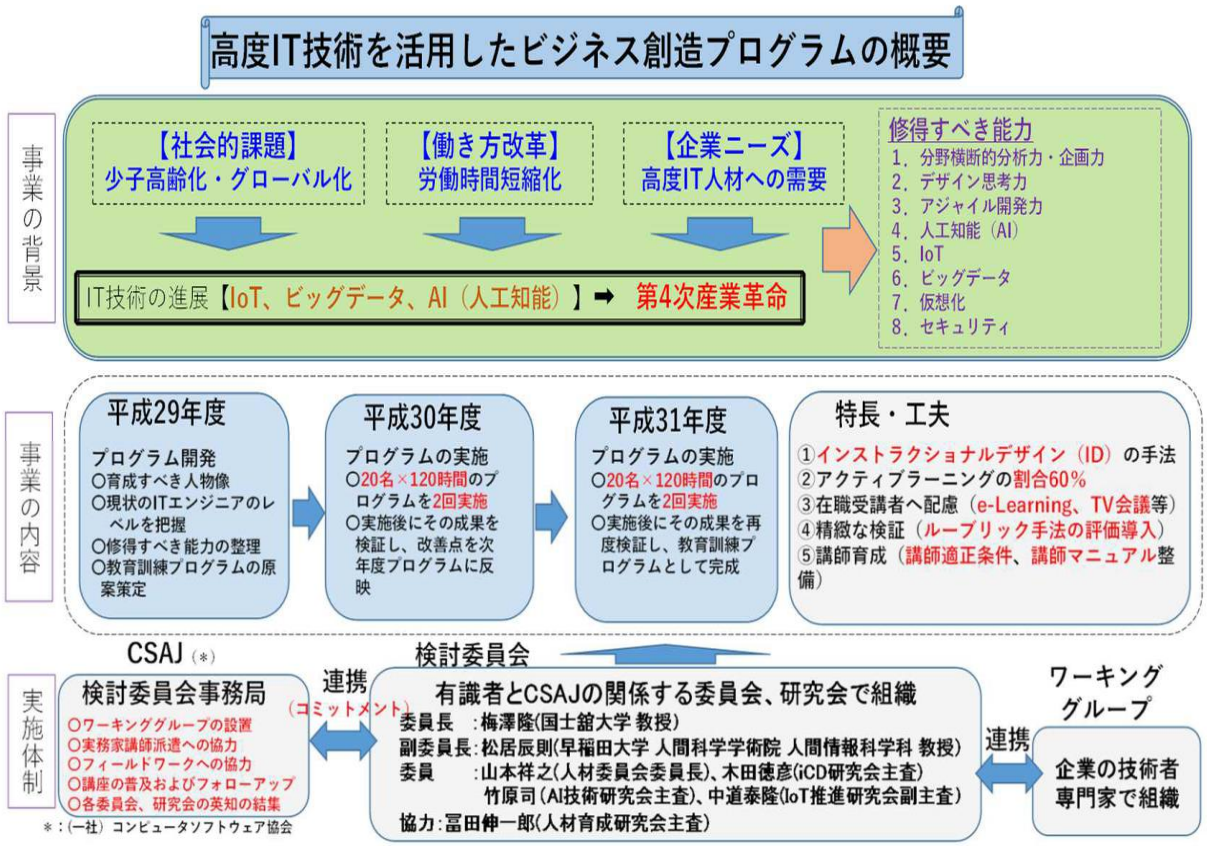
(資料1)
『新事業・新サービス創出に必要な能力・技術力とは？』
(IPA「IT 人材白書2016」)

「ビジネスアイデア構想力」「技術力」とは何かを詳細に見てみると、例えば「IoT に関わる人材に必要な能力は」というIPA のアンケート調査「IT 人材白書2016」（資料2）では「顧客分析・企画力」「データ解析」「ソフトウェア工学」「ネットワーク技術」「データベース技術」「情報セキュリティ」があげられる。



(資料2)
『IoT 人材に必要な技術力とは？』
（IPA「IT 人材白書2016」）

3.実施組織


高度IT技術を活用したビジネス創造プログラムの概要
事業の背景
【社会的課題】
少子高齢化・グローバル化
【働き方改革】
労働時間短縮化
【企業ニーズ】
高度IT人材への需要
IT技術の進展【IoT、ビッグデータ、AI（人工知能）】➡ 第4次産業革命
修得すべき能力
1．分野横断的分析力・企画力
2．デザイン思考力
3．アジャイル開発力
4．人工知能（AI）
5．IoT
6．ビッグデータ
7．仮想化
8．セキュリティ
事業の内容
平成29年度
プログラム開発
○育成すべき人物像
○現状のITエンジニアのレベルを把握
○修得すべき能力の整理
○教育訓練プログラムの原案策定
平成30年度
プログラムの実施
○20名×120時間のプログラムを2回実施
○実施後にその成果を検証し、改善点を次年度プログラムに反映
平成31年度
プログラムの実施
○20名×120時間のプログラムを2回実施
○実施後にその成果を再度検証し、教育訓練プログラムとして完成
特長・工夫
①インストラクショナルデザイン（ID）の手法
②アクティブラーニングの割合60％
③在職受講者へ配慮（e-Learning、TV会議等）
④精緻な検証（ルーブリック手法の評価導入）
⑤講師育成（講師適正条件、講師マニュアル整備）
実施体制
CSAJ（＊）
検討委員会事務局
○ワーキンググループの設置
○実務家講師派遣への協力
○フィールドワークへの協力
○講座の普及およびフォローアップ
○各委員会、研究会の英知の結集
＊：（一社）コンピュータソフトウェア協会
連携
（コミットメント）
有識者とCSAJの関係する委員会、研究会で組織
委員長　：梅澤隆(国士舘大学 教授)
副委員長：松居辰則(早稲田大学 人間科学学術院 人間情報科学科 教授)
委員　　：山本祥之(人材委員会委員長)、木田徳彦(iCD研究会主査)
竹原司(AI技術研究会主査)、中道泰隆(IoT推進研究会副主査)
協力：冨田伸一郎(人材育成研究会主査)
連携
ワーキング
グループ
企業の技術者
専門家で組織

4.高度IT 技術を活用したビジネス創造プログラムの構成

■各講座の時間割

コース名	講義	演習	小計
1．オリエンテーション	2:00	-	2:00
2．デザイン思考講座	4:00	6:00	10:00
3．仮想化講座	4:00	4:00	8:00
4．ビッグデータ講座	7:10	7:50	15:00
5．AI基礎講座	8:35	7:25	16:00
6．IoT活用講座	8:50	7:10	16:00
7．セキュリティ講座	6:30	4:30	11:00
8．アジャイル講座	5:40	6:20	12:00
9．顧客分析・企画力養成講座	4:50	13:10	18:00
10．フィールドワーク（セキュリティ・アジャイル開発・AI）	-	12:00	12:00

■フィールド・ワーク

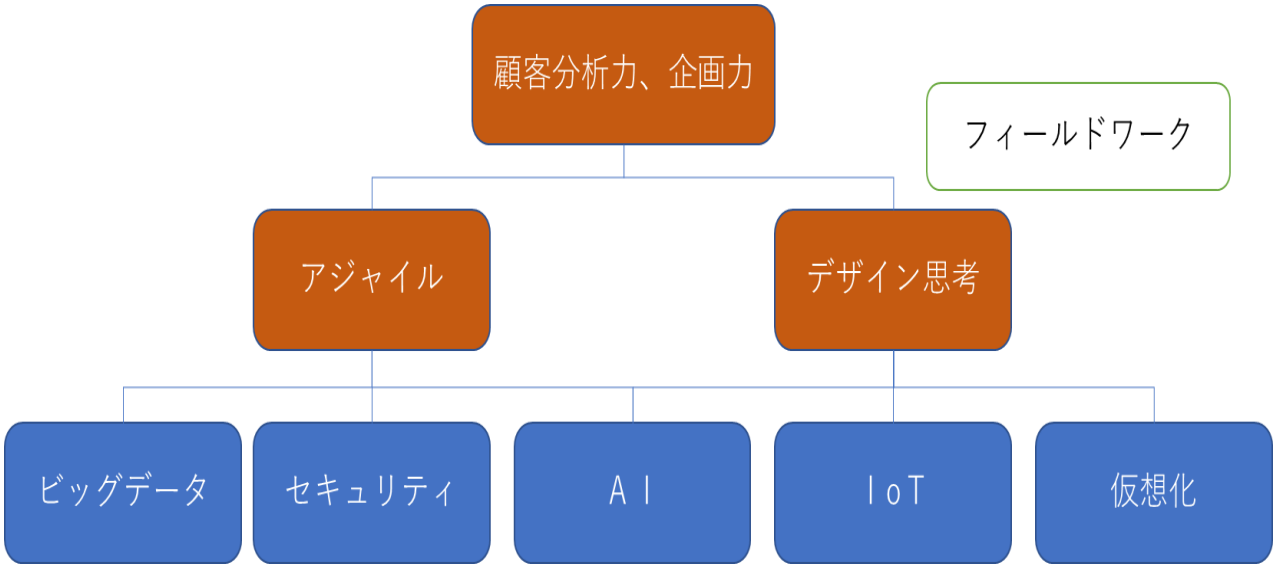
	関連講座	訪問先企業（予定）	学習概要
1	セキュリティ	株式会社ラック様	場所：本社ビル　永田町 内容：1.ラック様のセキュリティ業務の実際 2.JSOCS見学 3.体験ゲーム
2	アジャイル開発	KDDI株式会社様	場所：新宿 内容：1.開発現場の見学 2.KDDI様のアジャイルへの取り組み 3.体感ゲーム
3	Ai基礎	日本電気株式会社様	場所：三田 内容：1. NEC Future Creation Hub 見学 2:AI企画の概要と演習説明 3:【演習】データ・バリューチェーンの作成 4:まとめ

5.高度IT 技術を活用したビジネス創造プログラムの構成

講座の関係図

・赤枠：思考法・発想法

・青枠：IT技術知識・スキル



6. 教育訓練プログラム受講修了及び評価

■受講修了条件
●終了時間：120 時間
●終了時の能力像：「第4 次産業革命において、IT 系の必須技術を駆使し、新たな発想を持ってビジネスを創造できる知識・スキル」

【各講座の時間数】

No	講座名（モジュール名）	時間	No	講座名（モジュール名）	時間
1	オリエンテーション	2	6	IoT活用講座	16
2	デザイン思考講座	10	7	セキュリティ講座	11
3	仮想化講座	8	8	アジャイル講座	12
4	ビッグデータ講座	15	9	顧客分析・企画力養成講座	18
5	AI基礎講座	16	10	フィールドワーク	12
				総合計	120

■評価指標と基準点
下記の指標に対する基準点をすべてクリアする。

	実施タイミング	指標	合格基準
1	ー	出席率	80%以上
2	各講座	e-learning テスト	100%
3	各講座	理解度確認テスト	80%
4	各講座	ルーブリック仕様のアンケート	～ができる（下記に例示）
5	各講座	成果物評価	講師による評価：講師が正常稼働及び理解度を前提に総合的な判定を行う

ルーブリック仕様のアンケート（例）

自己評価		1	2（合格基準）	3
理解度	内容を理解している	メソッドのメリット・デメリットを理解している。	メリット・デメリットをメンバーに説明できる。	メリット・デメリットをについて自分の考えを説明できる
応用力	現場で活用できる	状況に応じてメリット・デメリットを説明できる。	情況にあったメソッドを選択できる。	情況にあったメソッドで課題解決に取り組める。

7.IoT 活用講座概要

ねらい	データ収集技術からデータ統合、クラウドサービスに至る最新の技法を学び、スキル強化を図る。				
開催日程	16 時間（ e-Learning　2 時間含む）				
受講条件	IT 技術者としての経験が 3 年以上、ICT の基礎知識を持っていること				
学習目標	本講座では、IoT を活用したビジネスに自社実践して活用できることを想定し、センサーモジュール（温度センサ、衝撃センサ等）や各種モジュール（LCD、モータ等）を利用・制御する方法、ネットワーク通信を実現する方法、IoT と連動するクラウドサービスなどの IoT の要素技術について総合的な開発実習を行う。				
	時間	講義	演習	学習概要	学習詳細
カリキュラム概要	1:30	1:00	0:30	IoT の概要	・Internet of Things (IoT) ・ネットワークとデータが創造する新たな価値 ・シンプルな IoT デバイスの例 ・Society 5.0 ・経済発展と社会的課題の解決を両立 ・第四次産業革命 ・第四次産業革命の世界的な潮流 ・データの利活用の進展とプロセス・プロダクトにおける進展の対応 ・業種ごとのプロセスの IoT 導入事例 ・業種ごとのプロダクトの IoT 導入事例 ・IoT における製品の高付加価値化の事例 ・IoT による新規事業・サービスの創出事例 ・IoT 関連技術に対する 1 社当たりの平均投資額（米国） ・米国大手企業における IoT/AI 関連の取り組みとしてメディアなど公開情報で取り上げられた主な事例の技術導入シェア ・IoT 活用分野 ・国内の IoT 市場規模の推移と予測 ・IoT の導入にあたっての課題 ・企業が AI・IoT の利活用を進める上での課題 ・日本企業におけるプロセス IoT 化率 ・日本企業におけるプロダクト IoT 化率 ・プロセス IoT 化を考えていない理由 ・プロダクト IoT 化を考えていない理由

					・IoT 化を考えていない理由の比較 ・IoT 推進コンソーシアム ・スマート IoT 推進フォーラム ・その他の委員会・コミュニティなど ・演習：IoT の事業モデル設計
	0:45	0:45	0:00	IoT に関連する主な通信技術	・主な予備知識 ・電波の種類 ・免許不要の無線局 ・Bluetooth・ZigBee・Wi-Fi ・ネットワークトポロジ ・Bluetooth Low Energy（BLE） ・Low Power Wide Area（LPWA） ・LPWA の特徴 ・主な LPWA 規格の位置付け ・LPWA の活用事例（日本） ・LPWA の活用事例（海外） ・IoT システムの主なプロトコル ・HTTP ・MQTT ・CoAP ・WebSocket
	0:45	0:45	0:00	電気回路の基礎	・電気回路に関する用語 ・電気の流れ ・LED の仕様 ・電圧と電流と電力の単位 ・感電した場合の危険度の目安 ・抵抗の色の読み方 ・電圧～電流～抵抗 ・オームの法則 ・並列接続における和分の積
	2:40	1:00	1:40	組込ボードの基礎	・IoT でよく使用される組込ボード ・Arduino とは ・Arduino のダウンロードとインストール ・Arduino のメニュー画面 ・Arduino のスケッチ例と動作検証 ・Arduino とブレッドボードによる配線

					・ブレッドボードの通電箇所 ・Arduino における回路設計 ・Arduino におけるオームの法則 ・演習：Arduino を使った電気回路の設計 ・センサ ・環境センサ ・入力モジュール ・出力モジュール ・演習：Arduino を使った電気回路の設計
	2:20	0:40	1:40	組込ボードとセンサ	・センサ ・環境センサ・ ・入力モジュール ・出力モジュール ・演習：Arduino とセンサを使った回路設計
	1:00	1:00	0:00	IoT のセキュリティ	・IoT デバイスを標的としたマルウェア ・Mirai ウィルス ・IoT セキュリティガイドライン ・IoT セキュリティガイドラインの目的 ・サービス提供者のための指針 ・一般利用者のための指針
	5:00	1:40	3:20	IoT プラットフォームを使ったデータ通信	・IoT プラットフォームの例 ・IoT プラットフォーム sakura.io ・sakura.io の特徴 ・さくらの LTE 通信モジュール ・さくらの通信モジュールの位置付け ・sakura.io の物理的構成 ・IoT システムの物理的構成 ・sakura.io 料金と通信ポイント ・ポイント管理例 ・ライブラリとマニュアル ・基本的な考え方 ・コード例 ・連携サービス ・WebSocket ・データ形式 ・JSON 例（データが単数）

					・JSON 例（データが複数） ・連携サービスの作成 ・WebSocket の URL と Token ・JSON 例（データが単数） ・JSON 例（データが複数） ・開発ツール Node-RED ・演習：さくら LTE モジュールの回路設計と利活用 演習：総合演習
合計時間	14:00	6:50	7:10		

8.IoT 活用講座詳細カリキュラム

時間	学習項目	学習項目の狙い	詳細内容
0:15	オリエンテーション	目的： この研修における目標を明確にし、研修への意欲を高める	**[講義]** **①オリエンテーション** 1.講師自己紹介 2.コースの目的（IoT のコアとなる知識と技術を習得する。プロトタイプを自作して自社で検証できる PL になってもらいたい） 3.注意点 ・駆け足で進む ・すべての技術背景をこの研修で取り扱うのは不可能 ・研修ではコア技術を集中的に実施 ・試験が研修と e-learning で 10 問ずつ 4.配布資料の確認 **②実機演習セットの確認** 1.各グループ実機演習セット内容を確認する。
			[演習] 1.聴講者同士で簡単な自己紹介（特に、講座を受講する理由） 聴講者間で情報共有し気付きを促す。
0:45	第 1 章 IoT の概要	目的： IoT の概念や事例について学ぶ ゴール： IoT の全体像を理解する。IoT のアイディアを身の回りの事象へ適用して企画の立案ができる。	**[講義]** **①IoT の概要と背景** 1.IoT が仮想世界と現実世界の橋渡しを担っている点を強調する。 2.第四次産業革命の潮流について確認してもらう。 【口頭質問】働いていて第四次産業革命というキーワードを見かけますか？（学習者が肌で感じているかどうかを確認） 3.IoT の事例と課題を紹介する。（e-learning で見てきている部分は軽く流す）

時間	学習項目	学習項目の狙い	詳細内容
			【口頭指示】後の演習で各自の身の回りの事例で IoT の導入案と課題を検討してもらうので、そのつもりで事例を見ておいてください。 4.IoT 推進コンソーシアムとスマート IoT 推進フォーラムにつちえ紹介する。
0:30			**[演習]** 1.各自の各自の身の回りの事例で IoT の導入案と課題を考えて用紙のまとめてもらう。 2.（時間があれば）各自で簡単に発表してもらって聴講者間で情報共有し気付きを促す。
0:45	第 2 章 IoT に関連する主な通信技術	目的： IoT に関連する通信技術について学ぶ ゴール： IoT に利用されているデータ通信技術について説明ができる。	**[講義]** **①e-learning 部分の確認** 1.今回の e-learning 部分は重要なので初めに簡単におさらいする。 2.注意点 ・後で利用するさくら IoT の通信技術について口頭で説明し確認してもらう。
0:45	第 3 章 電気回路の基礎	目的： 電気回路の設計の基礎について学ぶ ゴール： 簡単な電気回路の法則について説明ができる。オームの法則を使って抵抗の計算ができる。	**[講義]** **①e-learning 部分の確認** 1.今回の e-learning 部分は重要なので初めに簡単におさらいする。 2.注意点 ・感電の注意のスライドにて、回路をショートさせないことや、ぬれた手で扱わないなど、基本的な注意点も付け加える。 ・オームの法則の簡単な問題を解いて、抵抗の計算方法について確認する。
1:00	第 4 章 組込ボードの基礎	目的： 組込ボードとセンサの基	**[講義]** **①部品の取り扱いに潰え**

時間	学習項目	学習項目の狙い	詳細内容
		本的な取り扱い方（特に組込ボードを中心）を学ぶ。 ゴール： ・組込ボードの１つである Arduino を IDE から利用できる。 ・ブレットボードを使って基本的な配線ができる。 ・オームの法則を使って適切な抵抗を選ぶことができる。	1.配布資料の確認 配布物を確認しながら取り扱い方法について確認する。 2．注意点 ・後片付け考えて整理整頓を心がけてもらう。 ・可能であればタッパーなどの箱を用意し、利用する部品をそこに入れてまとめておくように促す。 **②Arduino の基本的な使い方** 1.（未実施の場合）Arduno をダウンロードしてインストールする。 2.Arduino の概要とメニュー画面、メニュー項目、基本的な使い方などを説明する。 3.動作検証用のプログラム（スケッチ）を書いて、Arduino に書き込んで Arduino が正しく動作（ボードの LED がプログラムに合わせて点滅）するかどうかを確認する。 4.注意点 ・USB での電源供給は不安定な点を注意しておく。→場合によっては外部電源を接続した方が動作が安定するケースがあることを紹介する。 ・［シリアルポート］の選択［COM 番号］に注意する。 ・インストール時にデバイスのインストールが求められた場合「はい」を選んでインストールしておかないと上手く動作しない点に注意する。 ・部品や USB ケーブルなどが物理的に動作しないことがあるので、それについて注意を促すと共に、予備の部品を用意しておく。 **③電気回路の設計** 1.ブレッドボードについて説明する。特に、通電箇所について説明する。 2.注意点 ・複雑な回路を作るときはブレッドボードの上側と下側にある＋と－

時間	学習項目	学習項目の狙い	詳細内容
			の回路をジャンパーピンで接続すると電源や GND の配線がやりやすくなる。 ・ブレッドボードは刺さっているようでもしっかりと刺さっていないことが多いので注意を促す。 ・必要であればテスターでチェックする。 ・場合によってはブレッドボードにハンダ付けしても良い点をアドバイスする。 ・e-learning で予習してもらったスライドのように感電すると危険な点を再度注意する。特に、電子部品はすぐに壊れるので注意を促す。 3.ケースバイケース ・LED の個数に余裕があれば、わざと 1 つ過電圧で壊しても良いかもしれない。←5V と GND に直接つなげばほとんどの LED は壊れる。 以上の 2.3.の注意から次のオームの法則の学習につなげる。 4.オームの法則について説明する。（電圧～電流～抵抗については e-learning で閲覧済み） 5.注意点 ・オームの法則の組み合わせで複雑な回路でも抵抗値を求めることができることを言及しておく。 ・スライドのオームの法則の問題の数値を変更して問題を出しても可
1:00			**[演習]** できるだけ学習者自身で問題を解決するように考えてもらう→試行錯誤が講習後に自分でやるときの糧になる。 **①LED が点灯する回路** ・LED の＋と－を間違えないように注意する。 ・ジャンパーピンを刺すブレッドボードの穴の列を間違えないように注意する。

時間	学習項目	学習項目の狙い	詳細内容
			②スイッチで LED を ON･OFF する回路 スイッチに正しい向きがあるので注意する。→電子部品を取り扱うときの全般の注意に促す。 **③スイッチを押したときに LED を ON する回路とプログラム** ・スケッチ（プログラム）がでてきたのでスケッチの流れや関数について説明する。 ・この回路は「プルダウン」という方式になっていることを説明する。 注意 ・「プルダウン」や「プルアップ」の簡単な説明が演習資料に載っているが、その場ですぐに理解することは難しいだろう. ・センサーの動作が安定しない場合、これらの利用を試してみることをすすめること。 **④アナログ出力による LED 点灯** ・学習者の進み具合をみてスケッチについて説明する。 ・LED の種類と抵抗の種類によって LED の点灯の様子が変わる点に注意しておくこと。 ・相性によっては LED の微妙な点灯の変化は確認できないかもしれない。
0:40			**［演習］** **⑤応用編：LED の種類や個数を変更** ・各自のレベルに合わせて、A)、B)、C）の 3 つから課題を 1 つ選んで取り組ませる。 ・大事なのは自分で回路を配線して Arduino から動かすことができるかどうか。適切な抵抗を選ぶことが出来るかどうか。 ・作成した回路の回路図を描き抵抗値を書き込ませる。 ・時間が余った学習者には 3 つの回路に挑戦させてもよい。
0:40	第 5 章 組込ボードとセンサ	目的： 組込ボードとセンサの基本的な取り扱い方（特にセンサを中心）を学	**[講義]** **①色々なセンサについて紹介する** 1.配布したセンサを種類別に分類しながら確認させる。 **②マニュアルなどを確認する**

時間	学習項目	学習項目の狙い	詳細内容
		ぶ。 ゴール： ・組込ボードの１つである Arduino と使った色々なセンサからデータを取得できる。 ・ブレットボードを使ってセンサごとに配線ができる。 ・センサに合わせたスケッチ（プログラム）を記述することができる。	1.OSOYOO の公式サイトや Kuman の公式マニュアルを例に出しながらセンサについて確認していく。 2.ポイント ・ただし、公式サイトや公式マニュアルにはすべてが書いてない点を確認させる。 ・特に、外国製品ではマニュアルが十分ではなく、日本語表記や英語表記がおかしい点や、場合によっては適切な情報が欠けていたりする点を確認する。 →自分で情報を集める必要がある。 3.配布資料の確認 4.注意 ・演習の時間を十分確保したいので、30 分未満でも問題なし。むしろ余った時間を演習に割り当てたい。
0:40			**[演習]** **①光センサの利用** 1.環境センサの例として光センサを利用してみる 2.新しく出てきたスケッチの関数を説明する。 **②光センサによる LED 点灯** 1.回路作成 ・回路が複雑になってくるので学習者が配線する時間を十分に確保する。 2.プログラミング スケッチ内のインプットとアウトプットのピン番号を確認させ、関数の引数として与えてある点を確認させる。 **③傾斜スイッチの利用** 1.入力モジュールの例として傾斜スイッチを利用してみる。 2.ポイント

時間	学習項目	学習項目の狙い	詳細内容
			・物理的な現象が数値化されている点を確認させる。 ・センサにも誤差があるので、それを体感で確認させる。 **④ロータリーエンコーダによる LED 点灯** 1.回路作成とプログラミング ・新しく出てきたスケッチの関数を説明する。 2.注意 ・OSOYOO のロータリーエンコーダの方が配線が楽なので、こちらを使う点を注意させる。 **⑤LCD 出力** 1.回路作成とプログラミング 出力モジュールの例として LCD を利用してみる。 2.ライブラリのダウンロード ライブラリのダウンロードがあるので講師と一緒に動作させる. 3.注意 ライブラリのインストールは頻繁にあるので、新しいセンサやモジュールを利用する際にはライブラリの有無を調べるように注意を促す。 4.ポイント ・LCD を利用する場合には、LCD のレジスタ（I2C アドレス）を指定する必要がある. ・I2C は I2C のことで、アイ・スクエアー・シー、アイ・スクエアード、シー、アイ・ツーシーなどと呼ばれる。
1:00			**［演習］** **⑥応用編：各自で色々なセンサやモジュールを組合せて利用** 1.回路作成 ・学習者に自由に発想させて回路を組み立てさせる。 ・時間を十分に確保する。 2.成果物報告

時間	学習項目	学習項目の狙い	詳細内容
			・最後に成果物を学習者に発表させて（もしくは皆で巡回して）情報共有して学習者に気付きを促す。
1:00	第 7 章 IoT のセキュリティ	目的： IoT のセキュリティに関して学習する。 ゴール： ・IoT のセキュリティに関して一般的な知識を知り、その知識を活用できる。	**[講義]** **①IoT のセキュリティについて** 1.IoT デバイスを標的としたマルウェアについての現状を説明する。 2.ポイント Mirai ウィルスについて紹介する。 3.セキュリティガイドラインとその目的をきちんと理解し、サービス提供者として一般利用者のための指針を知る。
0:30	第 8 章 IoT プラットフォームを使ったデータ通信	目的： IoT プラットフォームの利活用について学習する。 ゴール： ・さくら IoT プラットフォームを利用できる。 ・IoT からクラウドにデータをアップロードできる。	**[講義]** **①IoT プラットフォームの確認** 1.さくらの IoT プラットフォームはもとより、SORACOM や IIJ IoT サービスなどの他のサービスも例に出しながら IoT プラットフォームについて紹介する。 2.ポイント IoT サービスは萌芽期にあたり、頻繁にサービスが一新されているため、IoT のサービスプランは各 Web サイトから紹介する。 3.本研修で利用するさくら IoT のサービスについて詳細を紹介する。 4.注意 ・AWS などのとの連携はプラットフォーム側でもまだ確定していないので、本研修では紹介程度にとどめておく。 ・さくら IoT プラットフォームの料金体系については特にしっかり説明しておく。
0:30			**[演習]** **①さくらの IoT コントロールパネルで確認** 1.講師のナビゲートと一緒にコントロールパネルを操作する。

時間	学習項目	学習項目の狙い	詳細内容
			2.ユーザ ID とパスワードの配布 3.注意点 ・さくら LTE モジュールを Arduino につけると Pin ラベルが見えなくなるので、シールなどを使ってラベルを手書きで作成するとよい点を紹介する。 ・さくらの IoT コントロールパネルは IE を推奨している点を注意を促す。 ・場合によっては AC アダプタなどで Arduino に電源供給が必要かもしれない。→USB ケーブルは不安定なため業者は嫌がるケースもある. ・演習時にデータのアップロード間隔をあまり狭くしないように注意を促す。→料金がほぼ青天井。
0:30			**[講義]** **②WebSocket を JavaScript で取得して表示** 1.WebSocket の原理を説明し、JavaScript で扱う方法について説明する。 2.注意点 ・プログラミングの経験が乏しい者に対しては、コードをあらかじめ準備しておくなど必要な措置をとっておく。
0:40			**[演習]** **②WebSocket を JavaScript で取得して表示** 1. WebSocket と JavaScript を用いて、センサ情報の取得をプログラムとして実装する。 2. 注意点 ・余裕があればグ jQuery と Chart.js を紹介し、JavaScript でデータを可視化する方法について紹介する。（しかし、そろらく時間がタイトなので難しい場合は口頭の言及のみでよい）
0:30			**[講義]** **③Node-RED を使ったデータ通信** 1. 開発ツールの 1 つである Node-RED について説明する。 2. 講師用機材に LED を接続し、Incoming Webhook によっ

時間	学習項目	学習項目の狙い	詳細内容
			てインターネット上から LED の ON/OFF を行うデモンストレーションを見せる。
0:40			**[演習]** **③Node-RED を使ったデータ通信** 1.（未実施の場合）Node-RED をコマンド操作からインストールする。 2.Node-RED の概要とメニュー画面、メニュー項目、基本的な使い方などを説明する。 3.Node-RED のノードを設置して、データ送受信が正しく動作するかどうかを確認する。 4.注意点 ・Node-RED にはグラフ化や API などの様々な機能があるので、時間に余裕があれば紹介する。
1:30			**[演習]** 1.回路作成 ・学習者に自由に発想させて回路を組み立てさせる。 ・時間を十分に確保する。 ・最初の講義でやった現実世界と仮想世界の橋渡しとなるような IoT サービスを企画させて作成させる。 2.成果物報告 ・最後に成果物を学習者に発表させて情報共有して学習者に気付きを促す。
0:10			**[講義]** **①振り返り** 講義全体を振り返って学習した内容を確認する。

IoT活用

目次（1）

1-3章はE-Learning

目次（2）

目次（3）

目次（4）

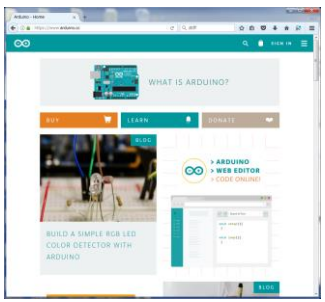
第４章 組込ボードの基礎

目的:
組込ボードとセンサの基本的な取り扱い方（特に組込ボードを中心）を学ぶ。

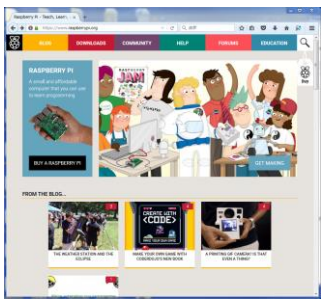
ゴール:
・組込ボードの1つであるArduinoをIDEから利用できる。
・ブレットボードを使って基本的な配線ができる。

4-1. IoTで使用される組込ボードの例

Arduino



Raspberry Pi



▼説明の流れ

組み込みボードの例を説明する。ここでは、Arduino、RasberryPiについて説明する。

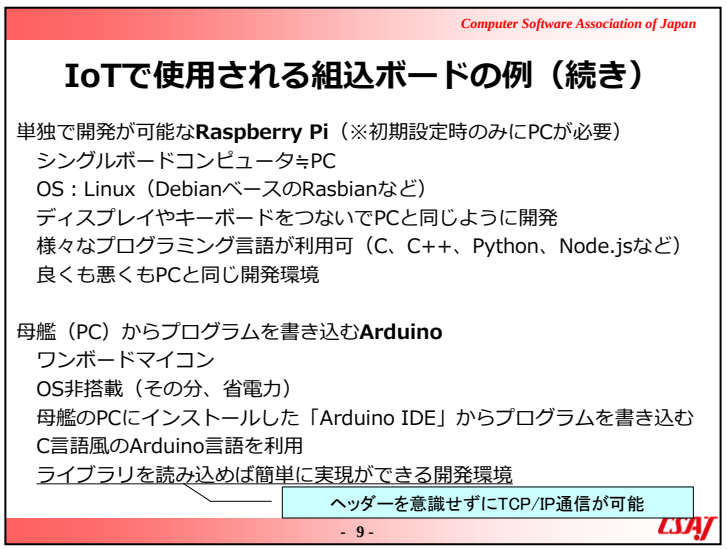
IoTで使用される組込ボードの例（続き）

STM32

IchigoJam

▼説明の流れ

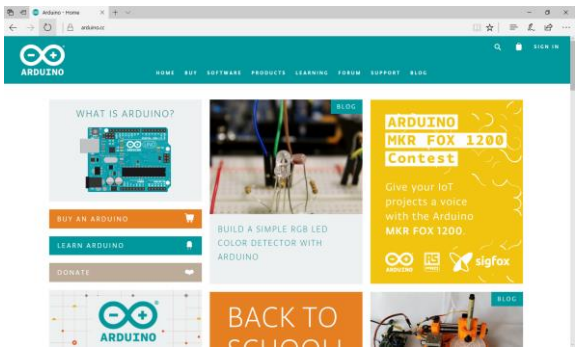
組み込みボードの例を説明する。(続き)ここでは、STM32、IchigoJamについて説明する。



▼説明の流れ

Raspberry Piについての説明を行なった後に、Arduinoの紹介に入る。

4-2. Arduino



（https://www.arduino.cc/）

▼説明の流れ

アルドゥイーノに関して概要を説明する。

Arduino（続き）

アルドゥイーノ

2005年に、Massimo Banzi・David Mellis（当時はイタリアのIDIIIの学生）、David Cuartiellesによってプロジェクトスタート、後にTom Igoeが加わる

- 派生元
  - 2003年　IDII修士論文プロジェクトWiring（Hernando Barragán）
  - 「Arduinoの語られざる歴史」より https://arduinohistory.github.io/ja.html

一枚のプリント基盤の上に、電子部品と入出力がついたマイクロコンピュータ

- Processingベースの開発環境（Javaアプリケーション）
- プログラミング言語：C++風言語（Arudino言語とも呼ばれる、元はWiring）

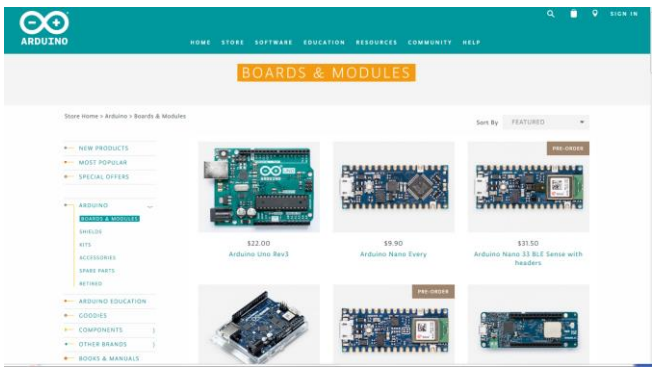
▼説明の流れ

アルドゥイーノに関して概要を説明する。（続き）

▼補足説明

Interaction Design Institute Ivrea (IDII)：　Ivreaインタラクションデザイン研究所

Arduino（続き）



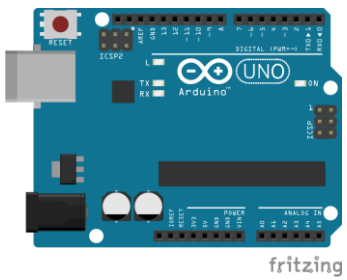
https://store.arduino.cc/usa/arduino/boards-modules

▼説明の流れ

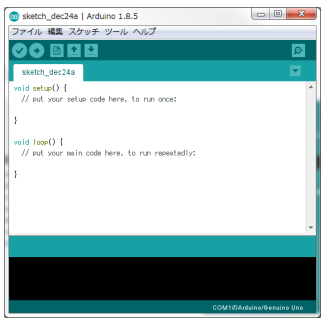
アルドゥイーノに関して概要を説明する。（続き）

Arduino（続き）

Arduinoのプリント基板



Arduinoの開発環境（IDE）



「開発者は「Arduino」という名称が商標の普通名称化となることを避けたいと考えており、許諾無く派生製品にArduinoを使うことを禁じている」とのことを紹介。したがって，〇〇inoという互換品が多く出回っている。

Arduino（続き）

IO
- デジタルIO　0～１３　　（最大負荷４０mA）
- アナログIO　0～5
  ※～のあるデジタルピンはPWM （Plus Width Modulation：パルス幅変調）が使えるピンを表す。通常、3、5、6、9、10、11でPWM出力が行える。

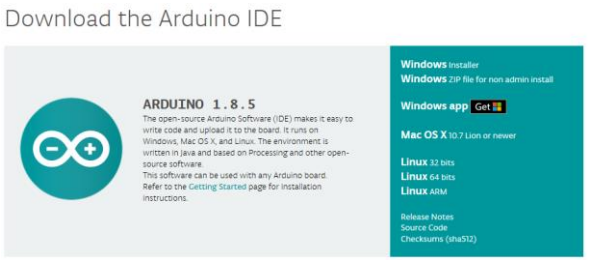
電源・・・外部電源またはUSB経由で供給
- 3.3V出力　　（最大負荷５０mA，一部１５０mA）
- 5V出力　　（最大負荷５０mA）

GND
- 電圧の基準（０V）
- ※電気が流れて帰ってくる場所のイメージ（下水）

▼補足説明

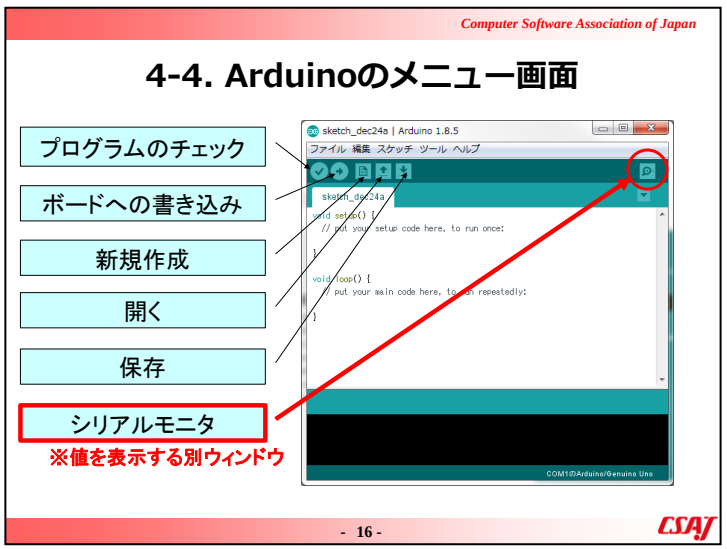
最大負荷とは、利用できる総量の説明でOK。

4-3. Arduino IDEのダウンロードとインストール



インストール版・ZIP版
※通常、インストールや解凍をすればすぐに利用できる
※もし必要がある場合は、PCのデバイスマネージャーから
Arduinoのデバイスを更新する

▼説明の流れ
アルドゥイーノ使用におけるインストール方法を説明する。



1）設定の確認

メニュー［ツール］を開きボードが「Arduino/Genuino Uno」の設定になっていることを確認する

2）ArduinoとPCをUSBケーブルで接続したときに必要な設定

ArduinoをUSBケーブルでPCに接続したとき、シリアルポートを自身の環境のCOM番号に合わせる

メニュー［ツール］→［シリアルポート］で確認or設定

（シリアルポートはデバイスマネージャで確認できる．通常はCOM3の場合が多い．）

3）ライブラリーのインストール時の設定

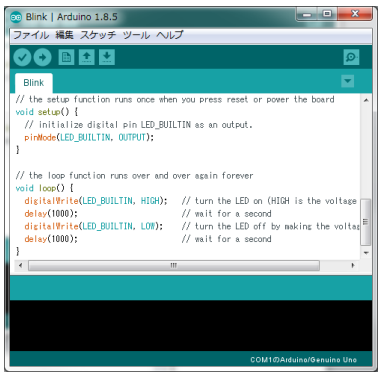
ライブラリをインクルードするときには、ZIP形式のライブラリをダウンロードして以下でインクルードする

メニュー［スケッチ］→［ライブラリをインクルード］→［.ZIP形式のライブラリをインクルード］

Computer Software Association of Japan

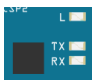
4-5. Arduinoのスケッチ例と動作検証

メニュー［ファイル］→［スケッチ例］→［Basics］→［Blink］



setup()
初期設定

loop()
繰返し処理



ボード上の
LEDが点滅
すればOK

CSAJ

▼説明の流れ

①メニュー［ファイル］→［スケッチ例］→［Basics］→［Blink］をクリックしてサンプルプログラムを開く

②PCとArduinoをUSBケーブルで接続する

メニュー［ツール］→［シリアルポート］で確認or設定

※必要があればシリアルポートを自身の環境のCOM番号に合わせる

③メニュー［→］を押してArduinoにプログラムを書き込む

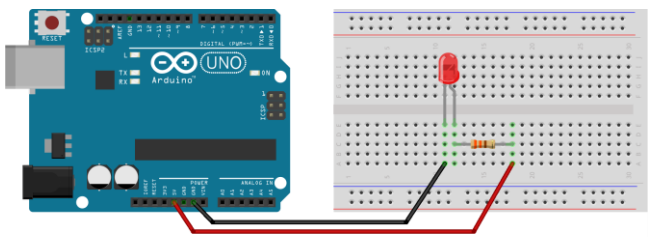
④Arduinoのボード内にあるLEDが点滅すればOK

⑤delay(1000); の数値1000の値を変えるとLEDの点滅する間隔が変わるのでやってみる

※注意点
・USBでの電源供給は不安定な点を注意しておく。→場合によっては外部電源を接続した方が動作が安定するケースがあることを紹介する。
・[シリアルポート]の選択[COM番号]に注意する。

4-6. Arduinoとブレッドボードによる配線

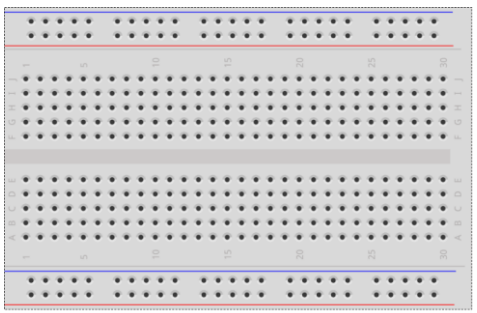
Arduino UNO　　ブレッドボード



▼説明の流れ

アルドゥイーノの配線について説明する。

4-7. ブレッドボードの通電箇所



▼説明の流れ

アルドゥイーノの配線について説明する。（続き）

※注意点

・複雑な回路を作るときはブレッドボードの上側と下側にある＋とーの回路をジャンパーピンで接続すると電源やGNDの配線がやりやすくなる。

・ブレッドボードは刺さっているようでもしっかりと刺さっていないことが多いので注意を促す。

・必要であればテスターでチェックする。

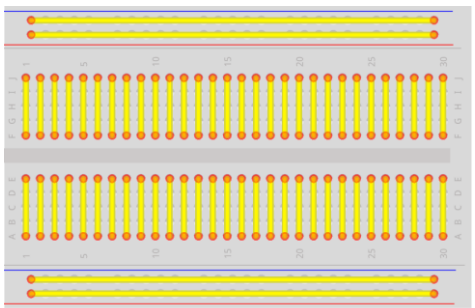
・場合によってはブレッドボードにハンダ付けしても良い点をアドバイスする。

・e-learningで予習してもらったスライドのように感電すると危

険な点を再度注意する。特に、電子部品はすぐに壊れるので注意を促す。

Computer Software Association of Japan

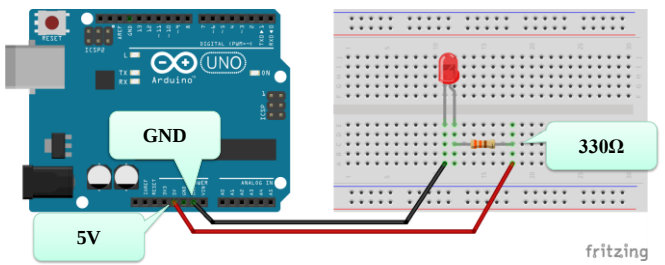
ブレッドボードの通電箇所（続き）



▼説明の流れ

アルドゥイーノの配線について説明する。（続き）

4-8. Arduinoにおける回路設計



▼説明の流れ

アルドゥイーノの配線について説明する。(続き)

▼補足説明

【注意】抵抗を入れないとLEDが壊れる。センサーなどにも必要な抵抗を設置する点に注意を促す。

再掲載

Computer Software Association of Japan

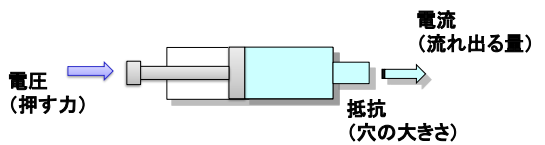
電圧～電流～抵抗

電圧（E）
- 電気を押す力（単位：V）

電流（I）
- 流れる電気の量（単位：A）

抵抗（R）
- 電気の出力の穴の大きさ（単位：Ω）

直列回路
電流はどこも同じ値
電圧の和＝全体の電圧
並列回路
電流の和＝全体の電流
電圧はどこも同じ値



CSAJ

電圧と電流と抵抗のイメージは水鉄砲をイメージするとよい
.

1）水を押し出す力が強いと水が勢いよく飛び出る＝電圧が大きいと電流が大きくなる

　　水を押し出す力が弱いと水は弱くでる＝電圧が小さいと電流も小さくなる

2）水を押す力が同じのとき穴の大きさが小さいと水は勢いよく飛び出る＝抵抗が小さいと電流が大きくなる

　　水を押す力が同じのとき穴の大きさが大きいと水は弱

く出る＝抵抗が大きいと電流は小さくなる

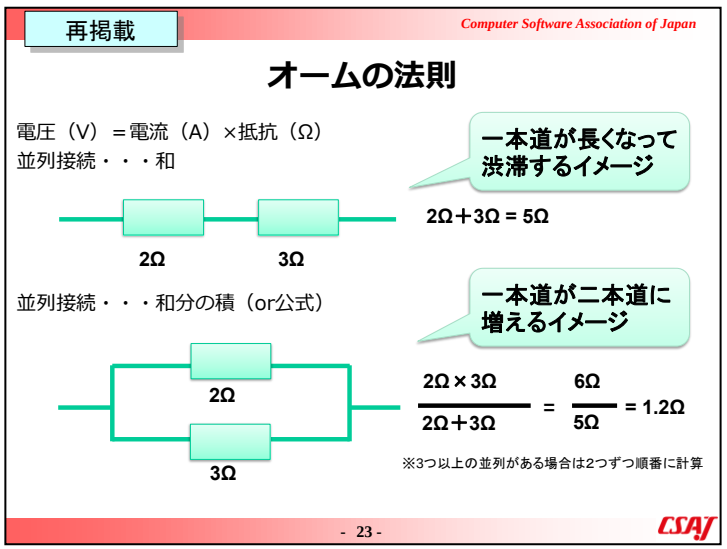
これらの特性を踏まえると、回路を流れる電流の大きさを調整することができる．つまり、

3）流したい電流が目標値より大きい場合、電流を今よりも小さくしたいので、電圧を下げるor抵抗を大きくする

流したい電流が目標値より小さい場合、電流を今よりも大きくしたいので、電圧を上げるor抵抗を小さくする

※通常、電圧は一定なので、抵抗を付け替えることによって電流の大きさを調整する

※例えば、大きな電圧と小さな電圧で同じ大きさの電流を流したい場合、「大きな電圧だが抵抗が大きい」＝「小さな電圧だが抵抗が小さい」で同じ値の電流を流すことができる．



並列接続の場合、和分の積はスライド右下の計算式のように、「分子が掛け算」で「分母が足し算」になる

和分の積は2つの並列接続のときに実施するものである点に注意する.

公式については、ここでは省略している.

【もし公式について説明するならば】
合成抵抗R0Ωを求める公式は、1÷R0＝1÷R1＋1÷R2＋1÷R3＋・・・＋1÷Rnとなる. つまり、抵抗の値をひっくり返したもの（逆数）を合計する. しかし、上記の和分の積を

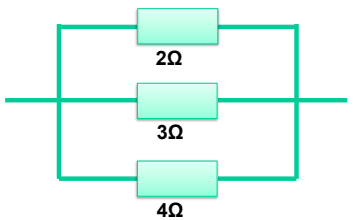
繰り返して2つの抵抗を1つに計算していくことで、合成抵抗を求めることができる.

