

Let's start electronics making with ESP32 Tranning kit



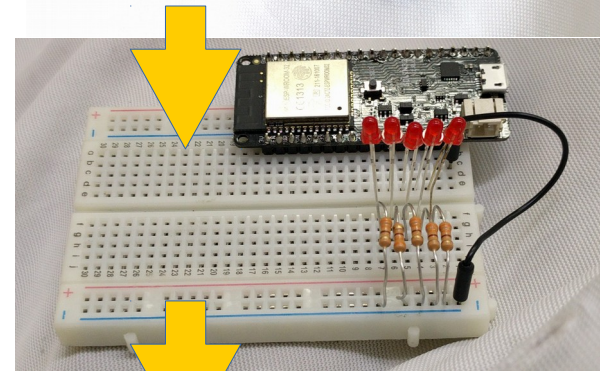
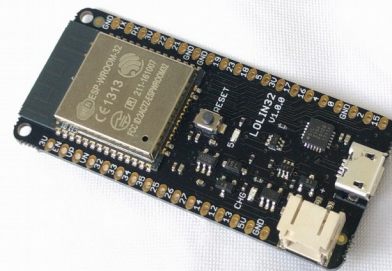
ESP32ではじめる  
はじめてのでんしこうさく

AKBROBOT

日本Androidの会秋葉原支部ロボット部・ジャパン



This text is CC 0 ( Public Domain )