再掲載

Computer Software Association of Japan

並列接続における和分の積

並列接続・・・３つ以上を和分の積で計算するのは間違い



間違い

$$\frac{2\Omega \times 3\Omega \times 4\Omega}{2\Omega + 3\Omega + 4\Omega}$$

CSAJ

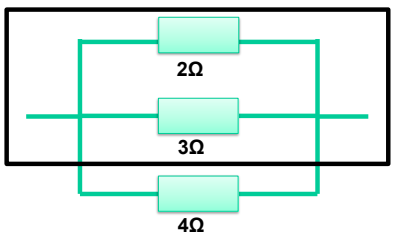
和分の積は2つの並列接続のときに実施するものである点に注意する．例えば、3つの場合は3つ同時に計算することはできない．つまり(2Ω×3Ω×4Ω)÷(2Ω＋3Ω＋4Ω)は間違いなので注意すること．3つの抵抗が並列接続している場合は、まず3つのうちの任意の2つを和分の積を用いて1つに計算する．その後、計算結果の1つと残りの1つの2つを再度和分の積で計算することになる．

再掲載

Computer Software Association of Japan

並列接続における和分の積（続き）

並列接続・・・まず一部分を和分の積で計算する



$$\frac{2\Omega \times 3\Omega}{2\Omega + 3\Omega} = \frac{6\Omega}{5\Omega} = 1.2\Omega$$

CSAJ

和分の積は2つの並列接続のときに実施するものである点に注意する．例えば、3つの場合は3つ同時に計算することはできない．つまり(2Ω×3Ω×4Ω)÷(2Ω＋3Ω＋4Ω)は間違いなので注意すること．3つの抵抗が並列接続している場合は、まず3つのうちの任意の2つを和分の積を用いて1つに計算する．その後、計算結果の1つと残りの1つの2つを再度和分の積で計算することになる．

再掲載

Computer Software Association of Japan

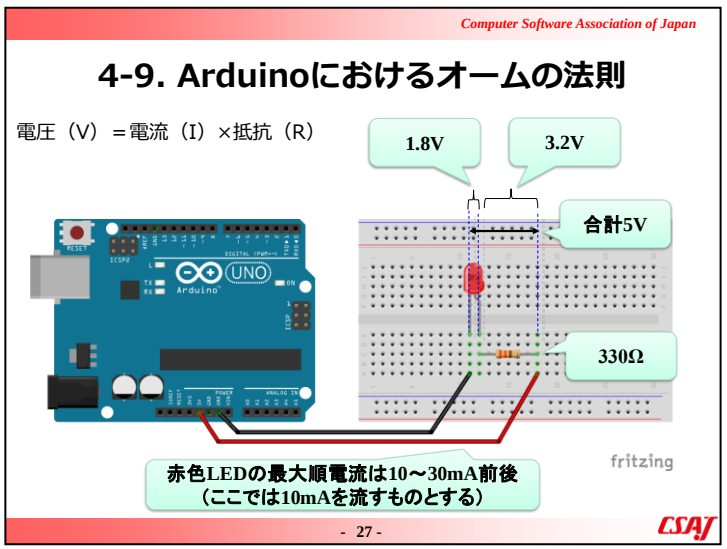
並列接続における和分の積（続き）

並列接続・・・残りの部分を２回目の和分の積で計算する

$$\frac{1.2\Omega \times 4\Omega}{1.2\Omega + 4\Omega} = \frac{4.8\Omega}{5.2\Omega} = 0.92\Omega$$

CSAJ

和分の積は2つの並列接続のときに実施するものである点に注意する．例えば、3つの場合は3つ同時に計算することはできない．つまり(2Ω×3Ω×4Ω)÷(2Ω＋3Ω＋4Ω)は間違いなので注意すること．3つの抵抗が並列接続している場合は、まず3つのうちの任意の2つを和分の積を用いて1つに計算する．その後、計算結果の1つと残りの1つの2つを再度和分の積で計算することになる．



▼説明の流れ

オームの法則を中心に説明する。（電圧～電流～抵抗についてはe-learningで閲覧済み）

▼補足説明

抵抗入りのLEDを利用する場合は、330Ωなどの抵抗を付けなくてよい場合もあることを説明する。

演習１　Arduinoを使った電気回路の設計

① LEDが点灯する回路
② スイッチでLEDをON・OFFする回路
③ スイッチを押したときにLEDをONする回路とプログラム
④ アナログ出力によるLED点灯
⑤ 応用編：LEDの種類や個数を変更
※作成した回路の回路図を描き抵抗値を書き込む

▼演習

できるだけ学習者自身で問題を解決するように考えてもらう
→試行錯誤が講習後に自分でやるときの糧になる。

①LEDが点灯する回路
・LEDの＋とーを間違えないように注意する。
・ジャンパーピンを刺すブレッドボードの穴の列を間違えないように注意する。
②スイッチでLEDをON・OFFする回路
スイッチに正しい向きがあるので注意する。→電子部品を取り扱うときの全般の注意に促す。
③スイッチを押したときにLEDをONする回路とプログラム
・スケッチ（プログラム）がでてきたのでスケッチの流れや関

数について説明する。
・この回路は「プルダウン」という方式になっていることを説明する。
※注意
・「プルダウン」や「プルアップ」の簡単な説明が演習資料に載っているが、その場ですぐに理解することは難しいだろう.
・センサーの動作が安定しない場合、これらの利用を試してみることをすすめること。
④アナログ出力によるLED点灯
・学習者の進み具合をみてスケッチについて説明する。
・LEDの種類と抵抗の種類によってLEDの点灯の様子が変わる点に注意しておくこと。
・相性によってはLEDの微妙な点灯の変化は確認できないかもしれない。

第５章 組込ボードとセンサ

目的:
組込ボードとセンサの基本的な取り扱い方（特に組込ボードを中心）を学ぶ。
ゴール:
・組込ボードの１つであるArduinoをIDEから利用できる。
・ブレットボードを使って基本的な配線ができる。
・オームの法則を使って適切な抵抗を選ぶことができる。

5-1. センサ

環境センサ
入力モジュール
出力モジュール

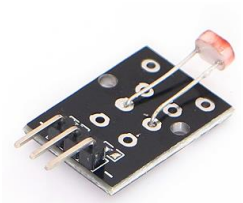
※次スライド以降はKumanのデータファイルよりの抜粋
※Kumanのマニュアルは付属のCD-ROM内にある

▼説明の流れ

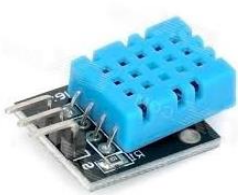
IoTにおけるセンサはデータの入り口に値することを説明する。

5-2. 環境センサ

光センサ



温湿度センサ



▼説明の流れ

配布したセンサを種類別に分類しながら確認させる。
OSOYOOの公式サイトやKumanの公式マニュアルを例に出しながらセンサについて確認していく。

※注意点

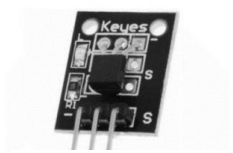
ただし、公式サイトや公式マニュアルにはすべてが書いてない点を確認させる。特に、外国製品ではマニュアルが十分ではなく、日本語表記や英語表記がおかしい点や、場合によっては適切な情報が欠けていたりする点を確認する。
→自分で情報を集める必要がある。

ここでは光センサと温湿度センサについて概要を説明する。

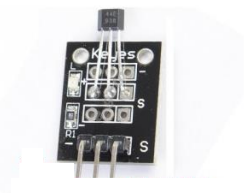
▼補足説明
具体的にどのような状況で光センサと温湿度センサが使用されるのかを話し合ってみる。

環境センサ（続き）

温度センサ



磁場センサ



▼説明の流れ

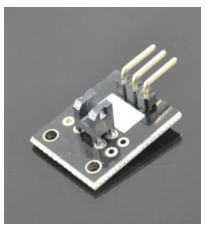
ここでは磁場センサと温湿度センサについて概要を説明。

▼補足説明

解説図について意味が分からない部分があれば解説する。

環境センサ（続き）

光遮断センサ



その他

- アナログ磁場センサ
- アナログ温度センサ
- 地磁気センサ
- 超音波センサ
- 赤外線センサ

など

▼説明の流れ

ここでは光遮断センサ、その他のセンサについて概要説明する。

5-3. 入力モジュール

ジョイスティック　　ロータリーエンコーダ

CSAJ

▼説明の流れ

入力モジュールについて概要を説明する。ロータリーエンコーダー、ジョイスティック。

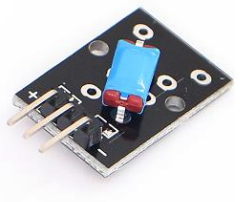
▼補足説明

ロータリーエンコーダはArduinoをはじめよう 初心者実験キット 基本部品セット20in1 UNO R3互換ボード (OSOYOO) の方が使いやすいのでそちらを推奨する。

入力モジュール（続き）

衝撃センサ

傾斜スイッチ

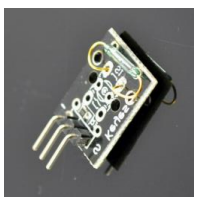


▼説明の流れ

入力モジュールについて概要を説明する。衝撃センサ、傾斜スイッチ。

入力モジュール（続き）

リードスイッチ



その他

- ボタン
- タッチセンサ
- 水センサ

など

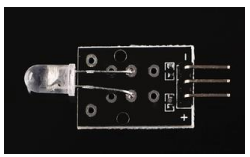
▼説明の流れ

入力モジュールについて概要を説明する。リードスイッチ、その他。

5-4. 出力モジュール

レーザ

7色LED

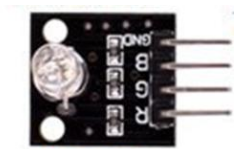


▼説明の流れ

出力モジュールについて概要を説明する。レーザー、7色LED。

出力モジュール（続き）

RGB LED



その他

- LCD
- サーボ
- モータ

など

▼説明の流れ

出力モジュールについて概要を説明する。RGB、その他。

演習２　Arduinoとセンサを使った回路設計

環境センサ
① 光センサの利用
② 光センサによるLED点灯

入力モジュール
③ 傾斜スイッチの利用
④ ロータリーエンコーダによるLED点灯

出力モジュール
⑤ LCD出力

⑥ 応用編：各自で色々なセンサやモジュールを組合せて利用

▼演習

①光センサの利用

1.環境センサの例として光センサを利用してみる　2.新しく出てきたスケッチの関数を説明する。

②光センサによるLED点灯

1. 回路が複雑になってくるので学習者が配線する時間を十分に確保する。
2. スケッチ内のインプットとアウトプットのピン番号を確認させ、関数の引数として与えてある点を確認させる。

③傾斜スイッチの利用

1.入力モジュールの例として傾斜スイッチを利用してみる。
- 物理的な現象が数値化されている点を確認させる。
- センサにも誤差があるので、それを体感で確認させる。

④ロータリーエンコーダによるLED点灯

1. 新しく出てきたスケッチの関数を説明する。
   ・OSOYOOのロータリーエンコーダの方が配線が楽なので、こちらを使う点を注意させる。

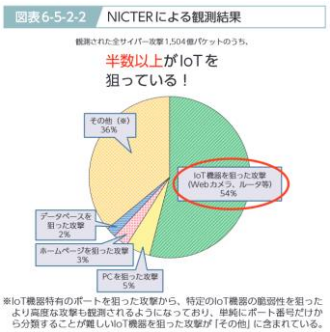
⑤LCD出力

1. 出力モジュールの例としてLCDを利用してみる。
2. ライブラリのダウンロードがあるので講師と一緒に動作させる.
   ・ライブラリのインストールは頻繁にあるので、新しいセンサやモジュールを利用する際にはライブラリの有無を調べるように注意を促す。
   ・LCDを利用する場合には、LCDのレジスタ（I2Cアドレス）を指定する必要がある.
   ・I2CはI2Cのことで、アイ・スクエアー・シー、アイ・スクエアード、シー、アイ・ツーシーなどと呼ばれる。

第６章　IoTのセキュリティ

6-1. IoTデバイスを標的としたマルウェア

IoTデバイスの普及に伴って、Miraiウィルスのようなデバイスを標的としたマルウェアが流行



NICT（国立研究開発法人情報通信研究機構）が運用するサイバー攻撃観測網（NICTER）が平成29年に観測したサイバー攻撃パケット、1,504億パケットのうち、半数以上がIoT機器を狙ったものであるという結果が示されている

情報通信白書平成30年版（総務省）
http://www.soumu.go.jp/johotsusintoke/whitepaper/ja/h30/pdf/30honpen.pdf

6-2. Miraiウィルス

Miraiウィルス
- IoTデバイスに感染しボットネットを作るマルウェア
- ボットネットから攻撃目標に対してDDoS攻撃を行う
- 2016年にMiraiウィルスのボットネットが発見される
- プロバイダやIT企業，ジャーナリストなどへの大規模かつ破壊的な攻撃が観測された

Miraiウィルスの挙動
- 対象外を除いてIPアドレスをランダムに走査
- 脆弱性のある機器を調査（工場出荷時・デフォルト状態、辞書攻撃など）
- 感染したデバイスはC&Cサーバ（指令＆制御）から遠隔操作
- DDoS攻撃を実行（UDPフラット攻撃、HTTPフラット攻撃、DNSフラット攻撃）
- 増殖を繰り返し感染を拡大

Miraiウィルス（続き）

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、スパイウェア、ボット、ランサムウェアなどの悪意のあるソフトウェアのこと
- 総合的な名称としてマルウェア（Malware）と呼ぶ

ボット
- 感染したコンピュータを外部から遠隔操作し不正アクセスの手足として利用し、迷惑メールの送信や特定サイトへの攻撃などを行うプログラム

ボットネット
- ボットに感染したコンピュータからなるネットワークはボットネットと呼ばれる
- ボットネットのコンピュータは特定サイトの一斉攻撃（DDos攻撃）などに利用される

Miraiウィルス（続き）

DDoS攻撃

- Distributed Denial of Service攻撃の略
- ウィルスに感染して遠隔操作可能な複数の端末から一斉にDoS攻撃（サービス拒否攻撃）を行う
- UDPフラット攻撃、HTTPフラット攻撃、DNSフラット攻撃など、通信プロトコルの手続きのパケットを一斉に大量に送りつけることで、相手が処理しきれなくなりサービスが停止してしまう

6-3. IoTセキュリティガイドライン

経済産業省及び総務省が「IoT推進コンソーシアム　IoTセキュリティワーキンググループ」を開催

- IoTを活用した革新的なビジネスモデルを創出
- 国民が安全で安心して暮らせる社会を実現
- 必要な取組等について検討

「IoTセキュリティガイドライン ver1.0」が策定（平成28年7月5日）

https://www.meti.go.jp/press/2016/07/20160705002/20160705002.html

6-4. IoTセキュリティガイドラインの目的

本ガイドラインの目的は、IoT特有の性質とセキュリティ対策の必要性を踏まえて、IoT機器やシステム、サービスについて、その関係者が**セキュリティ確保の観点から求められる基本的な取組を、セキュリティ・バイ・デザインを基本原則としつつ、明確化すること**によって、産業界による積極的な開発等の取組を促すとともに、利用者が安心してIoT機器やシステム、サービスを利用できる環境を生み出すことにつなげるもの。

なお、本ガイドラインの目的は、サイバー攻撃などによる被害発生時における関係者間の法的責任の所在を一律に明らかにすることではなく、むしろ**関係者が取り組むべきIoTのセキュリティ対策の認識を促す**とともに、その認識のもと、関係者間の相互の情報共有を促すための材料を提供することである。

本ガイドラインは、その対象者に対し、一律に具体的なセキュリティ対策の実施を求めるものではなく、**守るべきものやリスクの大きさ等を踏まえ、役割・立場に応じて適切なセキュリティ対策の検討**が行われることを期待する

IoTセキュリティガイドラインver1.0
https://www.meti.go.jp/press/2016/07/20160705002/20160705002-1.pdf

6-5. サービス提供者のための指針

	指針	主な要点
方針	IoTの性質を考慮した基本方針を定める	•経営者がIoTセキュリティにコミットする •内部不正やミスに備える
分析	IoTのリスクを認識する	•守るべきものを特定する •つながることによるリスクを想定する
設計	守るべきものを守る設計を考える	•つながる相手に迷惑をかけない設計をする •不特定の相手とつなげられても安全安心を確保できる設計をする •安全安心を実現する設計の評価・検証を行う
構築・接続	ネットワーク上での対策を考える	•機能及び用途に応じて適切にネットワーク接続する •初期設定に留意する •認証機能を導入する
運用・保守	安全安心な状態を維持し、情報発信・共有を行う	•出荷・リリース後も安全安心な状態を維持する •出荷・リリース後もIoTリスクを把握し、関係者に守ってもらいたいことを伝える •IoTシステム・サービスにおける関係者の役割を認識する •脆弱な機器を把握し、適切に注意喚起を行う

6-6. 一般利用者のための指針

- 問合せ窓口やサポートがない機器やサービスの購入・利用を控える
- 初期設定に気をつける
- 使用しなくなった機器については電源を切る
- 機器を手放す時はデータを消す

IoTセキュリティガイドラインver1.0
https://www.meti.go.jp/press/2016/07/20160705002/20160705002-1.pdf

第７章　IoTプラットフォームを使ったデータ通信

目的:
IoTプラットフォームの利活用について学習する。
ゴール:
・さくらIoTプラットフォームを利用できる。
・IoTからクラウドにデータをアップロードできる。

7-1. IoTプラットフォームの例

sakura.io

IIJ IoT

▼説明の流れ

さくらのIoTプラットフォームはもとより、SORACOMやIIJ IoTサービスなどの他のサービスも例に出しながらIoTプラットフォームについて紹介する。

IoTサービスは萌芽期にあたり、頻繁にサービスが一新されているため、IoTのサービスプランは各Webサイトから紹介する。

IoTプラットフォームの例（続き）

AWS IoT

SORACOM IoT

▼説明の流れ

AWS、SORACOMの紹介をする。

▼補足説明

AWSなどのとの連携はプラットフォーム側でもまだ確定していないので、本研修では紹介程度にとどめておく。

7-2. IoTプラットフォーム sakura.io

https://sakura.io/

▼説明の流れ

Sakura.io の紹介をする。

▼補足説明

さくらIoTプラットフォームの料金体系については特にしっかり説明しておく。

7-3. sakura.ioの特徴

低価格＆セキュア（閉域網を使用）
クラウド連携可能
最低月額料金 64円（税込み）

▼説明の流れ

Sakura.io では様々なモノやサービスに繋げることが可能なネットワークの提供を行っている。サービス概要を説明する。

▼説明の流れ

LTE通信を利用してセンサ等からサーバーへデータを送る際に使用するモジュール。1万円弱から提供可能である。

さくらのLTE通信モジュール（続き）

sakura.io Webサイト
https://sakura.io/product/

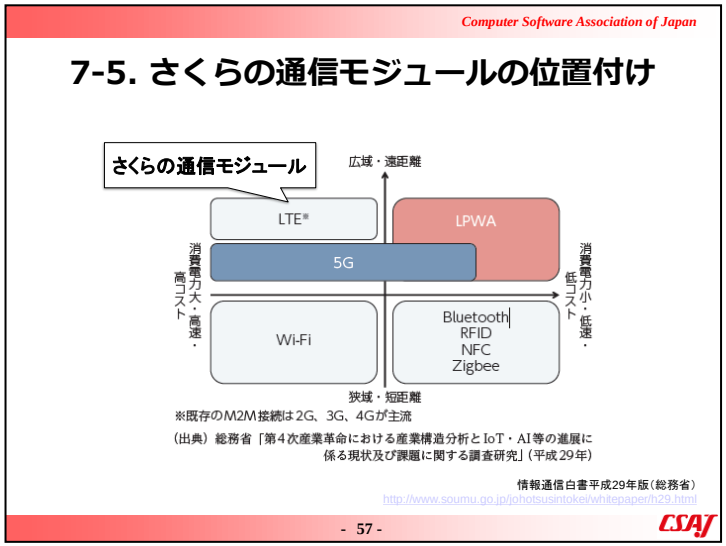
▼説明の流れ
無線通信を利用できるモジュールの説明をする。

さくらのLTE通信モジュール（続き）

sakura.ioモジュール
　LTE通信モジュール、LTEカテゴリー１（低速、小消費電力、IoT向き）

特徴（製品データシートより抜粋）
・sakura.ioにLTE網を通じてダイレクトに接続するため，ゲートウェイ装置がいらない
・コマンドのみでデータの送受信ができ，ホストMCU側で通信プロトコルを実装する必要がない
・ホストMCUインタフェースはI2C，SPI，およびUARTから選択可能
・小型モジュール（46W×34D×3H）内にLTEモデムやSIMなど必要な機能をすべて内蔵
・待ち受け時の消費電力が低い
・日本国内工事設計認証および電気通信端末機器認証済み

▼説明の流れ
Sakura.io モジュールの説明をする。

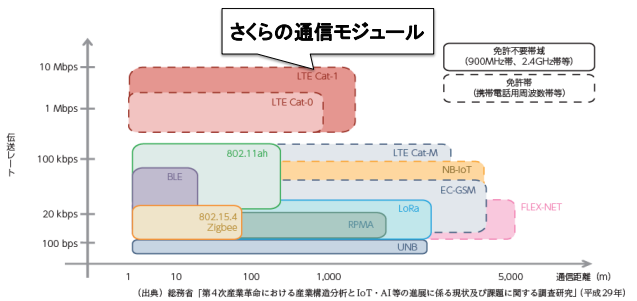


本演習で使用するのは、LPWAを活用したさくらインターネットの「さくらのIoT」

▼説明の流れ

Sakura.io で使用する通信モジュールの位置づけについて説明する。高性能であることがうかがえる。

さくらの通信モジュールの位置付け（続き）



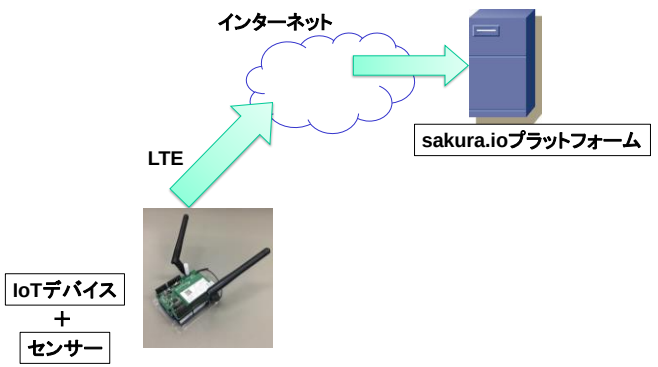
（出典）総務省「第４次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintoki/whitepaper/h29.html

▼説明の流れ

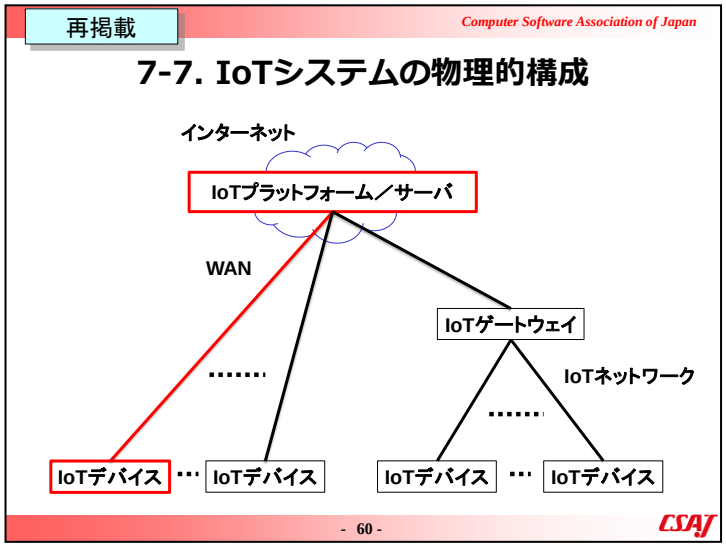
Sakura.io で使用する通信モジュールの位置づけについて説明する。（続き）

7-6. sakura.ioの物理的構成



▼説明の流れ

CSakura.io が管理するデータセンターのコンピューターにデータを集める。



▼説明の流れ

Sakura.io で使用するネットワークの基本構成を説明。途中を経由するネットワークはインターネットだけでなく、WAN回線やLANを利用してより効率的にデータを集約することが可能である。

再掲載

Computer Software Association of Japan

7-3. さくらのIoT Platformの特徴

低価格＆セキュア（閉域網を使用）
クラウド連携可能
最低月額料金 64円（税込み）

「電気信号」と「JSONデータ」の相互変換装置として動作し、これまでのIoTデバイス/サービス開発の
中間層を補完することで、モノ/サービスづくり・連携に注力が可能です。

CSAJ

（既出）

Sakura.io では様々なモノやサービスに繋げることが可能なネットワークの提供を行っている。サービス概要を説明する。

7-8. sakura.io 料金と通信ポイント

１ヶ月につき通信ポイントが10,000pt付与
100回の通信ごとに100pt消費（100回未満は切り上げ）
別途購入する場合は20,000pt/100円
都度消費ではなく、月末に通信回数によってポイント引き落とし。不足すればその分を精算
10,000pt = 10,000回の通信
5分に1度の通信 → 1時間で12回 → 1日 288回
→ 30日で8,640回
5分に1度の通信でも充分。データを貯めて定期的に送信することも可能

▼説明の流れ
Sakura.io におけるコストを説明する。

▼説明の流れ

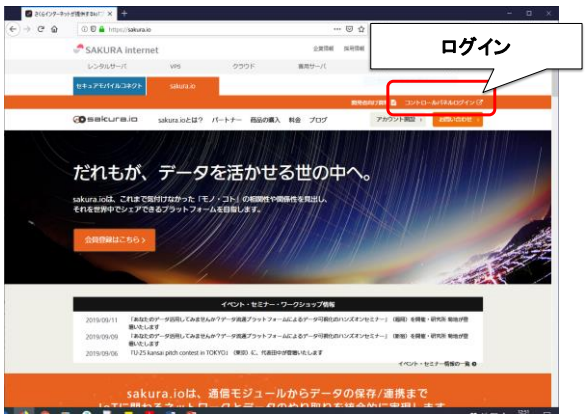
Sakura.io におけるポイントを説明する。

7-10. ライブラリとマニュアル

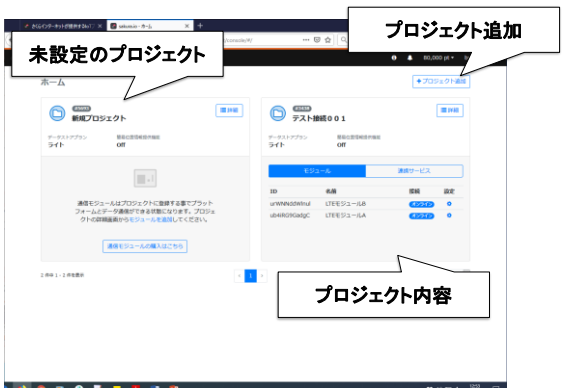
ライブラリ
https://github.com/sakuraio/SakuraIOArduino
マニュアル
https://sakura.io/docs/index.html

▼説明の流れ
Sakura.io の資料を説明する。

7-11. ログインとプロジェクト

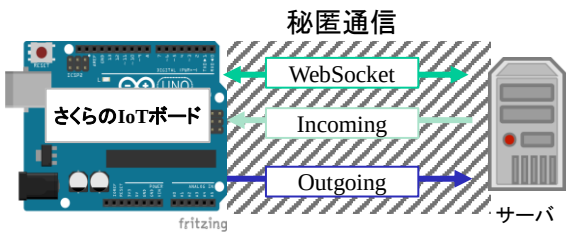


ログインとプロジェクト



7-12. 基本的な考え方

さくらIoTのライブラリを通じてデータの送受信を行う
Arduino側にTCP/IPスタックは必要ない



▼説明の流れ

Sakura.io の通信の基本的な仕組みを説明。これまで説明してきたWebSocketの構造を改めて振り返る。

7-13. コード例

接続

```
sakuraio.getConnectionStatus()
```

データ送信キューに貯める

```
sakuraio.enqueueTx()
```

データ送信

```
sakuraio.send()
```

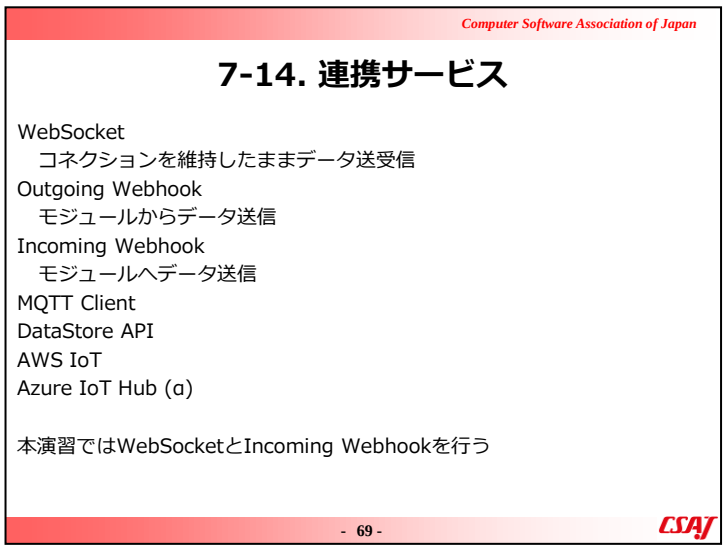
データ受信

```
sakuraio.getRxQueueLength()
```

ライブラリをインポートし、スケッチ例Standardを実行する

▼説明の流れ

Sakura.io で利用されている通信のコード例を説明する。



▼説明の流れ

Sakura.io の連携サービスを説明する。

7-15. WebSocket

従来のhttp等はコネクションレスの通信プロトコル
WebScoketはコネクションを維持したまま通信可能なプロトコル
さくらのIoTで最も簡単に扱える
10秒に1度keepaliveを送信し、コネクションを維持（keepaliveは課金されない）

PHPでWebSocketを扱うのは容易
ただし、コマンドを都度実行したり、Webブラウザで読み込み続ける必要がある

▼説明の流れ

WebSocketの原理を説明し、JavaScriptで扱う方法について説明する。jQueryとChart.jsを紹介し、JavaScriptでデータを可視化する方法について紹介する。
※注意点
・Chart.jsの他にも可視化ライブラリは多数あるが、ライセンスを確認し、業務使用に対する制限を確認する。

7-16. データ形式

データ形式はすべてJSON
送信できるデータ形式は決まっている
int型変数は、符号あり32bit整数のint32_tのみ
同じく符号無しのint型変数は、uint32_tのみ
floatやdoubleはそのままでよい

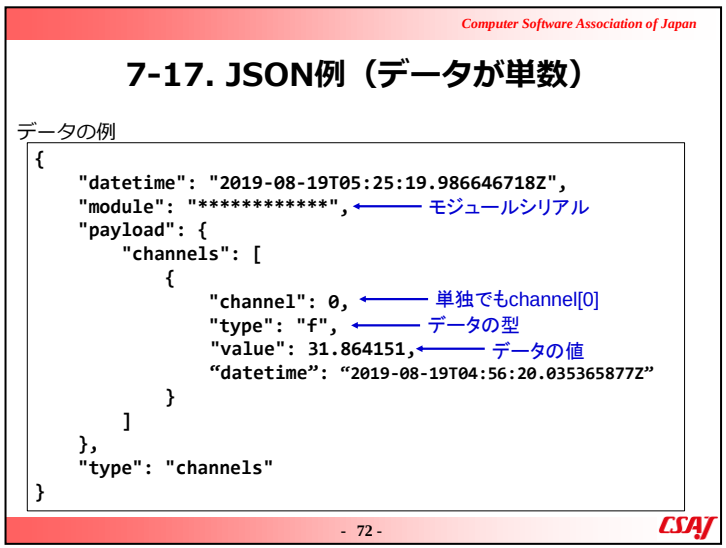
参照：
https://sakura.io/docs/pages/platform-specification/message.html

▼説明の流れ

Sakura.io で利用しているデータ形式を説明する。

▼補足説明

さくらIoTから送られてくる、またIoTモジュールに送信するデータはすべてJSON形式。Javascriptと非常に親和性が良い。

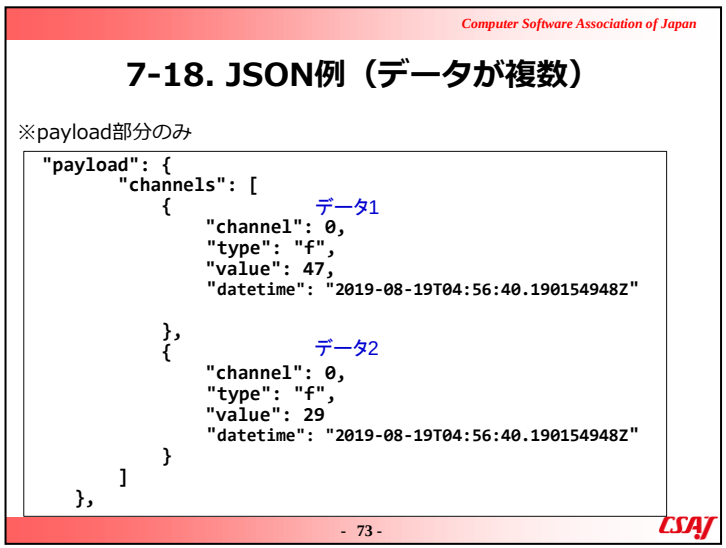


▼説明の流れ

Sakura.io のJSON例を説明する。

▼補足説明

データが単数の場合でも必ずchannel[0]になるので注意が必要である。



▼説明の流れ

Sakura.io のJSON例を説明する。

▼補足説明

データが複数の場合でもやはりchannel[0]になる。複数の場合は、1番目のchannel[0] 、2番目のchannel[0]として取得する。

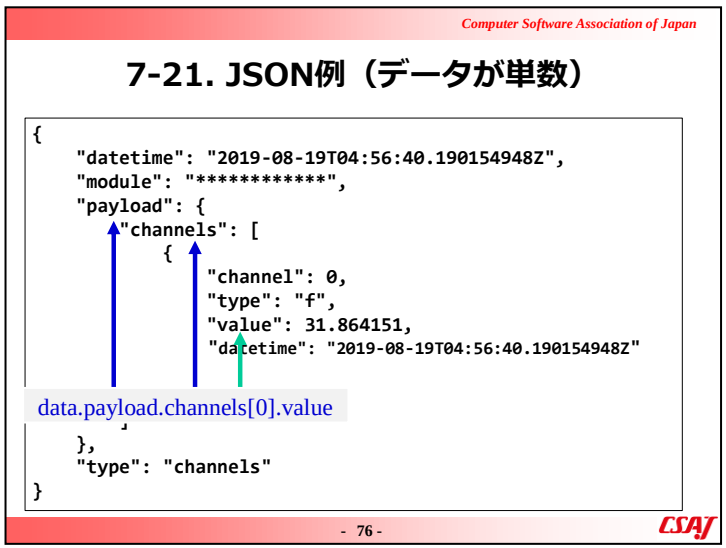
▼説明の流れ

Sakura.io の連携サービスの作成について説明する。

7-20. WebSocketのURLとToken

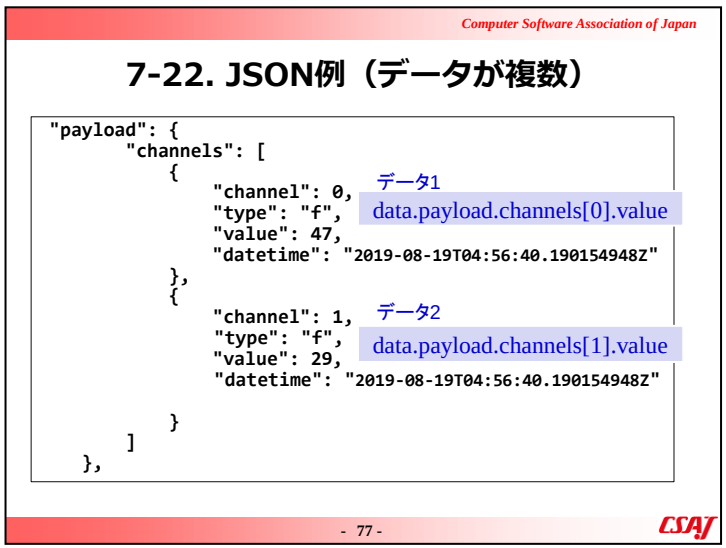
コントロールパネルで確認可能
外部からアクセスする際は、ここに表示されるURLとTokenが必要

▼説明の流れ
Sakura.io のWebSocketのURLとTokenについて説明する。



▼説明の流れ

Sakura.io のJSON例を説明する。

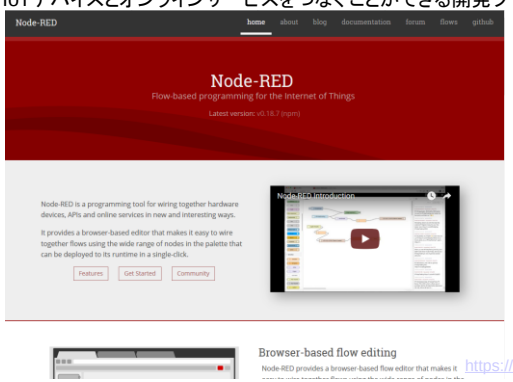


▼説明の流れ

Sakura.io のJSON例を説明する。

7-23. 開発ツール Node-RED

Flowエディタを使って、プラグイン/モジュールであるノードを視覚的に接続しながら、IoTデバイスとオンラインサービスをつなぐことができる開発ツール



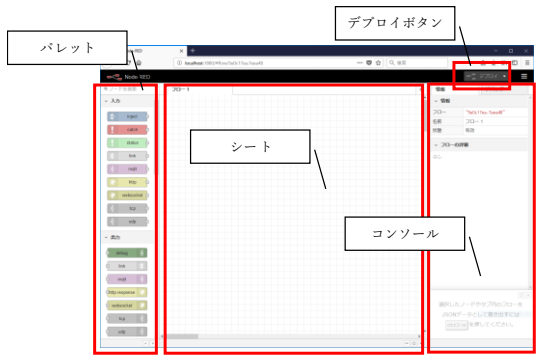
Node-RED https://nodered.org/

▼説明の流れ

Sakura.io の開発ツールNode-REDを説明する。

開発ツール Node-RED

Flowエディタを使って、プラグイン/モジュールであるノードを視覚的に接続しながら、IoTデバイスとオンラインサービスをつなぐことができる開発ツール



▼説明の流れ

Sakura.io の開発ツールNode-REDを説明する。

演習３　さくらLTEモジュールの回路設計と利活用

さくらのIoTコントロールパネルで確認

WebSocketをJavaScriptで取得して表示

Node-REDを使ったデータ通信

▼演習

①さくらのIoTコントロールパネルで確認

1.講師のナビゲートと一緒にコントロールパネルを操作する。
2.ユーザIDとパスワードの配布
3.注意点
・さくらLTEモジュールをArduinoにつけるとPinラベルが見えなくなるので、シールなどを使ってラベルを手書きで作成するとよい点を紹介する。
・さくらのIoTコントロールパネルはIEを推奨している点を注意を促す。
・場合によってはACアダプタなどでArduinoに電源供給が必要かもしれない。→USBケーブルは不安定なため業者は嫌がるケースもある.

・演習時にデータのアップロード間隔をあまり狭くしないように注意を促す。→料金がほぼ青天井。

②WebSocketをJavaScriptで取得して表示
③JavaScriptでデータのグラフ化
1. WebSocketとJavaScriptを用いて、センサ情報の取得やグラフによる可視化をプログラムとして実装する。

演習４　総合演習

これまで学んだものに基づいて各自のIoTシステムを構築

【必須】
・IoTデバイスに任意のセンサを利用する
・取得したセンサーの値をsakura.ioにアップロードする

【任意】
A. センサを複数にする／センサにアクチュエータをつける
B. sakura.ioに集めたデータをNode-Redを使って可視化する
C. IoTデバイスへのフィードバック機能を任意につける
D. その他

▼演習

1.回路作成
・学習者に自由に発想させて回路を組み立てさせる。
・時間を十分に確保する。
・最初の講義でやった現実世界と仮想世界の橋渡しとなるようなIoTサービスを企画させて作成させる。

2.成果物報告
・最後に成果物を学習者に発表させて情報共有して学習者に気付きを促す。

3.振り返り

講義全体を振り返って学習した内容を確認する。

IoT活用

目次（1）

1-3章はE-Learning

目次（2）

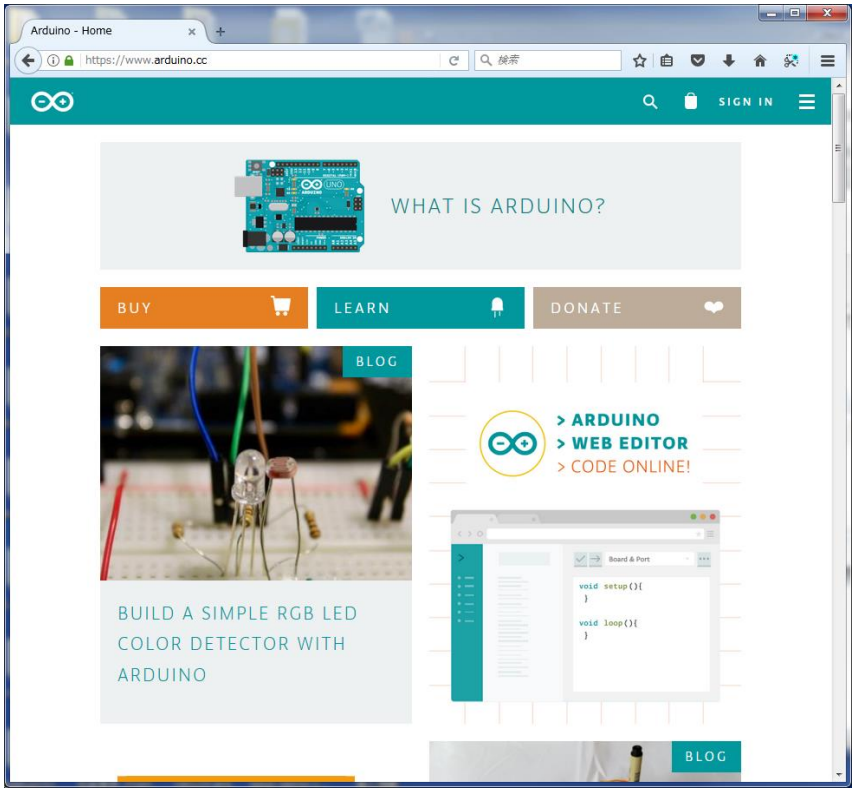
目次（3）

目次（4）

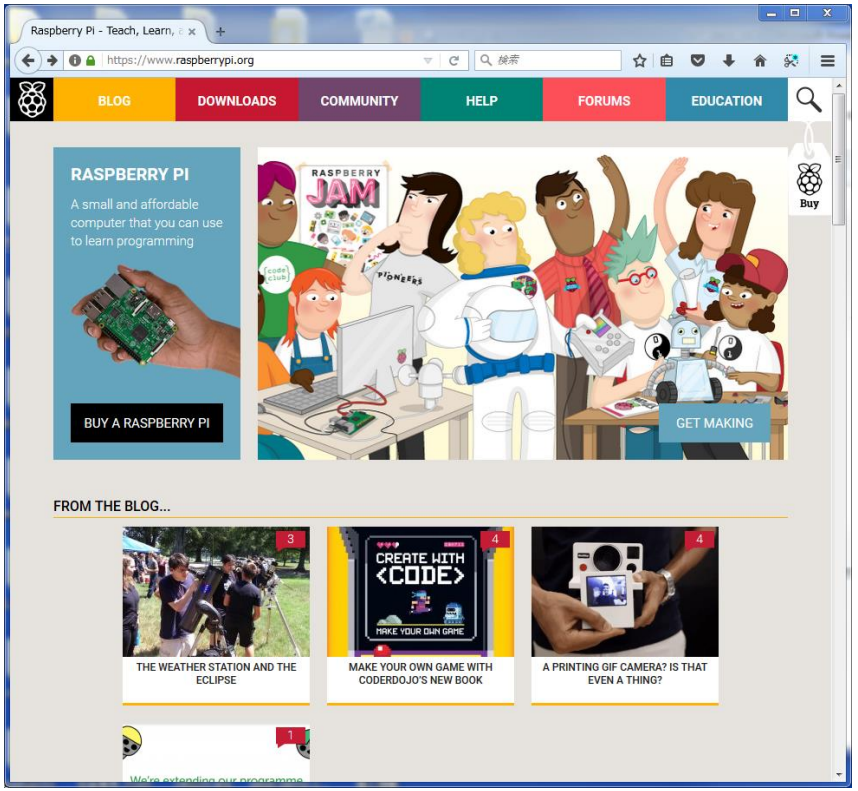
第４章 組込ボードの基礎

4-1. IoTで使用される組込ボードの例

Arduino

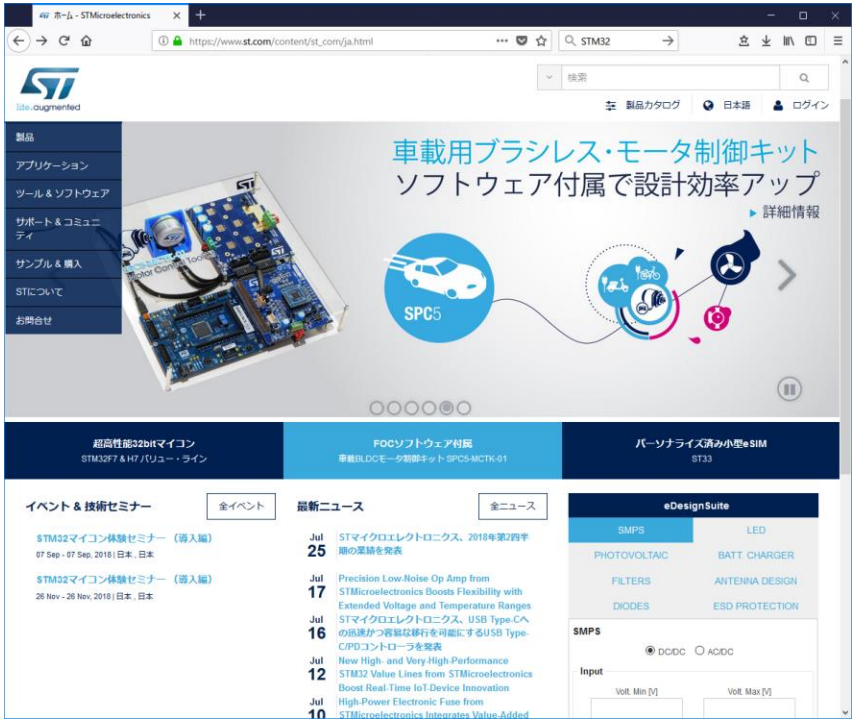


Raspberry Pi



IoTで使用される組込ボードの例（続き）

STM32



IchigoJam

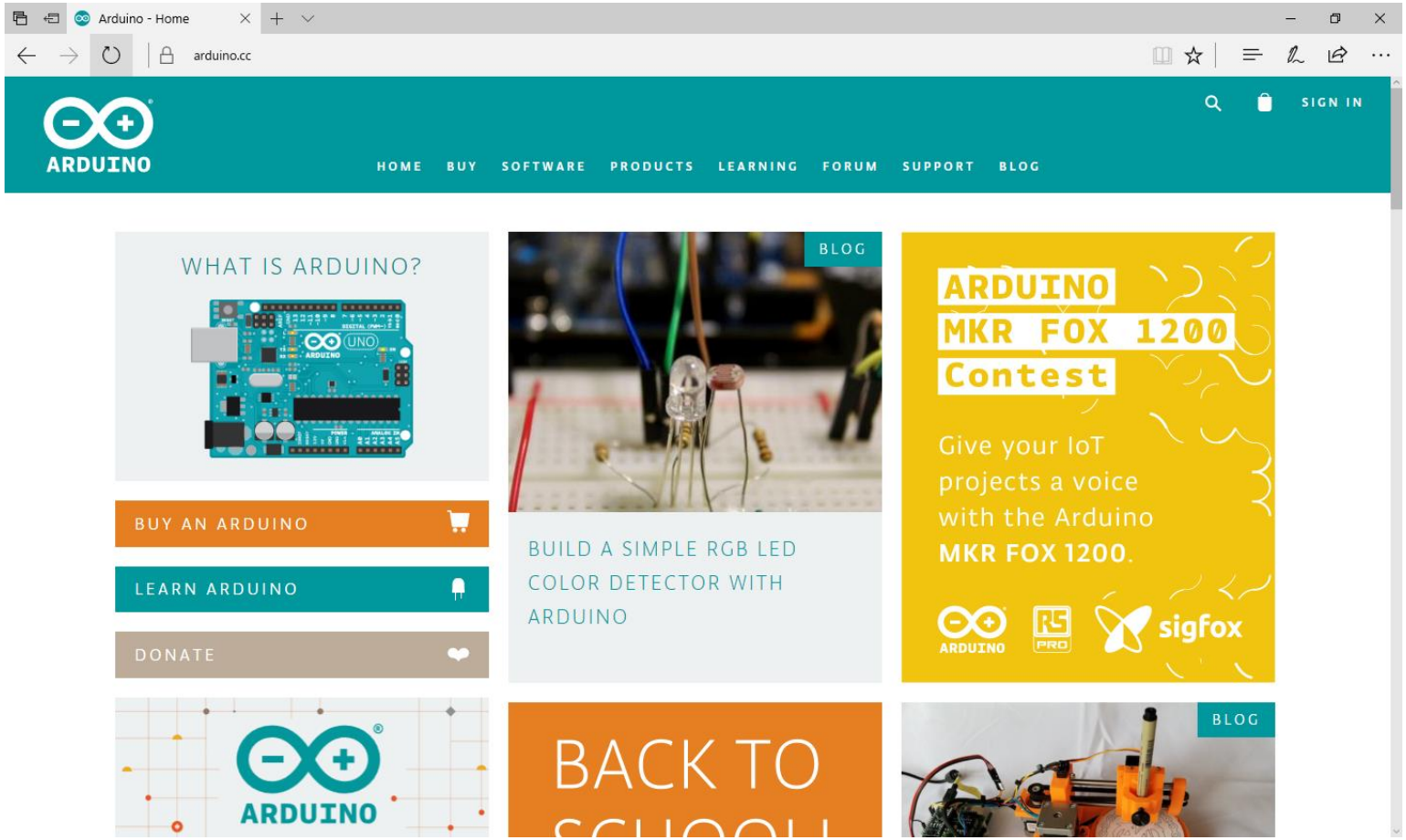
IoTで使用される組込ボードの例（続き）

単独で開発が可能な**Raspberry Pi**（※初期設定時のみにPCが必要）
- シングルボードコンピュータ≒PC
- OS：Linux（DebianベースのRasbianなど）
- ディスプレイやキーボードをつないでPCと同じように開発
- 様々なプログラミング言語が利用可（C、C++、Python、Node.jsなど）
- 良くも悪くもPCと同じ開発環境

母艦（PC）からプログラムを書き込む**Arduino**
- ワンボードマイコン
- OS非搭載（その分、省電力）
- 母艦のPCにインストールした「Arduino IDE」からプログラムを書き込む
- C言語風のArduino言語を利用
- ライブラリを読み込めば簡単に実現ができる開発環境

ヘッダーを意識せずにTCP/IP通信が可能

4-2. Arduino



（https://www.arduino.cc/）

Arduino（続き）

アルドゥイーノ

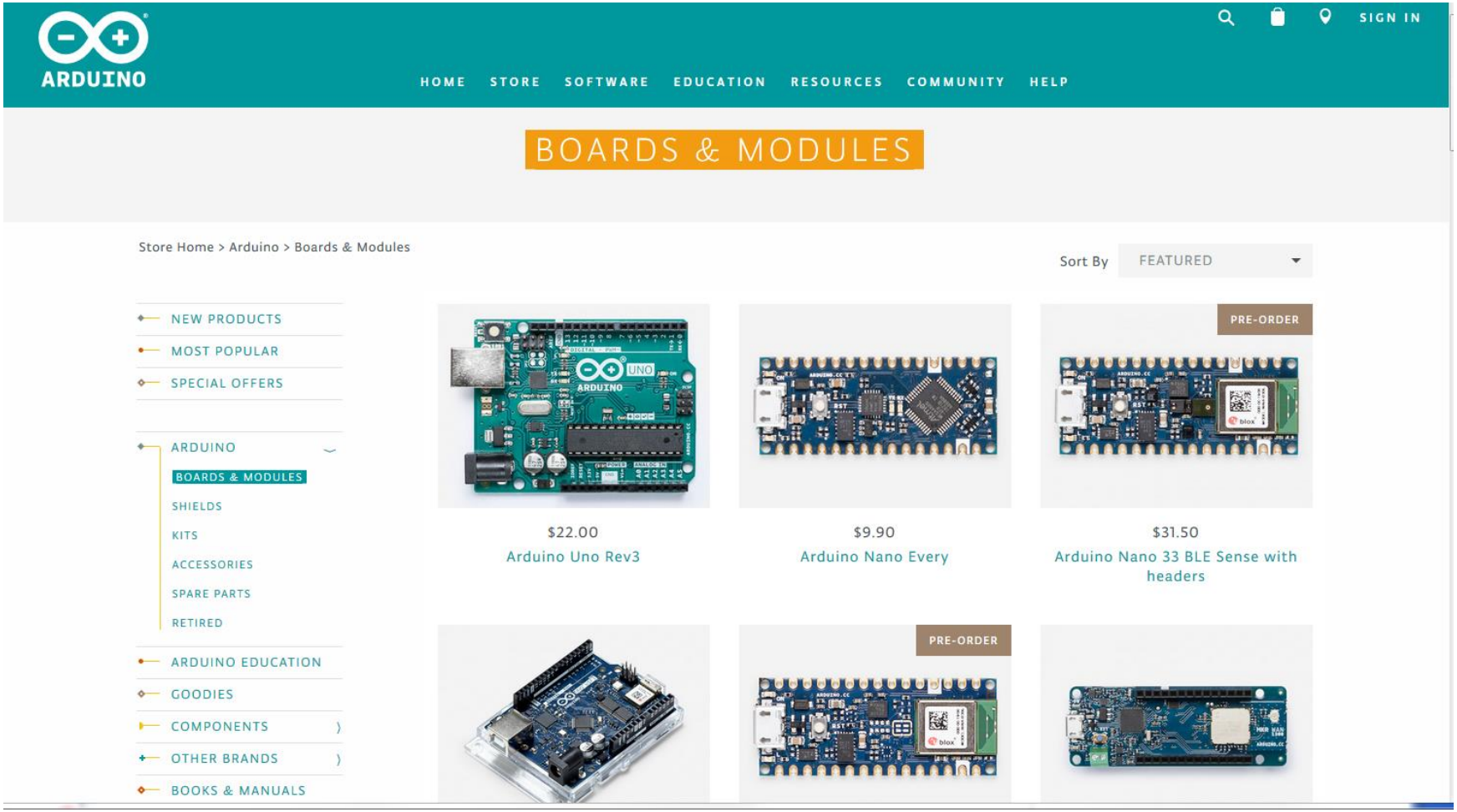
2005年に、Massimo Banzi・David Mellis（当時はイタリアのIDIIIの学生）、David Cuartiellesによってプロジェクトスタート、後にTom Igoeが加わる

- 派生元
  - 2003年　IDII修士論文プロジェクトWiring（Hernando Barragán）
  - 「Arduinoの語られざる歴史」より https://arduinohistory.github.io/ja.html

一枚のプリント基盤の上に、電子部品と入出力がついたマイクロコンピュータ

- Processingベースの開発環境（Javaアプリケーション）
- プログラミング言語：C++風言語（Arudino言語とも呼ばれる、元はWiring）

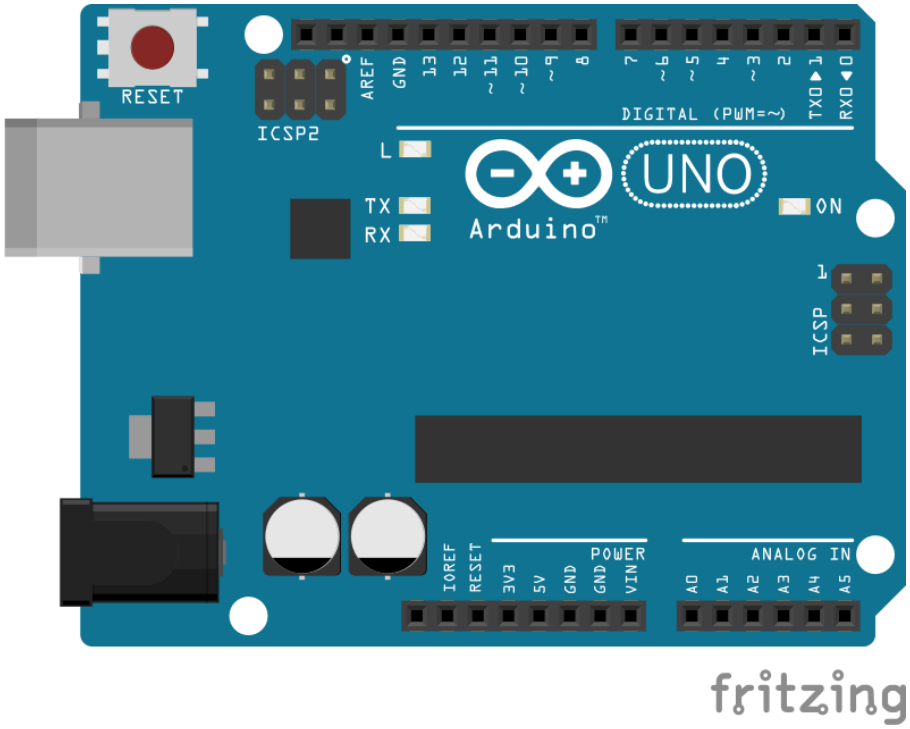
Arduino（続き）



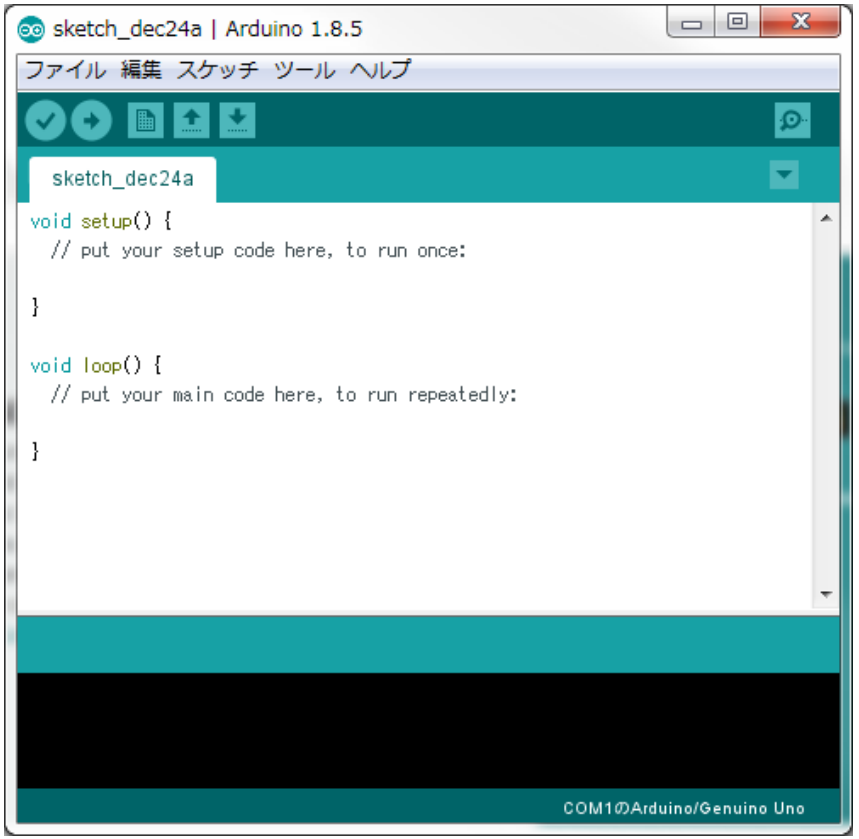
https://store.arduino.cc/usa/arduino/boards-modules

Arduino（続き）

Arduinoのプリント基板



Arduinoの開発環境（IDE）



Arduino（続き）

IO

デジタルIO　０～１３　　（最大負荷４０mA）

アナログIO　０～５

※～のあるデジタルピンはPWM （Plus Width Modulation：パルス幅変調）が使えるピンを表す。通常、3、5、6、9、10、11でPWM出力が行える。

電源・・・外部電源またはUSB経由で供給

3.3V出力　　（最大負荷５０mA，一部１５０mA）

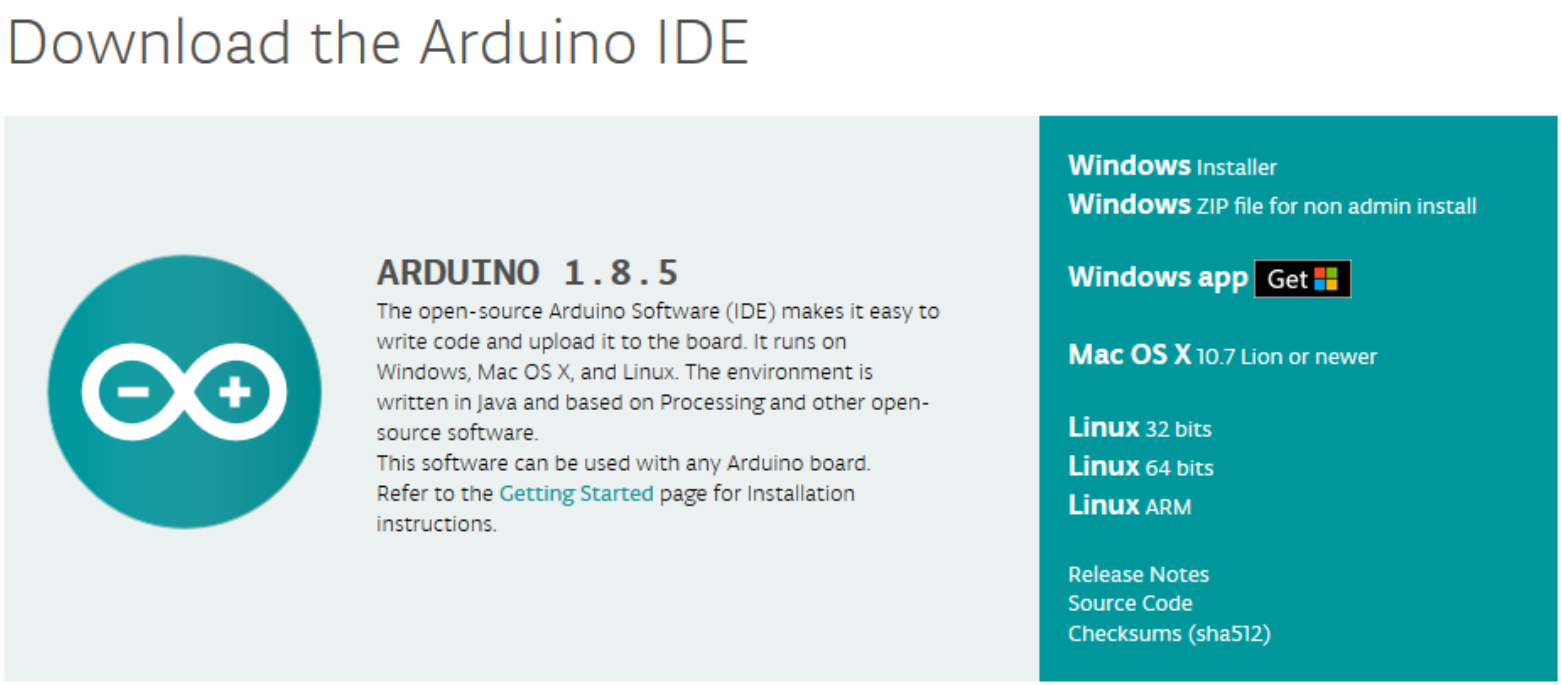
5V出力　　（最大負荷５０mA）

GND

電圧の基準（０V）

※電気が流れて帰ってくる場所のイメージ（下水）

4-3. Arduino IDEのダウンロードとインストール



インストール版・ZIP版

※通常、インストールや解凍をすればすぐに利用できる

※もし必要がある場合は、PCのデバイスマネージャーから
Arduinoのデバイスを更新する

4-4. Arduinoのメニュー画面

プログラムのチェック

ボードへの書き込み

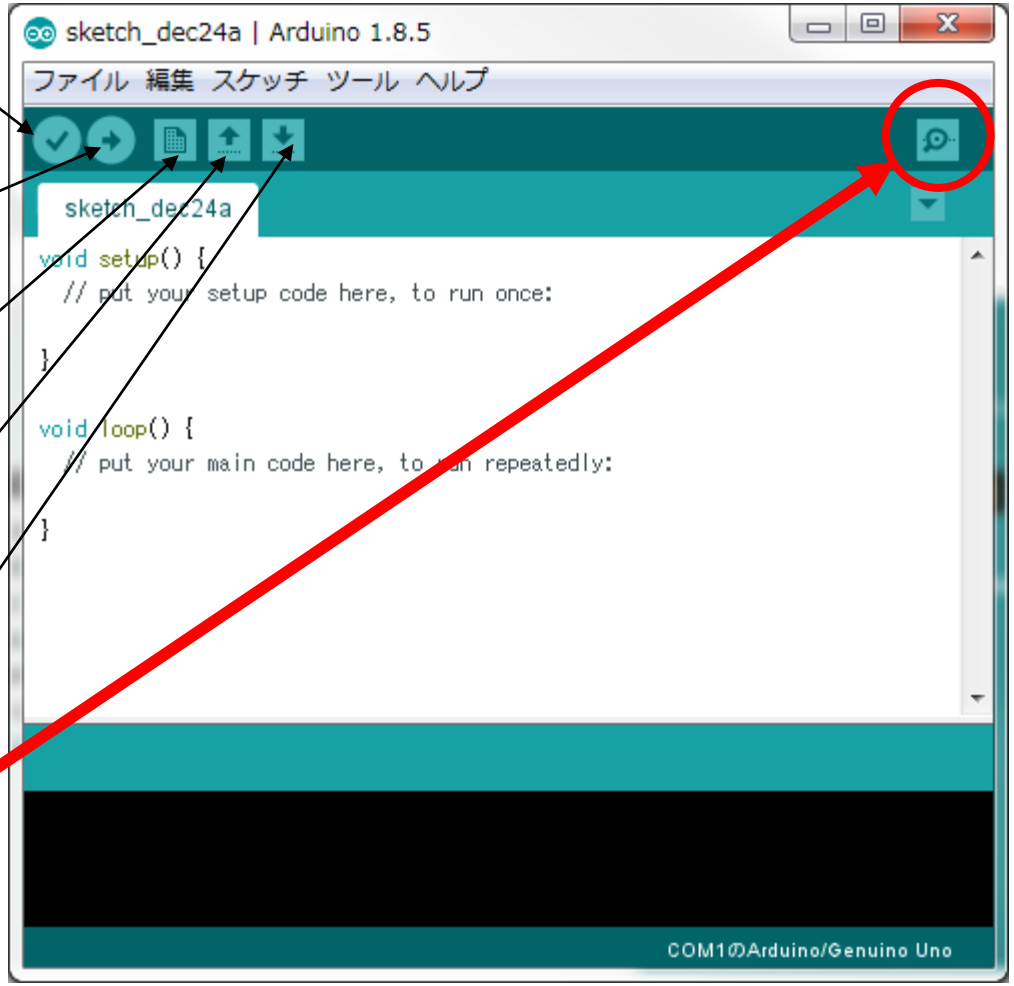
新規作成

開く

保存

シリアルモニタ

※値を表示する別ウィンドウ

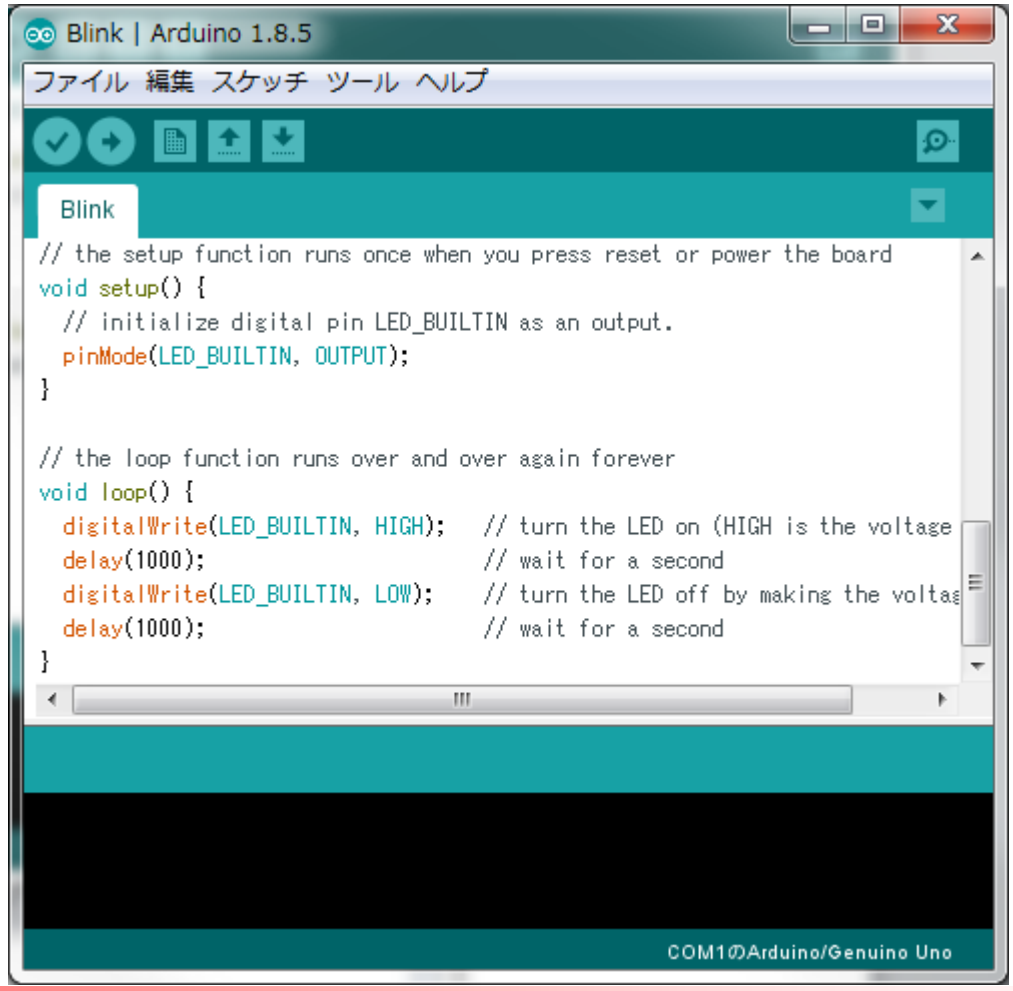


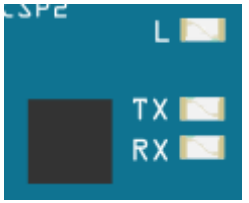
4-5. Arduinoのスケッチ例と動作検証

メニュー［ファイル］→［スケッチ例］→［Basics］→［Blink］

setup()
初期設定

loop()
繰返し処理



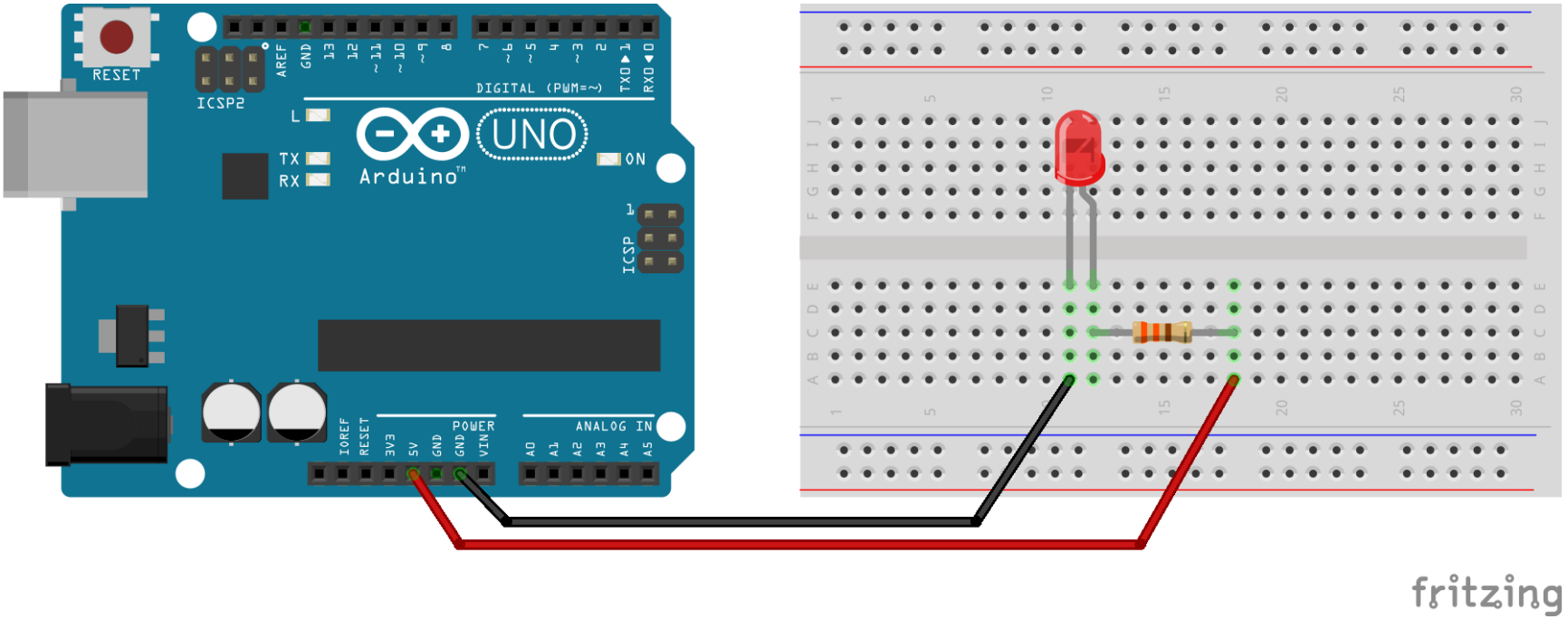


ボード上の
LEDが点滅
すればOK

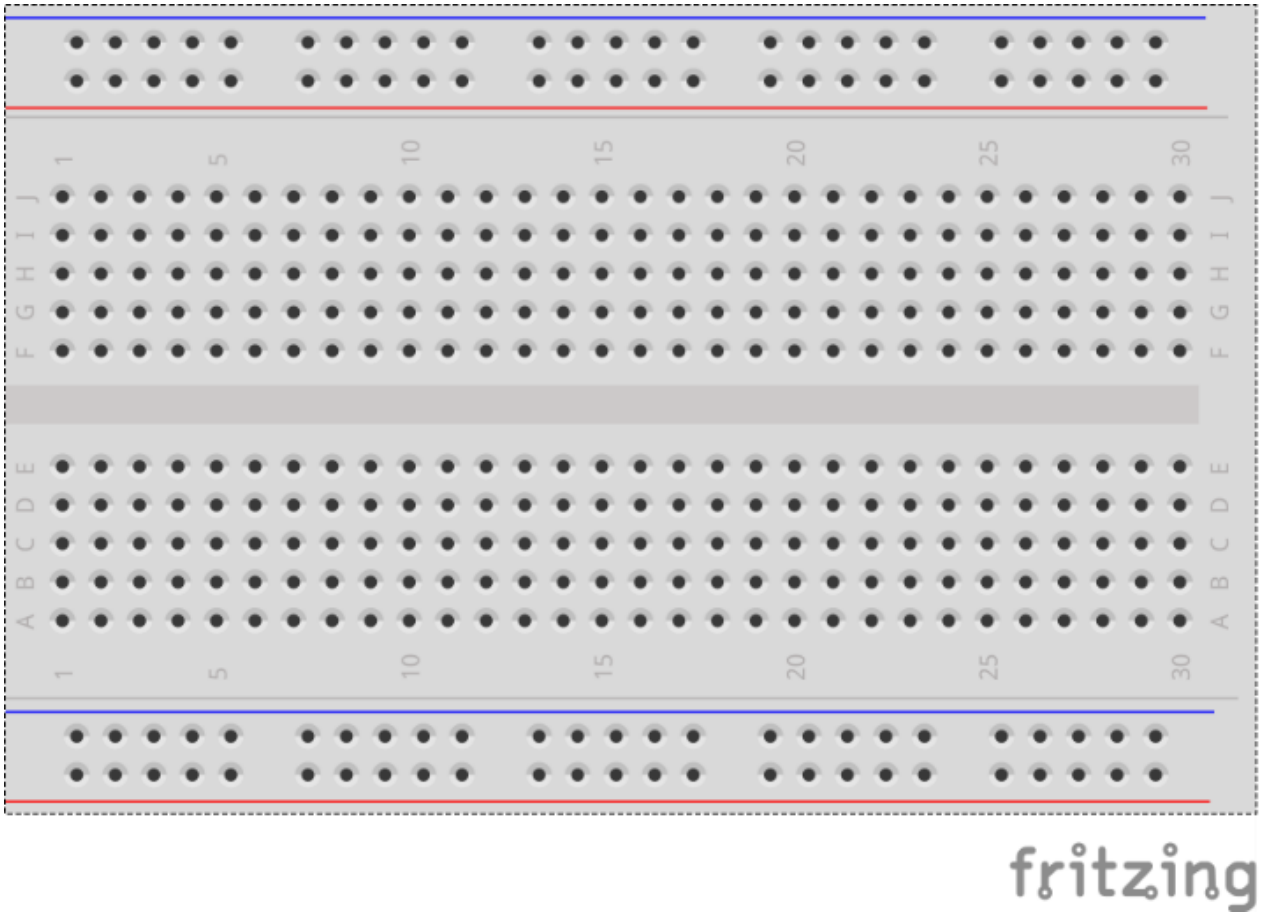
4-6. Arduinoとブレッドボードによる配線

Arduino UNO

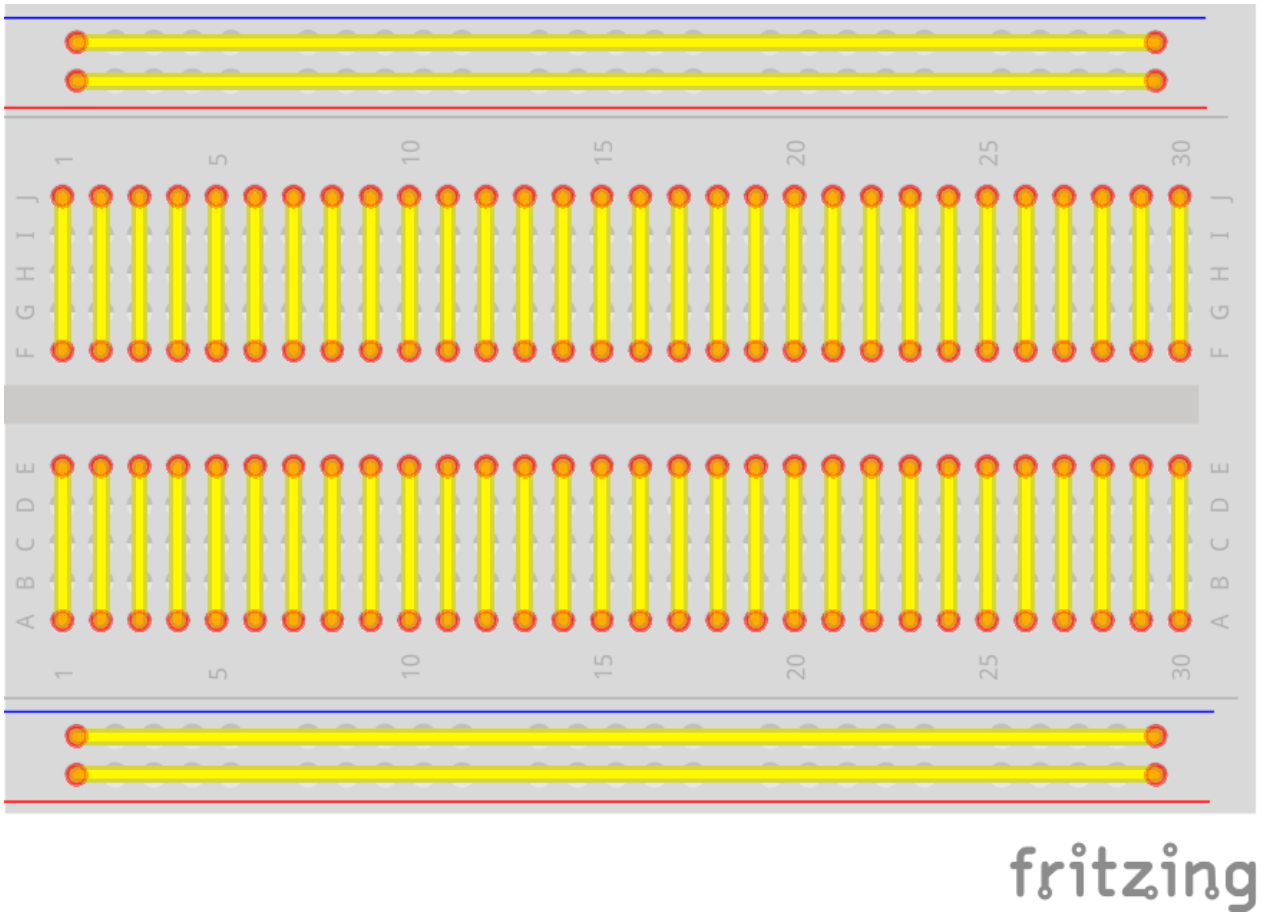
ブレッドボード



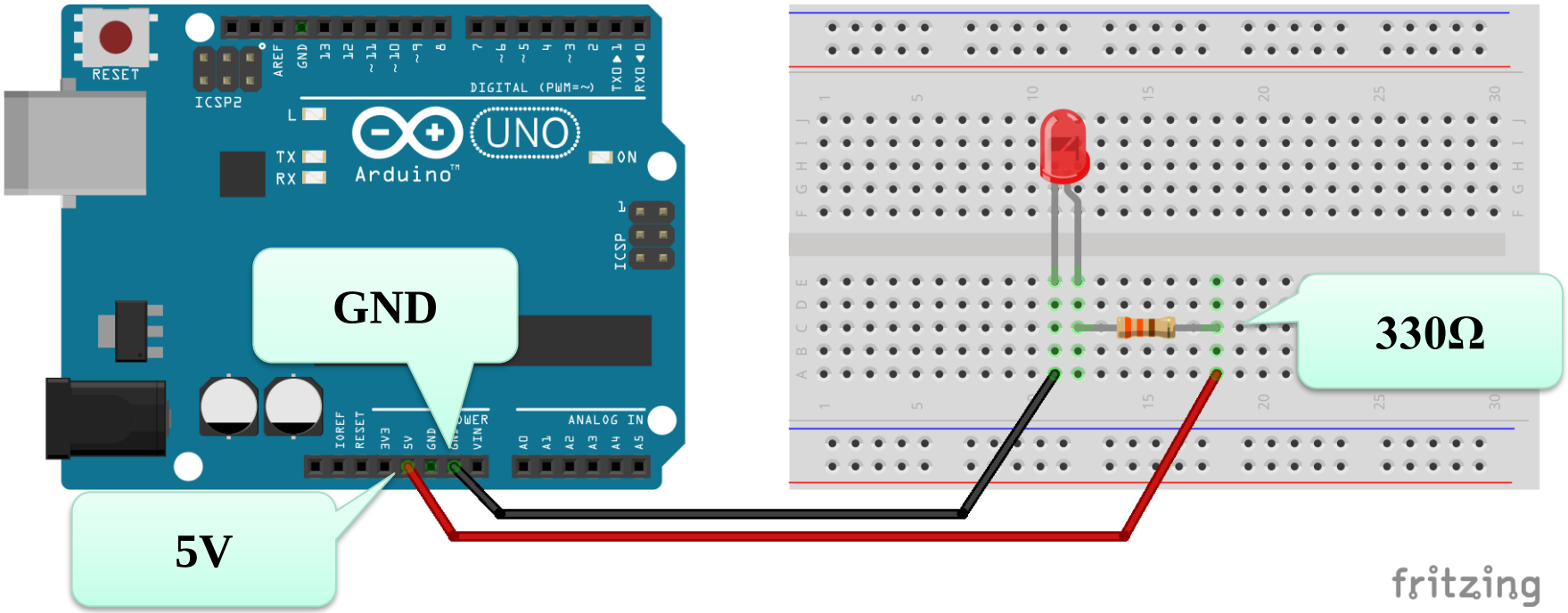
4-7. ブレッドボードの通電箇所



ブレッドボードの通電箇所（続き）



4-8. Arduinoにおける回路設計



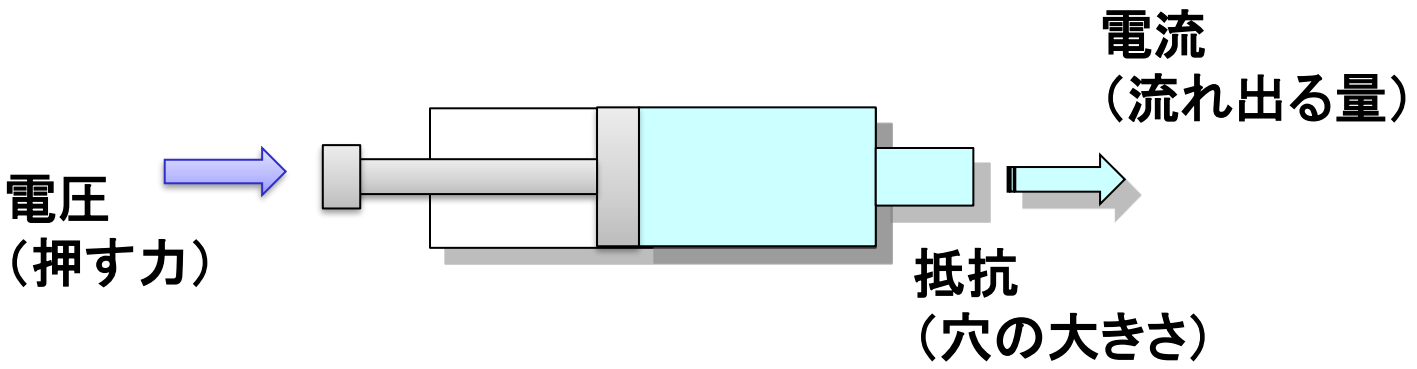
電圧～電流～抵抗

電圧（E）
- 電気を押す力（単位：V）

電流（I）
- 流れる電気の量（単位：A）

抵抗（R）
- 電気の出力の穴の大きさ（単位：Ω）

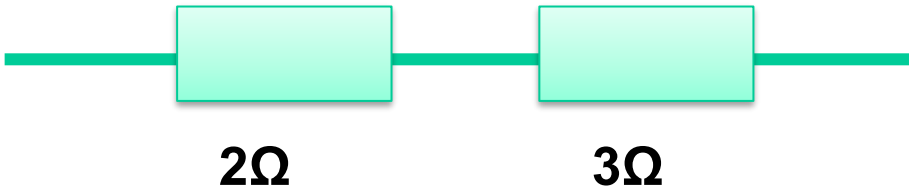
直列回路
電流はどこも同じ値
電圧の和＝全体の電圧
並列回路
電流の和＝全体の電流
電圧はどこも同じ値



オームの法則

電圧（V）＝電流（A）×抵抗（Ω）

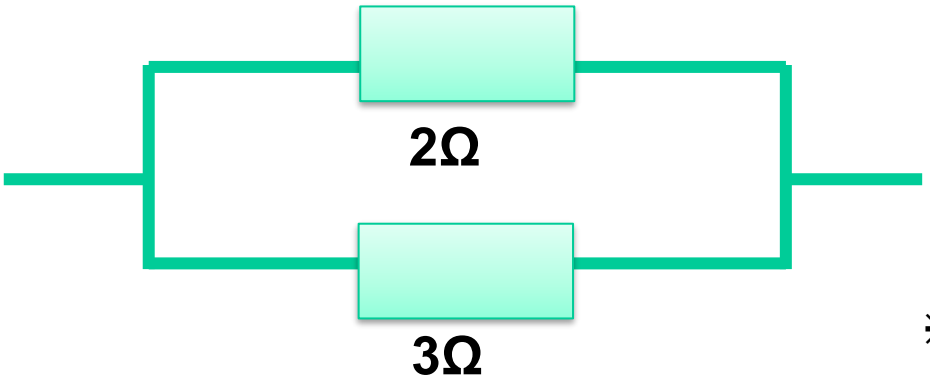
並列接続・・・和



2Ω＋3Ω = 5Ω

一本道が長くなって渋滞するイメージ

並列接続・・・和分の積（or公式）



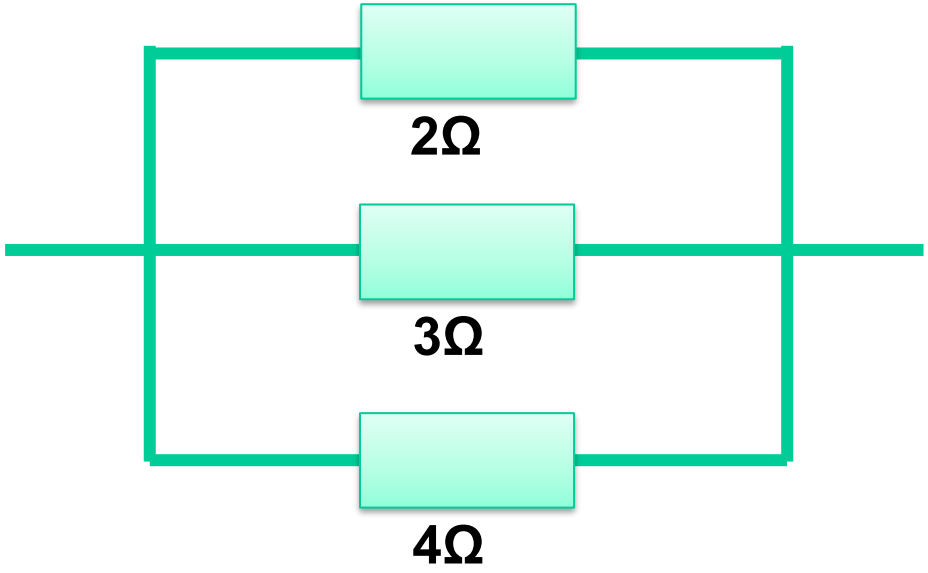
$$\frac{2\Omega \times 3\Omega}{2\Omega + 3\Omega} = \frac{6\Omega}{5\Omega} = 1.2\Omega$$

一本道が二本道に増えるイメージ

※3つ以上の並列がある場合は2つずつ順番に計算

並列接続における和分の積

並列接続・・・３つ以上を和分の積で計算するのは間違い

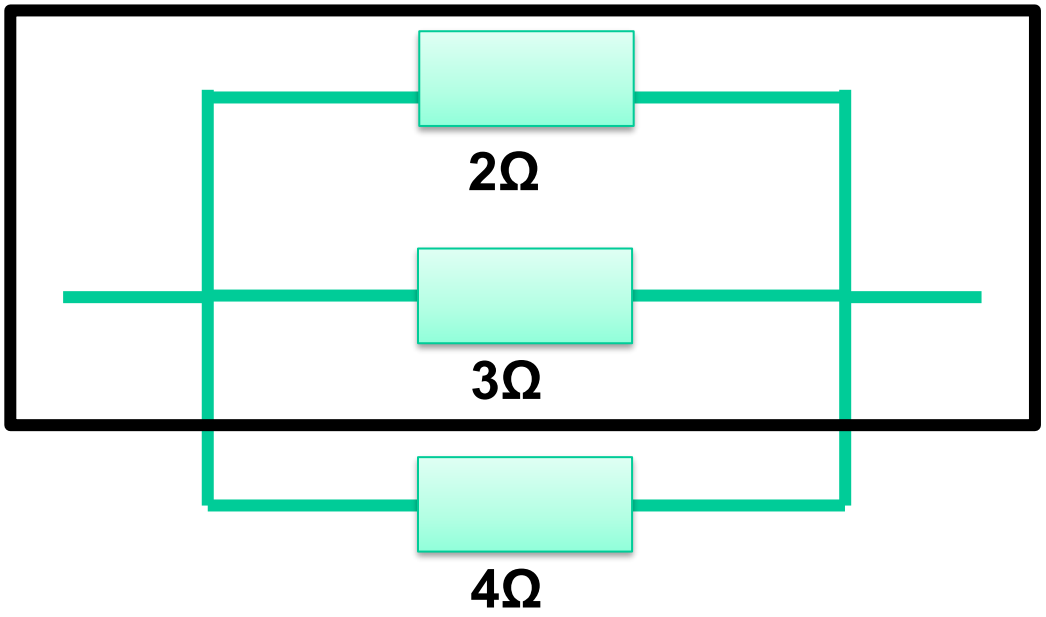


間違い

$$\frac{2\Omega \times 3\Omega \times 4\Omega}{2\Omega + 3\Omega + 4\Omega}$$

並列接続における和分の積（続き）

並列接続・・・まず一部分を和分の積で計算する



$$\frac{2\Omega \times 3\Omega}{2\Omega + 3\Omega} = \frac{6\Omega}{5\Omega} = 1.2\Omega$$

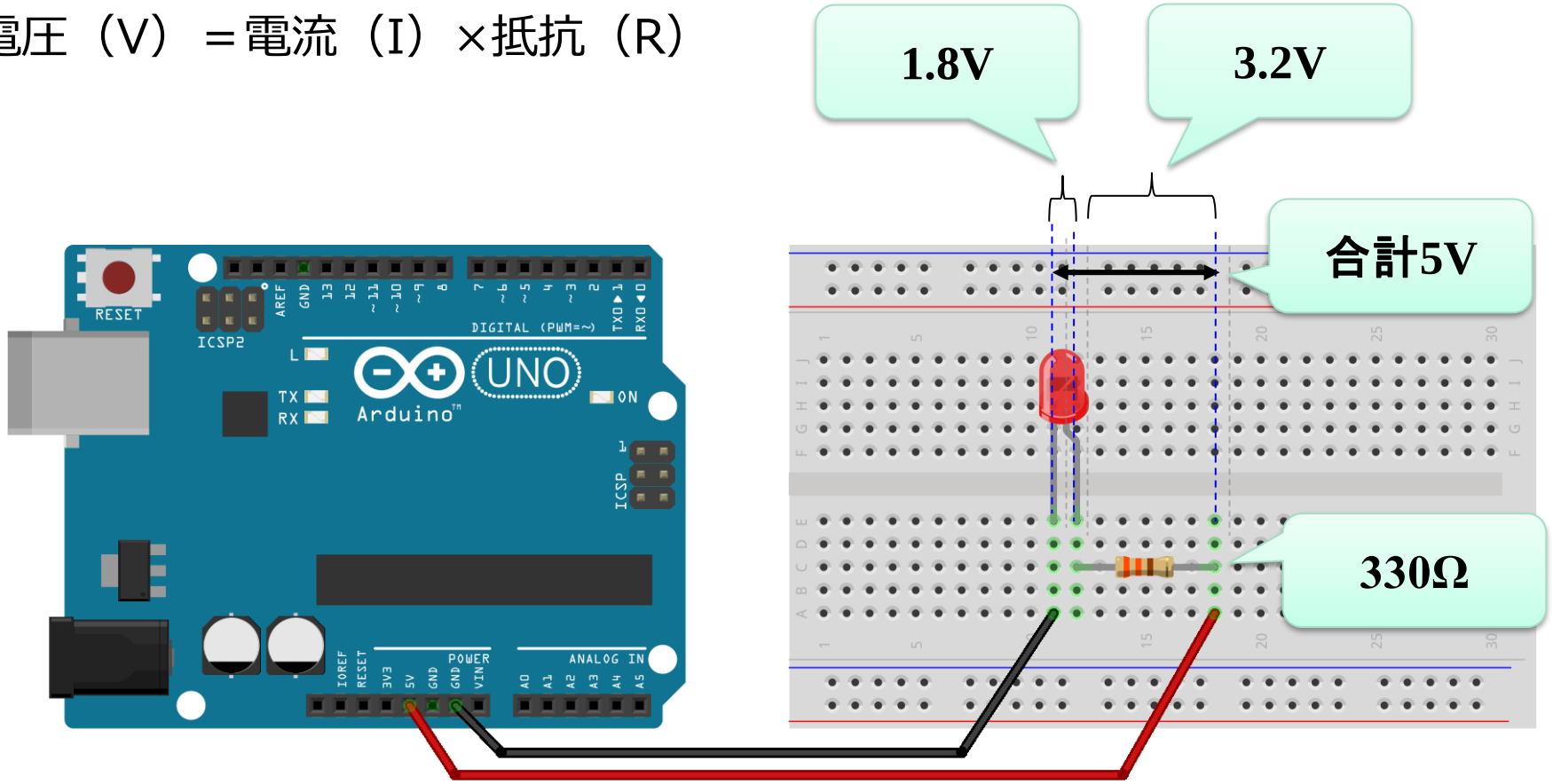
並列接続における和分の積（続き）

並列接続・・・残りの部分を２回目の和分の積で計算する

$$\frac{1.2\Omega \times 4\Omega}{1.2\Omega + 4\Omega} = \frac{4.8\Omega}{5.2\Omega} = 0.92\Omega$$

4-9. Arduinoにおけるオームの法則

電圧（V）＝電流（I）×抵抗（R）



赤色LEDの最大順電流は10～30mA前後
（ここでは10mAを流すものとする）

演習１　Arduinoを使った電気回路の設計

① LEDが点灯する回路
② スイッチでLEDをON・OFFする回路
③ スイッチを押したときにLEDをONする回路とプログラム
④ アナログ出力によるLED点灯
⑤ 応用編：LEDの種類や個数を変更
※作成した回路の回路図を描き抵抗値を書き込む

第５章 組込ボードとセンサ

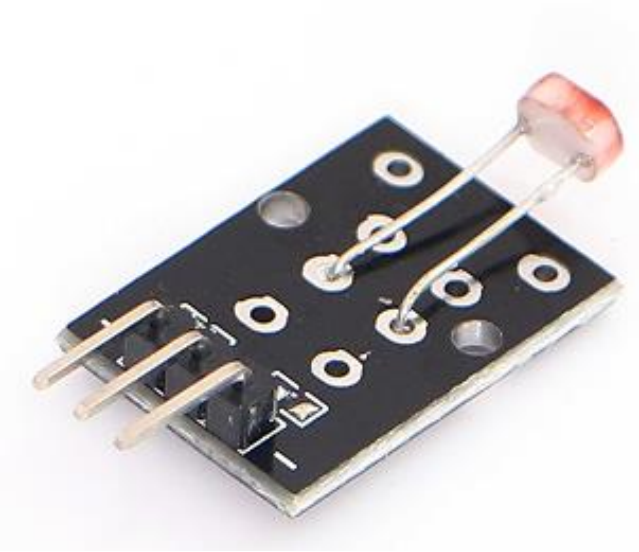
5-1. センサ

環境センサ
入力モジュール
出力モジュール

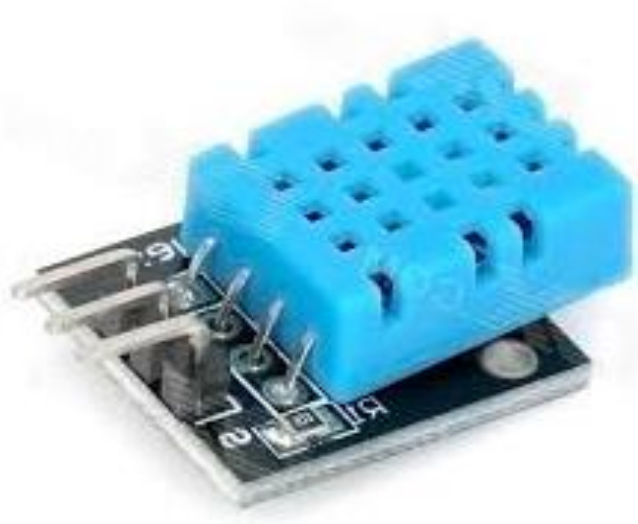
※次スライド以降はKumanのデータファイルよりの抜粋
※Kumanのマニュアルは付属のCD-ROM内にある

5-2. 環境センサ

光センサ

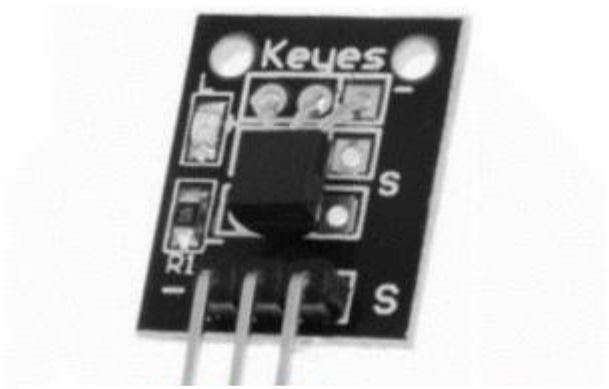


温湿度センサ



環境センサ（続き）

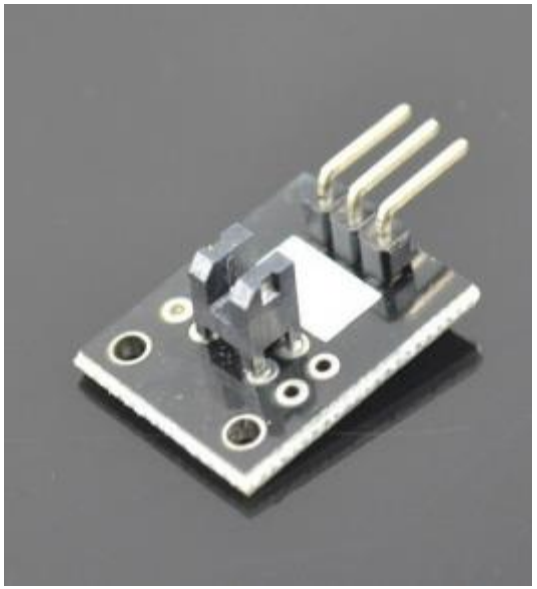
温度センサ



磁場センサ

環境センサ（続き）

光遮断センサ



その他

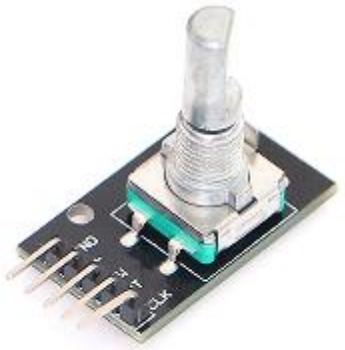
- アナログ磁場センサ
- アナログ温度センサ
- 地磁気センサ
- 超音波センサ
- 赤外線センサ

など

5-3. 入力モジュール

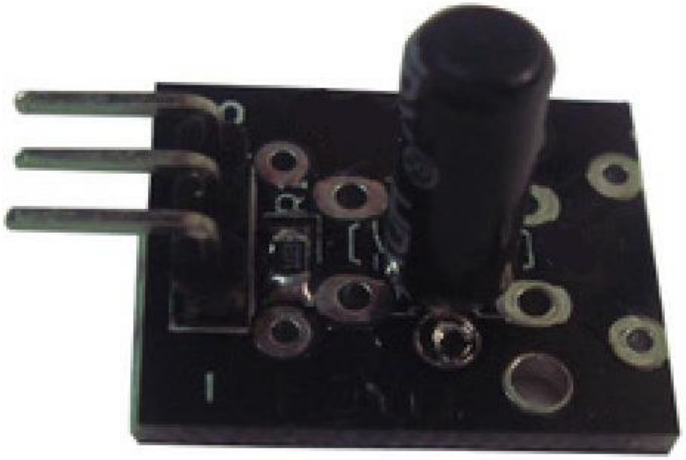
ジョイスティック

ロータリーエンコーダ

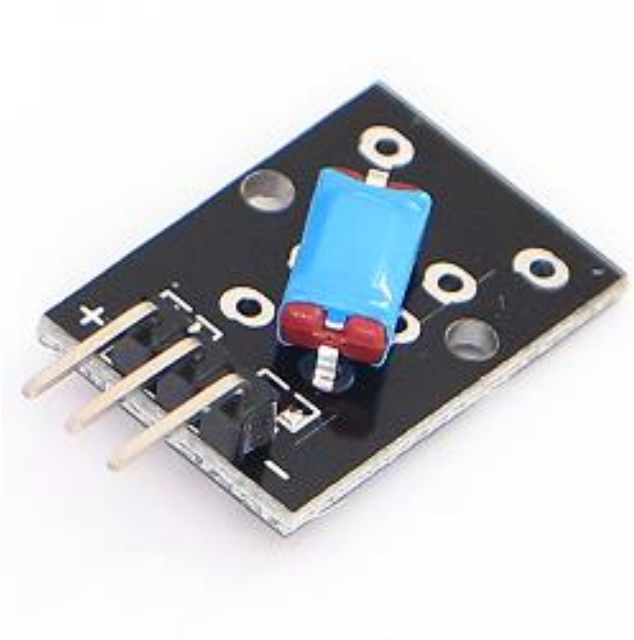


入力モジュール（続き）

衝撃センサ

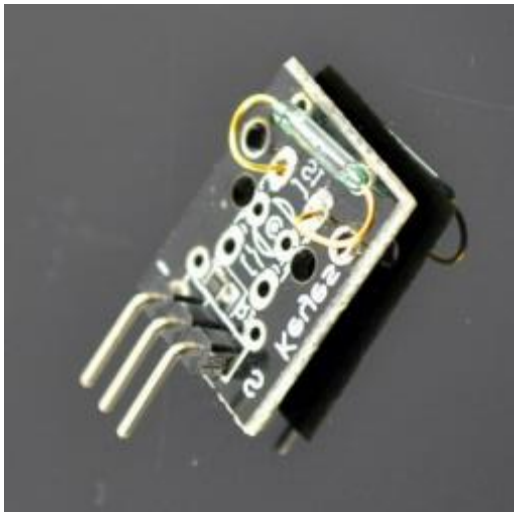


傾斜スイッチ



入力モジュール（続き）

リードスイッチ



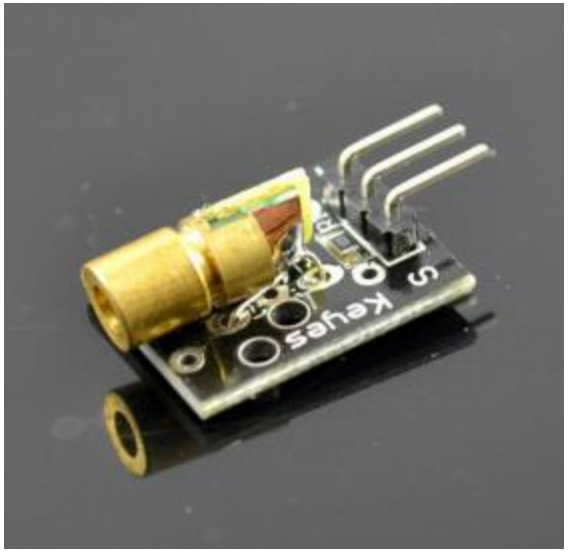
その他

- ボタン
- タッチセンサ
- 水センサ

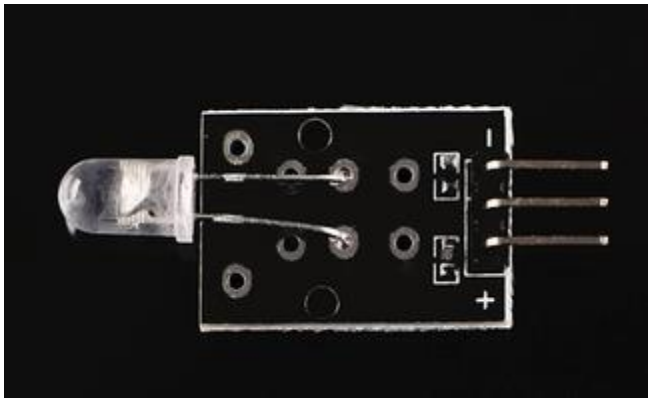
など

5-4. 出力モジュール

レーザ

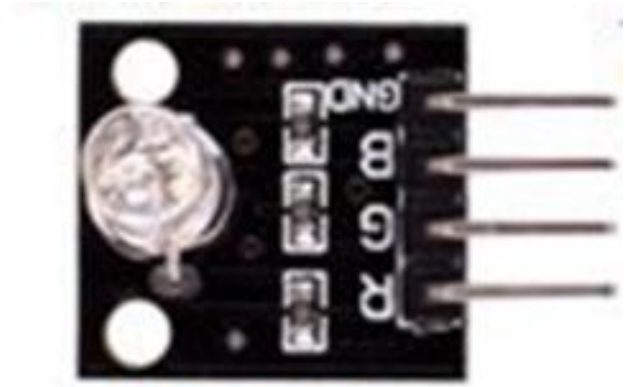


7色LED



出力モジュール（続き）

RGB LED



その他

- LCD
- サーボ
- モータ

など

演習２　Arduinoとセンサを使った回路設計

環境センサ

①　光センサの利用

②　光センサによるLED点灯

入力モジュール

③　傾斜スイッチの利用

④　ロータリーエンコーダによるLED点灯

出力モジュール

⑤　LCD出力

⑥　応用編：各自で色々なセンサやモジュールを組合せて利用

第６章　IoTのセキュリティ

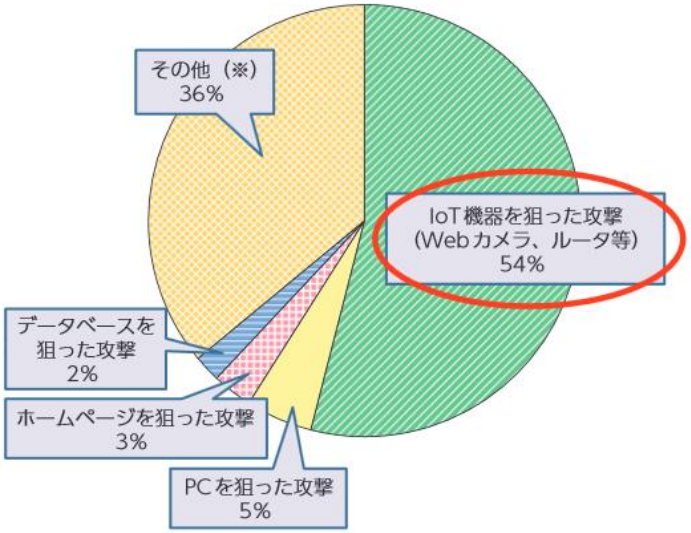
6-1. IoTデバイスを標的としたマルウェア

IoTデバイスの普及に伴って、Miraiウィルスのようなを標的としたマルウェアが流行

図表6-5-2-2 NICTERによる観測結果

観測された全サイバー攻撃1,504億パケットのうち、

半数以上がIoTを狙っている！



※IoT機器特有のポートを狙った攻撃から、特定のIoT機器の脆弱性を狙ったより高度な攻撃も観測されるようになっており、単純にポート番号だけから分類することが難しいIoT機器を狙った攻撃が「その他」に含まれている。

NICT（国立研究開発法人情報通信研究機構）が運用するサイバー攻撃観測網（NICTER）が平成29年に観測したサイバー攻撃パケット、1,504億パケットのうち、半数以上がIoT機器を狙ったものであるという結果が示されている

情報通信白書平成30年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/pdf/30honpen.pdf

6-2. Miraiウィルス

Miraiウィルス

- IoTデバイスに感染しボットネットを作るマルウェア
- ボットネットから攻撃目標に対してDDoS攻撃を行う
- 2016年にMiraiウィルスのボットネットが発見される
- プロバイダやIT企業，ジャーナリストなどへの大規模かつ破壊的な攻撃が観測された

Miraiウィルスの挙動

- 対象外を除いてIPアドレスをランダムに走査
- 脆弱性のある機器を調査（工場出荷時・デフォルト状態、辞書攻撃など）
- 感染したデバイスはC&Cサーバ（指令＆制御）から遠隔操作
- DDoS攻撃を実行（UDPフラット攻撃、HTTPフラット攻撃、DNSフラット攻撃）
- 増殖を繰り返し感染を拡大

Miraiウィルス（続き）

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、スパイウェア、ボット、ランサムウェアなどの悪意のあるソフトウェアのこと

総合的な名称としてマルウェア（Malware）と呼ぶ

ボット

感染したコンピュータを外部から遠隔操作し不正アクセスの手足として利用し、迷惑メールの送信や特定サイトへの攻撃などを行うプログラム

ボットネット

ボットに感染したコンピュータからなるネットワークはボットネットと呼ばれる

ボットネットのコンピュータは特定サイトの一斉攻撃（DDos攻撃）などに利用される

Miraiウィルス（続き）

DDoS攻撃

Distributed Denial of Service攻撃の略

ウィルスに感染して遠隔操作可能な複数の端末から一斉にDoS攻撃（サービス拒否攻撃）を行う

UDPフラット攻撃、HTTPフラット攻撃、DNSフラット攻撃など、通信プロトコルの手続きのパケットを一斉に大量に送りつけることで、相手が処理しきれなくなりサービスが停止してしまう

6-3. IoTセキュリティガイドライン

経済産業省及び総務省が「IoT推進コンソーシアム　IoTセキュリティワーキンググルループ」を開催

- IoTを活用した革新的なビジネスモデルを創出
- 国民が安全で安心して暮らせる社会を実現
- 必要な取組等について検討

「IoTセキュリティガイドライン ver1.0」が策定（平成28年7月5日）

https://www.meti.go.jp/press/2016/07/20160705002/20160705002.html

6-4. IoTセキュリティガイドラインの目的

本ガイドラインの目的は、IoT特有の性質とセキュリティ対策の必要性を踏まえて、IoT機器やシステム、サービスについて、その関係者が**セキュリティ確保の観点から求められる基本的な取組を、セキュリティ・バイ・デザインを基本原則としつつ、明確化すること**によって、産業界による積極的な開発等の取組を促すとともに、利用者が安心してIoT機器やシステム、サービスを利用できる環境を生み出すことにつなげるもの。

なお、本ガイドラインの目的は、サイバー攻撃などによる被害発生時における関係者間の法的責任の所在を一律に明らかにすることではなく、むしろ**関係者が取り組むべきIoTのセキュリティ対策の認識を促す**とともに、その認識のもと、関係者間の相互の情報共有を促すための材料を提供することである。

本ガイドラインは、その対象者に対し、一律に具体的なセキュリティ対策の実施を求めるものではなく、**守るべきものやリスクの大きさ等を踏まえ、役割・立場に応じて適切なセキュリティ対策の検討**が行われることを期待する

IoTセキュリティガイドラインver1.0
https://www.meti.go.jp/press/2016/07/20160705002/20160705002-1.pdf

6-5. サービス提供者のための指針

	指針	主な要点
方針	IoTの性質を考慮した基本方針を定める	•経営者がIoTセキュリティにコミットする •内部不正やミスに備える
分析	IoTのリスクを認識する	•守るべきものを特定する •つながることによるリスクを想定する
設計	守るべきものを守る設計を考える	•つながる相手に迷惑をかけない設計をする •不特定の相手とつなげられても安全安心を確保できる設計をする •安全安心を実現する設計の評価・検証を行う
構築・接続	ネットワーク上での対策を考える	•機能及び用途に応じて適切にネットワーク接続する •初期設定に留意する •認証機能を導入する
運用・保守	安全安心な状態を維持し、情報発信・共有を行う	•出荷・リリース後も安全安心な状態を維持する •出荷・リリース後もIoTリスクを把握し、関係者に守ってもらいたいことを伝える •IoTシステム・サービスにおける関係者の役割を認識する •脆弱な機器を把握し、適切に注意喚起を行う

6-6. 一般利用者のための指針

- 問合せ窓口やサポートがない機器やサービスの購入・利用を控える
- 初期設定に気をつける
- 使用しなくなった機器については電源を切る
- 機器を手放す時はデータを消す

IoTセキュリティガイドラインver1.0
https://www.meti.go.jp/press/2016/07/20160705002/20160705002-1.pdf

第７章　IoTプラットフォームを使ったデータ通信

7-1. IoTプラットフォームの例

sakura.io

IIJ IoT

IoTプラットフォームの例（続き）

AWS IoT

IoT とは

IoT (モノのインターネット) は生産性と効率性を高めるために、物理的な世界をインターネットと接続してデバイスからのデータを活用します。広範な接続オプションがあり、接続コストが下がっているうえに、より多くのデバイスがデータを取得できるため、様々なモノをインターネットに接続することが可能です。IoT アプリケーションが使用されているのは、冷蔵庫、監視カメラ、ケーブルテレビのセットトップボックスのような消費者製品、コンベアベルトや製造装置のような商業システム、信号機やスマートメーターのような商用デバイスなど多岐に渡ります。電源をオンにできるすべてのデバイスが IoT アプリケーションの一部なのです。

AWS IoT の紹介 (日本語字幕) (10:07)

消費者は自宅にあるすべてのモノ、運転するすべてのモノ、着用するモノにまで IoT の

SORACOM IoT

7-2. IoTプラットフォーム sakura.io

https://sakura.io/

7-3. sakura.ioの特徴

低価格＆セキュア（閉域網を使用）
クラウド連携可能
最低月額料金 64円（税込み）

「電気信号」と「JSONデータ」の相互変換装置として動作し、これまでのIoTデバイス/サービス開発の
中間層を補完することで、モノ/サービスづくり・連携に注力が可能です。

7-4. さくらのLTE通信モジュール

sakura.ioモジュール(LTE)
SCM-LTE-01

sakura.io シールド for Arduino
SCO-ARD-01

ブレイクアウトボード(検証ボード)
SCO-BB-01

sakura.io HAT for Raspberry Pi
SCO-RPI-01

IchigoSoda/IchigoJam for sakura.io
SCO-ICG-02

NEW

sakura.io評価ボード
SCO-EVB-01

sakura.io Webサイト
https://sakura.io/product/

さくらのLTE通信モジュール（続き）

sakura.io Webサイト
https://sakura.io/product/

さくらのLTE通信モジュール（続き）

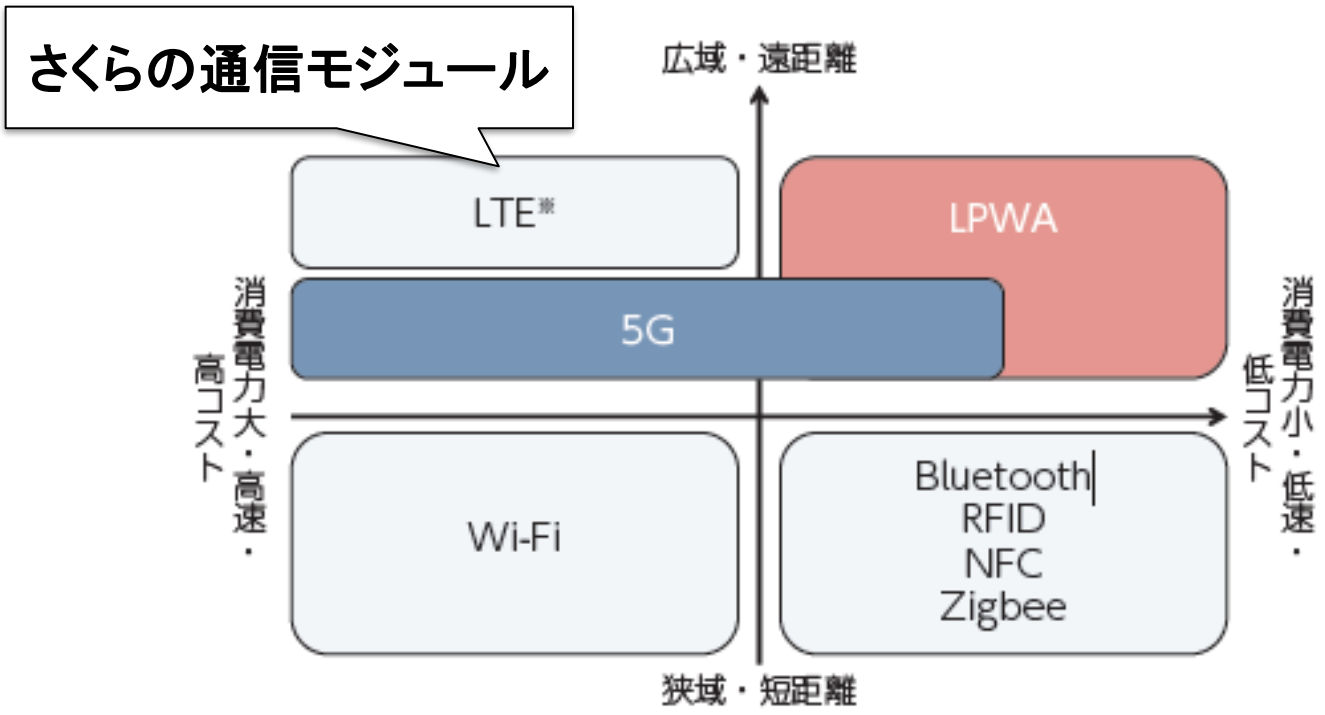
sakura.ioモジュール

　LTE通信モジュール、LTEカテゴリー１（低速、小消費電力、IoT向き）

特徴（製品データシートより抜粋）

・sakura.ioにLTE網を通じてダイレクトに接続するため，ゲートウェイ装置がいらない
・コマンドのみでデータの送受信ができ，ホストMCU側で通信プロトコルを実装する必要がない
・ホストMCUインタフェースはI2C，SPI，およびUARTから選択可能
・小型モジュール（46W×34D×3H）内にLTEモデムやSIMなど必要な機能をすべて内蔵
・待ち受け時の消費電力が低い
・日本国内工事設計認証および電気通信端末機器認証済み

7-5. さくらの通信モジュールの位置付け

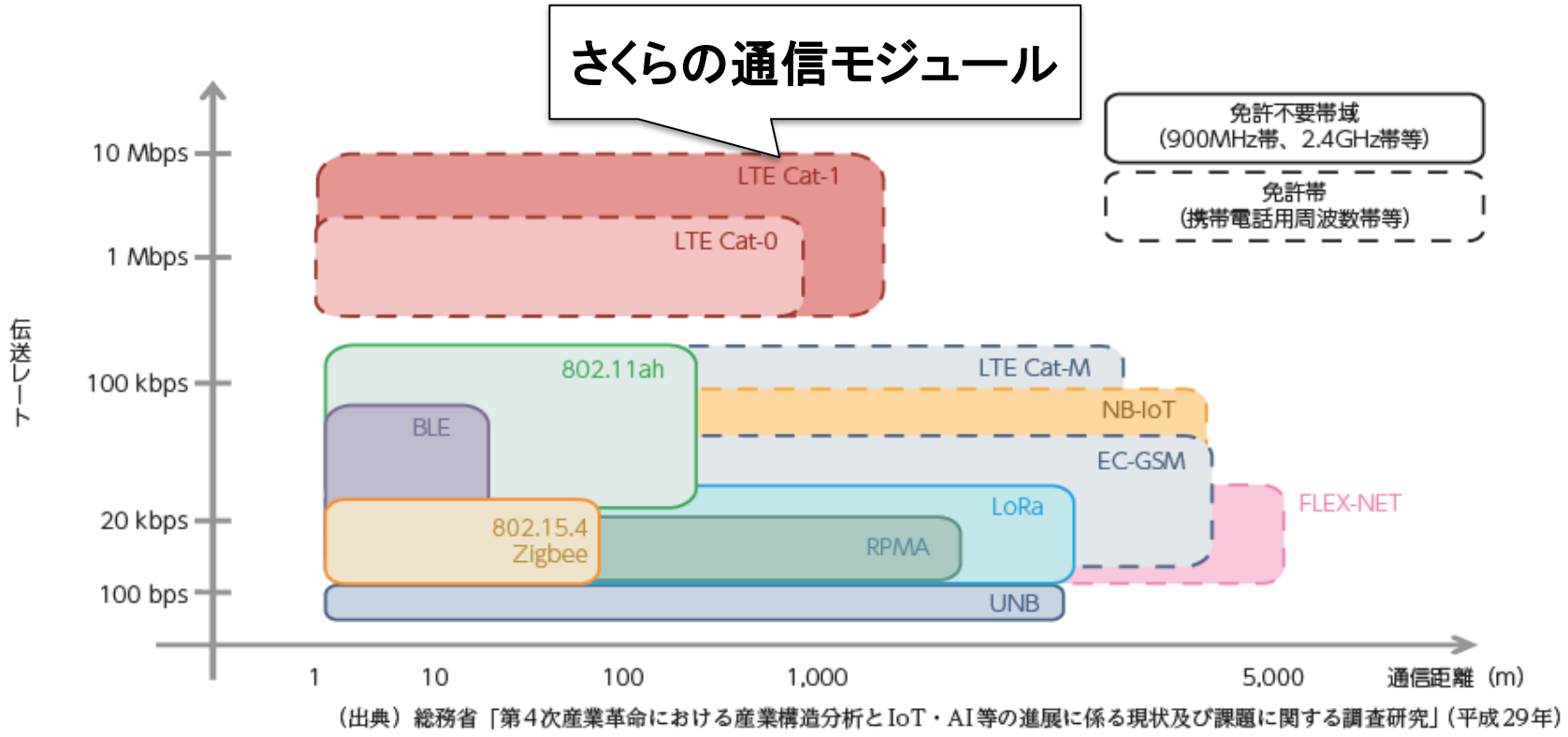


※既存のM2M接続は2G、3G、4Gが主流

（出典）総務省「第４次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

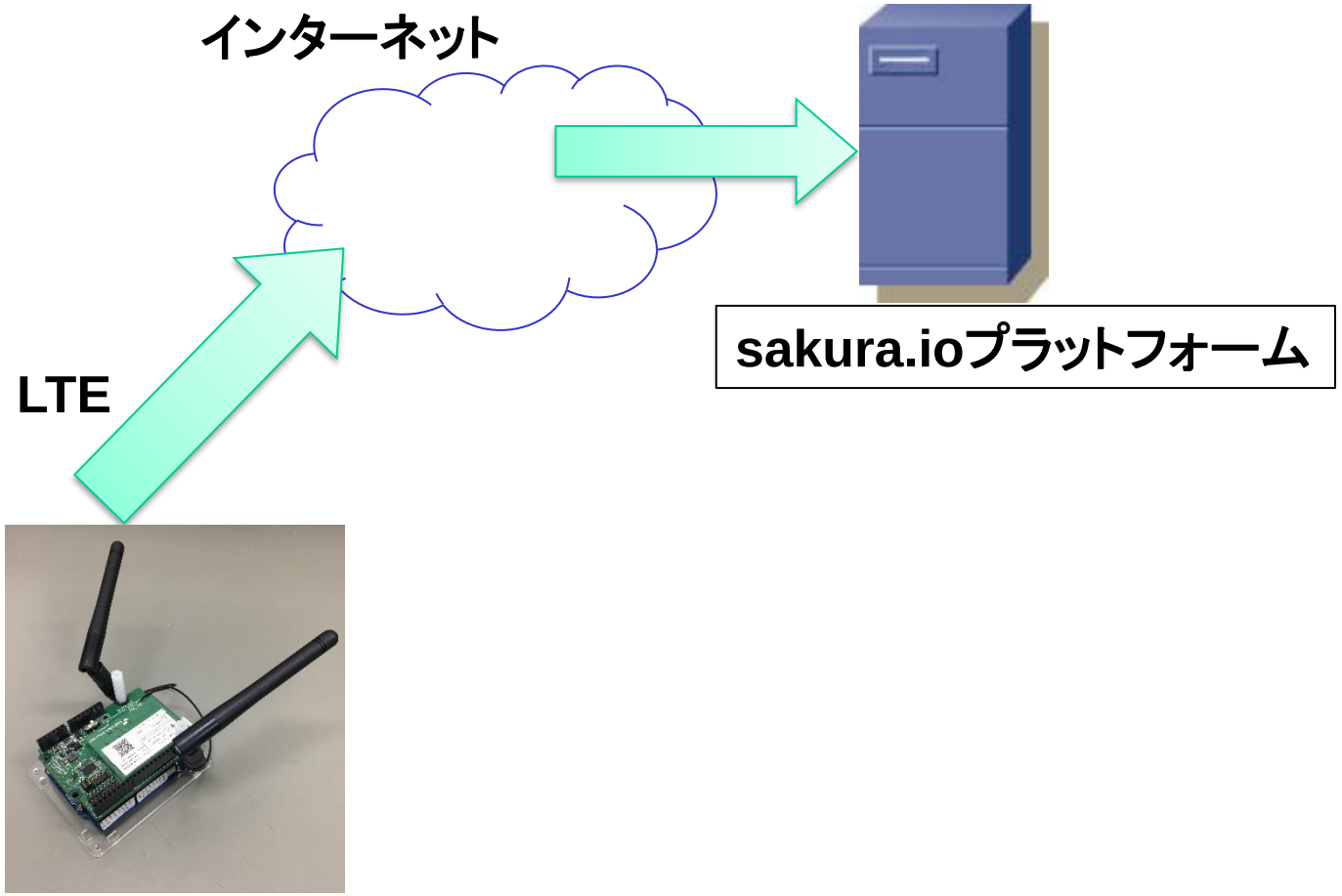
さくらの通信モジュールの位置付け（続き）



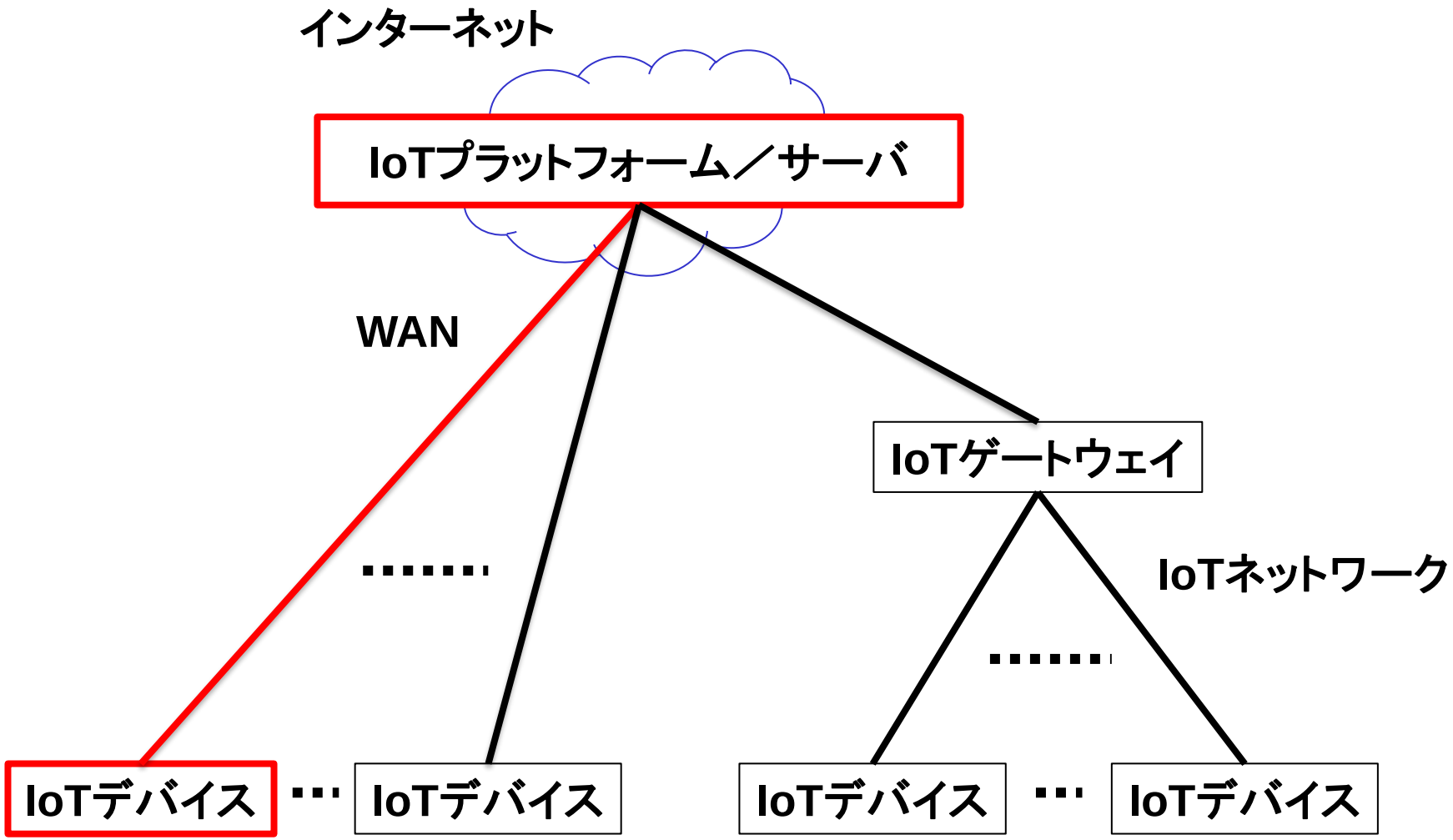
（出典）総務省「第４次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

7-6. sakura.ioの物理的構成



7-7. IoTシステムの物理的構成



7-3. さくらのIoT Platformの特徴

低価格＆セキュア（閉域網を使用）
クラウド連携可能
最低月額料金 64円（税込み）

「電気信号」と「JSONデータ」の相互変換装置として動作し、これまでのIoTデバイス/サービス開発の
中間層を補完することで、モノ/サービスづくり・連携に注力が可能です。

7-8. sakura.io 料金と通信ポイント

１ヶ月につき通信ポイントが10,000pt付与

100回の通信ごとに100pt消費（100回未満は切り上げ）

別途購入する場合は20,000pt/100円

都度消費ではなく、月末に通信回数によってポイント引き落とし。不足すればその分を精算

10,000pt = 10,000回の通信

5分に1度の通信 → 1時間で12回 → 1日 288回

→ 30日で8,640回

5分に1度の通信でも充分。データを貯めて定期的に送信することも可能

7-9. ポイント管理例

ポイント管理

現在のポイント　39,800pt　2018年7月末期限ポイント

10,000pt

有効期限	有償ポイント	無償ポイント
2018-07-31	0pt	10,000pt
2018-08-31	0pt	10,000pt
2018-09-30	0pt	10,000pt
2019-07-31	0pt	9,800pt

履歴

日付	区分	ポイント増減
2017-07-31	2017年7月 モジュール通信(200ポイント)	-200pt
2017-07-31	2017年8月 付与分	+10,000pt

7-10. ライブラリとマニュアル

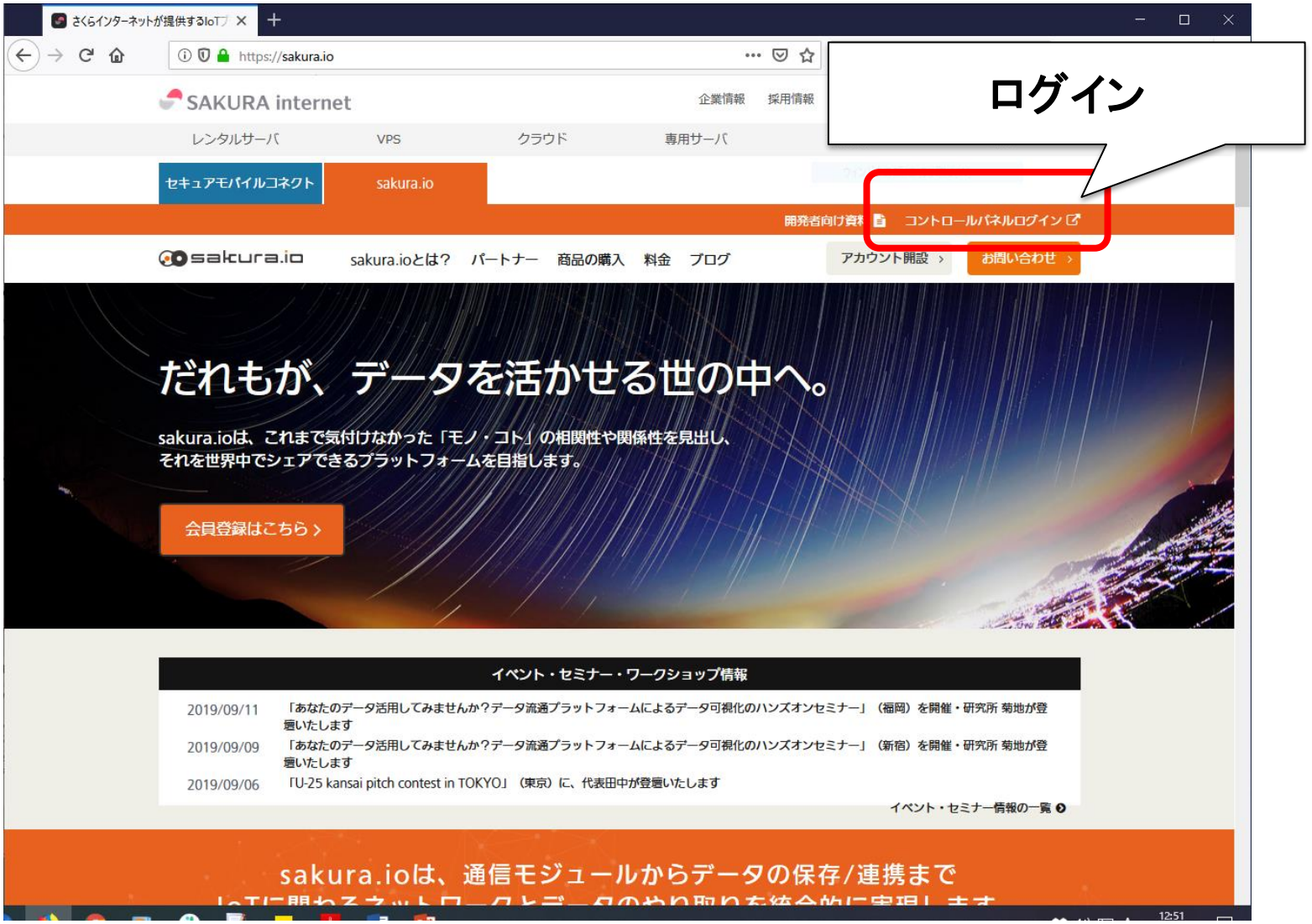
ライブラリ

https://github.com/sakuraio/SakuraIOArduino

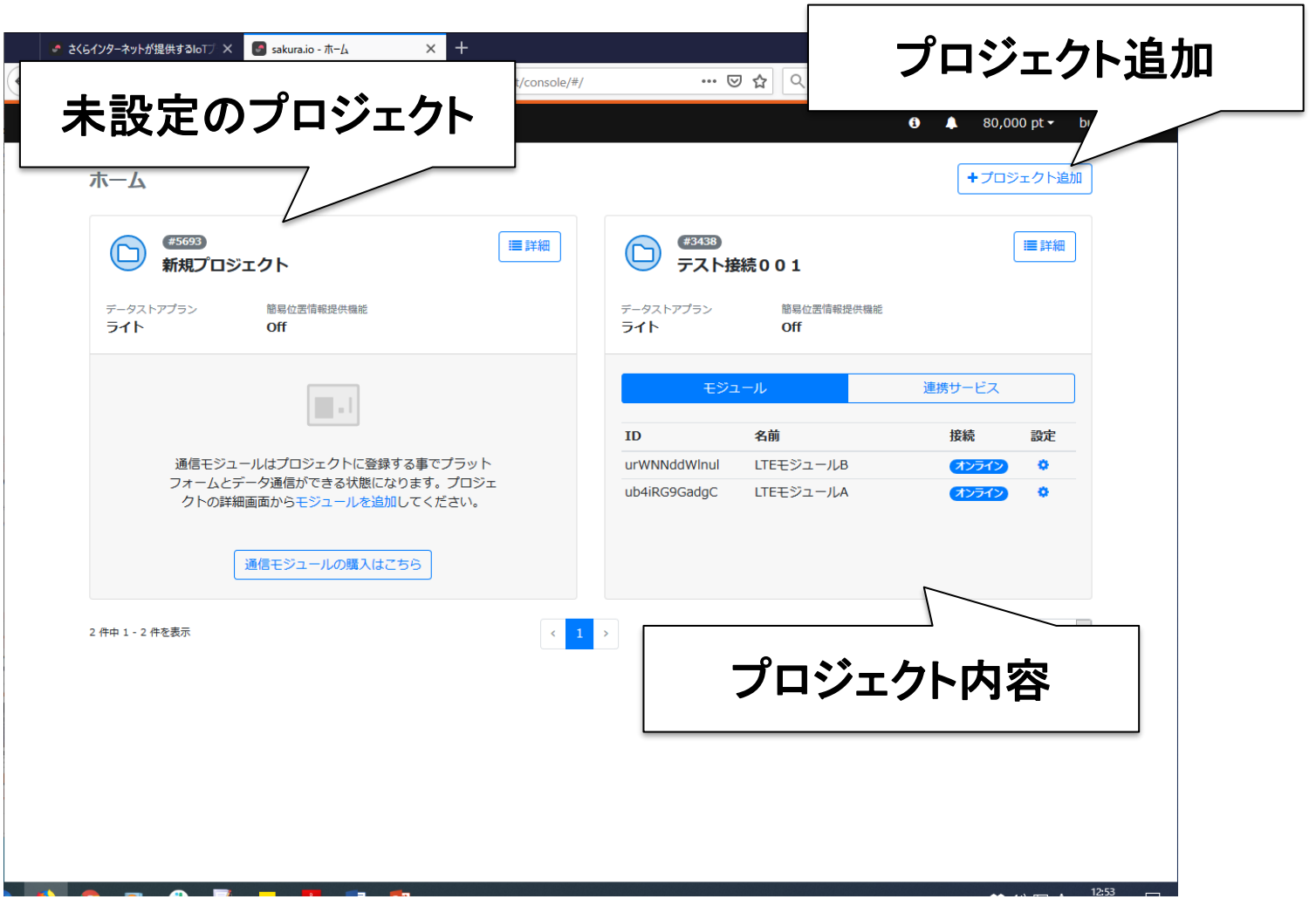
マニュアル

https://sakura.io/docs/index.html

7-11. ログインとプロジェクト

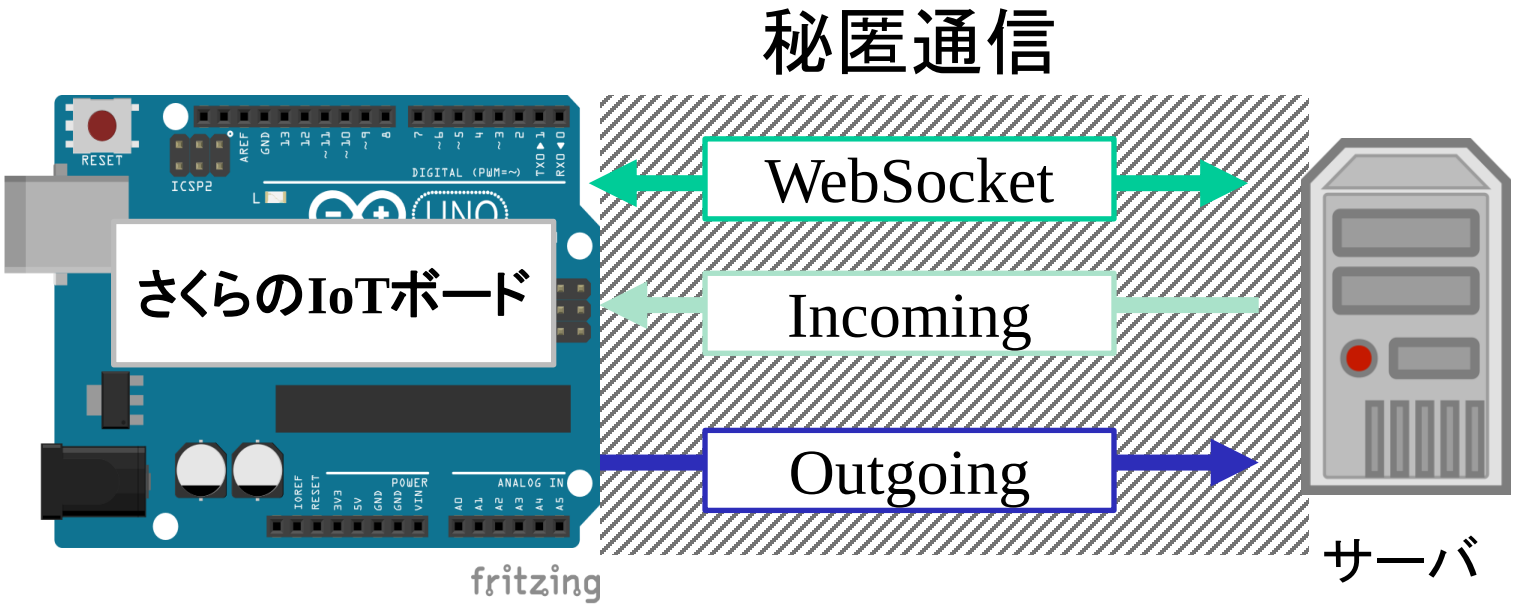


ログインとプロジェクト



7-12. 基本的な考え方

さくらIoTのライブラリを通じてデータの送受信を行う
Arduino側にTCP/IPスタックは必要ない



7-13. コード例

接続

```
sakuraio.getConnectionStatus()
```

データ送信キューに貯める

```
sakuraio.enqueueTx()
```

データ送信

```
sakuraio.send()
```

データ受信

```
sakuraio.getRxQueueLength()
```

ライブラリをインポートし、スケッチ例Standardを実行する

7-14. 連携サービス

WebSocket
　コネクションを維持したままデータ送受信
Outgoing Webhook
　モジュールからデータ送信
Incoming Webhook
　モジュールへデータ送信
MQTT Client
DataStore API
AWS IoT
Azure IoT Hub (α)

本演習ではWebSocketとIncoming Webhookを行う

7-15. WebSocket

従来のhttp等はコネクションレスの通信プロトコル

WebScoketはコネクションを維持したまま通信可能なプロトコル

さくらのIoTで最も簡単に扱える

10秒に1度keepaliveを送信し、コネクションを維持（keepaliveは課金されない）

PHPでWebSocketを扱うのは容易

ただし、コマンドを都度実行したり、Webブラウザで読み込み続ける必要がある

7-16. データ形式

データ形式はすべてJSON
送信できるデータ形式は決まっている
int型変数は、符号あり32bit整数のint32_tのみ
同じく符号無しのint型変数は、uint32_tのみ
floatやdoubleはそのままでよい

参照：
https://sakura.io/docs/pages/platform-specification/message.html

7-17. JSON例（データが単数）

データの例

```
{
    "datetime": "2019-08-19T05:25:19.986646718Z",
    "module": "************",←――― モジュールシリアル
    "payload": {
        "channels": [
            {
                "channel": 0, ←――― 単独でもchannel[0]
                "type": "f", ←――― データの型
                "value": 31.864151,←――― データの値
                “datetime”: “2019-08-19T04:56:20.035365877Z”
            }
        ]
    },
    "type": "channels"
}
```

7-18. JSON例（データが複数）

※payload部分のみ

```
"payload": {
        "channels": [
            {                   データ1
                "channel": 0,
                "type": "f",
                "value": 47,
                "datetime": "2019-08-19T04:56:40.190154948Z"

            },
            {                   データ2
                "channel": 0,
                "type": "f",
                "value": 29
                "datetime": "2019-08-19T04:56:40.190154948Z"
            }
        ]
    },
```

7-19. 連携サービスの作成

sakura.ioにログインし、コントロールパネルから作成

7-20. WebSocketのURLとToken

コントロールパネルで確認可能
外部からアクセスする際は、ここに表示されるURLとTokenが必要

7-21. JSON例（データが単数）

```
{
    "datetime": "2019-08-19T04:56:40.190154948Z",
    "module": "************",
    "payload": {
        "channels": [
            {
                "channel": 0,
                "type": "f",
                "value": 31.864151,
                "datetime": "2019-08-19T04:56:40.190154948Z"
            }
        ],
    },
    "type": "channels"
}
```

data.payload.channels[0].value

7-22. JSON例（データが複数）

```
"payload": {
        "channels": [
            {
                "channel": 0,
                "type": "f",
                "value": 47,
                "datetime": "2019-08-19T04:56:40.190154948Z"
            },
            {
                "channel": 1,
                "type": "f",
                "value": 29,
                "datetime": "2019-08-19T04:56:40.190154948Z"

            }
        ]
    },
```

データ1

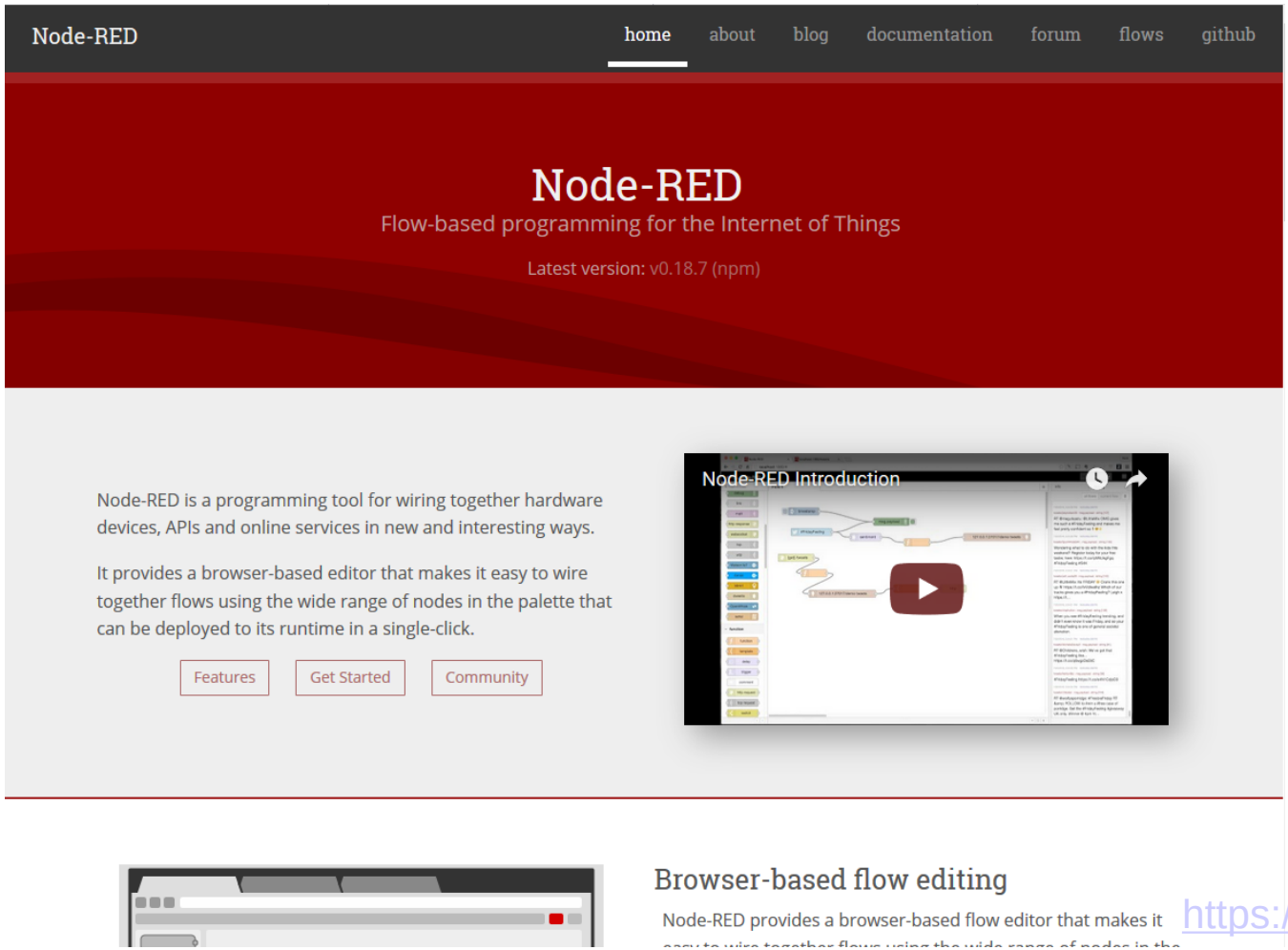
data.payload.channels[0].value

データ2

data.payload.channels[1].value

7-23. 開発ツール Node-RED

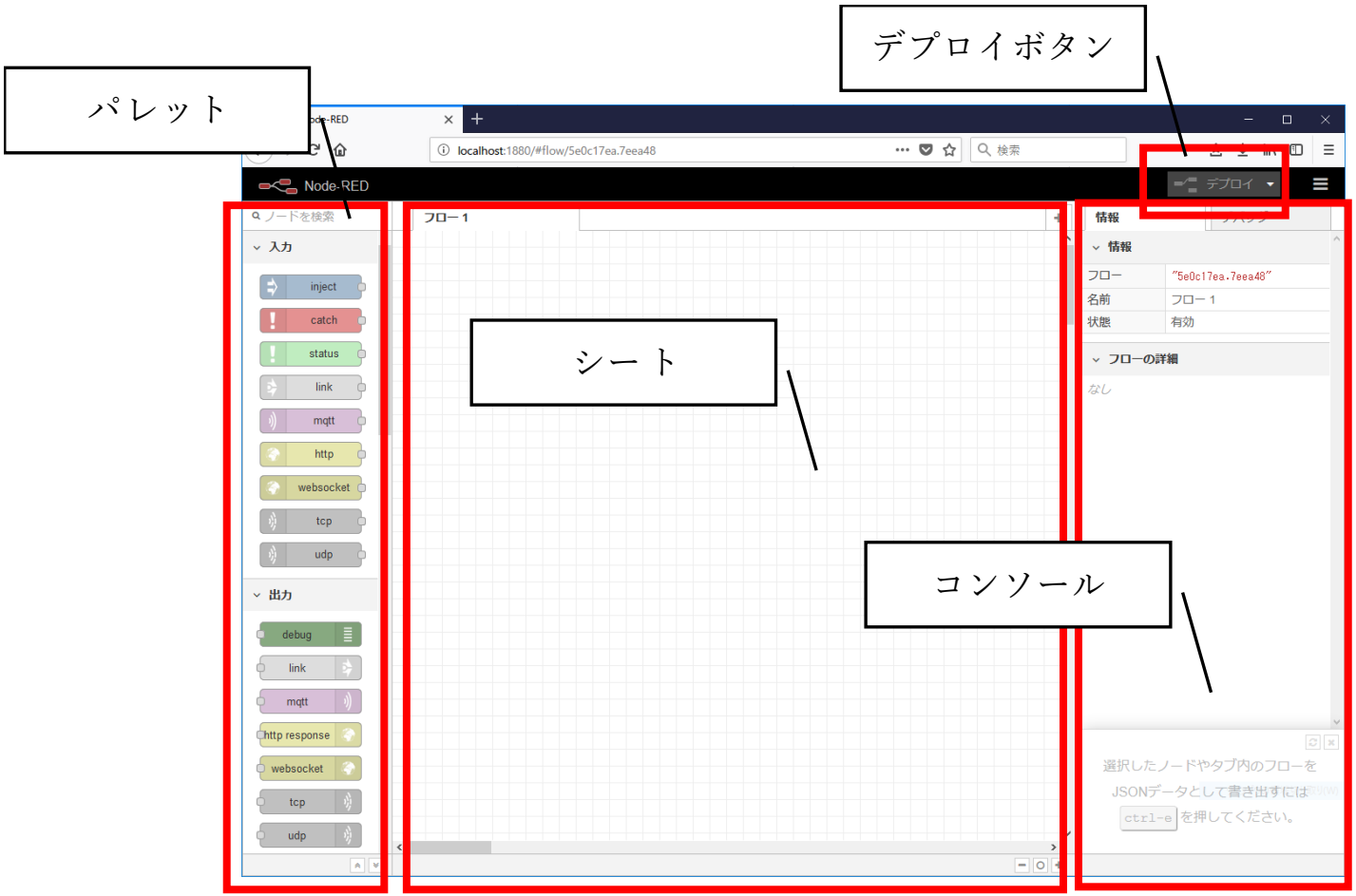
Flowエディタを使って、プラグイン/モジュールであるノードを視覚的に接続しながら、IoTデバイスとオンラインサービスをつなぐことができる開発ツール



Node-RED
https://nodered.org/

開発ツール Node-RED

Flowエディタを使って、プラグイン/モジュールであるノードを視覚的に接続しながら、IoTデバイスとオンラインサービスをつなぐことができる開発ツール



演習３　さくらLTEモジュールの回路設計と利活用

さくらのIoTコントロールパネルで確認

WebSocketをJavaScriptで取得して表示

Node-REDを使ったデータ通信

演習４　総合演習

これまで学んだものに基づいて各自のIoTシステムを構築

【必須】
・IoTデバイスに任意のセンサを利用する
・取得したセンサーの値をsakura.ioにアップロードする

【任意】
A. センサを複数にする／センサにアクチュエータをつける
B. sakura.ioに集めたデータをNode-Redを使って可視化する
C. IoTデバイスへのフィードバック機能を任意につける
D. その他

株式会社ウチダ人材開発センタ
UCHIDA HUMAN DEVELOPMENT CO.,Ltd.

IoT活用

目次

目次（1）

目次（2）

目次（3）

目次（3）

目次（4）

第1章：第1章 IoTの概要

1-1. Internet of Things (IoT)

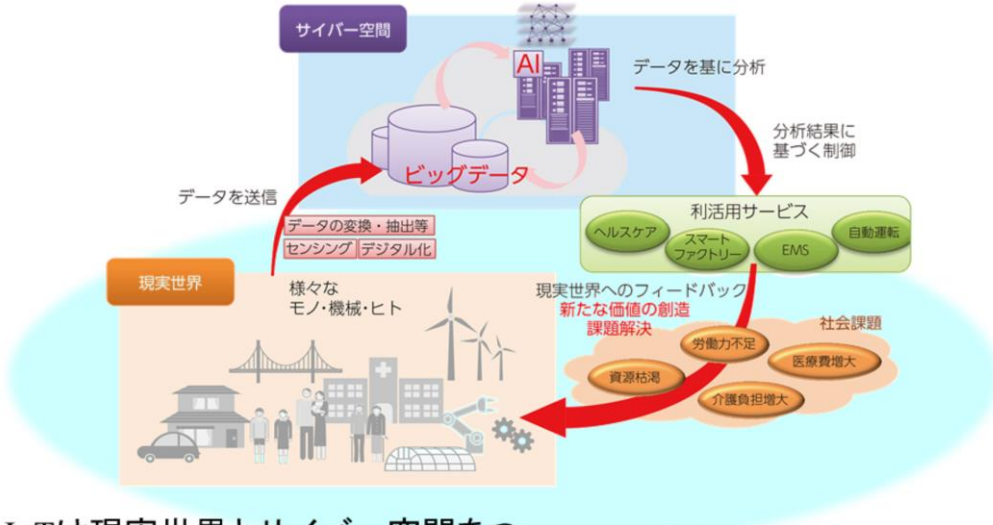
- Internet of Things
- モノのインターネット
- 様々なモノがインターネットにつながり新たなサービスを創造する

- 用語自体は1999年ごろに登場したといたと言われている

  Kevin Ashton（当時MIT）が使用していたInternet for Thingsという用語が起源と言われている
  当時の意味合いでは、赤外線通信がインターネットのように繋がるという意味
- コンセプト自体は1980年代からある

  機器：インターネットに接続された自動販売機
- ユビキタスネットワークの後継とも言える

1-2. ネットワークとデータが創造する新たな価値



IoTは現実世界とサイバー空間をつなぐ

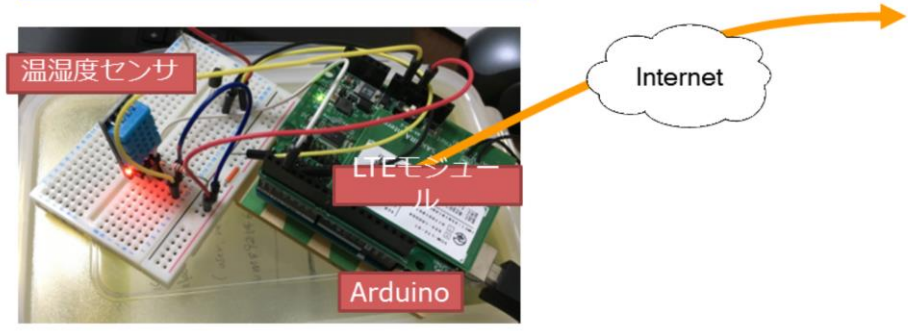
情報通信白書平成28年度版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

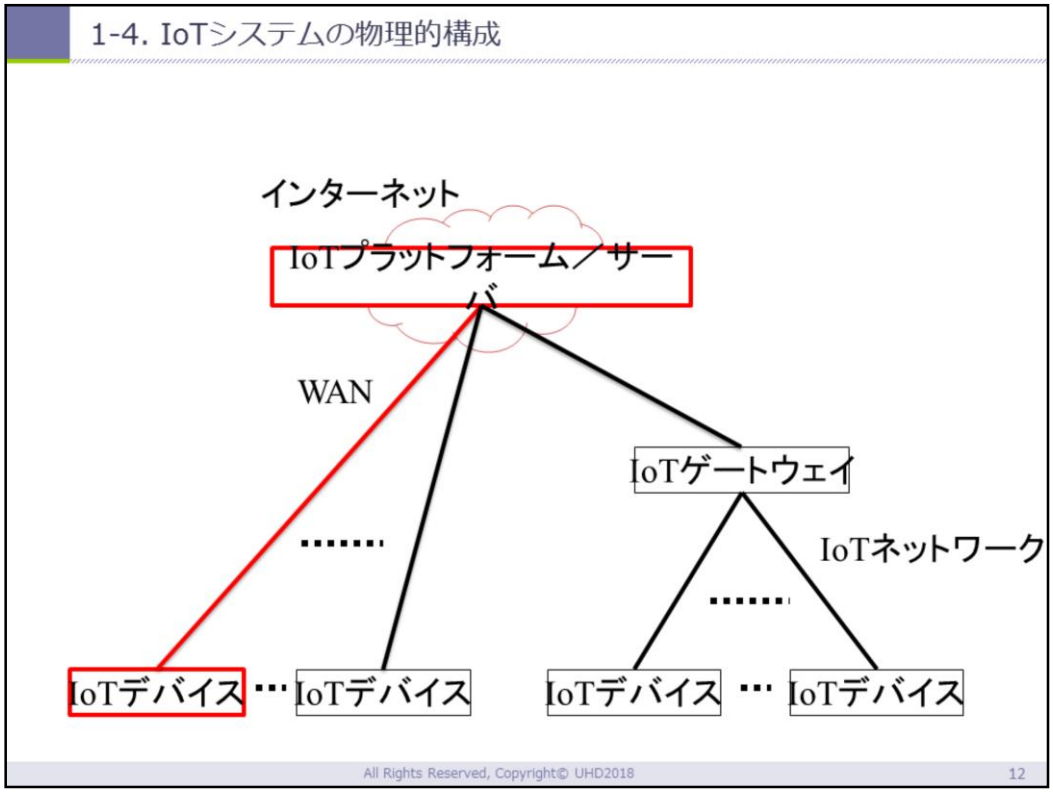
総務省での概要

1-3. シンプルなIoTデバイスの例

- 10分間隔で温度・湿度を計測してサーバへ送信

```
2017-09-14T23:26:28.498910794Z,51,25
2017-09-14T23:36:28.683556193Z,50,26
2017-09-14T23:46:28.845793168Z,48,27
2017-09-14T23:56:29.037074181Z,49,26
```





今回，さくらIoTを使って最終的につくるのは赤色の構成部分になることを説明する。

1-5. Society 5.0

政府広報オンライン 新聞広告(読売新聞2018年1月1日付)
https://www.gov-online.go.jp/cam/s5/

Web先から「そもそも、Society5.0ってなに？」の動画（１分22秒）を見せてイメージをつかんでもらう.

1-6. 経済発展と社会的課題の解決を両立

内閣府　科学技術政策　Society 5.0
http://www8.cao.go.jp/cstp/society5_0/index.html

時間があれば内閣府　科学技術政策　Society5.0のサイトの最下部の事例イラストを見てもらう.

1-7. 第四次産業革命（1）

第一次産業革命
- 18～19世紀初頭
- 蒸気機関、紡績機など（軽工業の機械化）

第二次産業革命
- 19世紀後半
- 石油、電力、重化学工業

第三次産業革命
- 20世紀後半
- インターネットの出現、ICTの急速な普及

第四次産業革命
- 21世紀極端な自動化、コネクティビティによる産業革新
  （ダボス会議UBS白書2016年1月）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

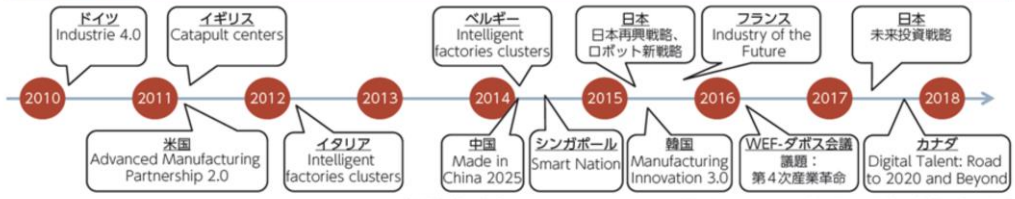
1-7. 第四次産業革命（2）

革命	特徴
第1次産業革命	18世紀後半、蒸気・石炭を動力源とする軽工業中心の経済発展および社会構造の変革。イギリスで蒸気機関が発明され、工場制機械工業が幕開けとなった
第2次産業革命	19世紀後半、電気・石油を新たな動力源とする重工業中心の経済発展および社会構造の変革。エジソンが電球などを発明したことや物流網の発展などが相まって、大量生産、大量輸送、大量消費の時代が到来。フォードのT型自動車は、第2次産業革命を代表する製品の１つといわれる
第3次産業革命	20世紀後半、コンピューターなどの電子技術やロボット技術を活用したマイクロエレクトロニクス革命により、自動化が促進された。日本メーカーのエレクトロニクス製品や自動車産業の発展などが象徴的である
第4次産業革命	2010年代現在、デジタル技術の進展と、あらゆるモノがインターネットとつながるIOTの発展により、限界費用や取引費用の低減が進み、新たな経済発展や社会構造の変革を誘発すると議論される

情報通信白書 平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

1-8. 第四次産業革命の世界的な潮流

図表3-1-3-1 主要国の取組等

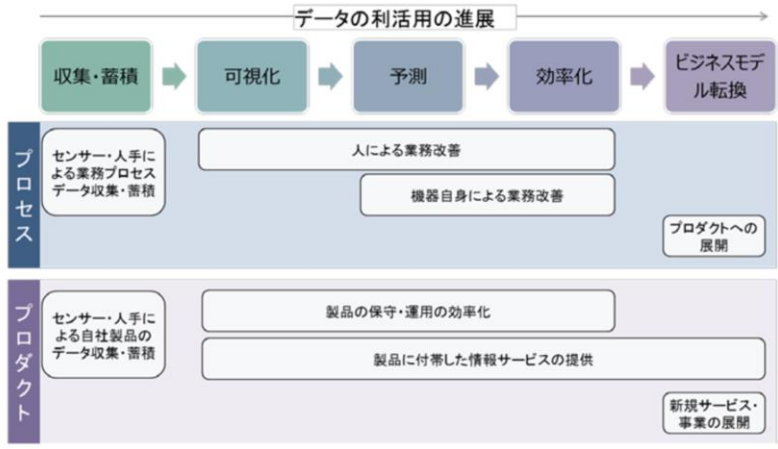


（出典）総務省「ICTによるイノベーションと新たなエコノミー形成に関する調査研究」（平成30年）

情報通信白書平成30年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h30.html

1-9. データの利活用の進展と プロセス・プロダクトにおける進展の対応

図表 3-3-2-3　データの利活用の進展とプロセス・プロダクトにおける進展の対応



IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

IoTで何ができるか

プロセスとプロダクトについては後述の事例で出てくる

1-10. 業種ごとのプロセスのIoT導入事例

プロセス					
業種	事業者	概要	業種	事業者	概要
農林水産・鉱業	JAやつしろ（日本）	ビニールハウス内のセンサから収穫した温度や炭酸ガス量等のデータをリアルタイムに監視し、育成に最適な環境を維持。	流通・小売り	日本郵船	SIMS(Ship Information Management System)の導入により、エンジンの回転数や燃料消費量などの船舶データと天候等の外部データを組み合わせて運行・配船を効率化し、約10%の省エネ効果を達成。
製造業	ボッシュ（ドイツ）	ホンブルク工場において、生産をソフトウェアで管理して電力消費量を効果的に抑制し、エネルギー需要の最適化を図り、ピーク時の負荷を最大で10%引き下げることに成功。	情報通信	Azercell（アゼルバイジャン）	アゼルバイジャンにある450か所の基地局の発電機等の設備のデータをリアルタイムで可視化し管理を効率化。
エネルギー・インフラ	中国電力（日本）	島根原子力発電所2号機のセンサ情報を基に、精度の高い予兆検知を実現。正常な状態を解析・分析し、「いつもと違う」状態に対してはアラームを発報。	サービス業	あきんどスシロー（日本）	皿をつけたICタブによる鮮度管理により、ICタグで何時何分にレーンに流したかを把握し、鮮度管理を徹底。合わせて、タッチパネルにより来店客の人数と大人、子どもの数をすることによるリアルタイムの需要予測を実施。

IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

1-11. 業種ごとのプロダクトのIoT導入事例

プロダクト					
業種	事業者	概要	業種	事業者	概要
農林水産・鉱業	MONSANT（米国）	MONSANTの一部門であるThe Climate corporation(MONSANTが2013年に買収)より、農場経営者に土壌の品質や気象データからのアドバイスや、生産リスク対策用の保険を提供。	流通・小売り	ネスレ（スイス）	自社の業務用コーヒーマシンをネットワークに接続し、稼働情報を収集、遠隔から機器を調節したり、異常発生時にサービスマンへアラートを発行。常に理想的な状態での稼働を実現。
製造業	GE（米国）	ジェットエンジンにセンサーを組み込み、効率的な保守サービスや最適な航路を提案するサービス、及びそれらに利用しているIOTプラットフォームを提供。	情報通信	SORACOM（日本）	IOTに向けの格安MVNOサービス「SORACOM Air」をはじめとした、IOT用通信プラットフォームを提供。
エネルギー・インフラ	東京電力（日本）	自社WEBサイト「でんき家計簿」にてスマートメーターで計測した30分ごとの電気利用料を時間別で可視化するサービスを提供。	サービス業	ウォルトディズニー（米国）	ウォルトディズニーワールド園内で入場券、ホテルの鍵、園内で財布代わりに使用可能な電子マネーなどして使えるウェアラブル端末「MagicBand],およびそれを統合したサービス「MyMagic+」を提供。

IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

1-12. IoTによる製品の高付加価値比の事例1

事例	メーカー	画像	概要
スマートペダル	Connected Cyde (フランス)		GPS機能付きの自転車に取り付けるだけでペダルが自動的にスピードや移動距離、乗車時間、消費カロリーなどを計測する。自転車盗難防止対策機能として万が一盗難があった場合はリアルタイムで居場所を追跡できる。自転車をこぐことで充電されるため、電池は不要である。
スマート傘	DAVEK （米国）		折りたたみ傘の中にBluetoothが内蔵されており、スマートフォンとペアリングして利用する。傘とスマートフォンの距離が一定以上離れると連動したスマートフォンに自動で通知が送られ傘の置き忘れを防ぐことができる。

IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」(平成28年、総務省)
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

1-12. IoTによる製品の高付加価値比の事例2

事例	メーカー	画像	概要
スマート吸入器	Qualcomm Life, ノバルティス（スイス）		吸引機にセンサーを内蔵し、患者の服薬状況や服薬時間といったデータを収集して、患者の服薬管理を支援する。飲み薬に比べて「吸入タイプの薬剤は服薬し忘れる患者が多い」ことに対応する。2019に発売予定である。
スマート衣類	AiQ（台湾）		スマート衣類「Bioman（バイオマン）」を着て運動すれば、心拍数、呼吸数、体温などのバイタルサインの情報がアプリに転送され、スマホやPC上でデータ分析し、健康状態の持続的なモニタリングが可能。

IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

1-13. IoTによる新規事業・サービスの創出事例1

事例	サプライヤ	画像	概要
スマート コントラクション	コマツ（日本）		ドローンによる工場現場の測定や測定結果に基づく施工計画の作成支援、その施工計画の通りに動くICT建機および全体工程の進捗管理システムまで一括で提供するソリューションである。ソリューションで使用するICT建機は現在、傘下のレンタル会社であるコマツレンタルを通じて貸し出しているが、2016年度中には販売も始める見通しである。
ドコモ・バイクシェア スマートシェアリング	ドコモ・バイクシェア（日本）		自転車にGPSを備え、自転車の利用状況をネット経由で把握できる仕組みを実現し、全無人でレンタル自転車の事業を運用いている。自転車はセンサーの情報をネットに送る通信機能を持ち、ネット上のサーバーに情報が常時蓄積される。自転車の状態をセンサーでリアルタイムに把握することができ、盗難や返却忘れなどにも対応可能である。

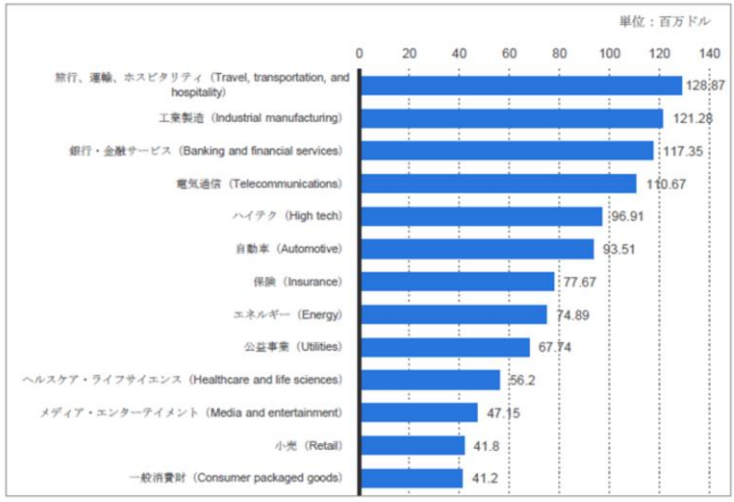
IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

1-13. IoTによる新規事業・サービスの創出事例2

事例	サプライヤ	画像	概要
PAY BY THE MILE	ミシュラン（フランス）		**タイヤにセンサーを組み込み、実際の走行距離に基づきタイヤのリース料金を請求する、「サービスとしてのタイヤ（Tire-as-a-Service）」を運送会社向けに提供している。**
クボタスマートアグリ	**クボタ（日本）**		**食味・収量測定機能を搭載したコンバインにより、圃場ごとの食味・水分・収量データを収集する。収集したデータに基づき圃場ごとに最適な施肥計画を立て、翌年度は堆肥自動調量機能を持つ農機によって、圃場ごとに計画通りの施肥を実施することができる。上記のサイクルを繰り返すことで、収穫・品質・食味の向上と安定化をサポートする。**

IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

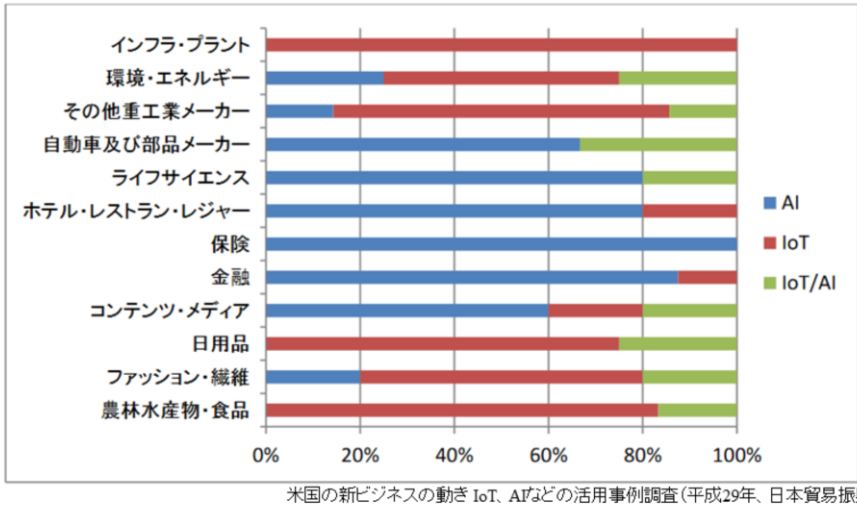
1-14. IoT関連技術に対する１社当たりの平均投資額（産業別、2015年5月）



米国の新ビジネスの動き IoT、AIなどの活用事例調査（平成29年、日本貿易振興機構）
https://www.jetro.go.jp/world/reports/2017/01/41f2ea19eaa8009f.html

アメリカでのIoTへの投資

1-15. 米国大手企業におけるIoT/AI関連の取り組みとしてメディアなど公開情報で取り上げられた主な事例の技術導入シェア（産業別）



米国の新ビジネスの動き IoT、AIなどの活用事例調査（平成29年、日本貿易振興機構）
https://www.jetro.go.jp/world/reports/2017/01/41f2ea19eaa8009f.html

アメリカでのIoTへの投資

1-16. IoT活用分野　（消費者向け）

顧客タイプ	分野	対象となるビジネス分野
消費者向け	家	・ホームオートメイション ・家の環境改良 ・エネルギー効率化
	ライフスタイル	・ウェアラブル ・エンターテイメント＆音楽 ・家族　　・レジャー ・ペット　　・おもちゃ ・ドローン
	ヘルスケア	・フィットネス ・モニタリング ・データ測定 ・診断
	移動	・コネクテッドカー ・e バイク

春特別号「米国におけるIoT 基盤の動向調査」(平成28年、日本貿易振興機構)
https://www.ipa.go.jp/about/NYreport/index_2016.html

1-17. IoTの活用分野　（ビジネス向け）1

ビジネス向け	小売り	・小売店 ・コンビニエンスストア
	医療（ヘルスケア）	・モニタリング ・データ計測 ・診断　・手術 ・患者ケア
	エネルギー	・送配電 ・化石燃料　・原子力 ・代替エネルギー源
	移動	・航空宇宙・空港 ・船舶 ・列車　・駅 ・自動車　・交通

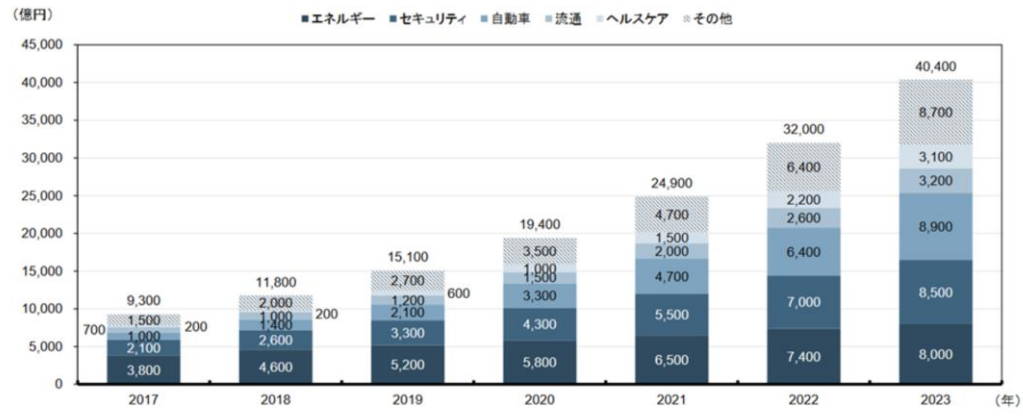
春特別号「米国におけるIoT 基盤の動向調査」（平成28年、日本貿易振興機構）
https://www.ipa.go.jp/about/NYreport/index_2016.html

1-17. IoTの活用分野　（ビジネス向け）2

ビジネス向け	都市	・インフラ ・水　・廃水 ・空調　・照明 ・セキュリティー　・安全
	製造	・鉱山 ・石油　・ガス ・生産 ・サプライチェーン
	公共サービス	・学校　・大学 ・政府　・行政 ・銀行　・保険 ・一般向けサービス
	その他	・環境 ・軍事 ・農業

春特別号「米国におけるIoT 基盤の動向調査」(平成28年、日本貿易振興機構)
https://www.ipa.go.jp/about/NYreport/index_2016.html

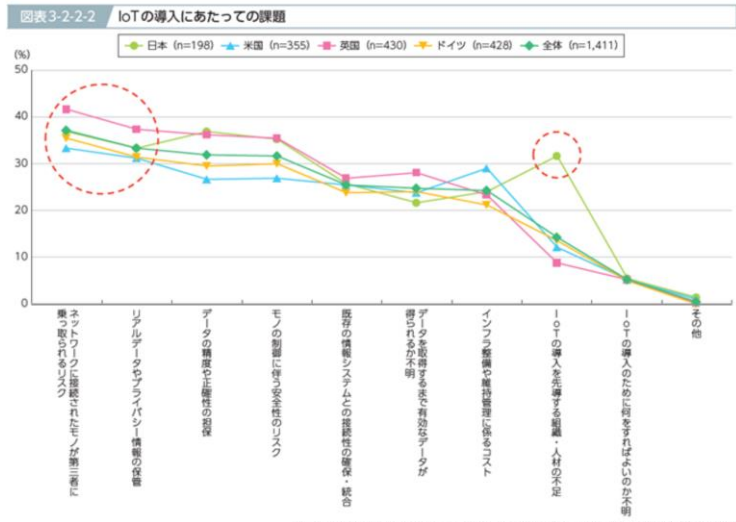
1-18. 国内のIoT市場規模の推移と予測



「ITナビゲーター2018年版」これからICT・メディア市場で何が起こるのか～2023年までの市場トレンドを予測（平成29年、野村総研）
http://www.nri.com/jp/event/mediaforum/2017/pdf/forum259.pdf

日本のIoT市場企業の推移と予想

1-19. IoTの導入にあたっての課題（平成30年度版）



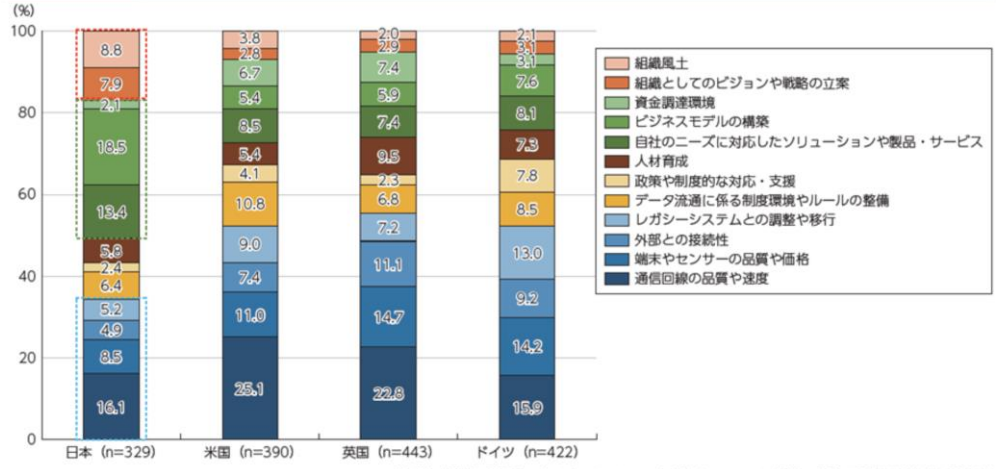
（出典）総務省「ICTによるイノベーションと新たなエコノミー形成に関する調査研究」（平成30年）

情報通信白書平成30年版（総務省）

http://www.soumu.go.jp/johotsusintokei/whitepaper/h30.html

1-20. 企業がAI・IoTの利活用を進める上での課題

図表3-2-2-4 企業がAI・IoTの利活用を進める上での課題

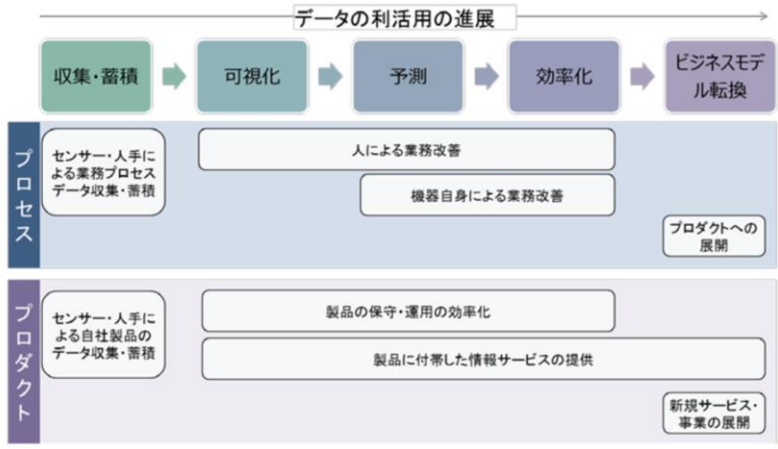


（出典）総務省「ICTによるイノベーションと新たなエコノミー形成に関する調査研究」（平成30年）
情報通信白書平成30年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h30.html

再掲載

利活用の進展と
ロダクトにおける進展の対応

図表 3-3-2-3　データの利活用の進展とプロセス・プロダクトにおける進展の対応

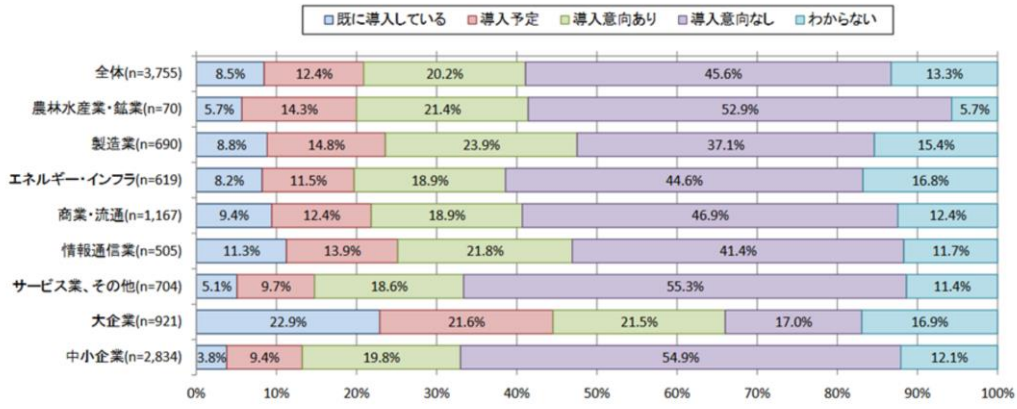


IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究（平成28年、総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

IoTで何ができるか

プロセスとプロダクトの事例についてはe-learning＆既出

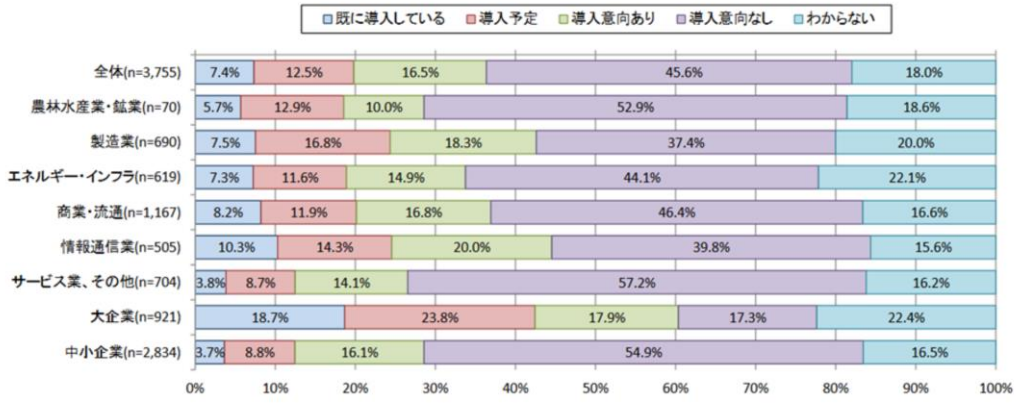
1-21. 日本企業におけるプロセスIoT化率



IoT時代におけるICT経済の諸課題に関する調査研究（平成29年、総務省）
http://www.soumu.go.jp/johotsusintokei/linkdata/h29_04_houkoku.pdf

- 人材不足、使い方、効果が疑問、、、というものが停滞理由の１つ
- 「この講座で使えるようになって、イメージ、検証ができるようになりましょう」とアピール

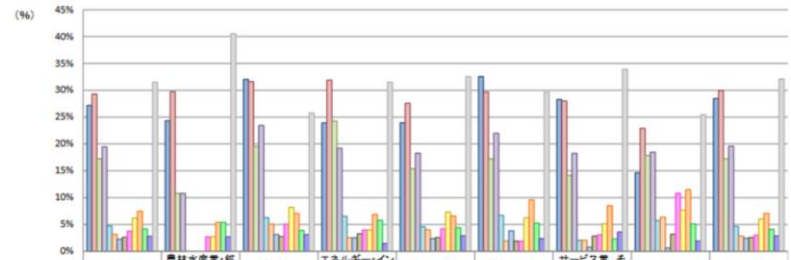
1-22. 日本企業におけるプロダクトIoT化率



IoT時代におけるICT経済の諸課題に関する調査研究（平成29年、総務省）
http://www.soumu.go.jp/johotsusintokei/linkdata/h29_04_houkoku.pdf

- 人材不足、使い方、効果が疑問、、、というものが停滞理由の１つ
- 「この講座で使えるようになって、イメージ、検証ができるようになりましょう」とアピール

1-23. プロセスIoT化を考えていない理由

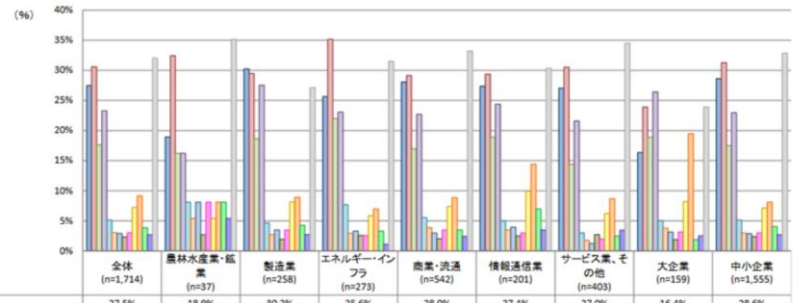


	全体 (n=1,714)	農林水産業・鉱業 (n=37)	製造業 (n=256)	エネルギー・インフラ (n=276)	商業・流通 (n=547)	情報通信業 (n=209)	サービス業、その他 (n=389)	大企業 (n=157)	中小企業 (n=1,557)
資金不足	27.2%	24.3%	32.0%	23.9%	23.9%	32.5%	28.3%	14.6%	28.5%
利用場面が不明	29.3%	29.7%	31.6%	31.9%	27.6%	29.7%	28.0%	22.9%	29.9%
人材不足	17.3%	10.8%	19.5%	24.3%	15.4%	17.2%	14.1%	17.8%	17.2%
効果に疑問	19.5%	10.8%	23.4%	19.2%	18.3%	22.0%	18.3%	18.5%	19.6%
ネットワークインフラの未整備	4.7%	0.0%	6.3%	6.5%	4.6%	6.7%	2.1%	5.7%	4.6%
ネットワークインフラの高度化・仮想化が不十分	3.2%	0.0%	5.1%	2.5%	4.0%	1.9%	2.1%	6.4%	2.8%
センサー・端末の未普及	2.3%	0.0%	3.1%	2.5%	2.4%	3.8%	0.8%	0.6%	2.4%
非標準化	2.6%	0.0%	2.7%	3.3%	2.6%	1.9%	2.8%	3.2%	2.6%
データ流通に係るルールの未整備	3.7%	2.7%	5.1%	4.0%	4.2%	1.9%	3.1%	10.8%	3.0%
新規市場が創出できない	6.2%	2.7%	8.2%	4.0%	7.3%	6.2%	5.1%	7.6%	6.0%
既存市場でのビジネスモデルが確立できない	7.5%	5.4%	7.0%	6.9%	6.6%	9.6%	8.5%	11.5%	7.1%
普及促進に係る政策・支援がない	4.2%	5.4%	3.9%	5.8%	4.4%	5.3%	2.3%	5.1%	4.1%
その他	2.8%	2.7%	3.1%	1.4%	2.9%	2.4%	3.6%	1.9%	2.9%
導入意向なし	31.5%	40.5%	25.8%	31.5%	32.5%	29.7%	33.9%	25.5%	32.1%

IoT時代におけるICT経済の諸課題に関する調査研究（平成29年、総務省）
http://www.soumu.go.jp/johotsusintokei/linkdata/h29_04_houkoku.pdf

- 人材不足、使い方、効果が疑問、、、というものが停滞理由の１つ
- 「この講座で使えるようになって、イメージ、検証ができるようになりましょう」とアピール

1-24. プロダクトIoT化を考えていない理由

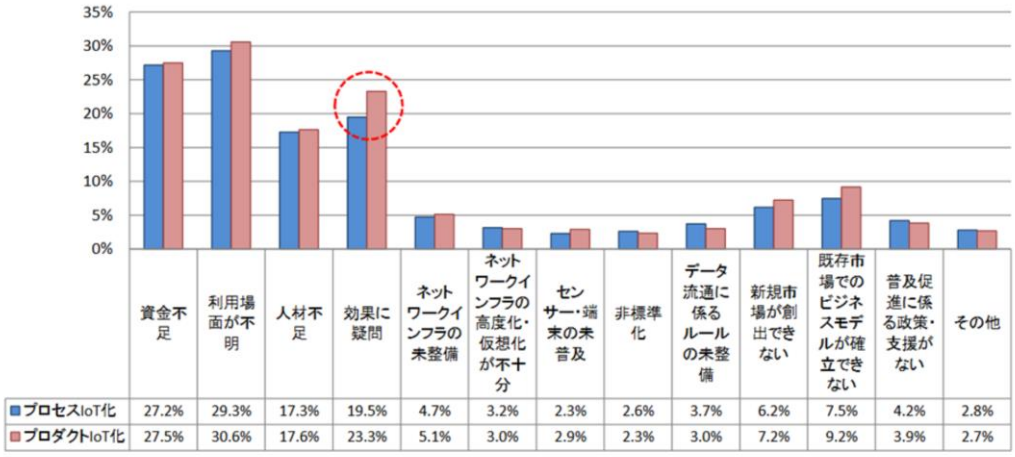


	全体 (n=1,714)	農林水産業・鉱業 (n=37)	製造業 (n=258)	エネルギー・インフラ (n=273)	商業・流通 (n=542)	情報通信業 (n=201)	サービス業、その他 (n=403)	大企業 (n=159)	中小企業 (n=1,555)
資金不足	27.5%	18.9%	30.2%	25.6%	28.0%	27.4%	27.0%	16.4%	28.6%
利用場面が不明	30.6%	32.4%	29.5%	35.2%	29.2%	29.4%	30.5%	23.9%	31.3%
人材不足	17.6%	16.2%	18.6%	22.0%	17.0%	18.9%	14.4%	18.9%	17.5%
効果に疑問	23.3%	16.2%	27.5%	23.1%	22.7%	24.4%	21.6%	26.4%	23.0%
ネットワークインフラの未整備	5.1%	8.1%	4.7%	7.7%	5.5%	5.0%	3.0%	5.0%	5.1%
ネットワークインフラの高度化・仮想化が不十分	3.0%	5.4%	2.7%	2.9%	3.9%	3.5%	1.7%	3.8%	3.0%
センサー・端末の未普及	2.9%	8.1%	3.5%	3.3%	3.0%	4.0%	1.2%	3.1%	2.9%
非標準化	2.3%	2.7%	1.9%	2.6%	2.0%	2.5%	2.7%	1.9%	2.4%
データ流通に係るルールの未整備	3.0%	8.1%	3.5%	2.6%	3.5%	3.0%	2.0%	3.1%	3.0%
新規市場が創出できない	7.2%	5.4%	8.1%	5.9%	7.4%	10.0%	6.2%	8.2%	7.1%
既存市場でのビジネスモデルが確立できない	9.2%	8.1%	8.9%	7.0%	8.9%	14.4%	8.7%	19.5%	8.1%
普及促進に係る政策・支援がない	3.9%	8.1%	4.3%	3.3%	3.5%	7.0%	2.5%	1.9%	4.1%
その他	2.7%	5.4%	2.7%	1.1%	2.4%	3.5%	3.5%	2.5%	2.7%
導入意向なし	32.0%	35.1%	27.1%	31.5%	33.2%	30.3%	34.5%	23.9%	32.9%

IoT時代におけるICT経済の諸課題に関する調査研究（平成29年、総務省）
http://www.soumu.go.jp/johotsusintokei/linkdata/h29_04_houkoku.pdf

- 人材不足、使い方、効果が疑問、、、というものが停滞理由の１つ
- 「この講座で使えるようになって、イメージ、検証ができるようになりましょう」とアピール

1-25. IoT化を考えていない理由の比較



	資金不足	利用場面が不明	人材不足	効果に疑問	ネットワークインフラの未整備	ネットワークインフラの高度化・仮想化が不十分	センサー・端末の未普及	非標準化	データ流通に係るルールの未整備	新規市場が創出できない	既存市場でのビジネスモデルが確立できない	普及促進に係る政策・支援がない	その他
プロセスIoT化	27.2%	29.3%	17.3%	19.5%	4.7%	3.2%	2.3%	2.6%	3.7%	6.2%	7.5%	4.2%	2.8%
プロダクトIoT化	27.5%	30.6%	17.6%	23.3%	5.1%	3.0%	2.9%	2.3%	3.0%	7.2%	9.2%	3.9%	2.7%

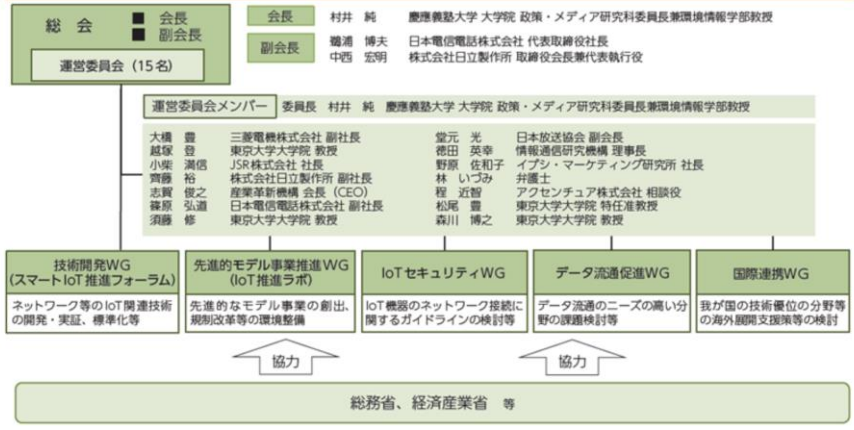
IoT時代におけるICT経済の諸課題に関する調査研究（平成29年、総務省）
http://www.soumu.go.jp/johotsusintokei/linkdata/h29_04_houkoku.pdf

- 人材不足、使い方、効果が疑問、、、というものが停滞理由の１つ
- 「この講座で使えるようになって、イメージ、検証ができるようになりましょう」とアピール

1-26. IoT推進コンソーシアム

図表6-1-2-3 IoT推進コンソーシアム

■ IoT/ビッグデータ/人工知能時代に対応し、企業・業種の枠を超えて産学官で利活用を促進するため、総務省及び経済産業省の共同の呼びかけのもと、民主導の組織として「IoT推進コンソーシアム」を設立。（平成27年10月23日（金）に設立総会を開催。）
■ 技術開発、利活用、政策課題の解決に向けた提言等を実施。
（会員法人数3,513社（平成30年3月13日現在））



情報通信白書平成30年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h30.html

1-27. スマートIoT推進フォーラム

トピックス

>>もっと見る

■ 2018年7月25日
「IoT機器等の電波利用システムの適正利用のためのICT人材育成事業 講習会」@島根県松江市（2018年8月23日）開催のご案内

https://smartiot-forum.jp/

1-28. IoT導入事例紹介

https://smartiot-forum.jp/iot-val-team/iot-case

1-29. その他の委員会・コミュニティなど

- 一般社団法人　情報通信技術委員会　・・・標準化委員会
  http://www.ttc.or.jp/

- IoMT学会　・・・Internet of Medical Things
  https://iomt.or.jp/

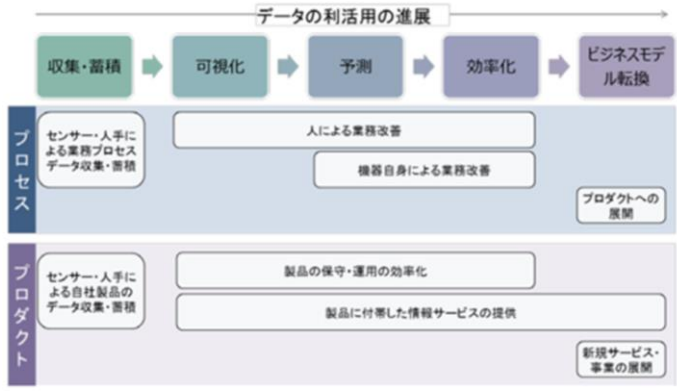
- 実践IoTラボ　・・・勉強会の任意団体
  http://piotlab.org/

- IoT縛りの勉強会！IoTLT　・・・勉強会のコミュニティ
  https://iotlt.connpass.com/

考察　IoTの事業モデル設計

- ①自社の課題や問題点、各自が考えるアイディアなどをデータの利活用の進展のプロセスに当てはめて書きまとめてみましょう。

図表 3-3-2-3　データの利活用の進展とプロセス・プロダクトにおける進展の対応



IoT時代におけるICT産業の構造分析とICTによる経済成長への多面的貢献の検証に関する調査研究」（平成28年，総務省）
http://www.soumu.go.jp/johotsusintokei/link/link03_h28.html

考察　IoTの事業モデル設計

- ②スマートIoT推進フォーラムのIoT導入事例紹介（p.43）から各自が考えるアイディアなどの参考になる事例のポイントを書き出してみましょう。

第２章 IoTに関連する
主な通信技術

2-1. 主な予備知識

- SSID
- IEEE 802.11シリーズ
- Wi-Fi
- 無線暗号化技術（WEP、WPA、WPA2）
- TCP/IPプロトコルの４層
- OSI参照モデルの７層
- 主要なプロトコルとポート番号
- IPアドレスとMACアドレス
- ループバックアドレス
- プライベートIPアドレスとグローバルIPアドレス
- NAT

2-2.電波の種類

周波数帯	波長	特徴
超長波	10～100km	地表面に沿って伝わり低い山をも越えることができる。また、水中でも伝わるため、海底調査にも応用できる。
長波	1～10km	非常に遠くまで伝わることができる。電波時計等に時間と周波数標準を知らせるための標準周波数局に利用されている。
中波	100～1000m	約100kmの高度に形成される電離層のE層に反射して伝わることができる。主にラジオ放送用といて利用されている。
短波	1～10m	約200～400kmの高度に形成されるF層に反射して、地表との反射を繰り返しながら地球の裏側まで伝わっていくことができる。遠洋の船舶通信、国際線航空機用の通信、国際放送及びアマチュア無線に広く利用されている。
超短波	10cm～1m	直進性があり、電離層で反射しにくい性質もあるが、山や建物の陰にもある程度回り込んで伝わることができる。防災無線や消防無線など多種多様な移動通信に幅広く利用されている。

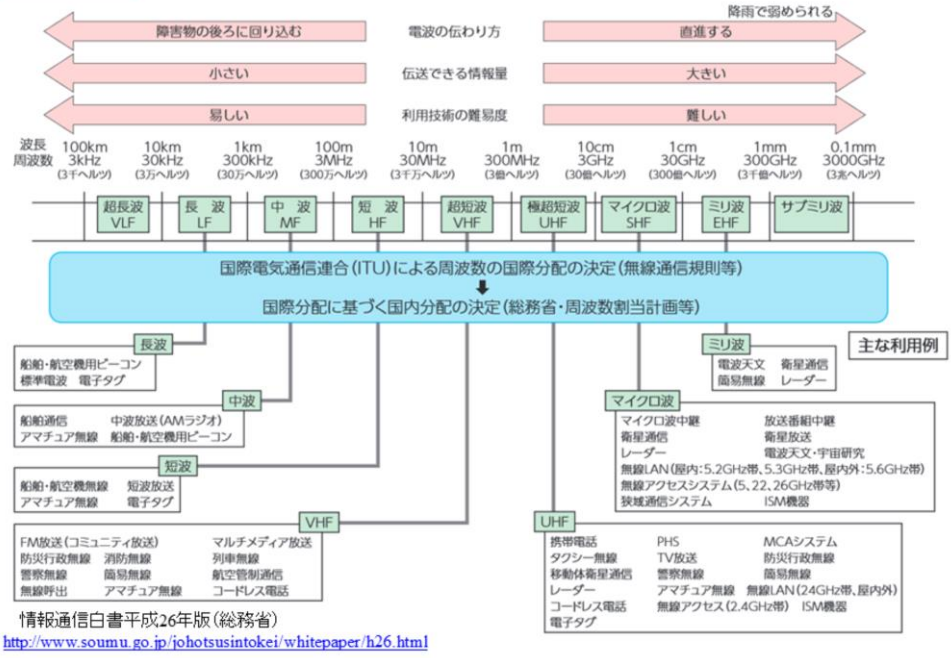
情報通信白書平成26年版(総務省)
http://www.soumu.go.jp/johotsusintoker/whitepaper/h26.html

電波の種類（続き）

周波数帯	波長	特徴
極超短波	10cm～1m	超短波に比べて直進性が更に強くなるが、多少の山や建物の陰には回り込んで伝わることができる。携帯電話をはじめとした多種多様な移動通信システムを中心にデジタルテレビ放送、空港監視レーダーや電子レンジ等に幅広く利用されている。
マイクロ波	1～10cm	直進性が強い性質を持つため、特定の方向に向けて発射するのに適している。主に固定の中継回線、衛星通信、衛星放送や無線LANに利用されている。
ミリ波	1mm～10mm	マイクロ波と同様に強い直進性があり、非常に大きな情報量を伝送することができるが悪天候時には雨や霧の影響を強く受けてあまり遠くへ伝わることができない。このため比較的短距離の無線アクセス通信や画像伝送システム、簡易無線、自動車衝突防止レーダー等に利用されている他、電波望遠鏡による天文観測が行われている。
サブミリ波	0.1mm～1mm	光に近い性質を持った電波。現在の技術では巨大な無線設備が必要で、また水蒸気による吸収が大きいという性質があるため、通信用としてはほとんど利用されていないが、一方では、ミリ波と同様に電波望遠鏡による天文観察が行われている。

情報通信白書平成26年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h26.html

図表5-7-1-1　我が国の周波数帯ごとの主な用途と電波の特徴通信衛星



情報通信白書平成26年版(総務省)
http://www.soumu.go.jp/johotsusintokei/whitepaper/h26.html

2-3. 免許不要の無線局

- 電波を使う通信システムは無線局として総務大臣の免許あるいは登録が必要。

- ただし、以下の無線局は免許不要となる。

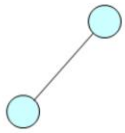
- ①微弱無線局
  - 主に322MHz以下の周波を利用
- ②特定小電力無線局
  - 技術基準適合証明を受けている無線局
- ③小電力データ通信システム
  - 技術基準適合証明を受けている無線局（送信電力が0.01W以下）
  - 無線LAN、Bluetooth、ZigBeeなどはここに属する

現在の技適マーク（Ｈ７．４～）　旧タイプの技適マーク（Ｓ６２．１０～）

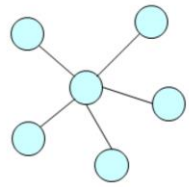
2-4. Bluetooth・ZigBee・Wi-Fi

	Bluetooth	ZigBee	Wi-Fi
周波数帯	2.4GHz帯	2.4GHz帯	2.4GHz帯・5GHz帯等
規格	IEEE802.15.1	IEEE802.15.4	IEEE802.11x
免許	不要	不要	不要
伝送速度	～24Mpbs	～250kbps	11～54Mbps, 600Mpbs等
通信距離	～10m	～3km	～100m
接続可能ノード数	20	65,536	13
トポロジ	P2P，メッシュ	P2P，ツリー，スター，メッシュ	P2P，ツリー，スター，メッシュ

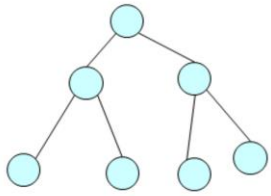
2-5. ネットワークトポロジ



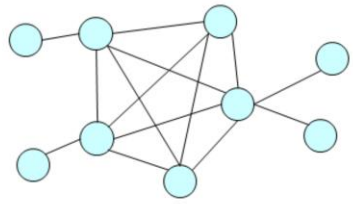
P2P（ピアツーピア）型



スター型



ツリー型



メッシュ型

2-6. Bluetooth Low Energy（BLE）

- Bluetooth 4.0の別名、従来の企画と互換性なし
- Bluetooth 4.1(2013)、4.2(2014)、5(2016)などのアップデート版がある

- Bluetoothの表記
- ・Bluetooth：従来のBluetoothにのみ対応した機器
- ・Bluetooth Smart：BLEのみに対応した機器
- ・Bluetooth Smart Ready：BluetoothとBLEの両方に対応した機器

- BLEの特徴
- ・通信可能距離が短い
- ・通信速度が低速

2-7. Low Power Wide Area（LPWA）

- LPWA
- 少ない電力消費で数kmの長距離通信ができる通信のこと
- 一定出力以下であれば免許が不要

名称	SIGFOX	LoRaWAN	WI-FI HaLow	WI-SUN	RPMA
bps	約100	約250-50k	約150k	約50k-400k	約40k
km	50	15	1	1	20

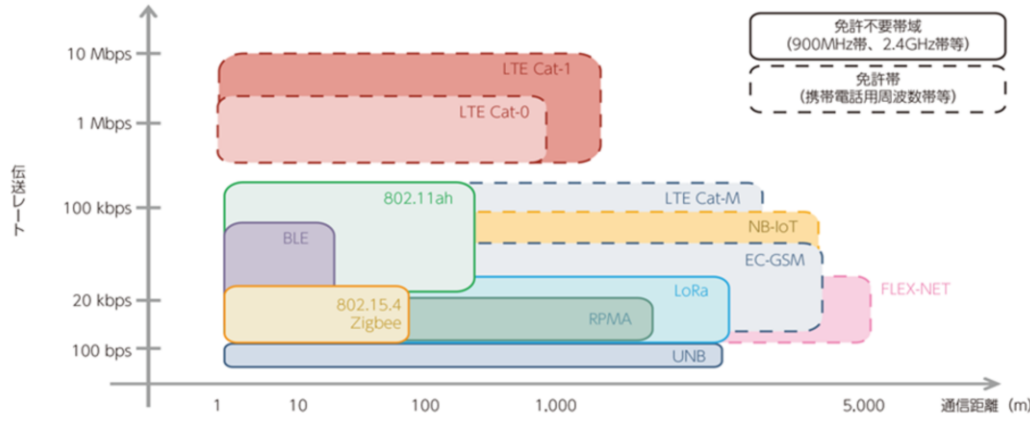
2-8. LPWAの特徴

特徴	内容
低電力	単一の小型フォームファクタバッテリで複数年のデバイス動作を実現
広域	都市や地下の環境などの複数のユースケースをカバーするために、全国的および国際的な携帯電話レベルのカバレッジを提供することが可能
その他の利点	エンドポイント密度が高い、ハードウェアコストが安い、接続コストが安い、データレートが低い、待ち時間が制限される、可動性

（出典）総務省「第4次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

2-9. 主なLPWA規格の位置付け



情報通信白書平成29年版(総務省)
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

2-10. LPWAの活用事例（日本）

区分		国・企業等	事例概要　（L：LoRa，S：Sigfox）
日本	実証	アズビル、日本IBM等7社	福岡市でガス・水道メータのデータ収集に関する実証実験を今年7月から実施。実用化に向けた課題を洗い出し。［L］
		NTTドコモ、ITベンチャーのハタプロ	長野県大町市で、市内の水源3か所、配水池11か所と市役所を結ぶネットワークを、LPWA方式で各施設の稼働状況を常時監視するシステムを実証。［L］
		日立システムズ	トミス、イートラスト、新潟市水道局、新潟市の協力を得て、マンホールの防犯・安全対策ソリューションの提供に向けた実証実験の一環として、マンホールの監視の検証を実施。［方式不明］
	実用	京セラコミュニケーションシステム	宅配ピザチェーン店で、ピザの生地を保護する冷蔵庫の温度管理を遠隔で行うシステムを導入。［S］

（出典）総務省「第4次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）
情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

2-11. LPWAの活用事例（海外）

区分	国・企業等	事例概要　（L：LoRa，S：Sigfox）	
海外	産業	米・Senet	米国では水道インフラの劣化が課題となっており、Senet社のLPWAネットワークを用いたインフラのモニタリングサービスが提供されている。［L］
		ペルー・マヌー国立公園	国立公園内にカメラとセンサーを設置し、環境状態をモニタリング。リルタイムな情報提供。［L］
		台湾・亜太電信	2016年から台北市を手始めに新北市、桃園市の計500か所に「LoRaホットスポット」を設置し、台湾をIoTスマートアイランドにする計画。［L］
	コンシューマ	フランス・La Poste	ボタンを押しだけで集荷や宅配を依頼できるボタン型デバイスを展開［S］
		オランダ・KPN	アムステルダムでLPWAのゲートウェイと3千個以上のビーコンを設置し、スマートシティーに向けたインフラを整備。既にAmesterdam Beacom Mileと呼ばれる観光客向けのサービスが提供されている。［L］

（出典）総務省「第4次産業革命における産業構造分析とIoT・AI等の進展に係る現状及び課題に関する調査研究」（平成29年）

情報通信白書平成29年版（総務省）
http://www.soumu.go.jp/johotsusintokei/whitepaper/h29.html

2-12. IoTシステムの主なプロトコル

- IoTシステムの機器やアプリケーションの種類などに応じてプロトコルを選択

- 2016年に公開されたoneM2M（シリーズ2）では、機械と機械の通信プロトコルとして以下が挙げられている。
- ・HTTP
- ・MQTT
- ・CoAP
- ・WebSocket

- ※oneM2M
- 2012年、世界の代表的な7つの標準化開発機関の代表が集まり、設

2-13. HTTP

- HTMLで書かれた文書を、Webサーバとクライアントの間でやり取りするプロトコル

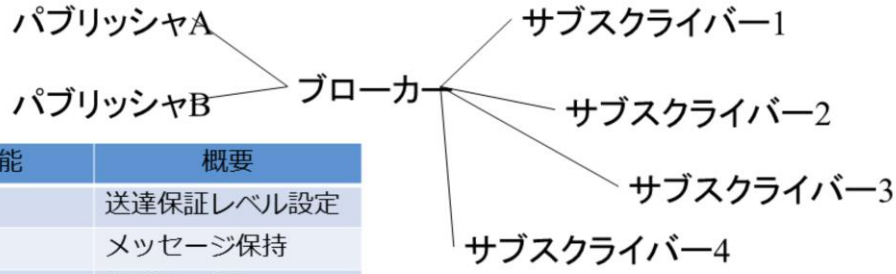
- メッセージを　ヘッダ情報＋（空行）＋ボディ　のフォーマットで送受信

- HTTPでは以下のような メソッドを利用

主なメソッド	機能
GET	リソース情報の取得
HEAD	リソース情報の取得（HTTPヘッダのみ）
POST	リソース情報の登録
PUT	登録済みリソースの変更
DELET	登録情報の削除
CONNECT	SSL等のトンネリング通信のプロキシの情報

2-14. MQTT

- MQ Telemetry Transportの略
- １対多のメッセージ通信が規定されたプロトコル
- メッセージを発行するパブリッシャーとメッセージ購読者であるサブスクライバー、メッセージの送受信を仲介するブローカーの３つで構成される



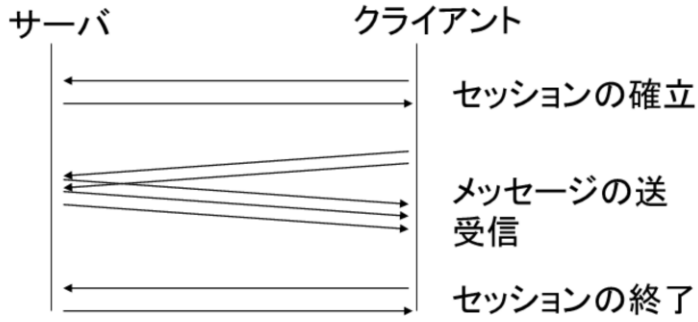
機能	概要
QoS	送達保証レベル設定
Retain	メッセージ保持
Will	切断時の通知

2-15. CoAP

- インターネットに関する技術の標準を定める任意団体（IETF）が定めたM2M通信向けのWeb転送プロトコル
- HTTPの圧縮化や簡素化をおこなったもの
- HTTPヘッダの情報を圧縮したり、UDPベースにすることで、HTTPと比較して通信量を削減している
- CoAPは、CPU能力が低い、メモリ容量が小さい、低消費電力が求められる、パケット損失率の高い無線ネットワークといった制約された環境での利用に適する

2-16. WebSocket

- 双方向通信を行うプロトコル
- HTTPを使ってクライアントーサーバ間でコネクションを確立した後に双方向通信を行っている
- WebSocketではフレーム単位でデータが送信される



第３章 電気回路の基礎

3-1. 電気回路に関する用語

- 電気回路・・・電流の流れる一巡りの回路。直流回路と交流回路がある。
- 直流　　・・・DC 電圧の大きさや流れる方向が一方向で変化しない。
- 交流　　・・・AC 電圧や電流の流れる方向が時間と共に変化する。
- トランス・・・変圧器。電圧の大きさを変える。交流でのみ利用。
- ダイオード・・整流器。電流の流れを一方向に制限する。許容範囲に注意。
- 　　以下のような記号で書かれる。電流の流れはアノードから

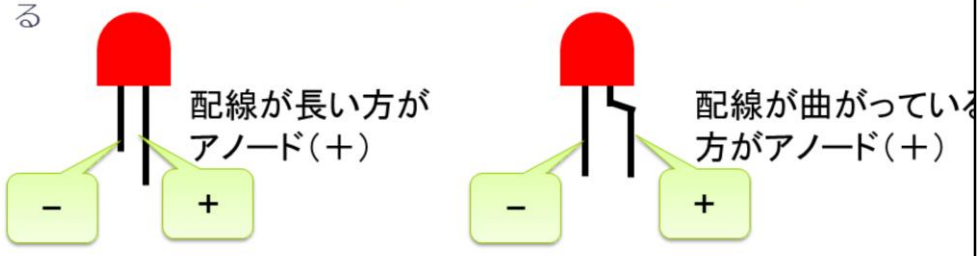
3-2. 電流の流れ

- 例えばLEDも電流が流れる方向（極性）が決まっている
- アノード（+）からカソード（ー）へ流れる方向を順方向とよび、その反対の方向を逆方向と呼ぶ
- ※逆方向に電流を流したとき、順方向に容量以上の電流を流したとき、素子を破壊する可能性があるので注意すること
- 順方向に流す電流や電圧を順方向電流（順電流）や順方向電圧（順電圧）とよぶ（IF：I Forward, VF：V Forward）
- 逆方向に流す電流や電圧を逆方向電流（逆電流）や逆方向電圧（逆電圧）とよぶ（IR：I Reverse, VR：V Reverse）

3-3. LEDの仕様

- 赤色LEDの場合、順方向電圧1.8~2.2V程度
- 黄色LEDの場合、順方向電圧1.9~2.2V程度
- 緑色LEDの場合、順方向電圧2.2~3.7V程度
- 青色LEDの場合、順方向電圧3.2~3.7V程度
- 白色LEDの場合、順方向電圧3.6~3.7V程度で点灯する

- 電流の最大容量が20~50mA程度、電圧の最大容量が3~9V程度である



3-4. 電圧と電流と電力の単位

- 電圧（ボルト）　・・・電気を押し出す力．単位：V
- 電流（アンペア）・・・電気の流れる量．単位：A
- 電力（ワット）　・・・電気エネルギー．単位：W
- 電力（W）＝電圧（V）×電流（A）
- 1kW ＝ 1000W　　1A ＝1000mA
- ※k（キロ）　1000

3-5. 感電した場合の危険度の目安

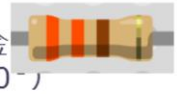
- 人体の電気抵抗を約５～１０KΩとすると、
- ・１mA　　びりっと感じる程度（電圧に換算すると５０V未満）
- ・５mA　　相当に痛い（約７５V）
- ・１０mA　耐えられないほど痛い（約１００V）
- ・２０mA　筋肉は硬直し呼吸も困難に（約１００～２００V）
- ・５０mA　短時間でも生命が相当危険（約２００～５００V）

3-6. 抵抗の色の読み方

- 黒＝0　茶＝1　赤＝2　橙＝3
  黄＝4　緑＝5　青＝6　紫＝7
  灰＝8　白＝9

- 1色目と2色目の色で数値を、3色目で0の数を表す＝10^xのx
  最初の3色で数値を、4色目で0の数を表す場合もあり）

- 例：　330Ω
  橙橙茶金
  （３３×10^1）

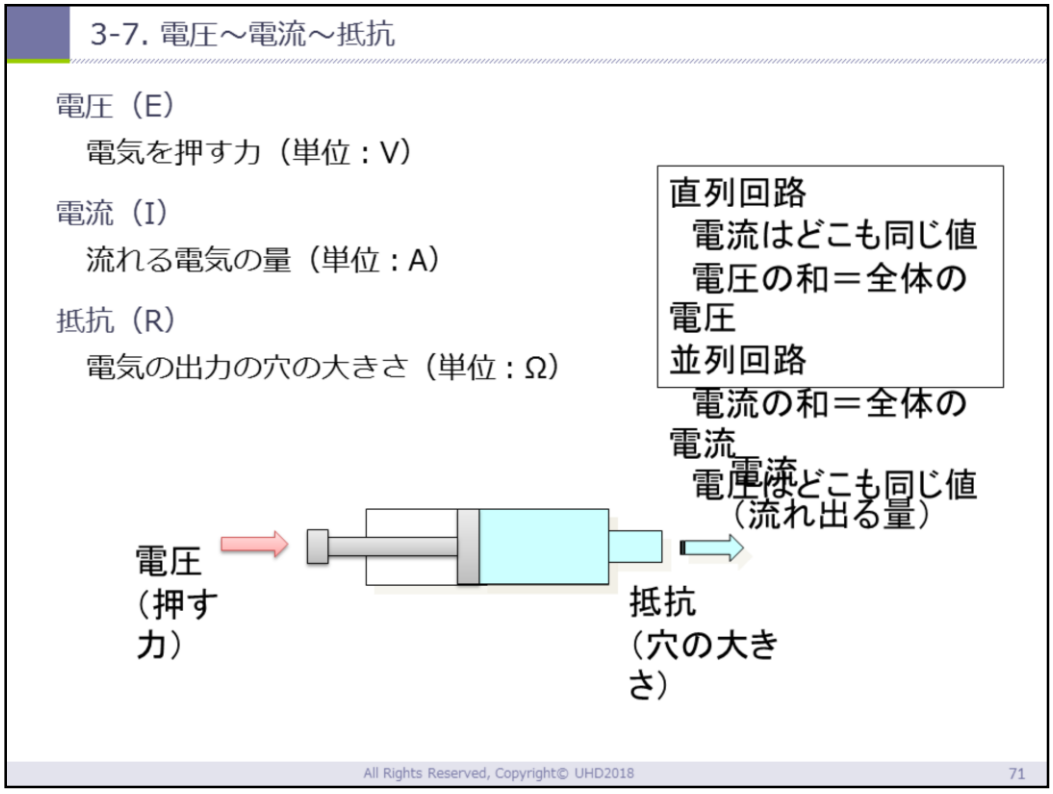


1kΩ
茶黒茶
（１０×10^2）

- 4色目は誤差（もしくは5色目の場合もある）
  （茶±1%, 赤±2%, 金±5%, 銀±10%, 無±20%）

覚え方　：くちあだきみあむはし....呪文のように覚えます！

く（黒＝0）、ち（茶＝1）、あ（赤＝2）、だ（橙＝3）、
き（黄＝4）、み（緑＝5）、あ（青＝6）、む（紫＝7）、
は（灰＝8）、し（白＝9）



電圧と電流と抵抗のイメージは水鉄砲をイメージするよい.

１）水を押し出す力が強いと水が勢いよく飛び出る＝電圧が大きいと電流が大きくなる

　　水を押し出す力が弱いと水は弱くでる＝電圧が小さいと電流も小さくなる

２）水を押す力が同じのとき穴の大きさが小さいと水は勢いよく飛び出る＝抵抗が小さいと電流が大きくなる

　　水を押す力が同じのとき穴の大きさが大きいと水は弱く出る＝抵抗が大きいと電流は小さくなる

これらの特性を踏まえると、回路を流れる電流の大

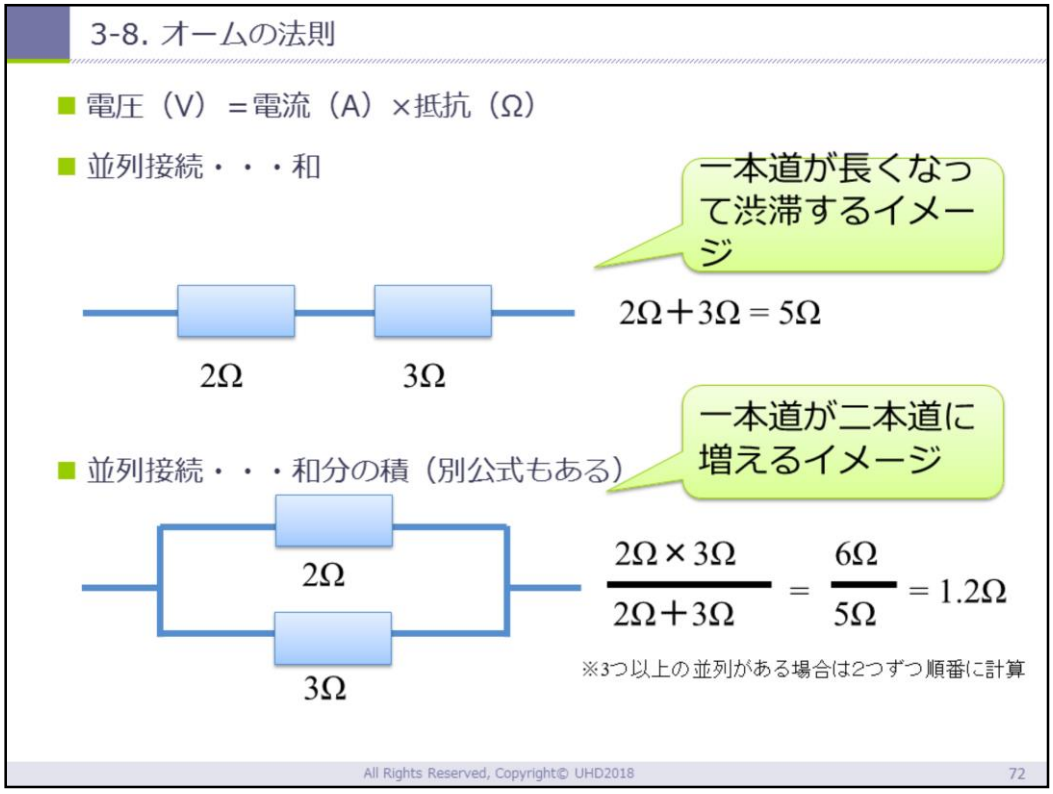
きさを調整することができる．つまり、

３）流したい電流が目標値より大きい場合、電流を今よりも小さくしたいので、電圧を下げるor抵抗を大きくする

　　流したい電流が目標値より小さい場合、電流を今よりも大きくしたいので、電圧を上げるor抵抗を小さくする

　　※通常、電圧は一定なので、抵抗を付け替えることによって電流の大きさを調整する

　　※例えば、大きな電圧と小さな電圧で同じ大きさの電流を流したい場合、「大きな電圧だが抵抗が大きい」＝「小さな電圧だが抵抗が小さい」で同じ値の電流を流すことができる.

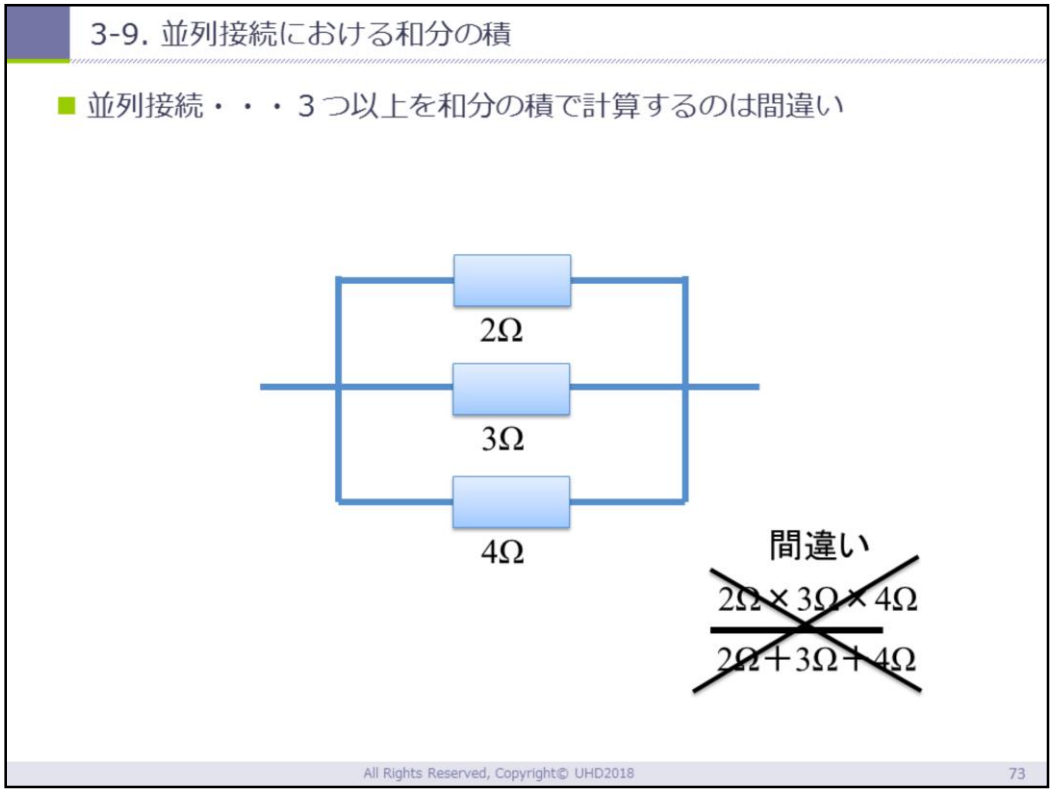


並列接続の場合、和分の積はスライド右下の計算式のように、「分子が掛け算」で「分母が足し算」になる

和分の積は２つの並列接続のときに実施するものである点に注意する.

公式については、ここでは省略している.

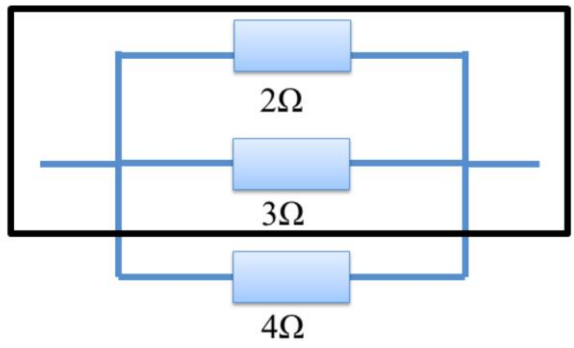
【もし公式について説明するならば】
合成抵抗R0Ωを求める公式は、１÷R0＝１÷R1＋１÷R2＋１÷R3＋・・・＋１÷Rnとなる．つまり、抵抗の値をひっくり返したもの（逆数）を合計する.
しかし、上記の和分の積を繰り返して２つの抵抗を１つに計算していくことで、合成抵抗を求めることができる.



和分の積は２つの並列接続のときに実施するものである点に注意する．例えば、３つの場合は３つ同時に計算することはできない．つまり（２Ω×３Ω×４Ω）÷（２Ω＋３Ω＋４Ω）は間違いなので注意すること．３つの抵抗が並列接続している場合は、まず３つのうちの任意の２つを和分の積を用いて１つに計算する．その後、計算結果の１つと残りの１つの２つを再度和分の積で計算することになる．

並列接続における和分の積（続き）

- 並列接続・・・まず一部分を和分の積で計算する

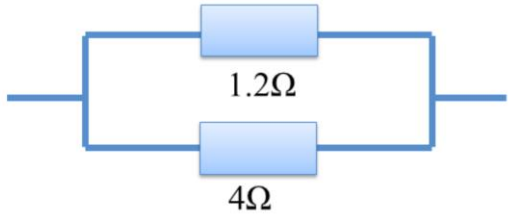


$$\frac{2\Omega \times 3\Omega}{2\Omega + 3\Omega} = \frac{6\Omega}{5\Omega} = 1.2\Omega$$

和分の積は２つの並列接続のときに実施するものである点に注意する．例えば、３つの場合は３つ同時に計算することはできない．つまり（２Ω×３Ω×４Ω）÷（２Ω＋３Ω＋４Ω）は間違いなので注意すること．３つの抵抗が並列接続している場合は、まず３つのうちの任意の２つを和分の積を用いて１つに計算する．その後、計算結果の１つと残りの１つの２つを再度和分の積で計算することになる．

並列接続における和分の積（続き）

- 並列接続・・・残りの部分を２回目の和分の積で計算する



$$\frac{1.2\Omega \times 4\Omega}{1.2\Omega + 4\Omega} = \frac{4.8\Omega}{5.2\Omega} = 0.92\Omega$$

和分の積は２つの並列接続のときに実施するものである点に注意する．例えば、３つの場合は３つ同時に計算することはできない．つまり（２Ω×３Ω×４Ω）÷（２Ω＋３Ω＋４Ω）は間違いなので注意すること．３つの抵抗が並列接続している場合は、まず３つのうちの任意の２つを和分の積を用いて１つに計算する．その後、計算結果の１つと残りの１つの２つを再度和分の積で計算することになる．

IoT
演習資料

はじめに・・・Arduino と類似製品

本家 Arduino と Arduino Nano

本講座でも使用する Arduino は、ハードウェア仕様が公開されているため、格安の互換品が多く出回っています。また、スロット互換の製品も多く登場しています。より小型の製品として、Arduino Nano がありますが、物理形状がまったく異なるため、ハードウェアの互換性はありません（ソフトウェア互換有り）。

ESP32-WROOM-32, ESP32-DevKitC

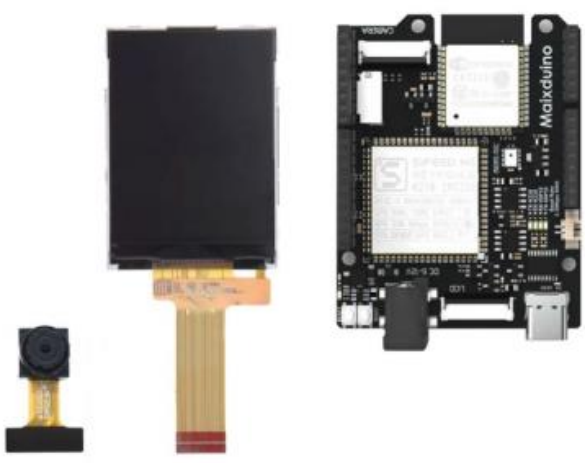
Arduino は通信機能を持たないため、Arduino に Wi-Fi 機能をプラスするために用いられるのが ESP32-WROOM32 です。ただし、ESP32-WROOM-32 は半田付けが必要で、Arduino とも多数の結線が必要になります。そのため、もっと簡単に使用できるようにピンヘッド等を接続したものが ESP32-DevKitC（技適取得済み）です。

ESP32-DevKitC は、これ単体で Wi-Fi や Bluetooth による通信機能を持ったワンチップマイコンとして機能します。Arduino とソフトウェア互換があるため、センサを接続し、ソフトウェア資産をそのまま流用できます。

M5Stack シリーズ

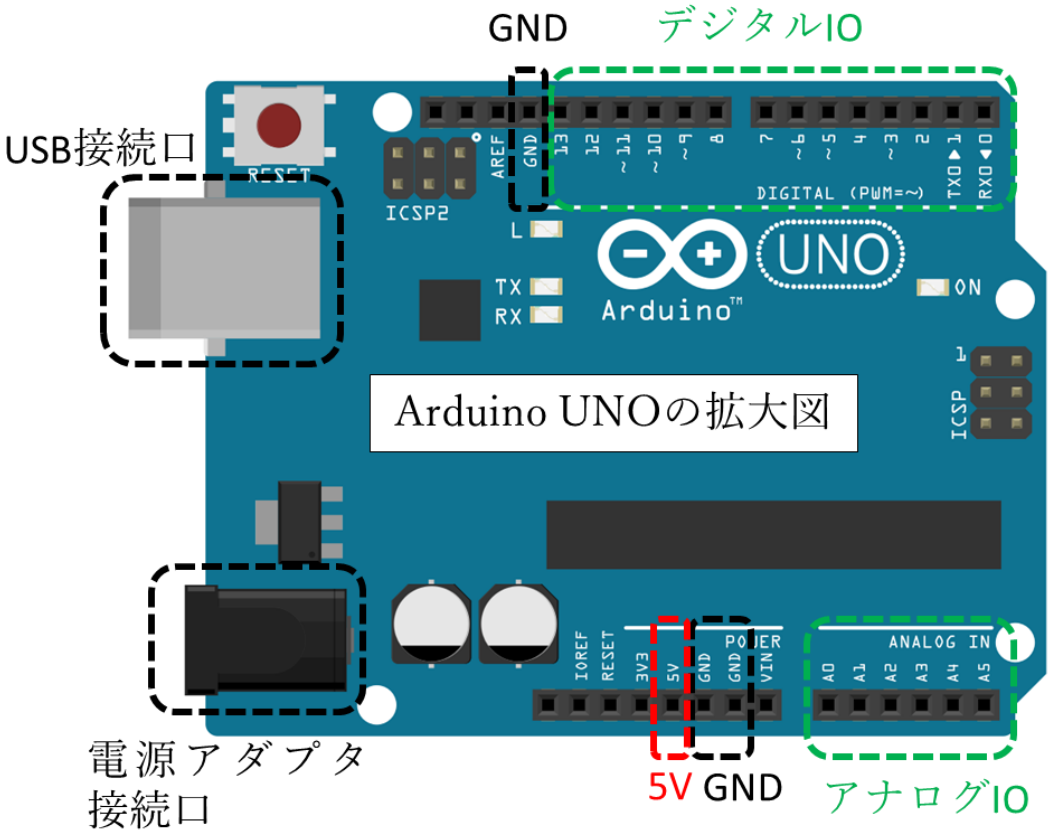
ESP32 に LCD や microSD スロット、スピーカを内蔵し、より便利な 1 台に仕上げたものが M5Stack シリーズです。9 軸センサなどを搭載した上位シリーズもあります。

M5StackV, Sipeed Maixduino



同じ M5Stack シリーズでも、M5StackV は RISC V チップに LCD、 K210(AI チップ)から構成されます。ただし、Wi-Fi は未搭載です。Maixduno は、RISC V、K210、LCD、Wi-Fi、カメラから構成されます。両者は従来の Arduino GUI も使用できますが、基本的に MaixPy IDE という専用統合環境を用います。使用するプログラミング言語は Python です。
K210 は単体で Deep Learning 学習を行うには非力ですが、PC 等で学習した重みデータを使用し、MobileNet や Tiny-YOLO といった推測エンジンを使用できます。

演習１　Arduino を使った電気回路の設計



～のマークあるデジタルピンは PWM　(Plus Width Modulation：パルス幅変調…疑似アナログ：256 段階）が使えるピンを表します。通常、3、5、6、9、10、11 で PWM 出力が行えます。

アナログピンでは 0～1023 の 1024 段階でのアナログ入力ができます。

センサなどの接続の＋側を 5V に、－側を GND に接続していきます。

電気の動きを理解することはなかなか一筋縄ではいきませんが、高いところ（5V）から低いところ（GND）に、（水と全く同じではないですが）水のように移動しているものとイメージすると想像しやすいかもしれません。

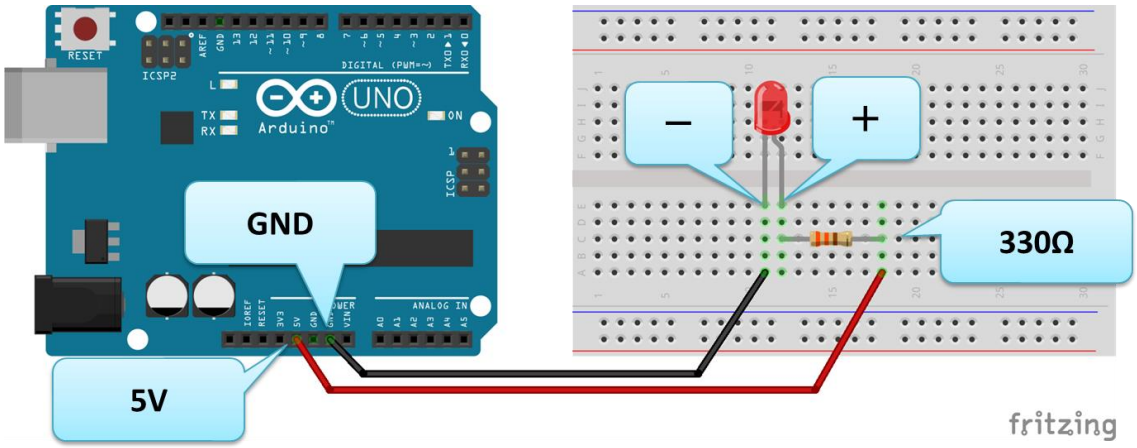
Arduino（×１）とブレッドボード（×１）をベースにして、次の課題の回路を順番に作成していきましょう。

① LED が点灯する回路

以下の部品のうち 330Ω抵抗以外は OSOYOO のセットに含まれています。330Ω抵抗が手元にない場合は OSOYOO のセットに含まれている 200Ω抵抗で代替しましょう。200Ω抵抗の場合、330Ω抵抗のときよりも LED への電力供給が増えるので、200Ω抵抗のときの方が LED が明るく光ります。

・赤色 LDE×１
・330Ω抵抗×１　　・・・330Ωが無い場合 200Ω抵抗でも可
・ジャンパーワイヤー×２　　・・・ワイヤーの外皮の色は何色でも可

下図にしたがって配線をすると自動的に電力が供給されて LED が点灯します。
抵抗を接続しないと LED が焼き切れることがあります。GND 側を先に接続するなど，接続する順番などにも注意しましょう。

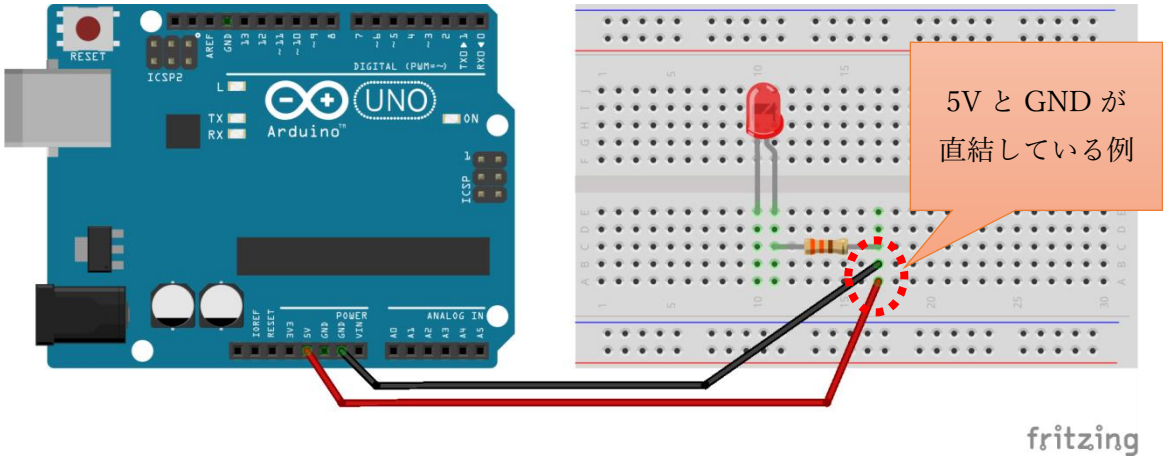


LED を物理的に点灯させる回路

電流が正しく流れない電気回路について

・ショートした回路

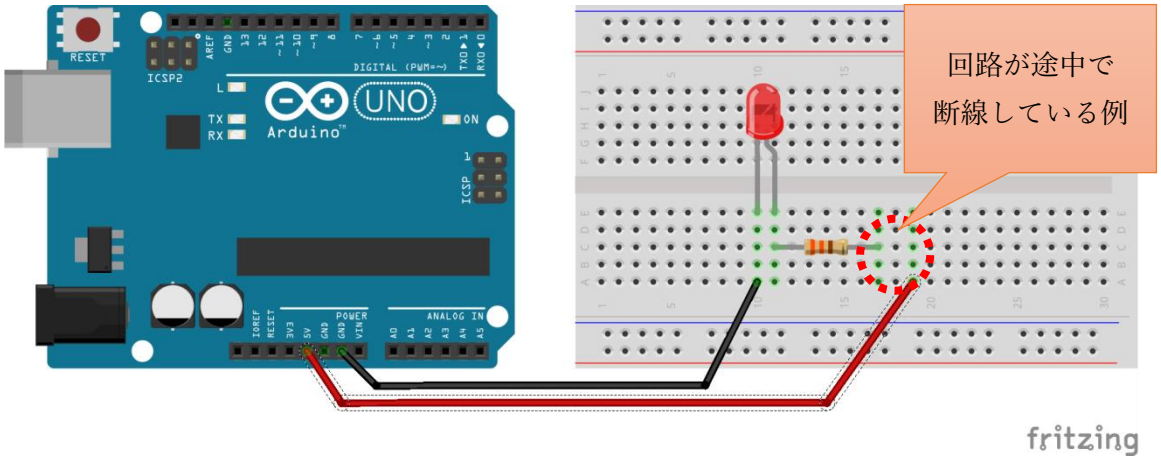
電流が正しい経路を通らずに近道してしまうものをショート（短絡）した回路と呼びます。ショートした回路では電流が流れすぎてしまい、発熱し火事につながることもあるので、このような回路は避けましょう。ショートしたときに電流が流れすぎるのを防ぐために、電流が流れすぎると溶断するヒューズやブレーカーなどをつけたりします。



ショートした回路の例

・オープンした回路

配線がうまくいっておらず回路が断線してしまっているものを「オープンしている」、「開放している」、「断線している」などと呼びます。英語では Open circuit と呼びます。オープンした回路では意図したように電流が正しく流れません。



オープンした回路の例

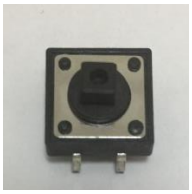
② スイッチで LED を ON・OFF する回路

赤色 LDE × 1

330Ω抵抗 × 1

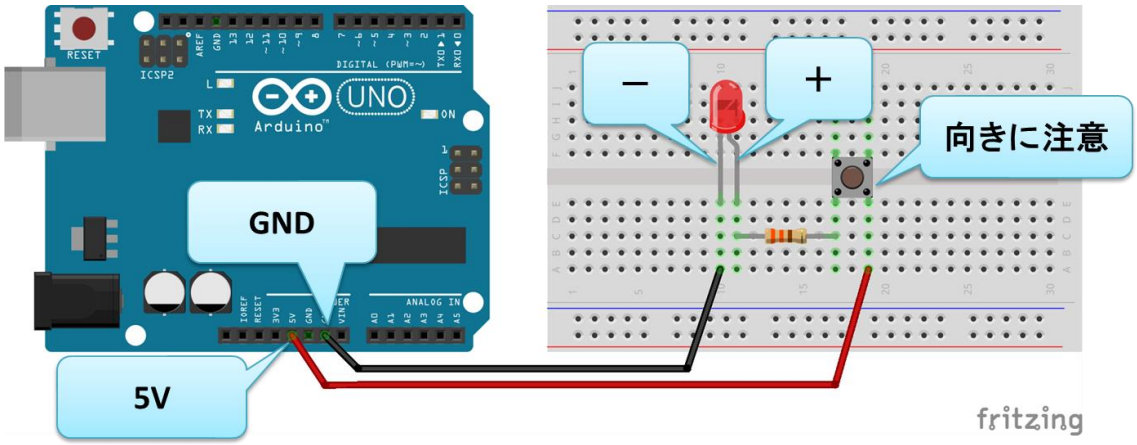
ジャンパーワイヤー × 2

タクトスイッチ × 1

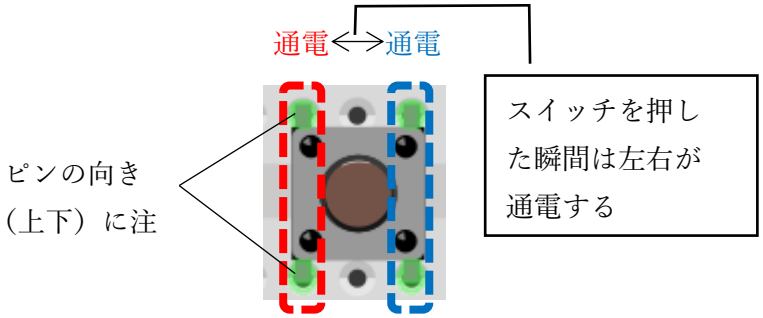


タクトスイッチ

下図のように配線すると、スイッチを押すと物理的に配線が接続されて電源が供給され LED が点灯します。



スイッチで LED を物理的に ON/OFF させる回路



タクトスイッチの通電

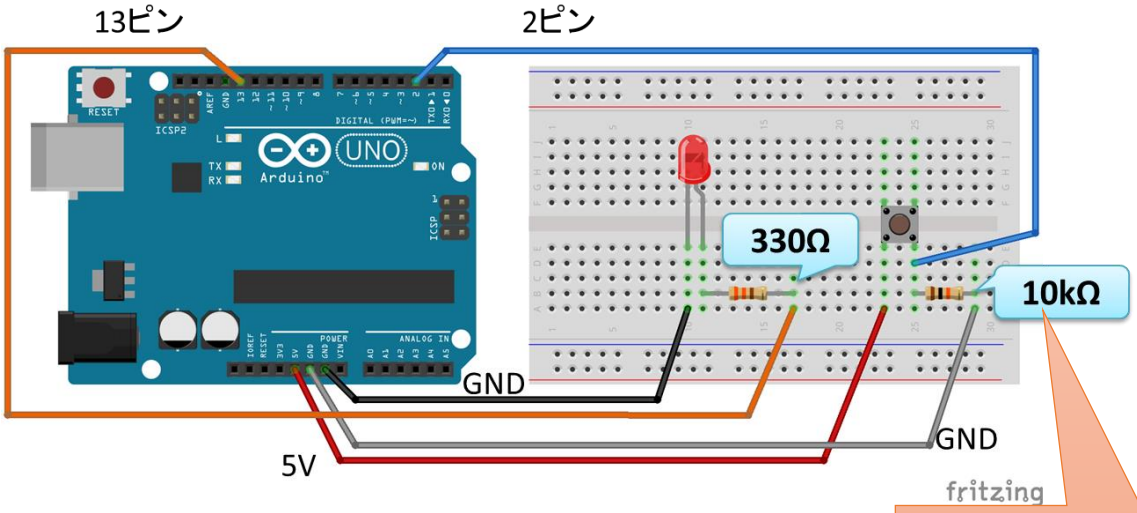
③　スイッチを押したときに LED を ON する回路とプログラム

赤色 LDE × １　　10kΩ抵抗 × １　　330Ω抵抗 × １

ジャンパーワイヤー × ５　　タクトスイッチ × １

Arduino　IDE から Arduino に書き込むプログラムは、Arduino　IDE のメニュー［ファイル］→［スケッチ例］→［Digital］→［Button］から読み込むことができます。プログラムの説明は次のサンプルプログラムのコメントを参考にしてください。

開いたファイルは別名（任意）のファイルに保存しましょう。



スイッチを押したときに LED を ON する回路

スイッチ OFF 時の挙動を安定化させるためのスイッチ～抵抗～GND への配線

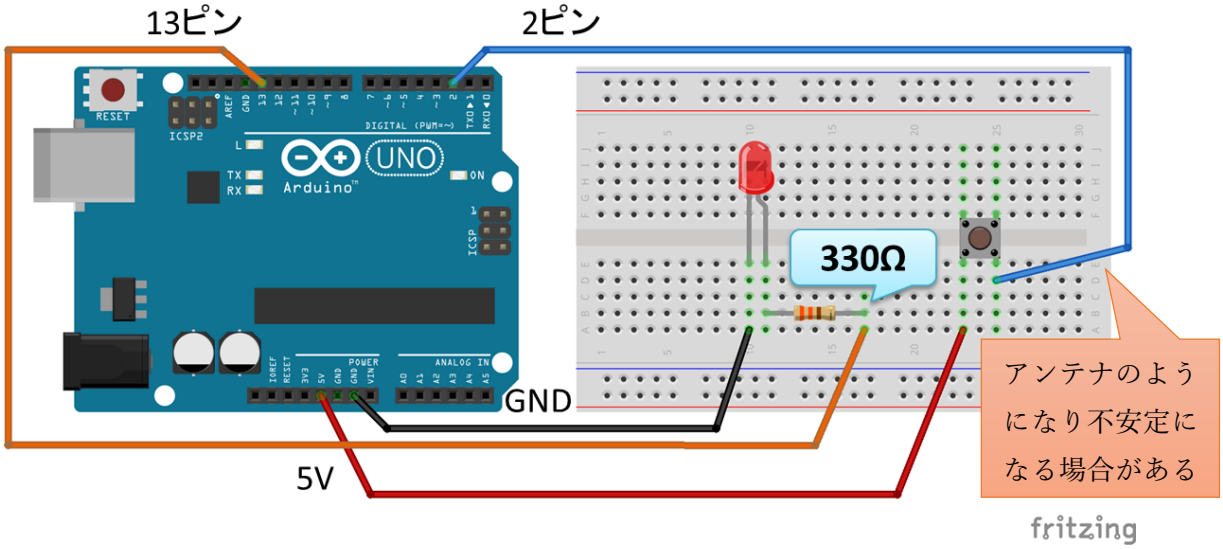
スイッチを押したときに LED を ON するプログラム

```
const int buttonPin = 2;     // ボタンを接続するピン
const int ledPin =  13;      // LED を接続するピン

int buttonState = 0;         // ボタンの状態変数を初期化

void setup() {
  pinMode(ledPin, OUTPUT);  // 13 ピンをアウトプットモードに設定
  pinMode(buttonPin, INPUT);  // 2 ピンをインプットモードに設定
}



void loop() {
  buttonState = digitalRead(buttonPin);  // 2 ピンからデジタルで読み取り
```

```
  if (buttonState == HIGH) {  // 2 ピンからの読み取りが HIGH の場合
    digitalWrite(ledPin, HIGH);  // 13 ピンの LED を点灯
  } else {
    digitalWrite(ledPin, LOW);  // その他の場合、13 ピンの LED を消灯
  }
}
```

※先ほどの回路ではスイッチに抵抗が接続されています。これは回路を安定化させるプルダウンという接続方法になります。

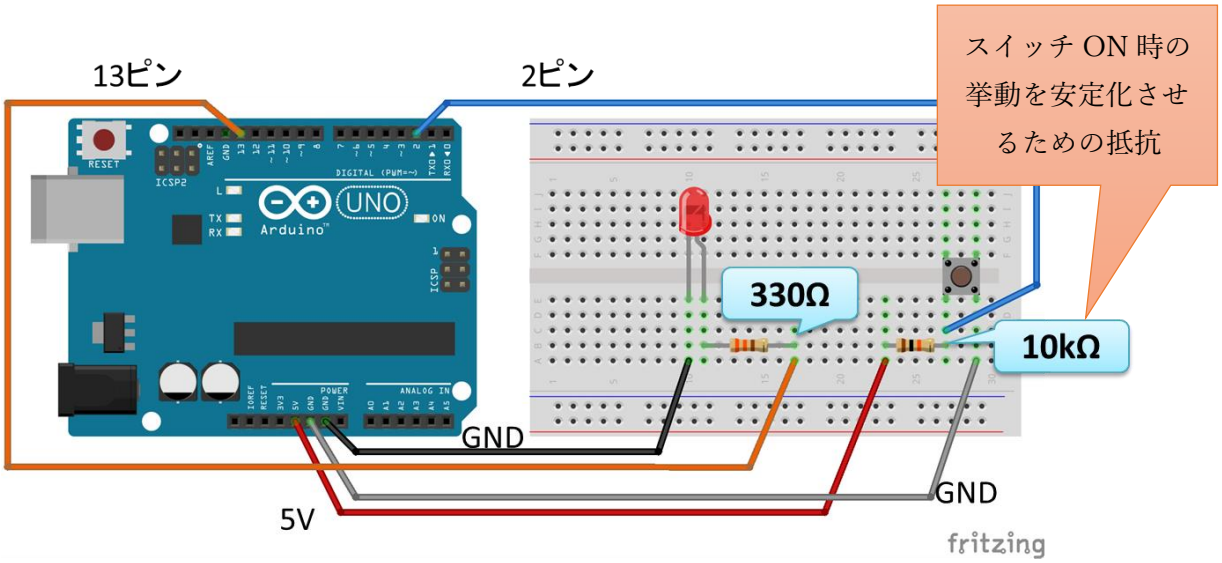
③の図で、もしスイッチに接続されている抵抗～GND の配線が下図のようになかった場合、プログラムが不安定になる可能性があります。③の回路でもし抵抗がない場合、インプットである 2 ピンからジャンパーピンによる配線が始まり、この配線がスイッチまで接続されて終了することになります。この状態では、スイッチが押されていない場合、スイッチから先はどこにも配線されておらず途中で切れている状態です。これは 2 ピンからスイッチまでのジャンパーピンがいわゆるアンテナのような役割をすることがあり、周りの環境によって 2 ピンにノイズが混入しプログラムが誤動作を引き起こす場合があります。



不安定になる回路の例

このような誤動作を防ぐために、③の回路ではスイッチから抵抗を経て GND に接続しています。この方式をプルダウンと呼びます。

プルダウンの方式の他に、プルアップという方式があります。
次の図はプルアップの回路です。この回路に先ほどの③のプログラム（p.9）を書き込んだ場合、ボタンを押したときに LED は消灯し、ボタンを離したときに LED は点灯します。



プルアップ方式の回路の例

プルダウンやプルアップは回路を安定させるためのテクニックです。回路設計のテクニックには、プルダウンとプルアップの他にも、ピンから外部へ電流を流すソース方式（吐き出し方式）や、ピンに向けて電流を流すシンク方式（吸い込み方式）といったように色々な回路設計の手法があります。**ここで、スイッチに接続された GND を外すと回路が不安定になり、場合よっては LED が点灯したままになります。**

Arduino では内部の抵抗（20k～50kΩ）と回路でプルアップ方式を実現できます。p.6 の回路とプログラムで、関数 setup()内の 2 ピンのモード設定の行を以下に変更するとボタンを押したときに OFF になり、ボタンを離したときが ON になるプルアップ方式が実現できます。次の演習課題で実際にやってみましょう。

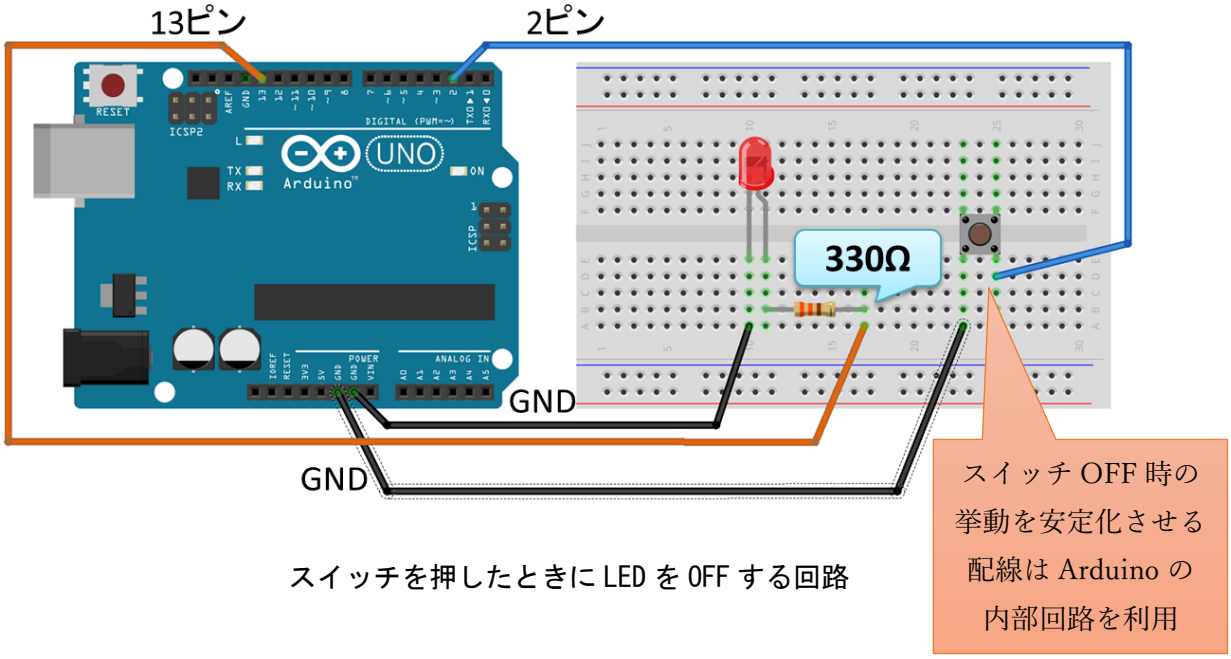
```
pinMode(buttonPin, INPUT_PULLUP);  //ピンモードをプルアップ方式に設定する
```

④　スイッチを押したときに LED を OFF する回路とプログラム

赤色 LDE × 1　　330Ω抵抗 × 1　　ジャンパーワイヤー × 4

タクトスイッチ × 1

Arduino IDE から Arduino に書き込むプログラムは、Arduino IDE のメニュー［ファイル］→［スケッチ例］→［Digital］→［Button］から読み込むことができます。プログラムの説明は次のサンプルプログラムのコメントを参考にしてください。



スイッチを押したときに LED を OFF する回路

スイッチを押したときに LED を OFF するプログラム

```
const int buttonPin = 2;     // ボタンを接続するピン
const int ledPin =  13;      // LED を接続するピン

int buttonState = 0;         // ボタンの状態変数を初期化

void setup() {
  pinMode(ledPin, OUTPUT);  // 13 ピンをアウトプットに設定
  pinMode(buttonPin, INPUT_PULLUP);  //2 ピンをプルアップ方式に設定する
}
```

③のプログラムで
この行のみを修正

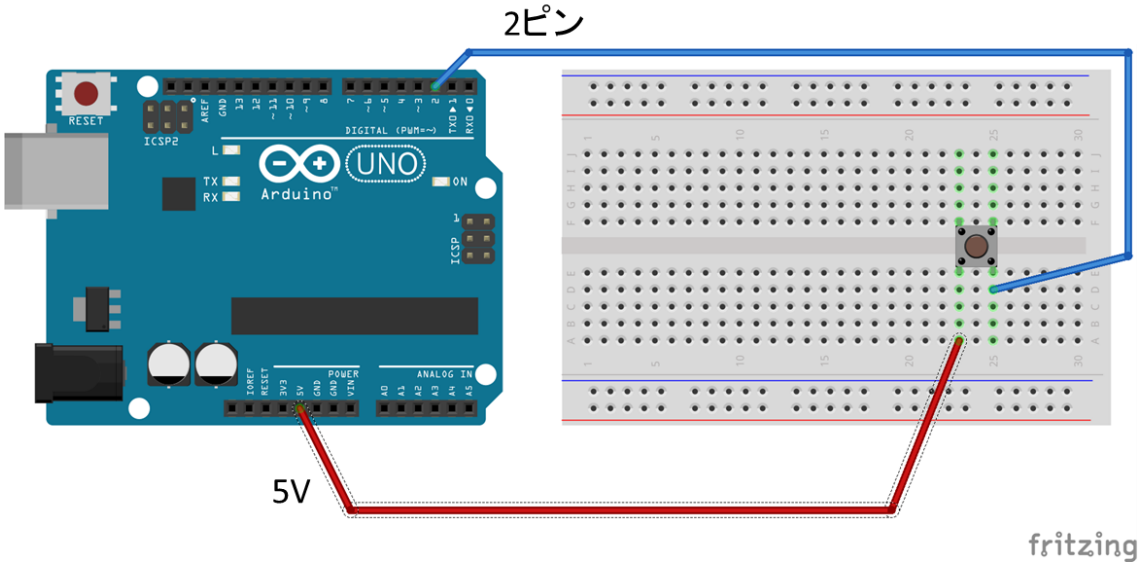
```
void loop() {
  buttonState = digitalRead(buttonPin);  // 2 ピンからデジタルで読み取り

  if (buttonState == HIGH) {  // 2 ピンからの読み取りが HIGH の場合
    digitalWrite(ledPin, HIGH);  // 13 ピンの LED を点灯
  } else {
    digitalWrite(ledPin, LOW);  // その他の場合、13 ピンの LED を消灯
  }
}
```

プルアップとプルダウンの基本的な考え方

１）ダメな回路１（オープンした回路）

「2 ピンをインプット用のピンに設定して High（5V）か Low（0V）を感知したい・・・」と考えて、下図のような回路をつくるのは結論から言うと「ダメ」です。スイッチが ON になっているときは OK ですが、OFF になっているとき、p.5 で紹介したダメな回路であるオープンな回路になってしまっています（特に 2 ピンが不安定になってしまいます）。

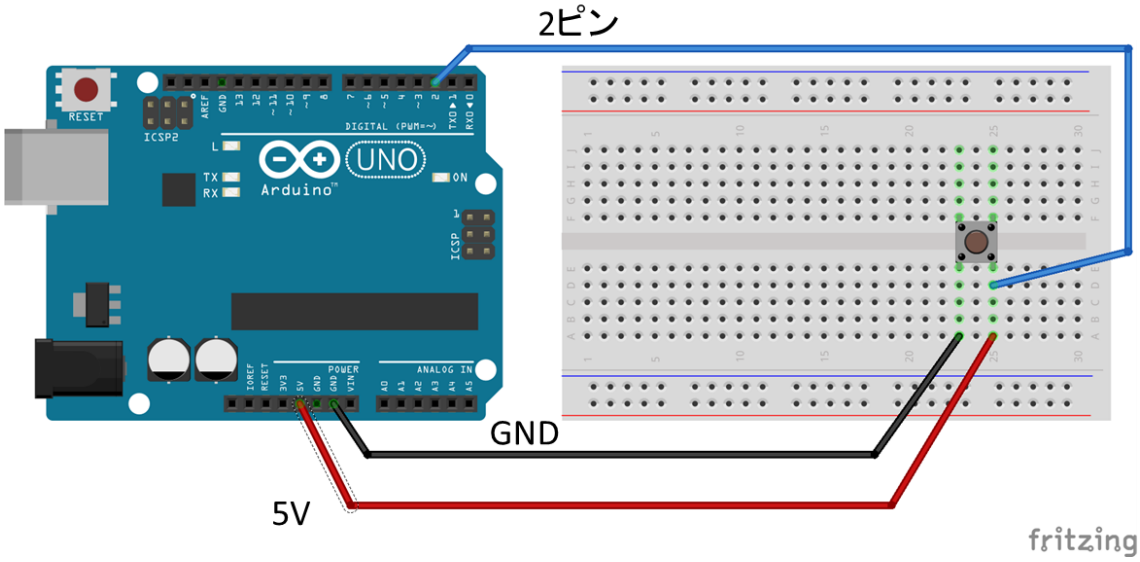
→解決版は３）を参照



オープンした回路

２）ダメな回路２（ショートした回路）

『上の１）でスイッチが OFF のときにオープンな回路になってしまっていた・・・。それではスイッチが OFF のときは GND につなげよう！』と考えて、次の図のような回路をつくるのも結論から言うと「ダメ」です。



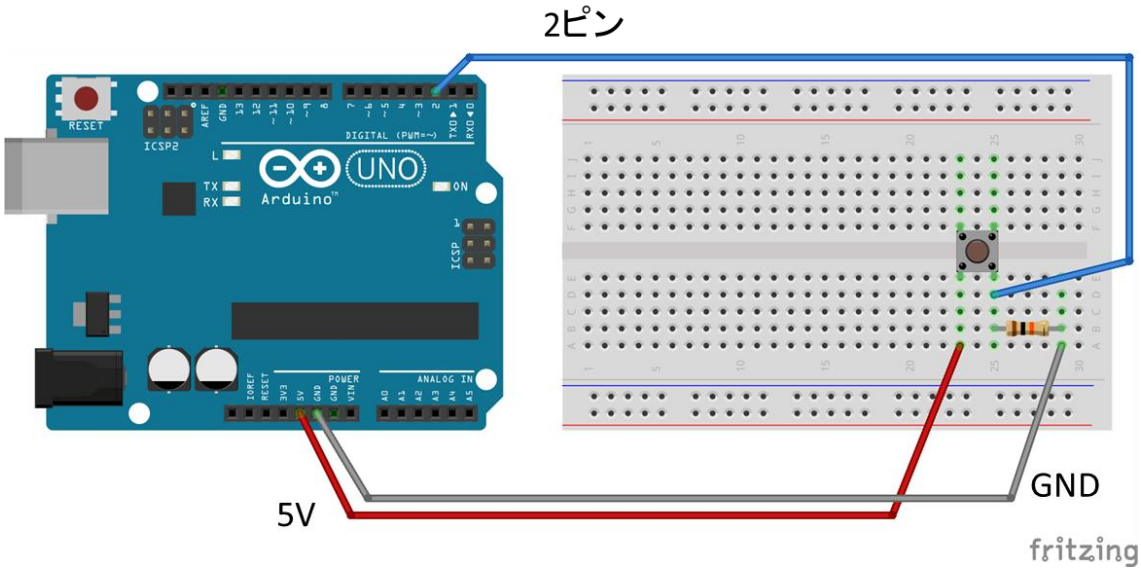
ショートした回路

この回路では、スイッチが ON のときスイッチ側にも 5V の電圧がかかることになります。スイッチは通常、ほとんど抵抗値をもっていません。したがって、オームの法則（V=R×I）を使ってスイッチに流れる電流を計算しようとすると､その計算式は I＝V／R になります。スイッチの抵抗は前述したようにほとんどありません（R≒0）。そうすると前述の計算式の分母の R が 0 に近づくとすると I は無限大に近づいてしまいます。無限大の電流がながれることは実際にはありませんが、それでも非常に大きな電流がスイッチに流れてしまい発熱などの危険があることが分かります。これがショートしたダメな回路になります。

→解決版は４）を参照

３）OK な回路１（プルダウン回路）

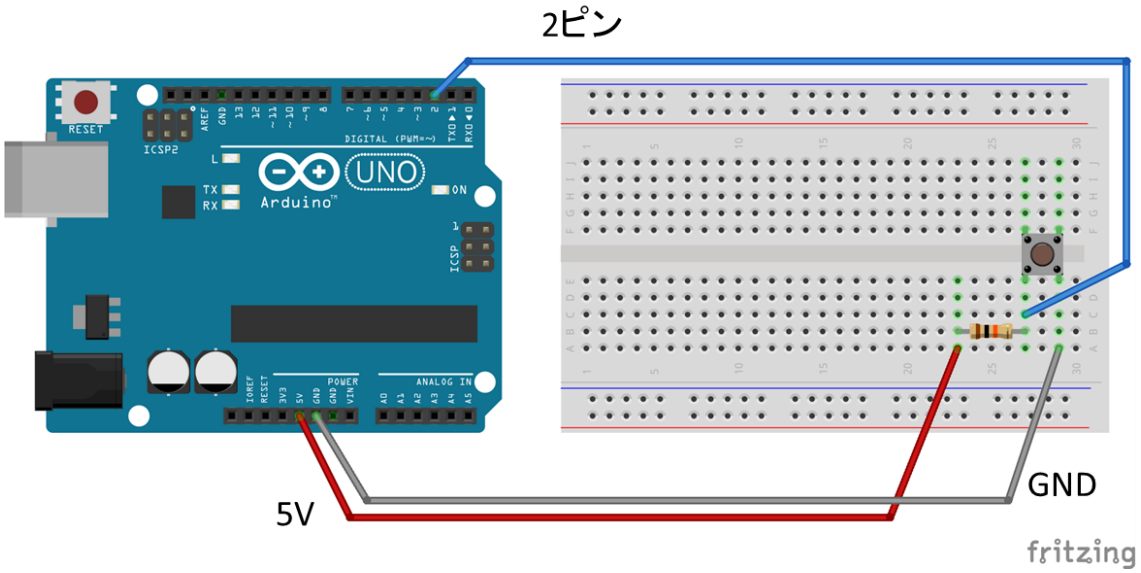
前述の１）ではスイッチが OFF のときオープンな回路になってしまっていました。それでは抵抗もつかってスイッチが OFF のときに GND につなげてやりましょう。その回路が次の図のようになります。この回路では抵抗のおかげでショートすることなく、スイッチが ON のとき 2 ピンには 5V の電圧がかかり High を検出します。逆に、スイッチが OFF のとき 2 ピンは GND につながり Low を検出します。



プルダウン回路の例

４）OK な回路２（プルアップ回路）

前述の２）ではスイッチが ON のとき、スイッチの抵抗値が 0 だったためにショートした回路になってしまっていました。それでは抵抗も使って回路をつくってやりましょう。その回路が以下になります。この回路では抵抗のおかげでショートすることなく、スイッチが ON のとき 2 ピンは GND につながり Low を検出します。逆に、スイッチが OFF のとき 2 ピンには 5V の電圧がかかり High を検出します。



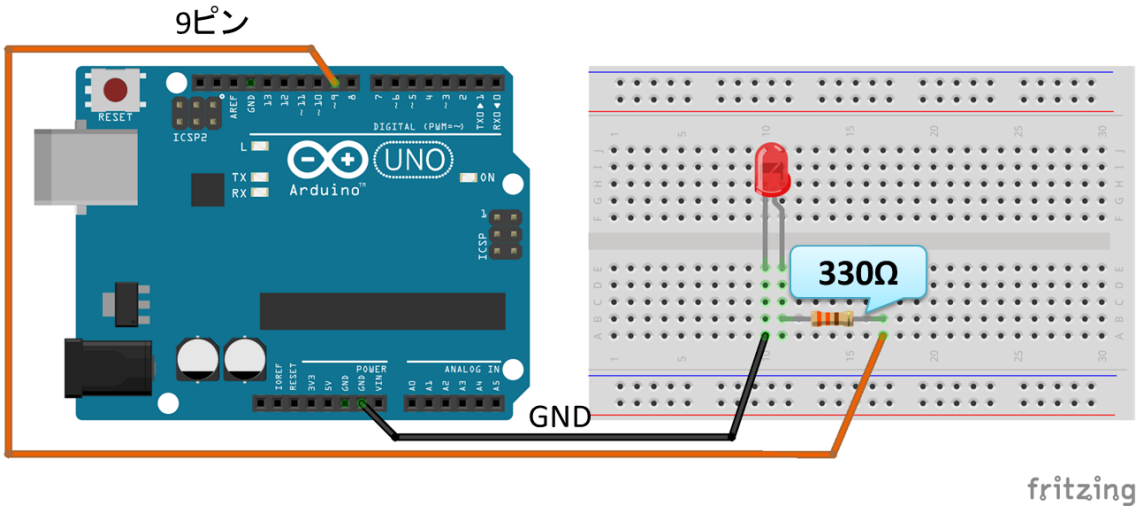
プルアップ回路の例

⑤　PWM を使ったアナログ出力による LED 点灯

赤色 LDE×１　　330Ω抵抗×１　　ジャンパーワイヤー×２

※プログラムは［ファイル］→［スケッチ例］→［Basics］→［Fade］から読み込む

※9 ピンは PWM マーク（～）があるので（疑似的な）アナログな操作が可能なピン



PWM を使ったアナログ出力による LED を自動で点灯させる回路

PWM を使ったアナログ出力による LED を自動で点灯させるプログラム

```
int led = 9;           // LED を接続するピン
int brightness = 0;    // LED の明るさ
int fadeAmount = 5;    // 変化量

void setup() {
  pinMode(led, OUTPUT);  // 9 ピンをアウトプットに設定
}

void loop() {
  analogWrite(led, brightness);  // 9 ピンに LED の明るさの値を設定

  brightness = brightness + fadeAmount;  // LED の明るさを変化量分変える

  if (brightness <= 0 || brightness >= 255) {//LED の明るさの範囲を 0～255 に設
定
```

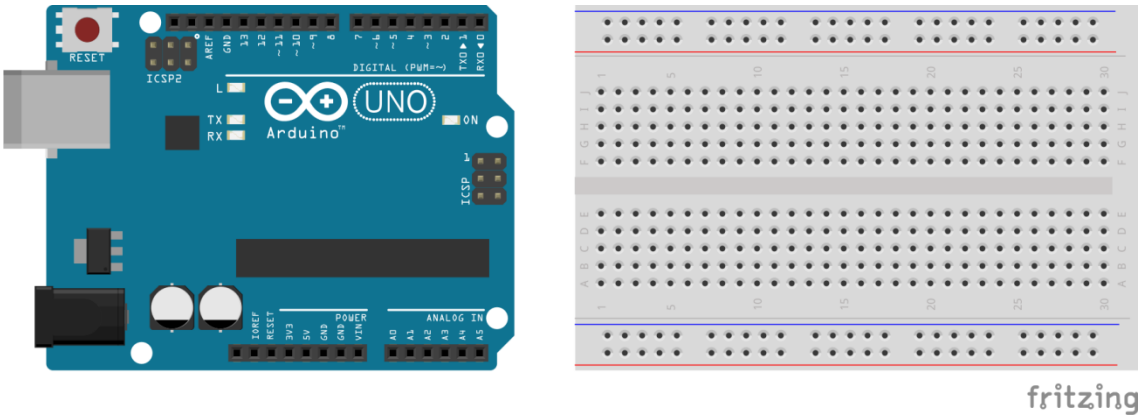
```
    fadeAmount = -fadeAmount;
  }
  delay(30);  // 30 ミリ秒待機
}
```

※delay 関数の引数は msec ですが、unsigned long 型です。32767 より大きい整数を指定するときは、値の後ろに UL をつけます。　例：delay(1000UL * 60 * 5) // 5 分待機

⑥　応用編：LED の種類や個数を変更
　　※作成した回路の回路図を描き抵抗値を書き込む

これまでやってきた①～③をもとに LED の種類を変えたり個数を変えたりして回路を作ってみましょう。以下の中から課題を選んで回路を作成し、回路図を描いた後に、それぞれの抵抗の値を計算して書き込んでみましょう。

A）LED の色を変えて適切な抵抗を選びましょう。
B）LED の個数を直列接続もしくは並列接続で増やして適切な抵抗を選びましょう。
C）LED の個数を直列接続と並列接続の 2 種類で増やして適切な抵抗を選びましょう。



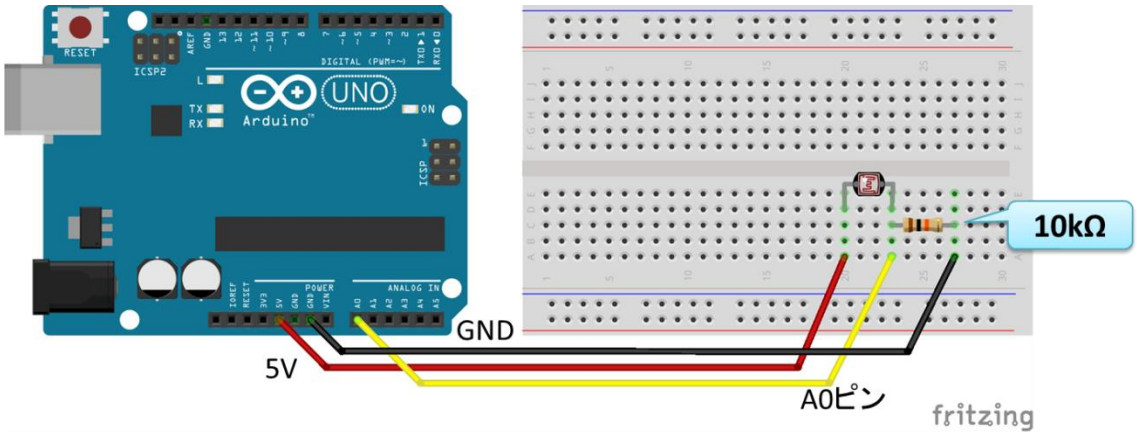
演習２　Arduino とセンサを使った回路設計

光センサ（アナログ

以下の回路を順番に作成していきましょう。

環境センサ
①光センサの利用
光センサ×１　　・・・±の接続はどちらでも OK
10Ω抵抗×１
ジャンパーワイヤー×３

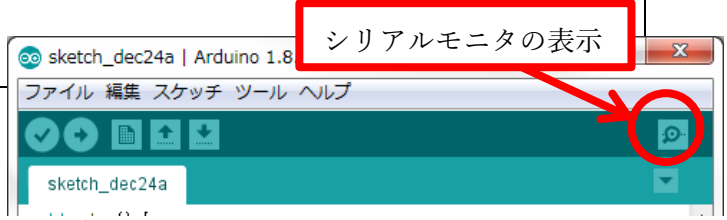
※プログラムは［ファイル］→［スケッチ例］→［Basics］→［AnalogReadSerial］から読み込む



光センサの値を読み込む回路

光センサの値を読み込むプログラム（シリアルモニタに値を表示）

```
void setup() {
  Serial.begin(9600);  // シリアルモニタに出力するためのシリアル通信の設定
}

void loop() {
  int sensorValue = analogRead(A0);  // A0 ピンから値を受け取る
  Serial.println(sensorValue);  //受け取った値をシリアルモニタに出力する
  delay(1);  // 1 ミリ秒待機
}
```

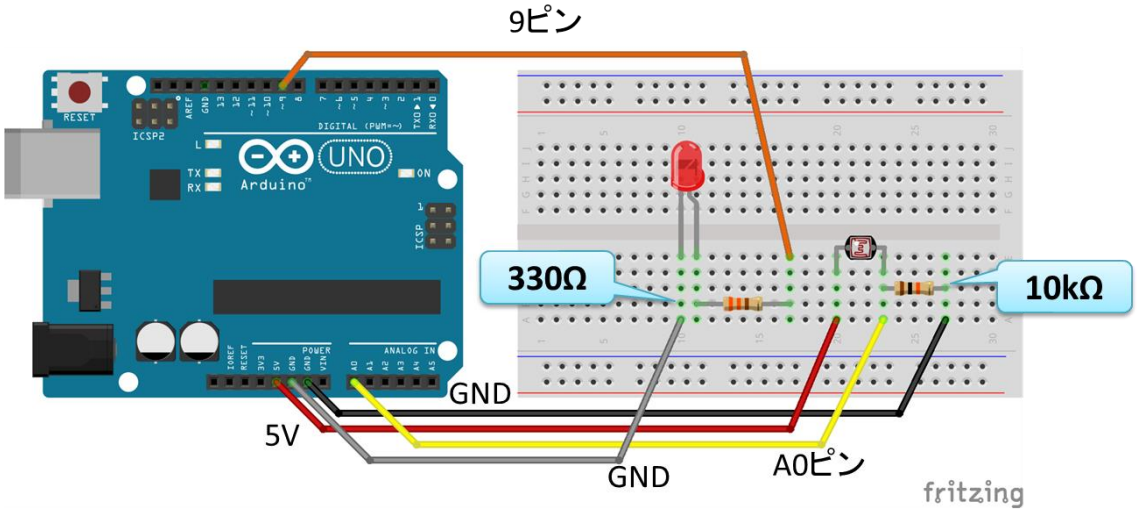


②光センサによる LED 点灯

光センサ×１　　赤色 LED×１　　10Ω抵抗×１

330Ω抵抗×１　　ジャンパーワイヤー×５

※プログラムは［ファイル］→［スケッチ例］→［Analog］→［AnalogInOutSerial］から読み込む



光センサの値に対応させて LED を点灯させる回路

※値を強調するなら、抵抗 10kΩを 200Ωなどの小さいものに入れ替える

光センサの値に対応させて LED を点灯させるプログラム

```
const int analogInPin = A0;  // 光センサを接続するピン
const int analogOutPin = 9;  // LED を接続するピン

int sensorValue = 0;         // 光センサの値
int outputValue = 0;         // LED にアウトプットする値

void setup() {
  Serial.begin(9600);   // シリアルモニタに出力するためのシリアル通信の設定
}

void loop() {
  sensorValue = analogRead(analogInPin);  // 光センサの値を受け取る
  outputValue = map(sensorValue, 0, 1023, 0, 255); // 光センサの値を 0-255 に変換
  analogWrite(analogOutPin, outputValue);  // LED にアウトプットの値を設定
```

値変換の調整をしていないのでセンサの感度と光度の対応に偏りがある可能性有り

```
  // シリアルモニタに値を出力する
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("¥t output = ");
  Serial.println(outputValue);
  delay(2);  // 2 ミリ秒待機
}
```

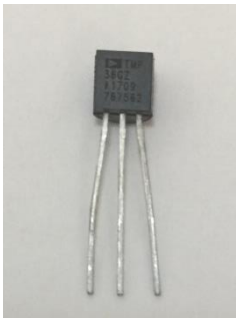
ここで、

outputValue = map(sensorValue, 0, 1023, 0, 255);

↓

outputValue = **255** - map(sensorValue, 0, 1023, 0, 255);

とするとどうなるでしょうか？



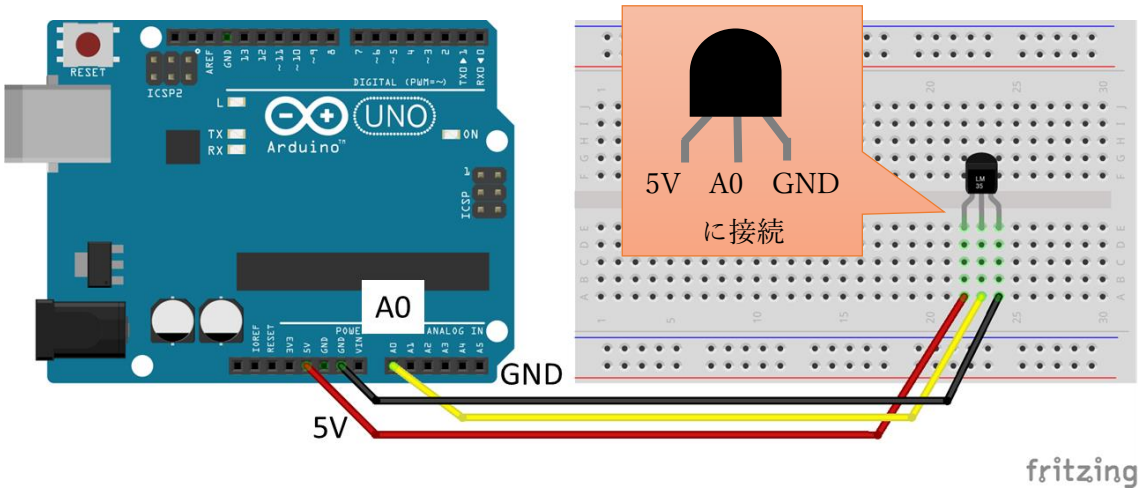
温度センサ（アナログ

③温度センサ TMP36 によるセンシング

温度センサ（TMP36/2N3904）×１

ジャンパーワイヤー×３

※プログラムは以下のサンプルプログラムを新規ファイルに書き込む（ファイル名は任意）



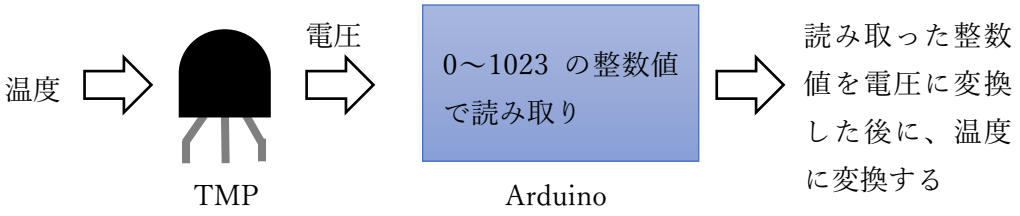
温度センサ TMP36 の値を読み取る回路

温度センサ TMP36 の値を読み取るプログラム（シリアルモニタに値を表示）

```
const int analogInPin = A0;  // 温度センサの出力に接続するピン

int sensorValue = 0;         // 温度センサの値

void setup() {
  Serial.begin(9600);   // シリアルモニタに出力するためのシリアル通信の設定
```

```
}

void loop() {
  float sensorValue = (float) analogRead(analogInPin); //温度センサの値を受け取る
  float ReadVolt = sensorValue * 5.0 / 1023.0; // 受け取った値を電圧に変換
  float Temperature = ReadVolt * 100.0 - 50.0; // 電圧を温度に変換

  // シリアルモニタに値を出力する
  Serial.print("Temperature = ");
  Serial.println(Temperature);

  delay(5000);  // 5秒待機
}
```

TMP36は温度を電圧に変換して出力しています。ArduinoはTMP36の出力している電圧を0～1023の整数値で読み取っているので、温度として認識するには整数値を変換し直す必要があります。



TMP36を経由した温度データの読み取りの流れ図

TMP36の仕様（の一部）はデータシートに以下のように記載してあります。

表4. TMP3xの出力特性

Sensor	Offset Voltage (V)	Output Voltage Scaling (mV/°C)	Output Voltage @ 25°C (mV)
TMP35	0	10	250
TMP36	0.5	10	750
TMP37	0	20	500

アナログ・デバイセズ社データシートより転載

https://www.analog.com/media/jp/technical-documentation/data-sheets/TMP35_36_37_jp.pdf

この表より、TMP36 は、オフセット電圧（入力がゼロの場合に流れてしまう電圧）が 0.5[V]、出力電圧は 1℃毎に 10[mV]（=0.01V）で変化し、出力電圧は 25℃で 750[mV]（=0.75V)となることが分かります。したがって、TMP36 のセンサの値は出力電圧を Vout[V]、温度を T[℃]とすると、

$$Vout[V] = 0.01[V/℃] * T[℃] + 0.5[V]$$

と表せることになります。さらにこの式を変形すると、

$$T = 100 * Vout - 50$$

となり、温度 T を求める計算式が導出できます。
また、Arduino のアナログピン A0～A5 は、10 ビット（＝1024）の AD（Analog to Digital）コンバータを持っています。したがって、5V の電圧の Arduino のボードの場合、入力電圧の範囲は 0V から 5V となり、関数 analogRead()は、0V から 5V の入力電圧を 0 から 1023 の 1024 個の整数値に変換して読み取っていることになります。したがって、analogRead()で読み取った値を N、出力電圧を Vout とすると、

$$Vout[V] = N * 5.0 / 1023.0$$

と表せることになります（読み取りの最大値が 1023 なので分母は 1023.0 です)。
したがって、先のプログラムでは、

```
float ReadVolt = sensorValue * 5.0 / 1023.0; // 受け取った値を電圧に変換
```

が

$$Vout[V] = N * 5.0 / 1023.0$$

に相当しています。また、

```
float Temperature = ReadVolt * 100.0 - 50.0; // 電圧を温度に変換
```

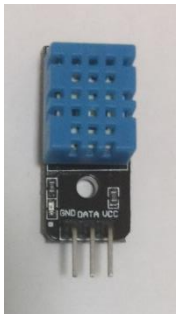
が

$$T = 100 * Vout - 50$$

に相当しています。

④　温湿度センサ DHT11 によるセンシング
温湿度センサ（DHT11）× 1
ジャンパーワイヤー× 3

※プログラムは公開されているライブラリと
サンプルプログラムを入手して利用する。

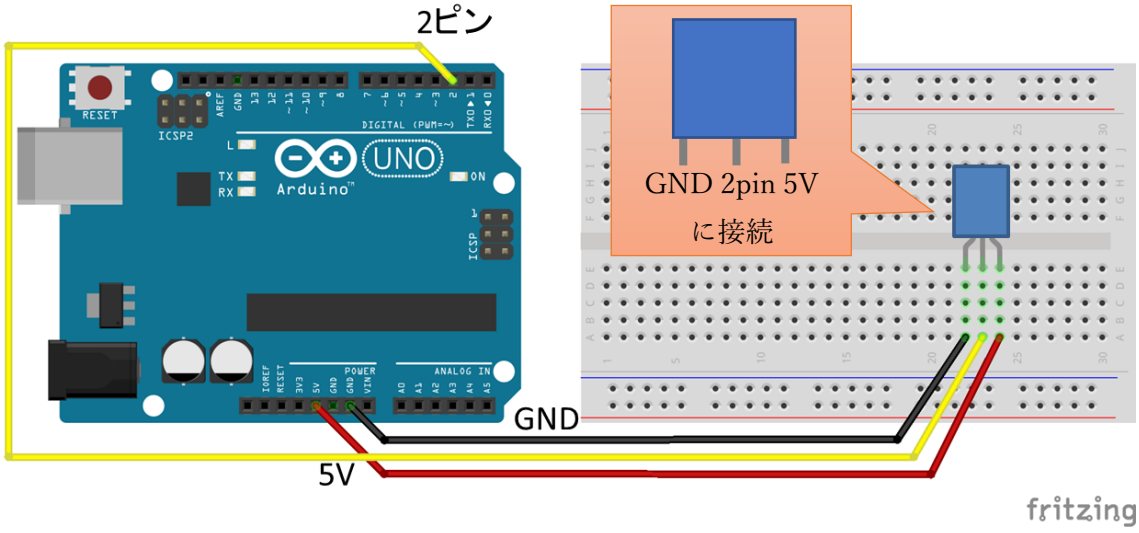


温湿度センサ（OSOYOO セット

１）GitHub に公開されているライブラリ「DHT-sensor-library-master.zip」を入手
https://github.com/adafruit/DHT-sensor-library
※上記の DHT 用のライブラリを利用するには別のライブラリ「Adafruit_Sensor-master.zip」も入手してインストールしておく必要あり（上記のサイトからも飛べます）
https://github.com/adafruit/Adafruit_Sensor

２）Arduino IDE のメニューから「スケッチ」→「ライブラリをインクルード」→「.ZIP 形式のライブラリをインストール…」を選択し、先ほどダウンロードしてきた zip ファイルを選択する。

３）プログラムは［ファイル］→［スケッチ例］→［DHT sensor library］→［DHTtester］から読み込む。



温湿度センサ DHT11 の値を読み取る回路

温湿度センサ DHT11 の値を読み取るプログラム（シリアルモニタに値を表示）

```
#include "DHT.h"  // 温湿度センサ DHT 用ライブラリの読み込み

#define DHTPIN 2     // 2 ピンに DHT 温湿度センサの出力ピンを接続

#define DHTTYPE DHT11   // DHT11 用の設定を有効にする ここのコメントアウトを外す
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321 用の設定をコメントアウト
//#define DHTTYPE DHT21   // DHT 21 (AM2301)用の設定をコメントアウト

DHT dht(DHTPIN, DHTTYPE);  // DHT 用のクラス DHT を定義

void setup() {
  Serial.begin(9600);  // シリアルモニタに出力するためのシリアル通信の設定
  Serial.println(F("DHTxx test!"));  // 説明文の出力

  dht.begin();  // dht の初期設定
}

void loop() {
  delay(2000);  // 2 秒待機

  float h = dht.readHumidity();  // 湿度データの読み込み
  float t = dht.readTemperature();  // 温度データの読み込み
  float f = dht.readTemperature(true); // 温度データ（華氏）の読み込み

  if (isnan(h) || isnan(t) || isnan(f)) {  // エラーの場合に出力
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  float hif = dht.computeHeatIndex(f, h);  // 華氏での体感温度の計算
  float hic = dht.computeHeatIndex(t, h, false);  // 摂氏での体感温度の計算

  // 湿度、温度、体感温度の値をシリアルモニタに表示
  Serial.print(F("Humidity: "));  // 湿度の表示
  Serial.print(h);
  Serial.print(F("%  Temperature: "));  // 温度の表示（摂氏と華氏）
  Serial.print(t);
  Serial.print(F("° C "));
  Serial.print(f);
  Serial.print(F("° F  Heat index: "));  //体感温度の表示（摂氏と華氏）
  Serial.print(hic);
  Serial.print(F("° C "));
  Serial.print(hif);
  Serial.println(F("° F"));
```

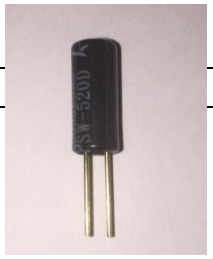
```
}
```

入力モジュール

⑤　傾斜スイッチの利用

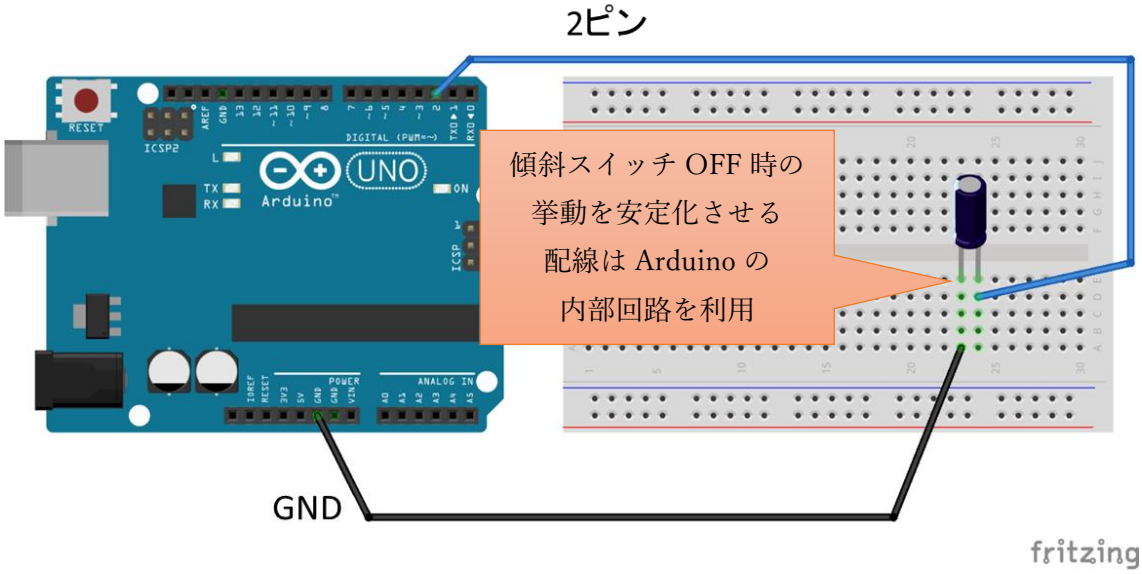
傾斜スイッチ×１

ジャンパーワイヤー×２



傾斜スイッチ（デジタル

※プログラムは［ファイル］→［スケッチ例］→［Digital］→［DigitalInputPullup］から読み込む・・・プルアップ方式の点に注意！



傾斜スイッチを傾けると内蔵 LED が消灯する回路

傾斜スイッチを傾けると内蔵 LED が消灯するプログラム

```
void setup() {
  Serial.begin(9600);   // シリアルモニタに出力するためのシリアル通信の設定
  pinMode(2, INPUT_PULLUP);  // 2 ピンからプルアップ方式で値を受け取る
  pinMode(13, OUTPUT);  //  基盤に内蔵されている LED にアウトプットする

}

void loop() {
  int sensorVal = digitalRead(2);  // 2 ピンから値を受け取る
  Serial.println(sensorVal);  //  受け取った値をシリアルモニタに出力する

  if (sensorVal == HIGH) {  // 傾斜時（＝傾斜スイッチ OFF 時）
```

```
    digitalWrite(13, LOW);  //  内蔵 LED を消灯
  } else {  // 通常時（＝傾斜スイッチ ON 時）
    digitalWrite(13, HIGH);  //  内蔵 LED を点灯
  }
}
```

※LED を 8pin に接続し、ソースコード中の 13 ピンの箇所を 8 ピンに変更してみましょう。（3 箇所あります）

⑥　ロータリーエンコーダによる LED 点灯

ロータリーエンコーダ×１　・・・回転スイッチ

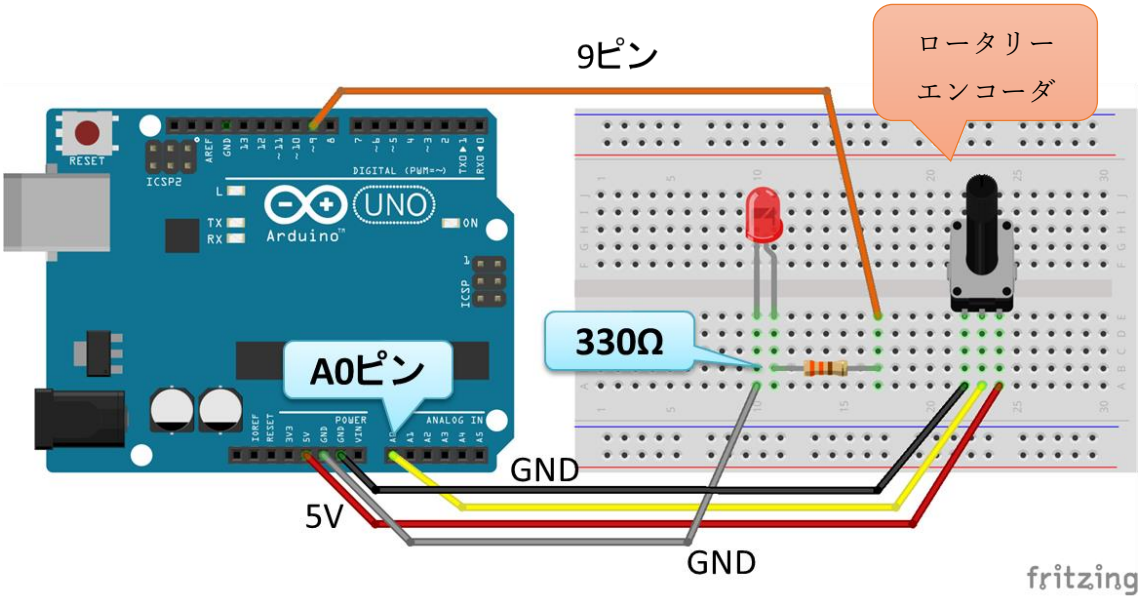
赤色 LED×１

330Ω抵抗×１

ジャンパーワイヤー×５

ロータリーエンコーダ（アナログ

※プログラムは［ファイル］→［スケッチ例］→［Analog］→［AnalogInOutSerial］から読み込む



回転スイッチで LED の光度を調整する回路

回転スイッチで LED の光度を調整するプログラム

```
const int analogInPin = A0;  // ロータリーエンコーダを接続するピン
const int analogOutPin = 9;  // LED を接続するピン
```

```
int sensorValue = 0;        // ロータリーエンコーダの値
int outputValue = 0;        // LED にアウトプットする値

void setup() {
  Serial.begin(9600);  // シリアルモニタに出力するためのシリアル通信の設定
}

void loop() {
  sensorValue = analogRead(analogInPin);  // ロータリーエンコーダの値を受け取る
  outputValue = map(sensorValue, 0, 1023, 0, 255); // ロータリーエンコーダの値
を
                                                    // 0-255 に変換する関数
  analogWrite(analogOutPin, outputValue);  // LED にアウトプットの値を設定

  // シリアルモニタに値を出力する
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("¥t output = ");
  Serial.println(outputValue);
  delay(2);  // 2 ミリ秒待機
}
```

⑦　赤外線コントローラから赤外線レシーバへの受信

赤外線リモコン×１　　赤外線レシーバ×１

赤色 LED×１　　330Ω抵抗×１

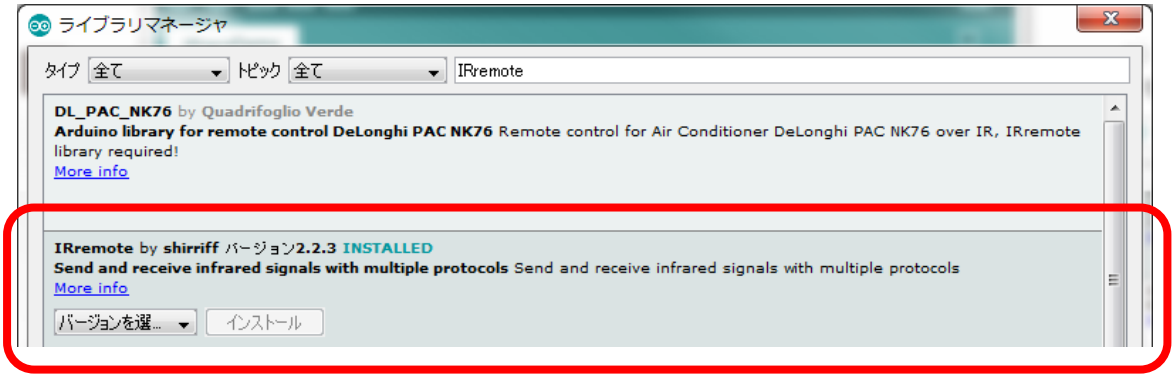
ジャンパーワイヤー×５

※プログラムは［ライブラリを管理...］から入手できるライブラリとサンプルプログラムを利用する。

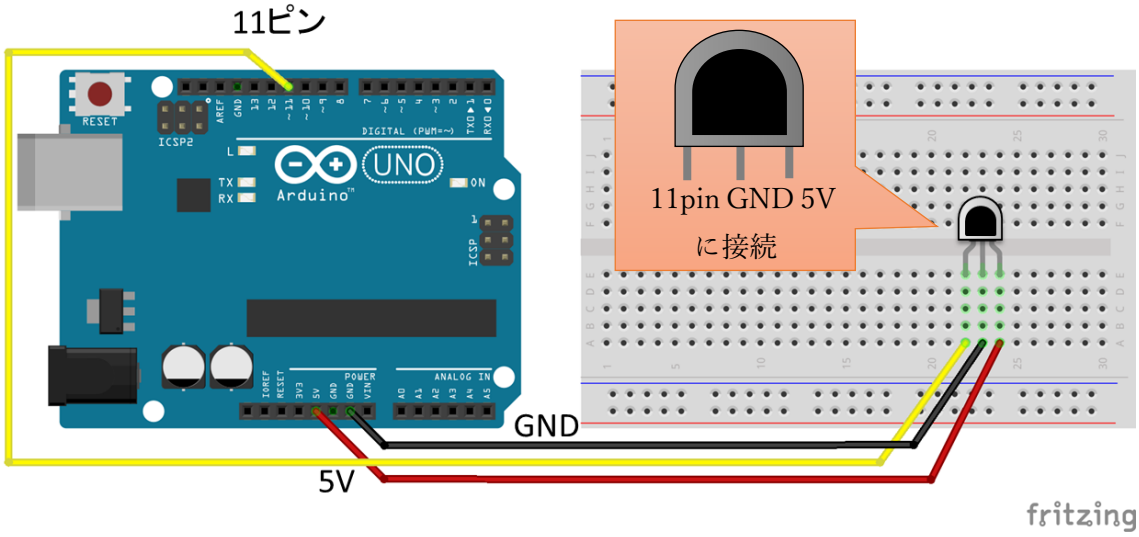
赤外線リモコンと赤外線レシーバ

１）Arduino IDE のメニューから「ツール」→「ライブラリを管理...」を選択し、ライブラリマネージャを開く。

２）ライブラリマネージャの検索機能を利用して、IRremote をキーワードにしてライブラリを検索し、IRremote by shirriff を選択しインストールする。



３）プログラムは［ファイル］→［スケッチ例］→［IRremote］→［IRrecvDemo］から読み込む。



赤外線コントローラから赤外線レシーバへ受信する回路

赤外線コントローラから赤外線レシーバへ受信するプログラム（受信値をシリアルモニタに表示）

```
#include <IRremote.h>  // 赤外線デバイス用ライブラリの読み込み

int RECV_PIN = 11;  // 赤外線レシーバの出力を 11 ピンに接続

IRrecv irrecv(RECV_PIN);  // 赤外線用のクラスを生成

decode_results results;  // 受信結果を格納する変数

void setup()
{
  Serial.begin(9600);  // シリアルモニタに出力するためのシリアル通信の設定
  Serial.println("Enabling IRin");  // 受信開始前
  irrecv.enableIRIn(); // 赤外線の受信開始
```

```
  Serial.println("Enabled IRin");  // 受信開始後
}

void loop() {
  if (irrecv.decode(&results)) {  //受信結果を 16 進数でシリアルモニタに表示
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
  delay(100);
}
```

受信した信号（16 進数）の先頭に 0x を付けて IF 関数すればボタン作が可能にな

DC5V ブザー

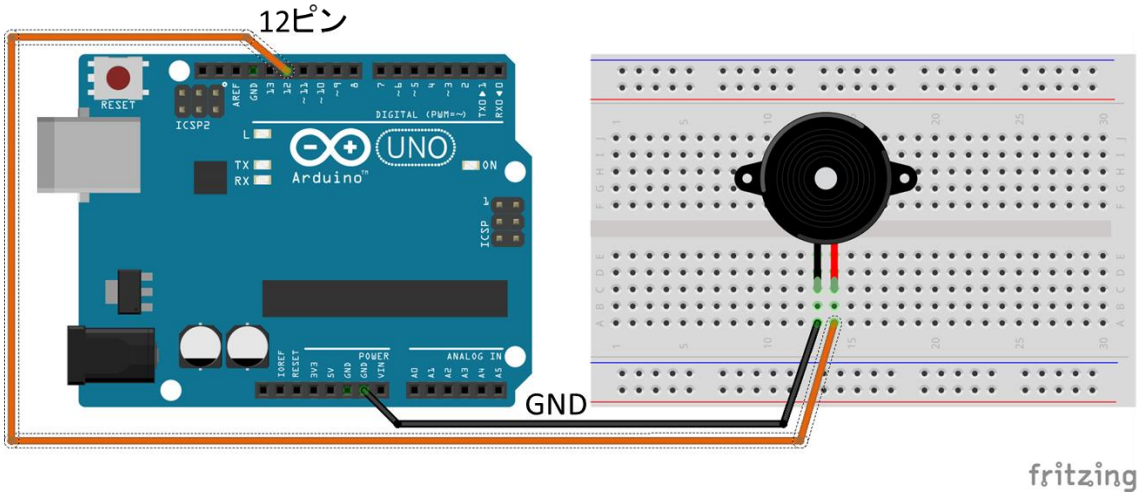
出力モジュール

⑧　ブザーによる音出力

DC5V ブザー×１

ジャンパーワイヤー×２

※プログラムは以下のサンプルプログラムを新規ファイルに書き込む（ファイル名は任意）



ブザーで音を出力する回路

ブザーで音を出力するプログラム

```
int pin = 12;      // ブザーの＋を 12 ピンに接続

void setup() {
}

void loop() {
     tone(pin,262,1000) ;  // ド
     delay(1500) ;         // 1500-1000 ほど待機
```

```
      tone(pin,294,1000) ;  // レ
      delay(1500) ;
      tone(pin,330,1000) ;  // ミ
      delay(1500) ;
      delay(1000) ;         // 1秒待機
}
```

⑨　LCDによるテキスト出力

LCD×１

ジャンパーワイヤー（凸～凹）×４

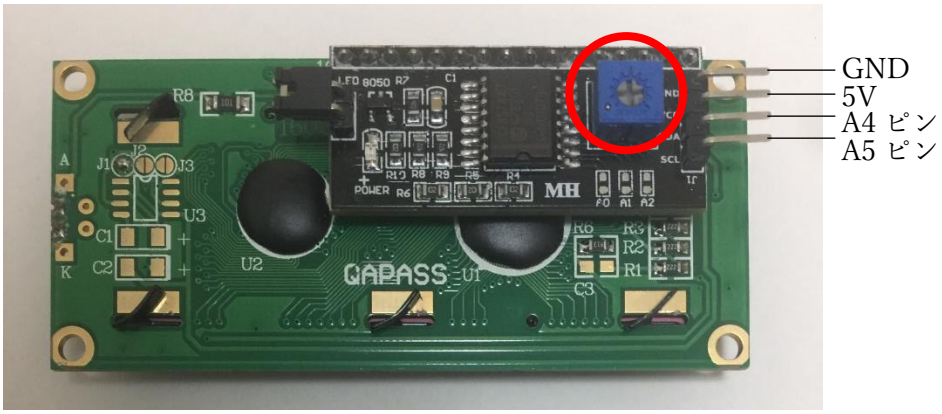
LCD（Liquid Crystal Display）

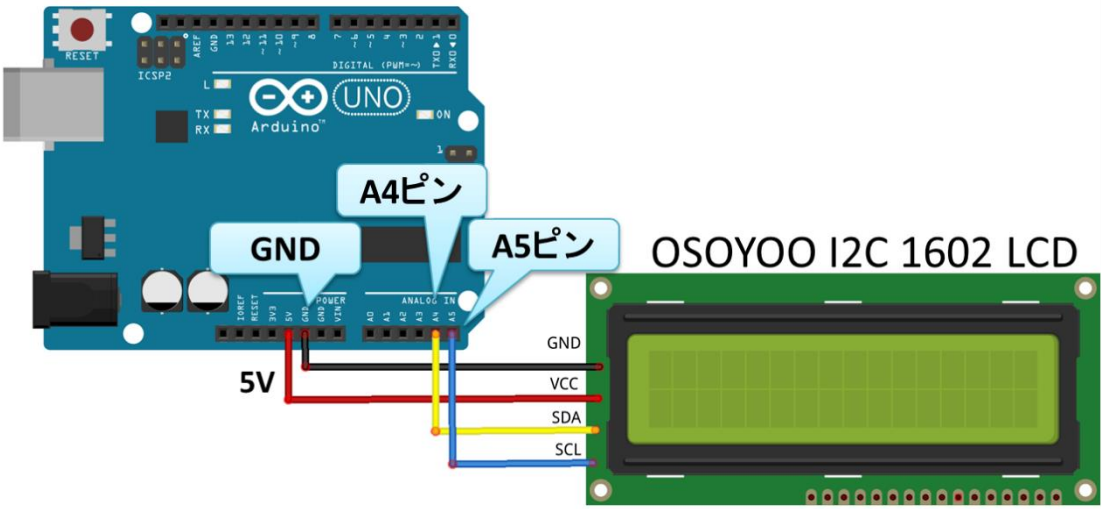
１）以下のOSOYOOのサポートサイトからライブラリ(LiquidCrystal_I2C.zip)をダウンロードする。 ※短縮URL: http://bit.ly/osoyooLCD
http://osoyoo.com/ja/2014/12/07/16x2-i2c-liquidcrystal-displaylcd/

２）Arduino IDEのメニューから「スケッチ」→「ライブラリをインクルード」→「.ZIP形式のライブラリをインストール…」を選択し、先ほどダウンロードしてきたzipファイルを選択する。

３）プログラムは以下のサンプルプログラムを新規ファイルに書き込む（ファイル名は任意）。
※もしくは、Arduino IDEのメニューから[ファイル]→[スケッチ例]→[LiquidCrystal_I2C]→[HelloWorld]から読み込んでもOKです(下のサンプルプログラムとは若干異なります)。

※LCDの裏面にコントラストを調整するつまみがあります。開封時等は、マイナスドライバーでつまみを左右に回して文字が表示されるように調整してください。コントラストが適切に設定されていないと、テキストが正しく出力されていても目視することができません。





LCD にテキストを表示する回路

LCD にテキストを表示するプログラム

```
#include <Wire.h>  // I2C デバイスとの通信用ライブラリ
#include <LiquidCrystal_I2C.h> // LCD 用ライブラリ

LiquidCrystal_I2C lcd(0x27,16,2);  // LCD の設定(設定用アドレス, 16 文字, 2 行)

void setup()
{
  lcd.init();  // LCD の初期化

  lcd.setCursor(0,0);  // 0 列 0 行に表示位置を移動、左上左端が(0, 0)
                       // 2 行目は(0, 1)
  lcd.backlight();  //LCD のバックライトを点灯
  lcd.print("Hello, world!");  // LCD への出力
}
```

注） 0x27, 0x3F など機器によって異なる

```
void loop()
{

}
```

lcd.clear();で表示を消去できる
表示を変えるときに利用

注）
OSOYOO のサポートページから ic2_scanner というファイルをダウンロードして Arduino でロードしましょう（または、ic2_scanner のプログラムを Arduino にコピペします）。
プログラムを Arduino に書き込んだあと、Arduino の操作画面でシリアルモニタ（「Tools」-「serial monitor」）を開くと I2C アドレスが表示されます。0x27、0x3F など機種によって異なります。詳細は次の OSOYOO LCD サポートページを参照してください。
OSOYOO LCD サポートページの URL
http://osoyoo.com/ja/2014/12/07/16x2-i2c-liquidcrystal-displaylcd/

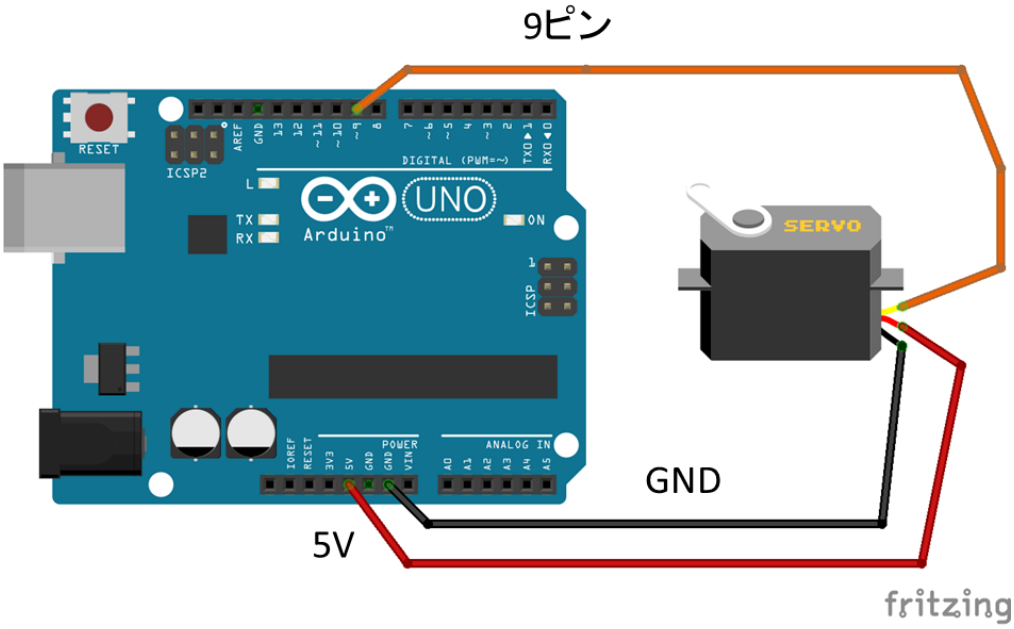
ic2_scanner ファイルの URL
http://osoyoo.com/wp-content/uploads/samplecode/ic2_scanner.txt

アクチュエータ
⑨ サーボの制御
マイクロサーボ×１
ジャンパーワイヤー×３

中央が 5V
茶色が GND
オレンジが 9 ピン

マイクロサーボ

※プログラムは［ファイル］→［スケッチ例］→［Servo］→［Sweep］から読み込む。



サーボを自動操作する回路

サーボを自動操作するプログラム

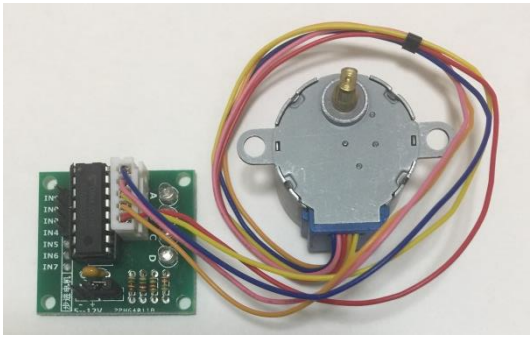
```
#include <Servo.h>

Servo myservo;  // サーボを動作させるクラスを定義

int pos = 0;    // サーボの角度を初期化

void setup() {
  myservo.attach(9);  // サーボの制御ピンを 9 ピンに接続
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // 0 度から 180 度に 1 度ずつ移動
    myservo.write(pos);              // 指定した角度に移動
    delay(15);                       // 0.015 秒待機
  }
  for (pos = 180; pos >= 0; pos -= 1) { // 180 度から 0 度に-1 度ずつ移動
    myservo.write(pos);              // 指定した角度に移動
    delay(15);                       // 0.015 秒待機
```

```
    }
}
```

⑩ モータの制御

ステッピングモータ×１

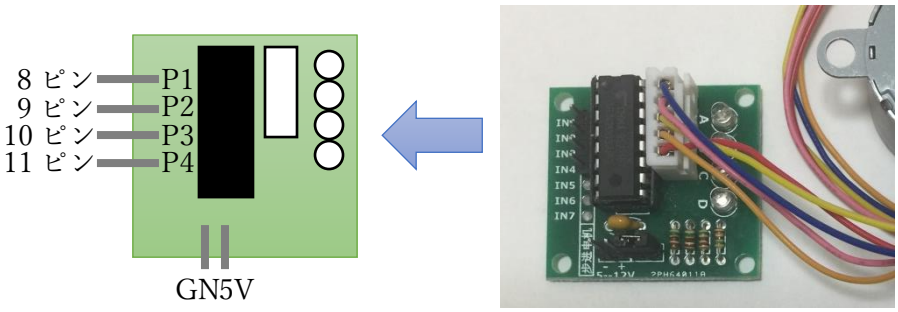
ジャンパーワイヤー×６



ステッピングモータ

※プログラムは、以下の OSOYOO のサポートサイトのサンプルプログラムを Arduino の新規ファイルにコピペする

http://osoyoo.com/wp-content/uploads/samplecode/stepper.txt



モータのドライバと Arduino の接続

⑪　応用編：各自で色々なセンサやモジュールを組合せて利用

※作成した回路の回路図を描く

これまでやってきた①～⑤をもとに様々なセンサやモジュールを組合せて回路を作ってみましょう。

A）センサ値を LCD に出力

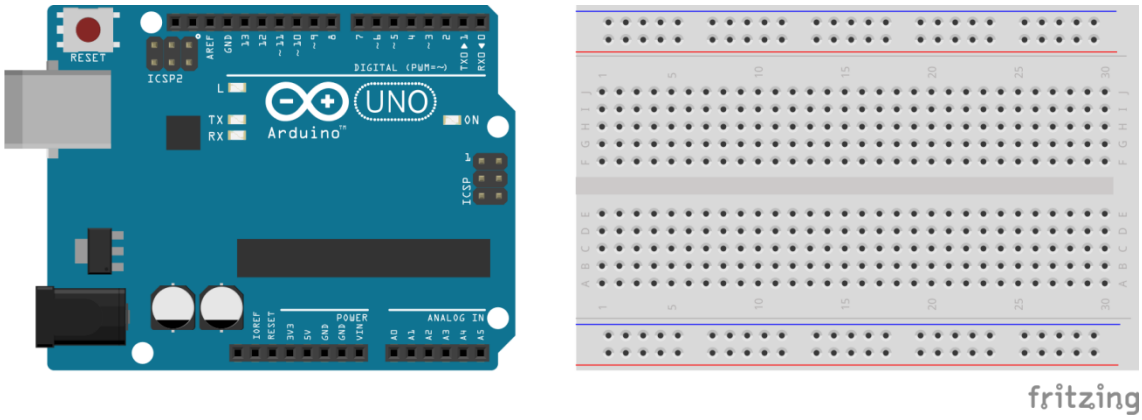
B）赤外線コントローラで LED を光らせる or ブザーで音を鳴らす

音階と周波数

	ド	レ	ミ	ファ	ソ	ラ	シ
1	131	147	165	175	196	220	247
2	262	294	330	349	392	440	494
3	523	587	659	698	784	880	988

C）ロータリーエンコーダでサーボやモータを動かす

D）その他



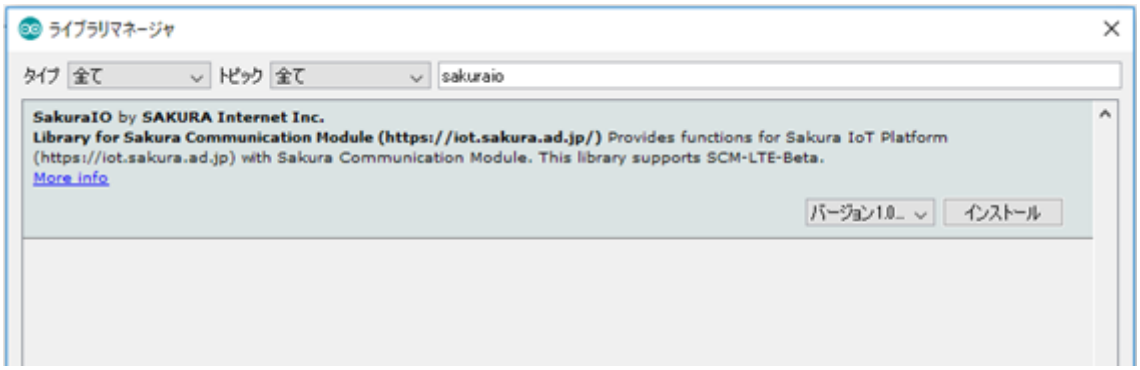
「OSOYOO 公式チュートリアル」Web サイト http://osoyoo.com/ja/2014/12/06/arduino-starter-kit/

も参考にしてみましょう。

演習３　さくら LTE モジュールの回路設計と利活用

sakura.io リファレンス
https://sakura.io/docs/

① 通信モジュールの接続とデータ送信
sakura.io リファレンスを参考に、LTE ボードの取り付けとモジュール登録を行います。Arduino を起動し、[スケッチ]＞[ライブラリをインクルード]＞[ライブラリを管理…]から検索機能を使用し、SakuraIO ライブラリをインストールします。



ライブラリをインストール後、連携サービスを登録します。sakura.io コントロールパネル（https://secure.sakura.ad.jp/iot/login）からプロジェクト内の［詳細］に進み、詳細の中の［連携サービス］のタブから［連携サービス追加］を選らんで、WebSocket を選択し適当な名前を付けて登録します。


#3438
テスト接続００１
データストアプラン
ライト
簡易位置情報提供機能
Off
ファイル配信
編集
削除
連携サービス
連携サービス追加
種類
名前
AWS test001
sakuratest2
Incomingtest
testservice
test
5 件中 1 - 5 件を表示
表示件数
50
①連携サービス
②連携サービス追加


ホーム › プロジェクト詳細 › 連携サービスカタログ
外部サービスとsakura.ioを連携し、データのやり取りを行います。
詳しくはドキュメントをご覧ください。 sakura.ioドキュメント - 連携サービス仕様
WebSocket
Outgoing Webhook
Incoming Webhook
MQTT Client
Datastore API
AWS IoT
Azure IoT Hub(α)：正式版提供に伴い廃止予定
Google Cloud Pub/Sub Publisher
Azure Event Hubs
Azure IoT Hub


連携サービス追加 - WebSocket
リアルタイムの双方向通信を行う連携サービスです。
詳しくはドキュメントをご覧ください。 sakura.ioドキュメント - WebSocket
名前
sakuratest1
①名前
②追加
追加


ホーム › プロジェクト詳細 › 連携サービス詳細
リアルタイムの双方向通信を行う連携サービスです。
詳しくはドキュメントをご覧ください。 sakura.ioドキュメント - WebSocket
#14986
WebSocket
名前
sakuratest1
URL 情報
URL
wss://api.sakura.io/ws/v1/d070
Token 情
Token
d070
編集
削除
最終到着データ(50件)
チャンネル別到着データ
接続
時刻
モジュール
タイプ
ペイロード
データはありません


ホーム › プロジェクト詳細
#3438
テスト接続００１
データストアプラン
ライト
簡易位置情報提供機能
Off
ファイル配信
編集
削除
モジュール
連携サービス
イベントアラート
連携サービス
連携サービス追加
種類
名前
設定
websocket
sakuratest1
プロジェクト詳細画面から上の連携サービス詳細への移動

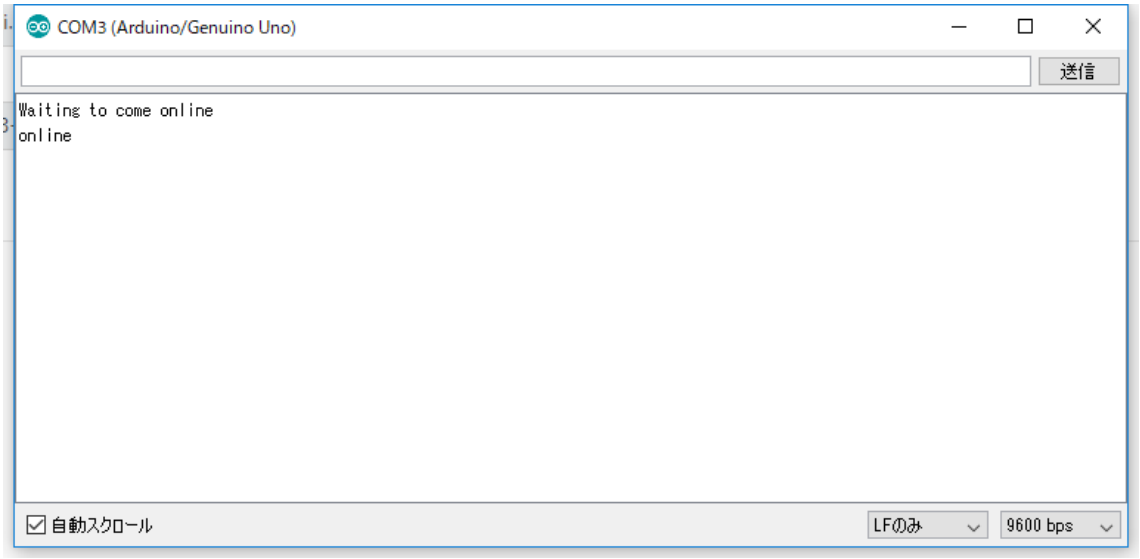
これで WebSocket の連携サービスが登録できました。次に、サンプルプログラムを入力し、実行して下さい。

Arduino 側のサンプルプログラム

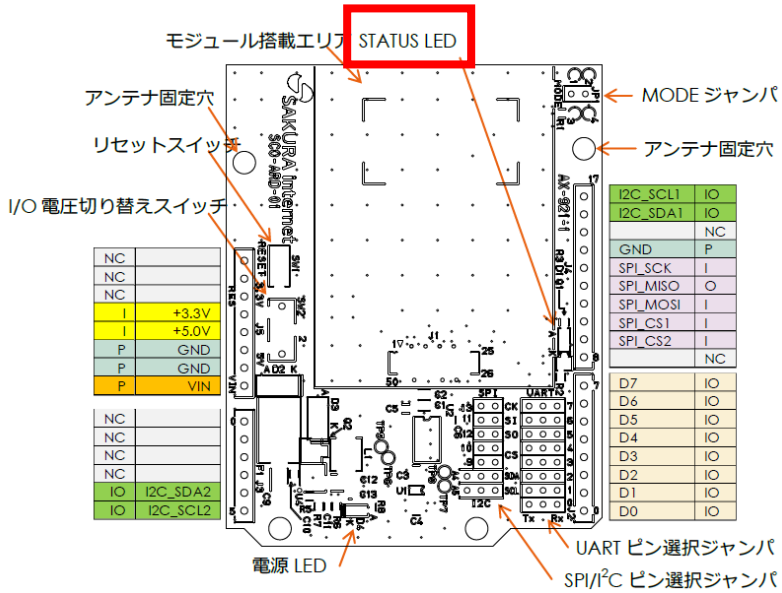
```
#include <SakuraIO.h>

SakuraIO_I2C sakuraio;
uint32_t cnt;

void setup() {
  Serial.begin(9600);
  Serial.print("Waiting to come online");
  for(;;){//接続
    if( (sakuraio.getConnectionStatus() & 0x80) == 0x80 ) break;
    Serial.print(".");
    delay(1000);
  }
  Serial.println("");
  Serial.println("online");
}

void loop() {
  cnt++;
  sakuraio.enqueueTx(0,cnt); //1 つ目のダミーデータの送信
  sakuraio.enqueueTx(1,cnt); //2 つ目のダミーデータの送信
  sakuraio.enqueueTx(2,cnt); //3 つ目のダミーデータの送信
  sakuraio.send();
  delay(10000);
}
```

ここでは意図にダミーデータを使っています.
ぜひ後で、各自でセンサを接続してセンサ値の送信にチャレンジし

シリアルモニタに、次のように online と表示されれば通信成功です。



なお、シリアルモニタに online が表示されたにも関わらずデータが上手く送信できていない場合，AC アダプなどで電源を供給すると上手く通信できることがあります。また、STATUS LED の状態も以下のように 1 回点滅しているか確認してください。



点滅回数	意味
3 回	起動中
2 回	オフライン
1 回	オンライン

このサンプルプログラムは、10 秒ごとに WebSocket で 3 つのチャンネルにカウンタデータを送信しています。setup 部では sakura.io の接続を行い、loop 部で送信しています。実際にデータが受信できているかをチェックするには、コントロールパネルから確認できます。連携サービスから先ほど作成した名前のサービスをクリックして下さい。次のように、ペイロードのデータがリアルタイムで表示されます。

・最終到着データの表示

時刻	モジュール	タイプ	ペイロード
2019-08-19 13:21:21		keepalive	

2019-08-19 13:21:15	ub4iRG9GadgC	channels	{
			"channels": [
			{
			"channel": 0,
			"type": "I",
			"value": 22,
			"datetime": "2019-08-19T04:21:14.982015863Z"
			},
			{
			"channel": 1,
			"type": "I",
			"value": 22,
			"datetime": "2019-08-19T04:21:15.004015863Z"
			},
			{
			"channel": 2,
			"type": "I",
			"value": 22,
			"datetime": "2019-08-19T04:21:15.025015863Z"
			}
			]
			}

2019-08-19 13:21:11		keepalive	

・チャンネル別到着データの表示

時刻	モジュール	チャンネル
2019-08-19 13:21:56	ub4iRG9GadgC	ch0: 26 ch1: 26 ch2: 26

② jQuery による WebSocket のデータ取得

次に、この WebSocket の情報をローカル PC から HTML で取得して Web ブラウザで表示させましょう。次ページの HTML ソースをテキストエディタで作成し、html ファイル（名前は任意）で保存しましょう。ここでは、jQuery を使用して簡単に WebSocket の情報を取得します。サンプルソース 6 行目は、jQuery のライブラリ URL を直接指定して読み込んでいます。次に 8 行目で、各モジュールに発行される wss://から始まる固有の URL を埋め込みます。インターネット上からは、この URL で各モジュールにアクセスできます。

Arduino 側では、3 つのデータを送信しています。これらは、次の JSON 形式で送られてきます。

{"module":"モジュールシリアル番号","type":"channels","datetime":"2019-08-19T04:48:16.645267212Z","payload":{"channels":[{"channel":0,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.58026841Z"},{"channel":1,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.60226841Z"},{"channel":2,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.62426841Z"}]}}

Arduino 側では、sakuraio.enqueueTx(0,cnt)が「ch0 に変数 cnt の内容を送信せよ」という内容です。すなわち、JSON の内容と対応づけると次のようになります。

```
sakuraio.enqueueTx(0,cnt);
 ↓
{"channel":0,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.58026841Z"}

sakuraio.enqueueTx(1,cnt);
 ↓
{"channel":1,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.60226841Z"}

sakuraio.enqueueTx(2,cnt);
 ↓
{"channel":2,"type":"I","value":183,"datetime":"2019-08-19T04:48:16.62426841Z"}
```

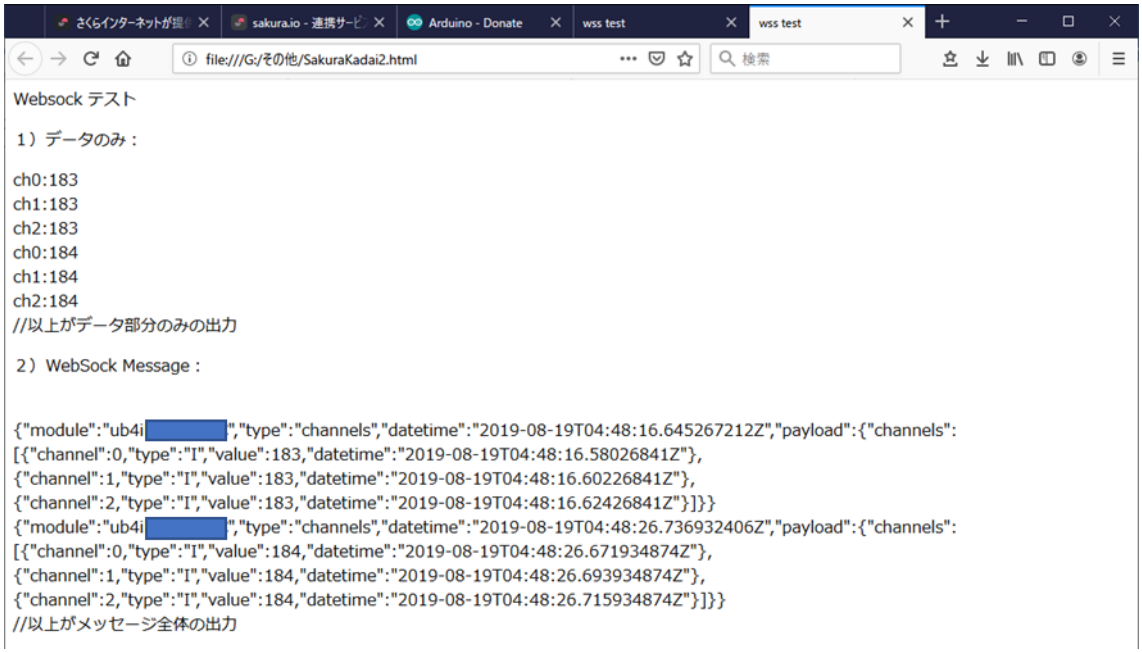
実際の値は value に格納されているため、data.payload.channels[0].value を読み出すことで、値の読み出しが可能です。

サーバ側のサンプルプログラム（今回はローカル環境で可）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<title>wss test</title>
</head>
<script src="http://code.jquery.com/jquery-latest.min.js"></script>
<script type="text/javascript">
        var webSocket = new WebSocket('モジュールの URL');

        //エラー出力
        webSocket.onerror = function(event) {
                onError(event)
        };
        webSocket.onopen = function(event) {
                onOpen(event)
        };
        webSocket.onmessage = function(event) {
                onMessage(event)
        };

        function onMessage(event) {
                //websock を受け取る
                var data = $.parseJSON(event.data);  // JSON 形式からデータ部分
を取り出す
                var module = data.module; // モジュールシリアルの代入
                if(module == "モジュールシリアル"){  // 任意のモジュールのみ抽
出
                    document.getElementById("data").innerHTML += "ch0:"+ data.
payload.channels[0].value +"<br />";
                    document.getElementById("data").innerHTML += "ch1:"+ data.
payload.channels[1].value +"<br />";
                    document.getElementById("data").innerHTML += "ch2:"+ data.
payload.channels[2].value +"<br />";
                    document.getElementById("dump").innerHTML += "<br />" + ev
ent.data;
                }
        }
        function onError(event) {
                alert(event.data);
        }
</script>
</head>
<body>
        Websock テスト
        <p>
        １）データのみ：
        <div id="data"></div> //以上がデータ部分のみの出力
        <p>
        ２）WebSock Message：
```

```
        <div id="dump"></div> //以上がメッセージ全体の出力
        <p>
</body>
</html>
```

実行例：



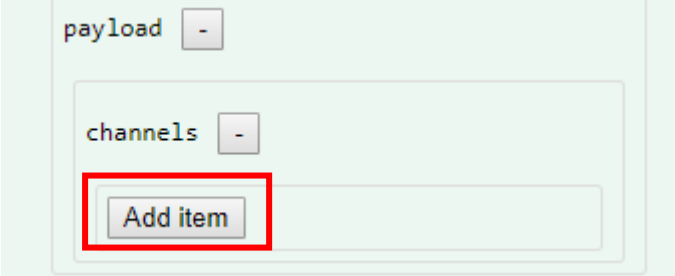
なお、この HTML はローカル PC がインターネットに接続していればローカル環境から実行できます。

③ Curl コマンドによる LED 点灯

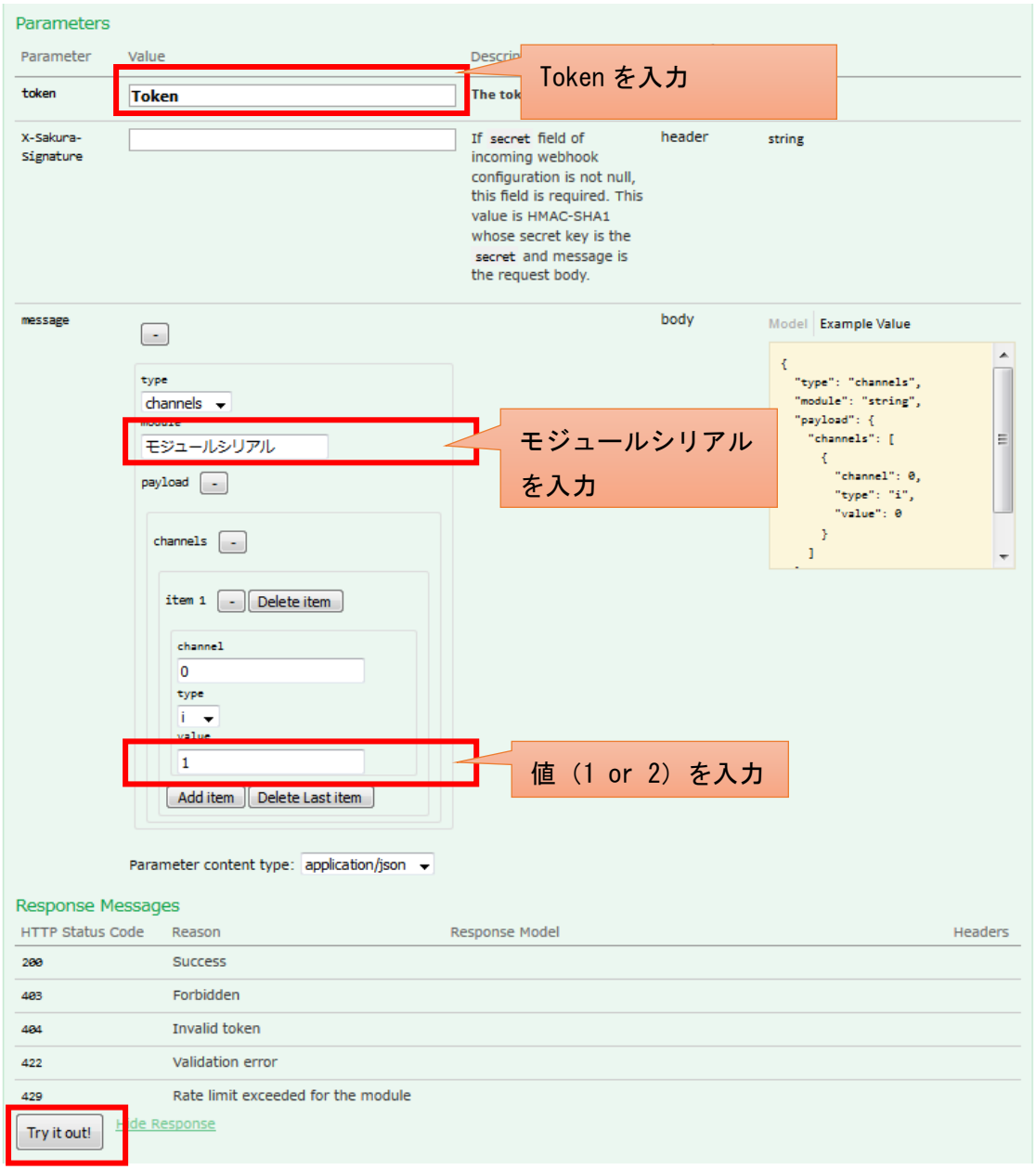
Incoming Webhook を使用することで、インターネットを経由して PC からボードを制御することができます。データ形式は JSON です。

まず WebSocket 同様に Incoming Webhook のサービス連携を登録します。コントロールパネルから適当な名前を入力して登録しますが、この時に Token と URL を控えておきます。

次に、データを送信する JSON を組み立てますが、さくらインターネットが JSON 生成用フォームを用意しているのでこれを使用しましょう。sakura.io Incoming Webhook（https://api.sakura.io/incoming/v1/docs/）にアクセスし、ページ中央やや下の[Add item]をクリックします。



ここでは、例として int 型データ 1 を送信する Incoming Webhook を組み立てることとします。次の例を参考に、Token と Secret、モジュールシリアルを入力し、type を i(int 型)、value に値の 1 を入力して画面下の[Try it out!!]をクリックします。



これでヘッダー情報や JSON、URL などが生成されました。実際の JSON は、Curl の枠に出力された次の内容のうち、以下の太字の部分が JSON に相当します。なお、以下は Linux ベースの PC から Curl コマンドを使って JSON を送信するコマンドになります（Windows 版は後述）。このコマンドは JSON 部分をメッセージとして、ヘッダー情報をつけ POST 送信するコマンドになります。

curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' -d '**{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}**' '**URL**'

JSON に相当する部分を抜き出すと以下になります（上記太字部分)。

{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}

では、次に Arduino 側にこの通信を受けるプログラムを Arduino IDE から書き込みます。ここでは、4 番のデジタルピンに白の LED を、7 番のデジタルピンに赤の LED を接続し、Incoming Webhook の内容でそれぞれを点灯させるようにします。

Arduino 側の Incoming Webhook の受信プログラム

```
#include <SakuraIO.h>
SakuraIO_I2C sakuraio;

void setup() {
  pinMode(7,OUTPUT);
  pinMode(4,OUTPUT);
  // シリアル通信を開始
  Serial.begin(9600);

  // オンラインになるまで待つ
  while ((sakuraio.getConnectionStatus() & 0x80) != 0x80) {
    Serial.print(".");
    delay(1000);
  }

  Serial.println("");
  Serial.println("Online");
}

void loop() {
  // 受信キューの状態を取得
  uint8_t rxAvailable;
  uint8_t rxQueued;
  sakuraio.getRxQueueLength(&rxAvailable, &rxQueued);

  digitalWrite(7,LOW);
  digitalWrite(4,LOW);
  // 受信キューにたまっているメッセージの数だけ繰り返す
  for (uint8_t i = 0; i < rxQueued; i++) {
    uint8_t channel;
    uint8_t type;
```

```
    uint8_t values[8];
    int64_t offset;

    // キューからのメッセージを取り出しに成功したら以下を実行
    uint8_t ret = sakuraio.dequeueRx(&channel, &type, values,
                                     &offset);
    if (ret == CMD_ERROR_NONE) {
      Serial.print("channel: ");
      Serial.println(channel);
      Serial.print("type: ");
      Serial.println((char)type);
      Serial.print("value: ");

      if (type == 'i') {
        // 32 ビット整数型の場合
        int32_t value = 0;
        memcpy(&value, &values, sizeof(int32_t));
        Serial.println(value);

        if (value == 1) { // value が 1 だったら 7 番ピンの LED を光らせる
          digitalWrite(7,HIGH);
          sakuraio.enqueueTx(0,7UL);
          sakuraio.send();
        }
        if (value == 2) { // value が 2 だったら 7 番ピンの LED を光らせる
          digitalWrite(4,HIGH);
          sakuraio.enqueueTx(0,4UL);
          sakuraio.send();
        }

      }
    } else {
      Serial.println("ERROR");
    }
  }

  delay(3000);
}
```

この例では、インターネット上から int 型整数が送信されてきて、その値が 1 なら 7 ピン（赤色 LED）を、値が 2 なら 4 ピン（白色 LED）のデジタルポートを HIGH に、すなわち LED を点灯させます。ここで、Curl コマンドが使用できる PC から先ほどの生成された Curl コマンドを含んだ JSON を Curl コマンドとして発行すると、i = 1 なので赤色 LED が点灯します。

PC のコマンドプロンプトで

curl -V

と入力して Curl コマンドが動作するかどうか確認しましょう。Curl が動作する場合は先ほどの Curl コマンドをコマンドプロンプトから実行してみましょう。ただし、Windows から Curl コマンドを実行するには、'（シングルコーテーション）を"（ダブルコーテーション）に変換し、""内の"の前に¥（バックスラッシュ）を付ける必要があります。その修正をしたコマンドが以下になります。

curl -X POST --header "Content-Type: application/json" --header "Accept: application/json" -d "{¥"type¥":¥"channels¥",¥"module¥":¥"モジュールシリアル¥",¥"payload¥":{¥"channels¥":[{¥"channel¥":0,¥"type¥":¥"i¥",¥"value¥":1}]}}" "URL"

Curl が動作しない場合は、公式ページ（https://curl.haxx.se/）から exe ファイルをダウンロードしてきて、exe ファイルがあるフォルダで上記のコマンドを実行しましょう。
※Curl コマンドは、色々なプロトコルを使ってデータ転送をするコマンドで、cURL（Clinent for URL）とも表記されます。オープンソースソフトウェアです。

この Curl コマンドをそのまま流用して、Curl を扱える PHP などで Web プログラミングを作成すれば（例えば、HTML のフォームを POST メソッドで送り、受け取った内容を JSON 形式に組み立てて Curl を発行するなど）、Web ページ上のボタンなどからインターネット経由で LED を点灯させることができます。後述の演習課題で類似課題を実施しましょう。

なお、secret キーを設定した場合は、JSON 部分をメッセージとし、Secret を Key とする HMAC-SHA1 を計算します。そして計算して算出した HMAC-SHA1 を X-Sakura-Signature ヘッダーに入れて、POST リクエストを送ることになります。詳細は以下の sakura.io ドキュメントを参照してください。

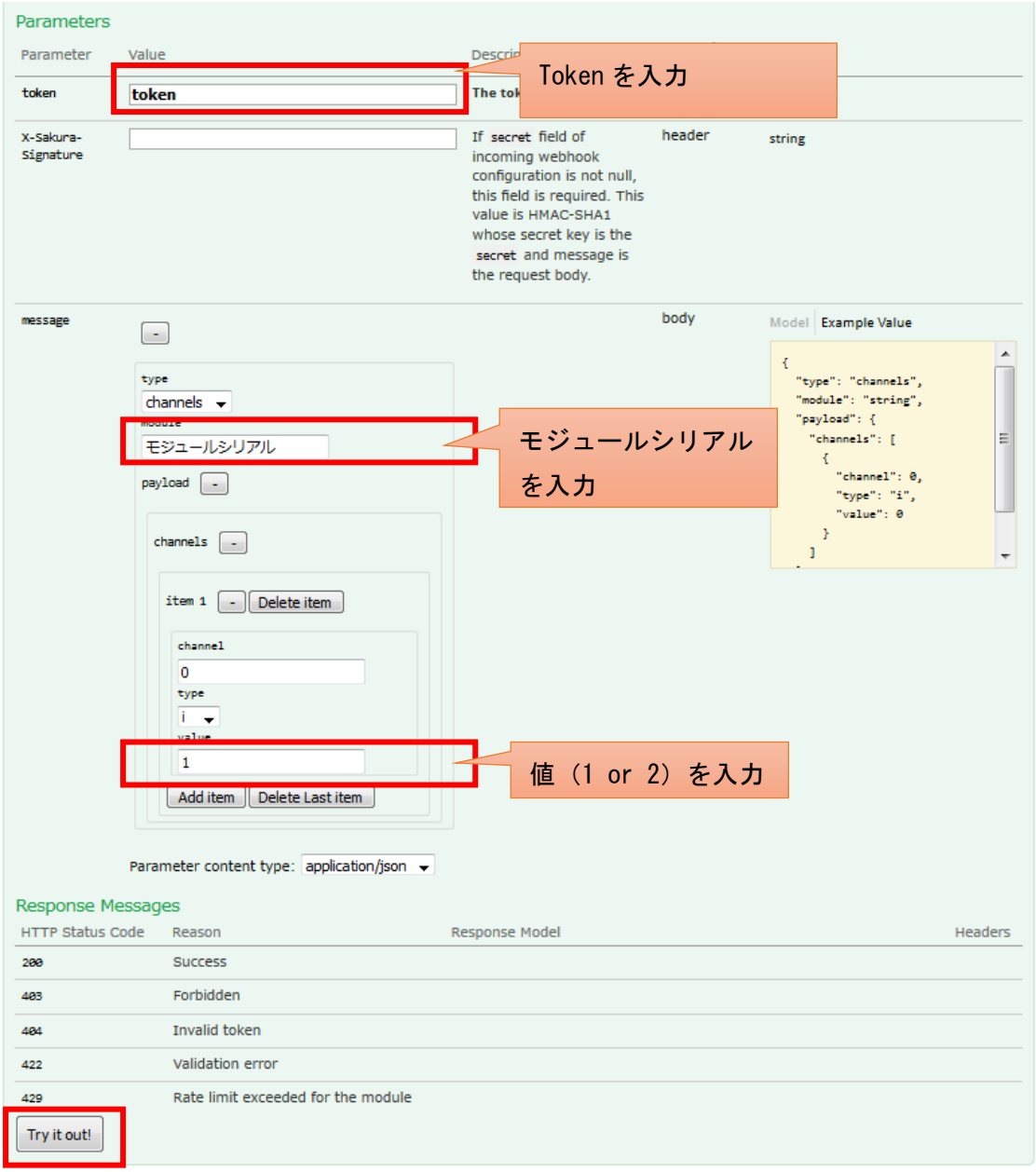
Docs 》API 利用ガイド 》連携サービス API
https://sakura.io/docs/pages/guide/api-guide/integrated-service-api.html

③’ Secret を入力した Curl コマンドによる LED 点灯

それでは Secret を指定して、暗号ハッシュ関数 HTMAC-SHA1 を利用したメッセージ認証による通信を行ってみましょう。ここでは先ほどの③の 1 と 2 を送る通信に認証機能を追加させます。まず以下のように Secret を入力した新しい Incoming Webhook の連携サービスを追加しましょう。

次にリクエストボディをメッセージとし、Secret を Key とする HMAC-SHA1 を計算します。リクエストボディ（JSON 部分）を確認するために、③と同様に、**さくらインターネットが用意している JSON 生成用フォームを使用しましょう。**sakura.io Incoming Webhook

（https://api.sakura.io/incoming/v1/docs/）にアクセスし、token、モジュールシリアル、value を設定しましょう（ここではまだ Secret は未入力です）。

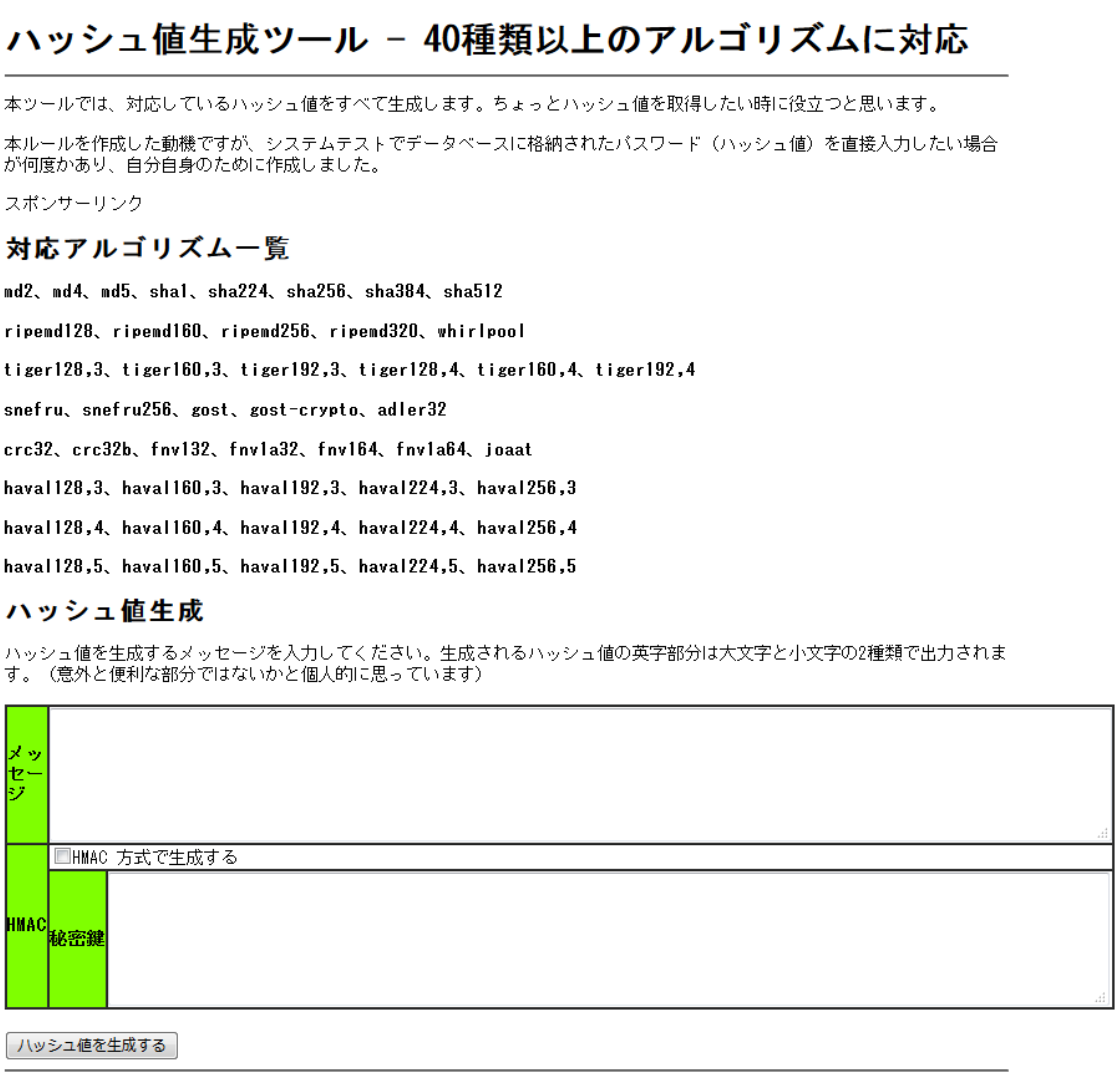


［Try it out!］を押すと以下のような JSON 生成してくれます。この背景色が黄色の部分が暗号化のためのメッセージ部分になります。

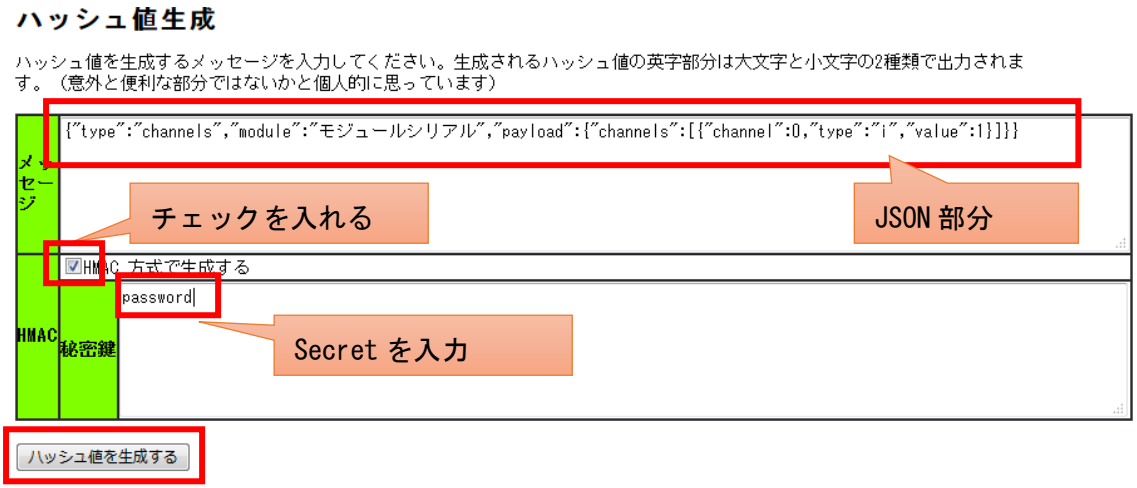
curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' -d '{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}' 'URL'

次に HMAC-SHA1 アルゴリズムに準拠して先ほどのメッセージと Secret からハッシュ値を作成します。ここでは Web ページ上でハッシュ値生成ができる以下のサイトを利用します。

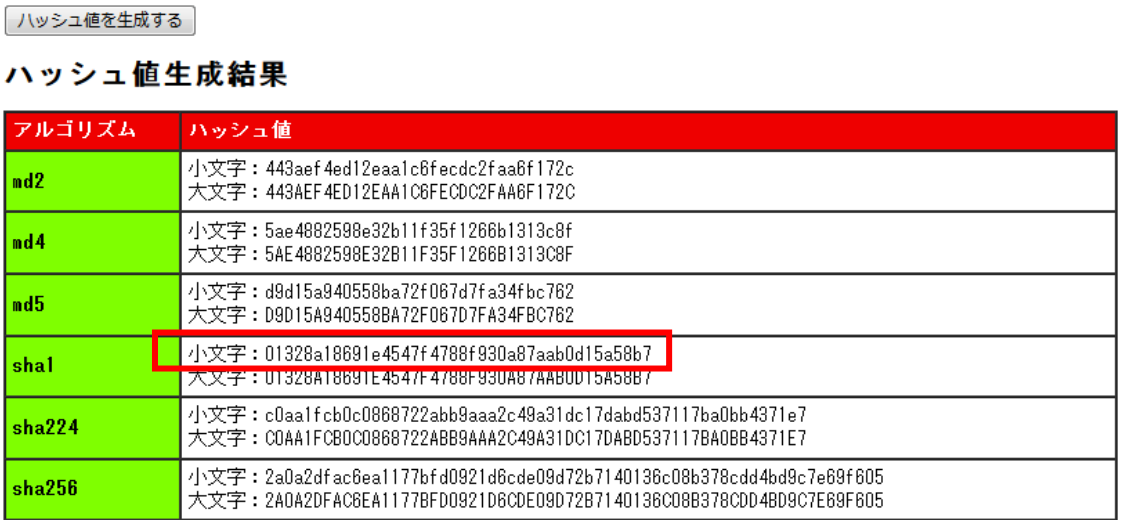
http://hensa40.cutegirl.jp/software/create_hash/



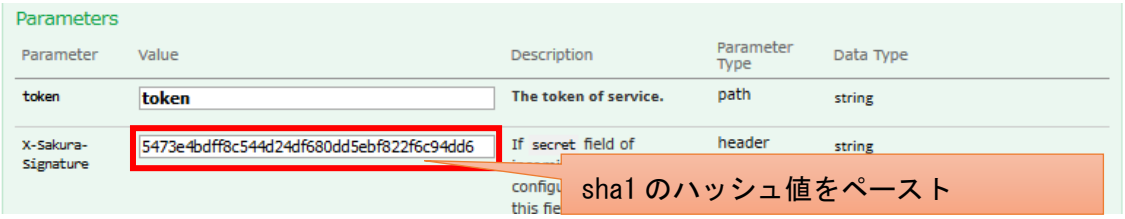
ここで、メッセージは先ほど生成した JSON 部分「{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}」とし、「HMAC 方式で生成する」のチェックボックスにチェックを入れ、秘密鍵に先ほど Incoming Webhook の連携サービスを作成したときに入力した Secret（ここでは password）を入力して［ハッシュ値を生成する］を押します。



ボタンを押すとハッシュ値が生成されるので、sha1 の小文字のハッシュ値をコピーします。



次に、先ほど利用したさくらインターネットが用意しているJSON生成用フォームsakura.io Incoming Webhook（https://api.sakura.io/incoming/v1/docs/）の［X-Sakura-Signature］にハッシュ値をペーストして［Try it out!］を押すとCurlの欄に完成したコマンドが表示されます。



・生成された Curl コマンド

curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' --header 'X-Sakura-Signature: 5473e4bdff8c544d24df680dd5ebf822f6c94dd6' -d '{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}' 'URL'

・Windows 用に加工した Curl コマンド

curl -X POST --header "Content-Type: application/json" --header "Accept: application/json" --header "X-Sakura-Signature: 5473e4bdff8c544d24df680dd5ebf822f6c94dd6" -d "{¥"type¥":¥"channels¥",¥"module¥":¥"モジュールシリアル¥",¥"payload¥":{¥"channels¥":[{¥"channel¥":0,¥"type¥":¥"i¥",¥"value¥":1}]}}" "URL"

③と同様に Windows 用に加工した Curl コマンドをコマンドプロンプト上で実行すると、Arduino の対応する LED が点灯します。

同様のやり方でもう片方の LED を作成させる Curl コマンドを作成してみましょう。具体的には、先の Curl コマンドの中で value が 1 から 2 に変更する必要があります。つまりハッシュ値を生成する基となるメッセージが変わってしまうので、またハッシュ値を作りなおす必要があります。

curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' --header 'X-Sakura-Signature: ~~5473e4bdff8c544d24df680dd5ebf822f6c94dd6~~' -d '{"type":"channels","module":"モジュールシリアル ~~yload~~":{"channels":[{"channel":0,"type":"i","value":2}]}}' 'URL'

ハッシュ値を作り直して書き換える必要がある

1 から 2 に変わる

④ センサデータの送信と取得

A) ①の課題を参考にして任意のセンサデータを Arduino に接続して、計測したセンサデータを sakura.io に送信してみましょう。

B) ②の課題を参考にして sakura.io に送信した任意のセンサデータをローカル環境から HTML を利用して取得してみましょう。

⑤ Node-RED を使ったデータ通信

A) データの受信とファイルへの書き出し

Node-RED は IBM を中心として開発された開発環境です。Flow エディタを使って、プラグイン/モジュールであるノードを視覚的に接続しながら、IoT デバイスとオンラインサービスをつなぐことができる開発ツールになります。

Windows では Node.js のサイト **https://nodejs.org/** からまず Node.js (Ver 10.x) をインストールし、次に管理者権限のコマンドプロンプトか PowerShell 上で次のコマンドで Node-RED をインストールします。

C:¥> **npm install -g --unsafe-perm node-red**

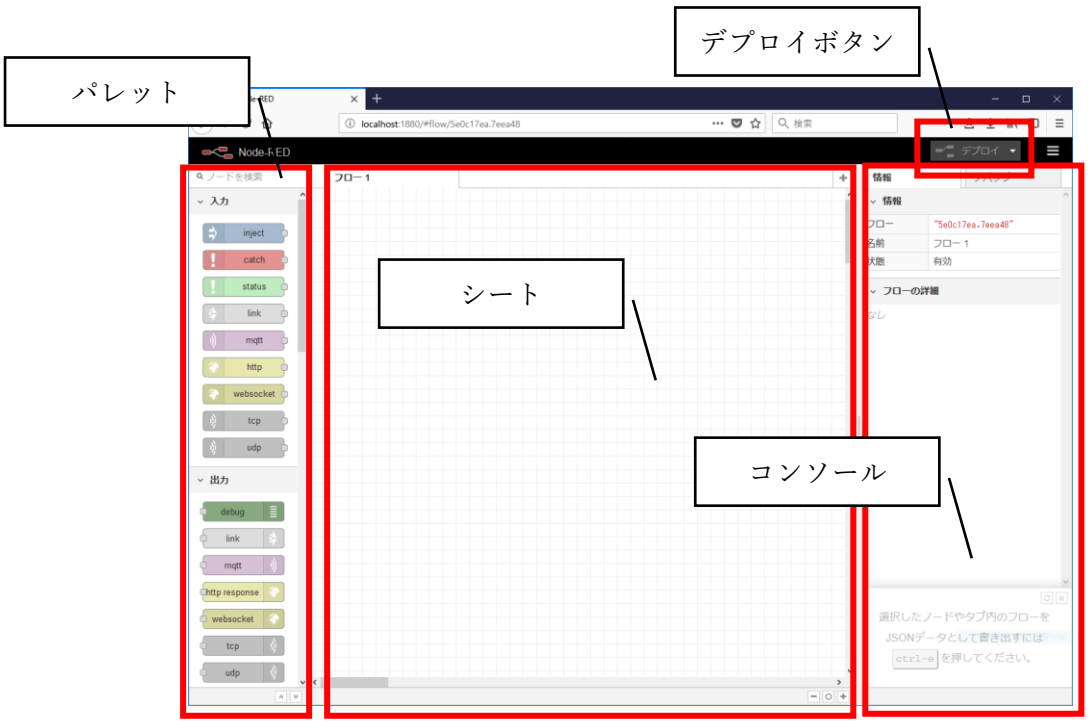
Node-RED の起動は次のコマンドで行います。

C:¥> **node-red**

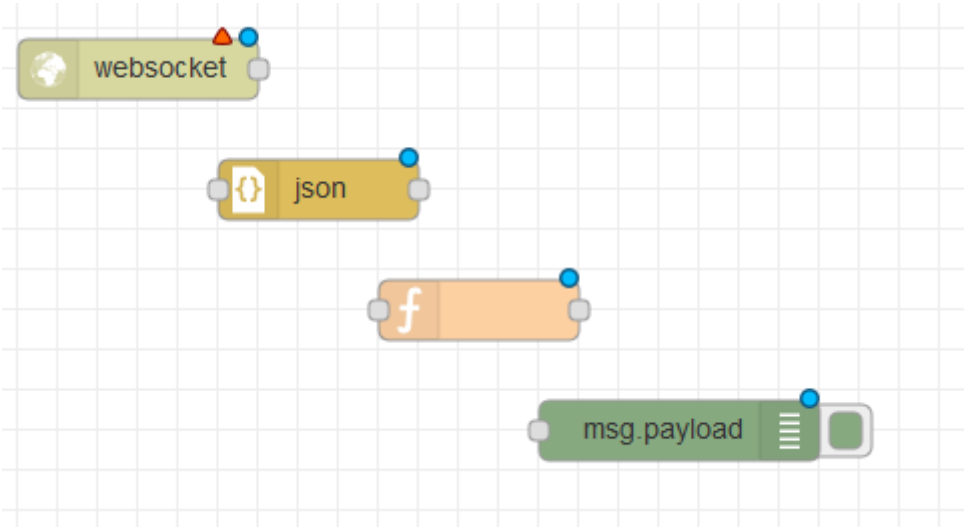
起動後に Web ブラウザで http://127.0.0.1:1880 または http://localhost:1880 にアクセスすると Node-RED が開きます。もし動作しない場合は次のコマンドを試して下さい。

C:¥> **node C:¥Users¥<ユーザ名>¥AppData¥Roaming¥npm¥node_modules¥node-red¥red.js**

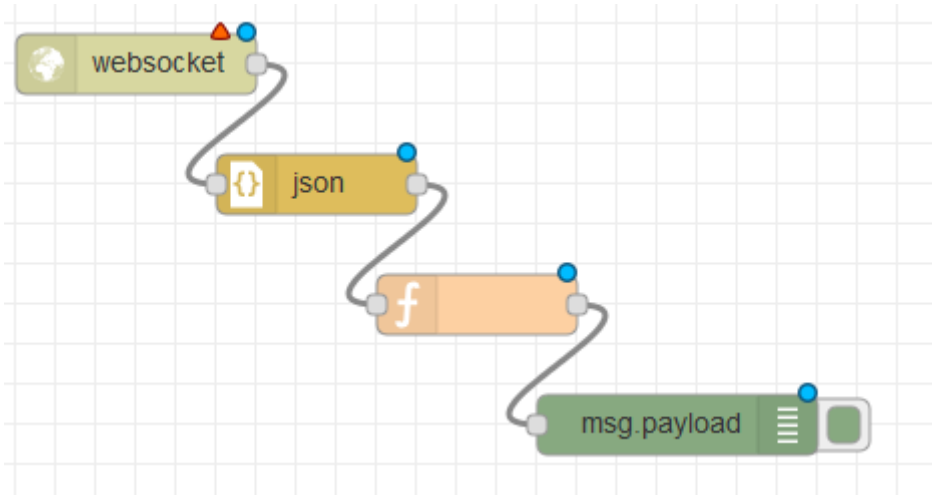
設定ファイルは、**C:¥Users¥<ユーザ名>¥.node-red¥settings.js** になります。



まず、左のノードのリストがあるパレットから Websocket、json、function、debug の 4 つをドラッグ＆ドロップで中央のシートに配置します。



それぞれのノードはコネクタで接続することができます。



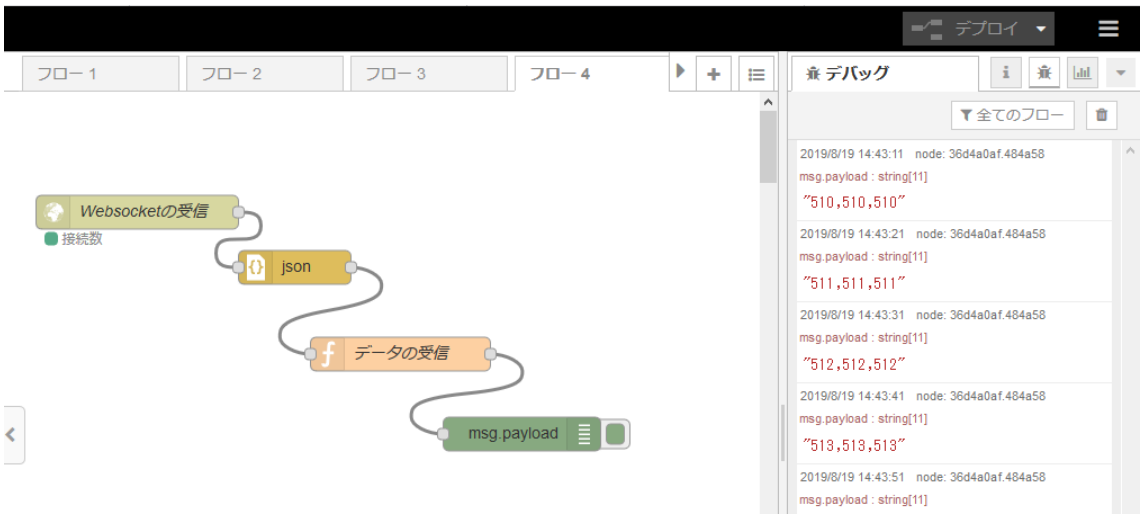
ノードをダブルクリックするとそれぞれ設定できます。

まず最初に、Websocket のノードをダブルクリックして、モジュールの URL と名前(Websocket の受信）を設定します。

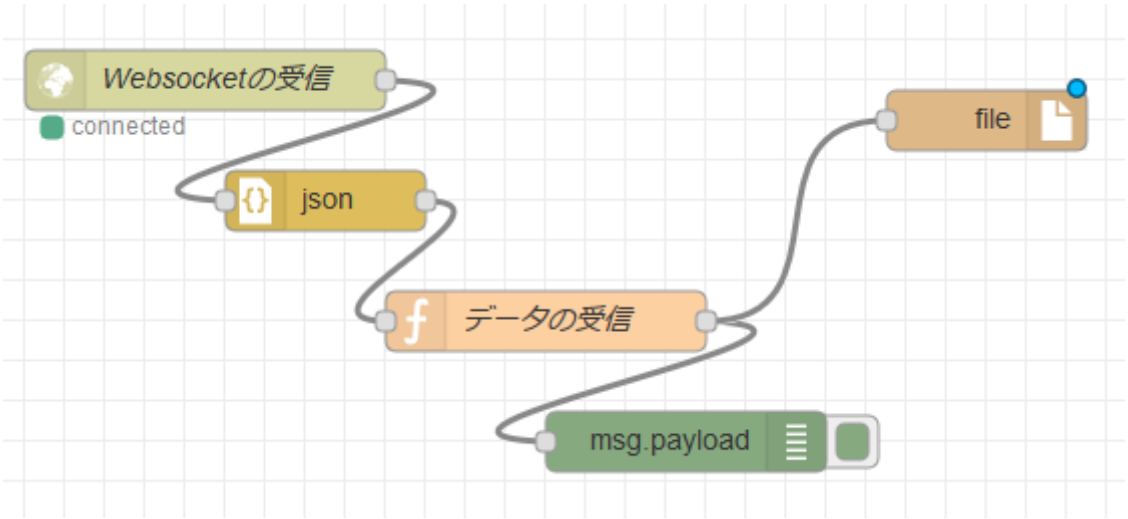
次に function のノードをダブルクリックして、名前とデータ受信用のプログラムを入力します。

```
if(msg.payload.module == "xxxxxxxxxx"){
    if(msg.payload.type =="channels"){
        var d1 = msg.payload.payload.channels[0].value;
        var d2 = msg.payload.payload.channels[1].value;
        var d3 = msg.payload.payload.channels[2].value;
        msg.payload = d1 + "," + d2 +"," + d3;
        return msg;
    }
}
```

設定したあと右上の[デプロイ]ボタンを押すとデータ受信を開始し、debug(msg.payload)にて受信したデータを確認することができます。



次にデータをファイルに書き出すために、file のノードを設置してデータの受信(function)と接続します。



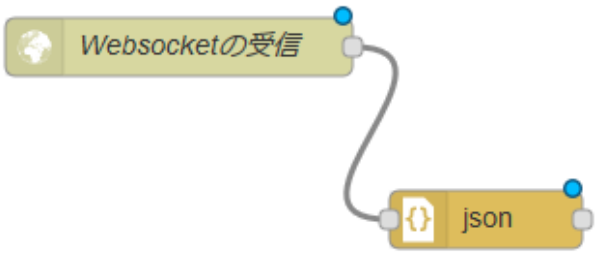
次に file のノードをダブルクリックして、名前と書き出しようのファイル名をフルパスで入力します。

最後にデプロイボタンを押すと、データを受信して書き出すプログラムが動作し始めます。

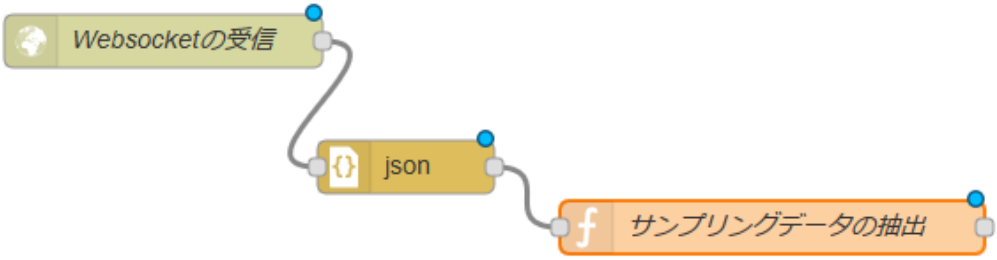
B) データの受信とリアルタイムのグラフ化

ここでは IoT デバイスから受信したデータをリアルタイムに可視化していきます。

まず、先ほどの演習 A)と同様の Websocket、json を準備します。ただし、タブでフロー画面を切り替えて作成する場合（つまり、先の演習課題を動作させながら残したまま、この演習課題を実施する場合）、Websocket のプロパティからパスを設定する時に「新規に websoket-lisener を追加」から Websocket の URL を再度登録してください。例えば、フロー１とフロー２で同じ websoket-lisener を利用していた場合、両方で同一のものが使われるため場合によってはエラーが発生してしまいます。同じ URL を登録する場合でも別の websoket-lisener として新規登録して動作させることでフロー１とフロー２で独立して動作できるようになります。

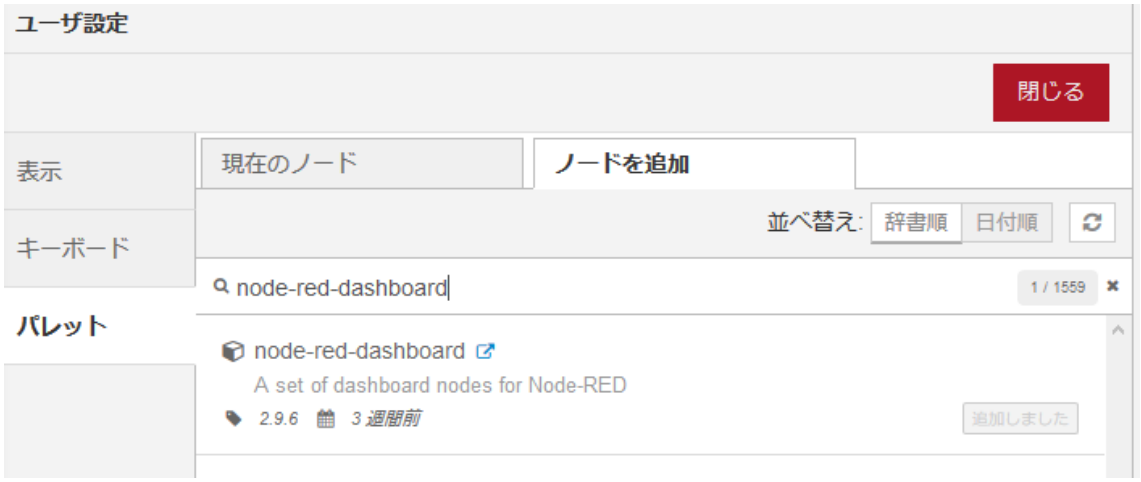


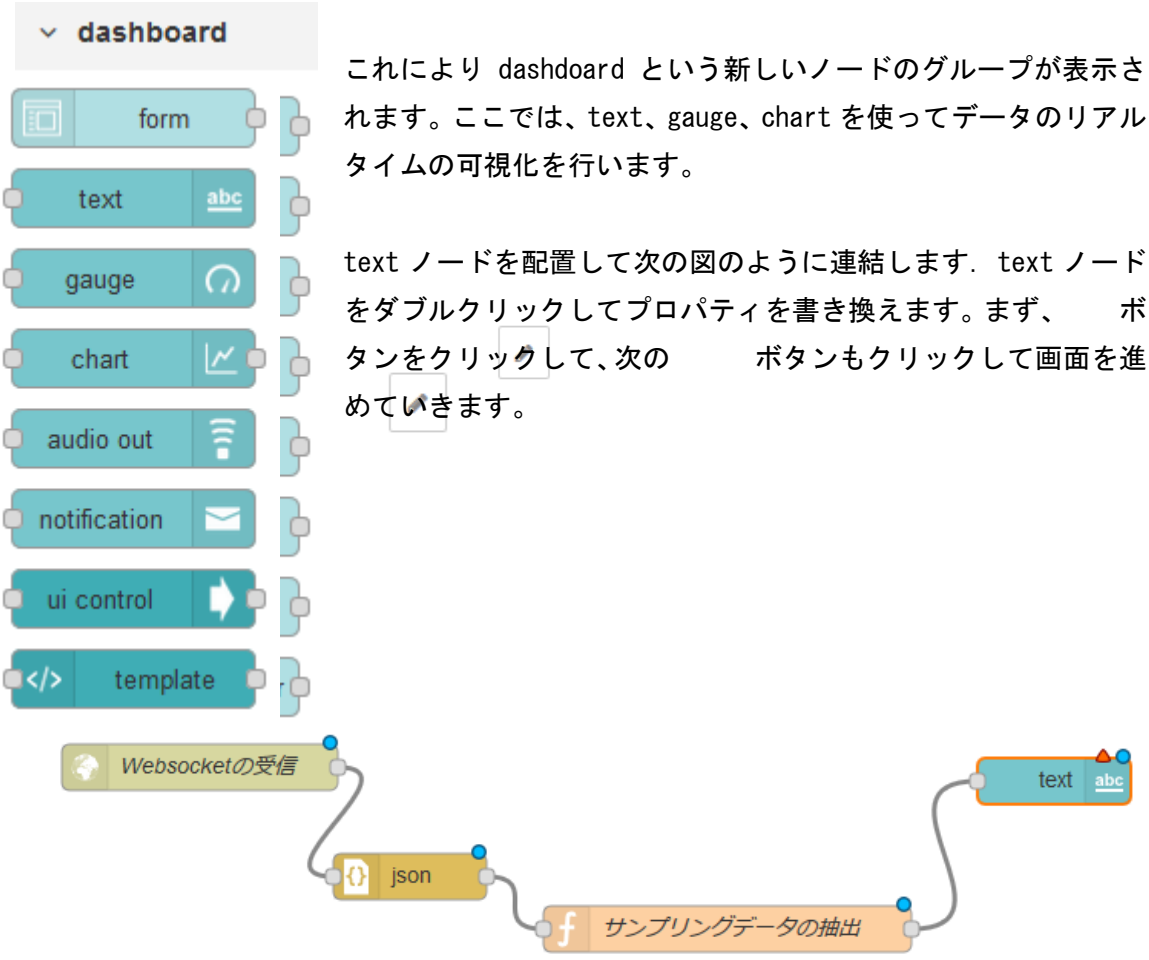
次に、function でリアルタイムに可視化する任意の１つのデータを指定します。




```
if(msg.payoad.module == "xxxxxxxxxx"){
    if(msg.payload.type =="channels"){
        var d1 = msg.payload.payload.channels[0].value;
        msg.payload = d1;
        return msg;
    }
}
```

次に右上メニューの設定を選び、開いたユーザ設定画面からパレットタブに切り替えます。
さらにノードの追加タブから node-red-dashboard を検索して追加します。





これにより dashdoard という新しいノードのグループが表示されます。ここでは、text、gauge、chart を使ってデータのリアルタイムの可視化を行います。

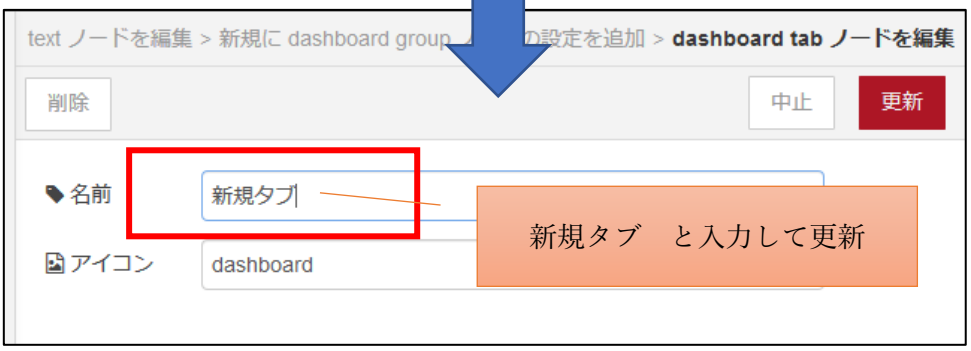
text ノードを配置して次の図のように連結します．text ノードをダブルクリックしてプロパティを書き換えます。まず、　　ボタンをクリックして、次の　　ボタンもクリックして画面を進めていきます。


text ノードを編集
削除
中止
完了
プロパティ
選択
Group
新規に ui_group を追加...
Size
自動
Label
text
Value format
{{msg.payload}}
Layout
label value
labelvalue
labelvalue
label value
label value
Name


text ノードを編集 > 新規に dashboard group ノードの設定を追加
中止
追加
名前
デフォルト
選択
タブ
新規に ui_tab を追加...
幅
6
グループ名を表示する
Allow group to be collapsed

（次のウィンドウが開く）

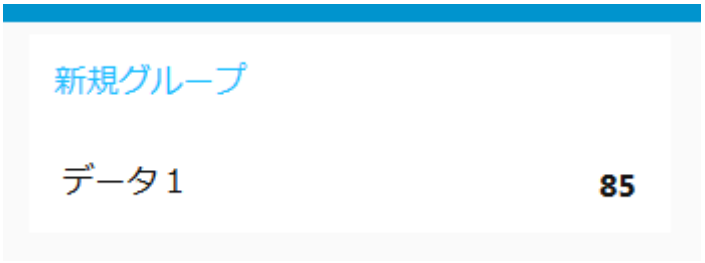

text ノードを編集 > 新規に dashboard group ノードの設定を追加 > 新規に dashboard tab ノードの設定を追加
中止
追加
名前
ホーム
アイコン
dashboard


text ノードを編集 > 新規に dashboard group
の設定を追加 > dashboard tab ノードを編集
削除
中止
更新
名前
新規タブ
アイコン
dashboard
新規タブ　と入力して更新

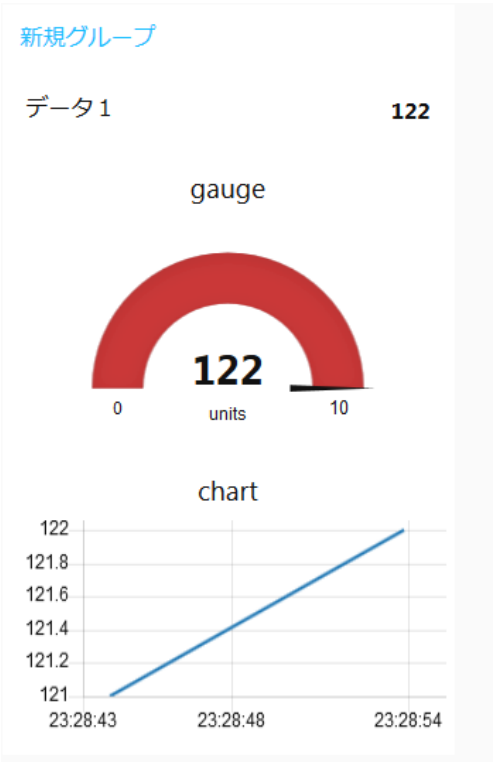

text ノードを編集 > 新規に dashboard group ノードの設定を追加
中止
追加
名前
新規グループ
タブ
新規タブ
幅
6
グループ名を表示する
Allow group to be collapsed
新規グループ　と入力して追加

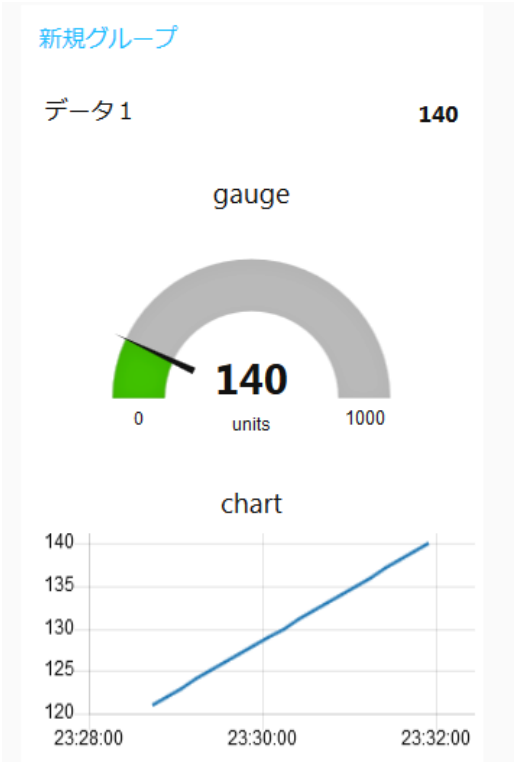
そうすると、右端のコンソールのダッシュボードタブに「新規タブ」と「新規グループ」、新しいデータの名前「データ１」が階層構造で表示されます。

以上で設定が終了したので、デプロイした後にダッシュボードタブのサイトボタンを押すと http://localhost:1880/ui が Web ブラウザが開かれて次のようにテキスト情報でデータをリアルタイムに可視化することができます。



同様に gauge と chart のノードを配置した後に、それぞれ先ほど設定した「新規グループ[新規タブ]」に設定して完了するとプロパティの設定が終了しまう。なお、プロパティを開くとあらかじめ「新規グループ[新規タブ]」が選ばれているように見えますが、完了ボタンを押して設定を終了してはじめて設定が完了となります。そうするとサイトボタンを押して開かれるダッシュボードに、左下図のようにテキストとゲージ、折れ線グラフの 3 つが表示されます。guage の range を設定すると右下図のように range で設定した範囲のゲージ内に値が収まります。なお、折れ線グラフは時間経過と共にグラフの横軸が拡張されていきます。





データの可視化に関しては、データを他のサービスと連携することで可視化を実現することもできます。例えば、クラウドサービスの Thinger.io（https://thinger.io/）では、最大デバイス数：2 や最大ダッシュボード数：4 などの条件付きの無料サービスがあります。

C) Node-RED からの LED 点灯

③で実施した LED の点灯を Node-RED 上で実施してみましょう。Node-RED では、WebSocket の双方向通信を使って実施します。前述の演習 3 の①で既に WebSocket の連携サービスを作成している人はそれをここでも使いましょう。作っていない人は演習 3 の①を参照して連携サービスを作成しましょう。

また、送信用の JSON 形式は演習 3 の③で作成したものを参考にしましょう（以下参照）。

{"type":"channels","module":"モジュールシリアル","payload":{"channels":[{"channel":0,"type":"i","value":1}]}}

では、次に Arduino 側にこの通信を受けるプログラムを Arduino IDE から書き込みます。ここでは、4 番のデジタルピンに白の LED を、7 番のデジタルピンに赤の LED を接続し、Incoming Webhook の内容でそれぞれを点灯させるようにします。

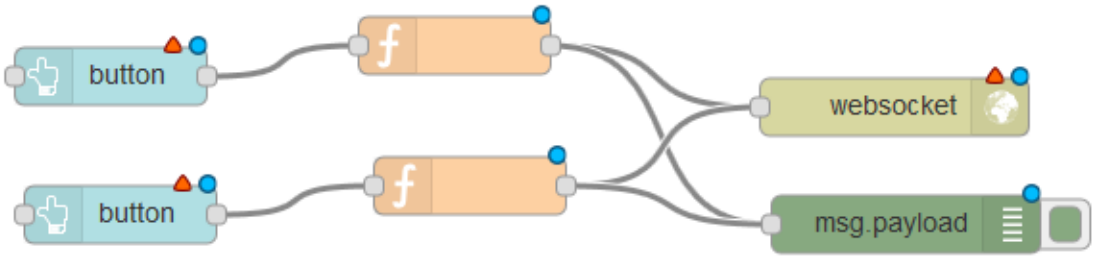
Arduino 側の Incoming Webhook の受信プログラム（再掲載）

```
#include <SakuraIO.h>
SakuraIO_I2C sakuraio;

void setup() {
  pinMode(7,OUTPUT);
  pinMode(4,OUTPUT);
  // シリアル通信を開始
  Serial.begin(9600);

  // オンラインになるまで待つ
  while ((sakuraio.getConnectionStatus() & 0x80) != 0x80) {
    Serial.print(".");
    delay(1000);
  }

  Serial.println("");
  Serial.println("Online");
}

void loop() {
  // 受信キューの状態を取得
  uint8_t rxAvailable;
  uint8_t rxQueued;
  sakuraio.getRxQueueLength(&rxAvailable, &rxQueued);

  digitalWrite(7,LOW);
  digitalWrite(4,LOW);
```

```
  // 受信キューにたまっているメッセージの数だけ繰り返す
  for (uint8_t i = 0; i < rxQueued; i++) {
    uint8_t channel;
    uint8_t type;
    uint8_t values[8];
    int64_t offset;

    // キューからのメッセージを取り出しに成功したら以下を実行
    uint8_t ret = sakuraio.dequeueRx(&channel, &type, values,
                                     &offset);
    if (ret == CMD_ERROR_NONE) {
      Serial.print("channel: ");
      Serial.println(channel);
      Serial.print("type: ");
      Serial.println((char)type);
      Serial.print("value: ");

      if (type == 'i') {
        // 32 ビット整数型の場合
        int32_t value = 0;
        memcpy(&value, &values, sizeof(int32_t));
        Serial.println(value);

        if (value == 1) { // value が 1 だったら 7 番ピンの LED を光らせる
          digitalWrite(7,HIGH);
          sakuraio.enqueueTx(0,7UL);
          sakuraio.send();
        }
        if (value == 2) { // value が 2 だったら 7 番ピンの LED を光らせる
          digitalWrite(4,HIGH);
          sakuraio.enqueueTx(0,4UL);
          sakuraio.send();
        }

      }
    } else {
      Serial.println("ERROR");
    }
  }

  delay(3000);
}
```

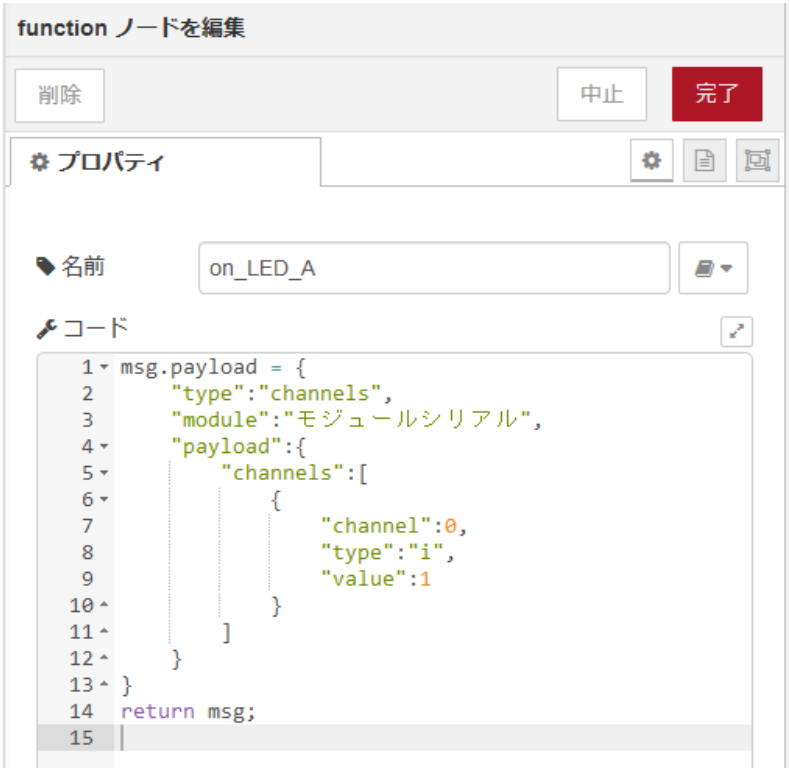
この例では、インターネット上から int 型整数が送信されてきて、その値が 1 なら 7 ピン（赤色 LED）を、値が 2 なら 4 ピン（白色 LED）のデジタルポートを HIGH に、すなわち LED を点灯させます。ここで、Curl コマンドが使用できる PC から先ほどの生成された Curl コマンドを含んだ JSON を Curl コマンドとして発行すると、i = 1 なので赤色 LED が点灯します。

次に、Node-RED のノード（button、function、websocket 出力、debug）を以下のように配置し、それぞれのノードを設定していきます。この例は value の 1 と 2 を送信するサンプルになります。



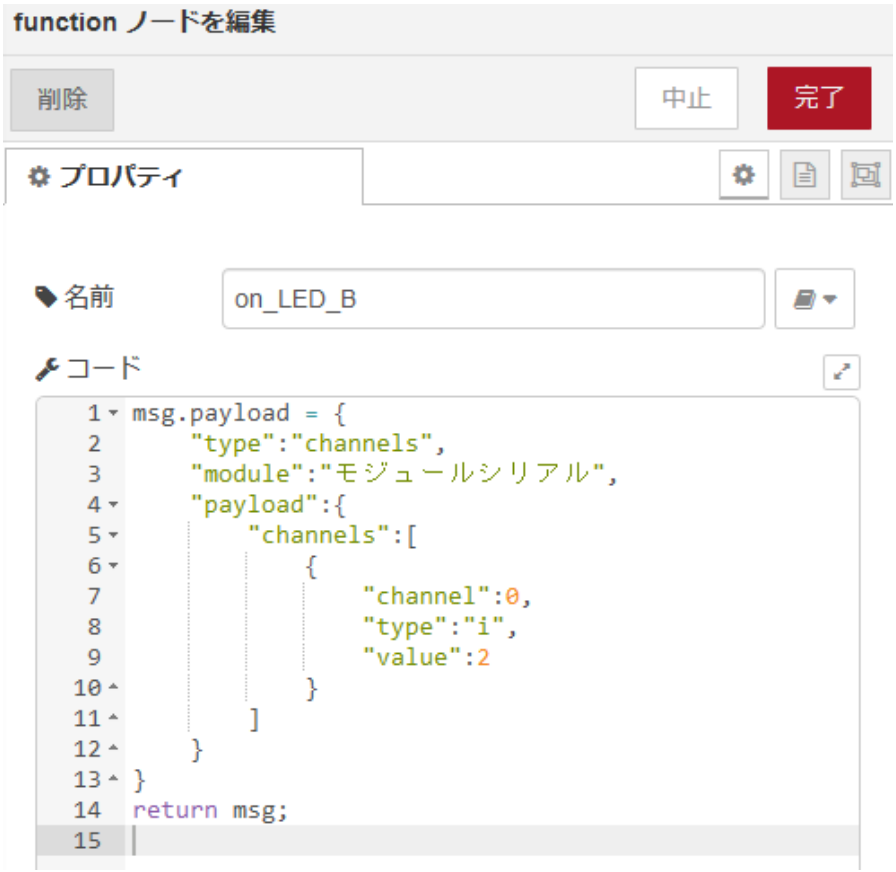
上側の button ノードと function ノードの設定を次のようにします。ここでは、グループを前の B)で作成したグループに登録しています。


button ノードを編集
削除
中止
完了
プロパティ
Group
新規グループ [新規タブ]
Size
自動
Icon
optional icon
Label
on_button_A
Colour
optional text/icon color
Background
optional background color
When clicked, send:
Payload
Topic
If msg arrives on input, pass through to output:
Name
グループ
名前


function ノードを編集
削除
中止
完了
プロパティ
名前
on_LED_A
コード
msg.payload = {
"type":"channels",
"module":"モジュールシリアル",
"payload":{
"channels":[
{
"channel":0,
"type":"i",
"value":1
}
]
}
}
return msg;

```
msg.payload = {
    "type":"channels",
    "module":"モジュールシリアル",
    "payload":{
        "channels":[
            {
                "channel":0,
                "type":"i",
                "value":1
            }
        ]
    }
}
return msg;
```

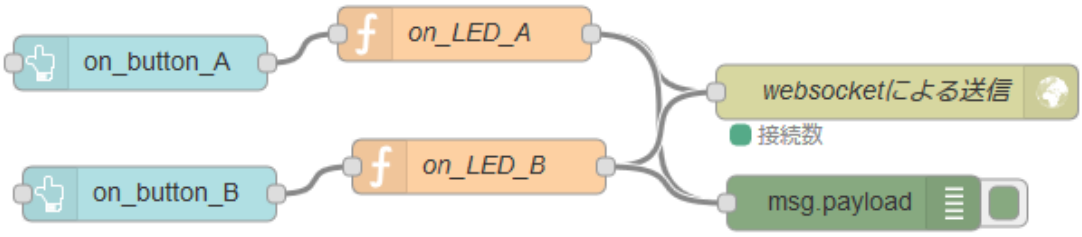
以上を参考に下側の button ノードと function ノードも設定していきます。



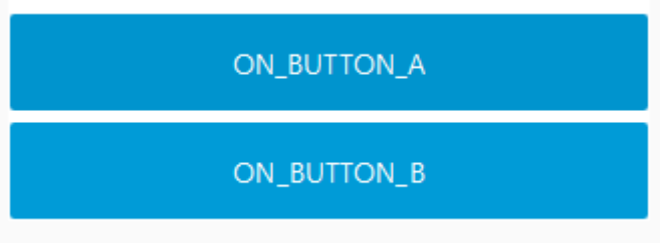

```
msg.payload = {
    "type":"channels",
    "module":"モジュールシリアル",
    "payload":{
        "channels":[
            {
                "channel":0,
                "type":"i",
                "value":2
            }
        ]
    }
}
return msg;
```

次に websocket 出力のノードの設定を以下のように行います。種類を［接続］、名前を任意に入力し、URL に websocket の URL を追記します。

設定が終わってデプロイするとノードの表示が以下のように変わります。



先の B)と同じようにダッシュボードのサイト画面を開くと以下のようなボタンが新しく追加されています。ボタンを押すとそれぞれに対応した Arduino に連結された LED が点灯します。



演習４　総合演習

これまで学んだものに基づいて各自の IoT システムを構築してみましょう。

【必須】
・IoT デバイスに任意のセンサを利用しましょう
・取得したセンサの値を sakura.io にアップロードしてみましょう

【任意】
A. センサを複数にする
B. sakura.io に集めたデータを可視化する
C. IoT デバイスへのフィードバック機能を任意につける
D. IoT デバイスにアクチュエータをつける
E. その他
・IoT を利用したビジネスモデルについて考えてみる。
・sakura.io 以外で自社利用に適した IoT プラットフォームを探してみる。
・復習として演習課題の最初から最後まで一人で作成してみる。
・まだ使っていないセンサやアクチュエータを試用してみる。
・複数のセンサでデータを集めてそれらの値を Node-RED のダッシュボードに表示すると共に、何らかの統計値も一緒に表示させるシステムを作成してみる。
・Node-RED でファイルに書き出したセンサのデータを Chart.js などの JavaScript ライブラリを使ってグラフ化してみる。
・Node-RED の Tweet ノードを利用してノードレットから Tweet してみる。
※ツイッターのアカウントは各自で準備してください。
・「OSOYOO 公式チュートリアル」Web サイトを参考にして、Node-RED でダッシュボードの数字が書かれたボタンを押されると、Arduino に接続した 1 桁 LED デジタル表示管にボタンの数字が表示されるシステムを作成してみる。
・XAMPP（https://www.apachefriends.org/jp/index.html）などの PHP 開発環境を利用して Web サーバなどを立ち上げ、Arduino と連携した任意の Web システムを作成してみる。

「OSOYOO 公式チュートリアル」Web サイト
http://osoyoo.com/ja/2014/12/06/arduino-starter-kit/