# もくじ

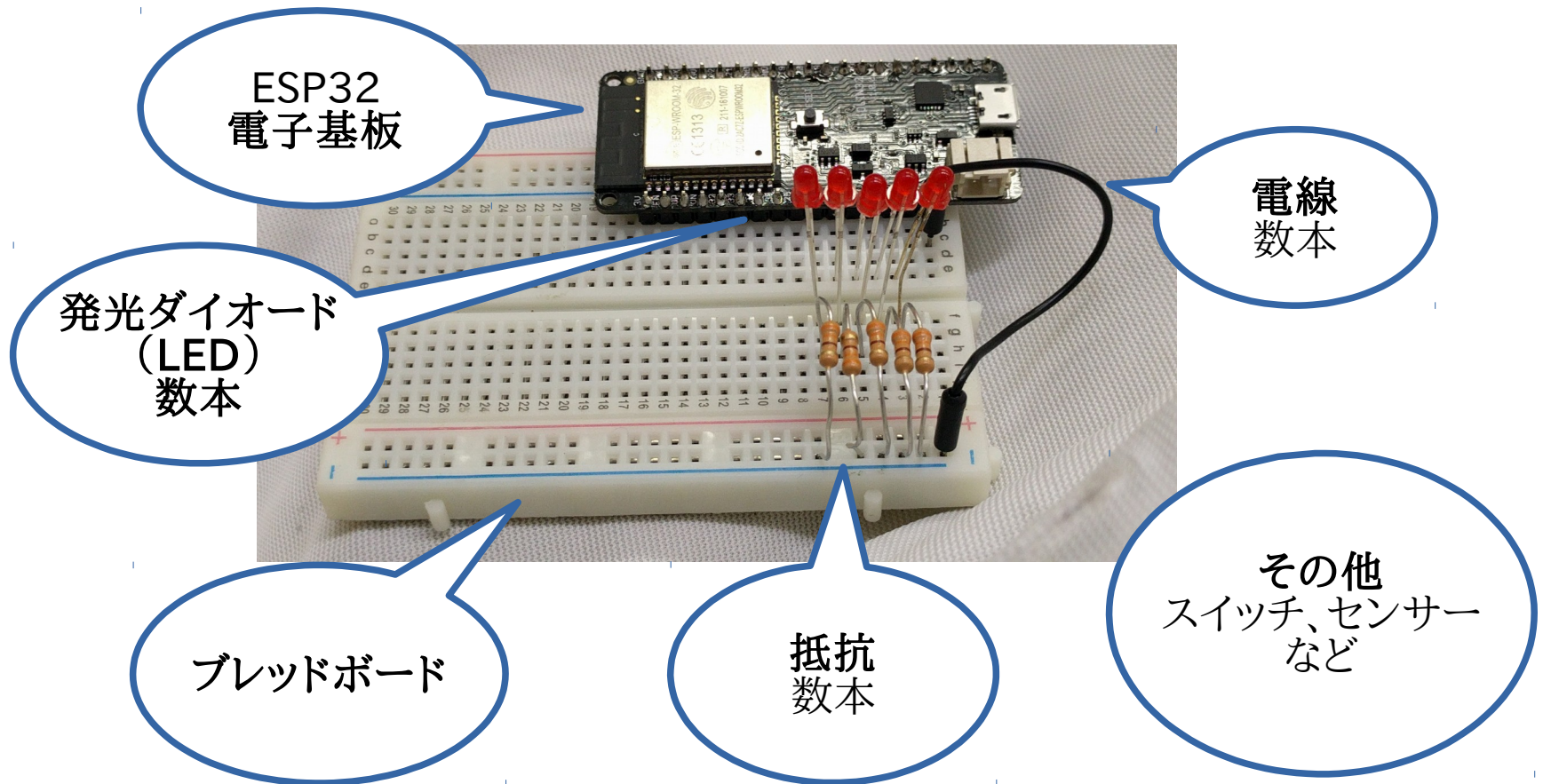
ページ 3... 「実験キット AKBONE ESP32」

ページ 5... ワークショップその1「光の流れ」 日本Androidの会秋葉原支部ロボット部

ページ 14... ワークショップその2「IoTの実験」 秘密結社オープンフォース / GEEKLAB.NAGANO

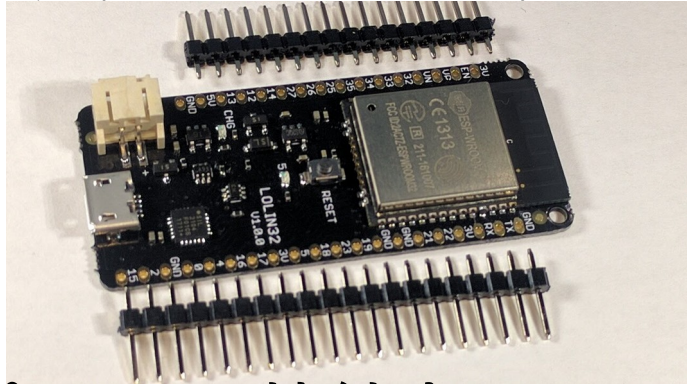
ページ 27... 「本格的に実験するには」

# 実験キット 「AKBONE ESP32」

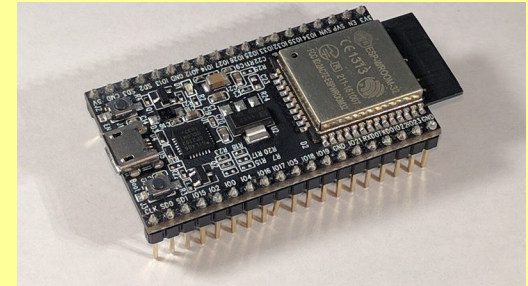


# ESP32ボードを組み立てる

キットには、WEMOS LOLIN32が入っています。  
実験する前にピンヘッダをハンダ付けしてください



付属のLOLIN32ではなく、  
ESP32-DevKitCを代わりに  
使うこともできます  
(はんだづけは必要ありません)



## パソコンに接続する

MicroUSBケーブル(別売)を使ってパソコンに接続して下さい。  
パソコンがMac OSXの場合は、以下からドライバをインストールして下さい。

<http://jp.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx#mac>



## Arduino起動、ArduBlock起動

# ワークショップ その1

「ハードウェア」と「ソフトウェア」

## 「光の流れ」テキスト

日本アンドロイドの会秋葉原支部ロボット部

ESP32を使った小さな電子基板を  
Arduino互換基板として使用して、  
LEDを光るプログラムを作ります



メーリングリスト: <http://groups.google.com/group/robot-android-group-japan-aku>

Webサイト: <https://sites.google.com/site/akbrobot/>

# 「光の流れ」ワークショップ

## 概要

目的： 「ハードウェア」と「ソフトウェア」に触れる

概要： 光の流れをつくる

対象： 小学生以上

コース時間： 30分～

コース方式： 個別指導

## 時間

(1) コース説明 3分

(2) ハードウェア工作 10分

(3) 課題1(1つ点減) 5分

(4) 課題2(光の流れ) 10分

(5) まとめ、質問 3分

(6) 追加の問題(流れを逆にするなど) 15分

# 全体のつながり

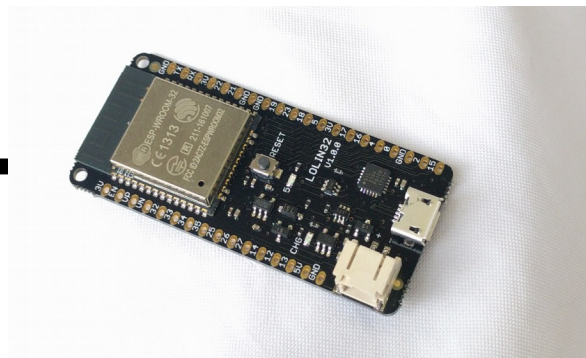
パソコン



USB接続

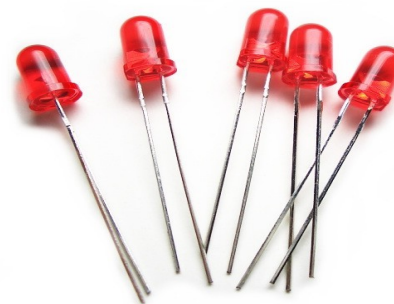
- ・ソフトウェアを作成する
- ・ソフトウェアを電子基板に送る
- ・電気を供給する

電子基板



- ・ソフトウェアを保存する
- ・ソフトウェアを実行する

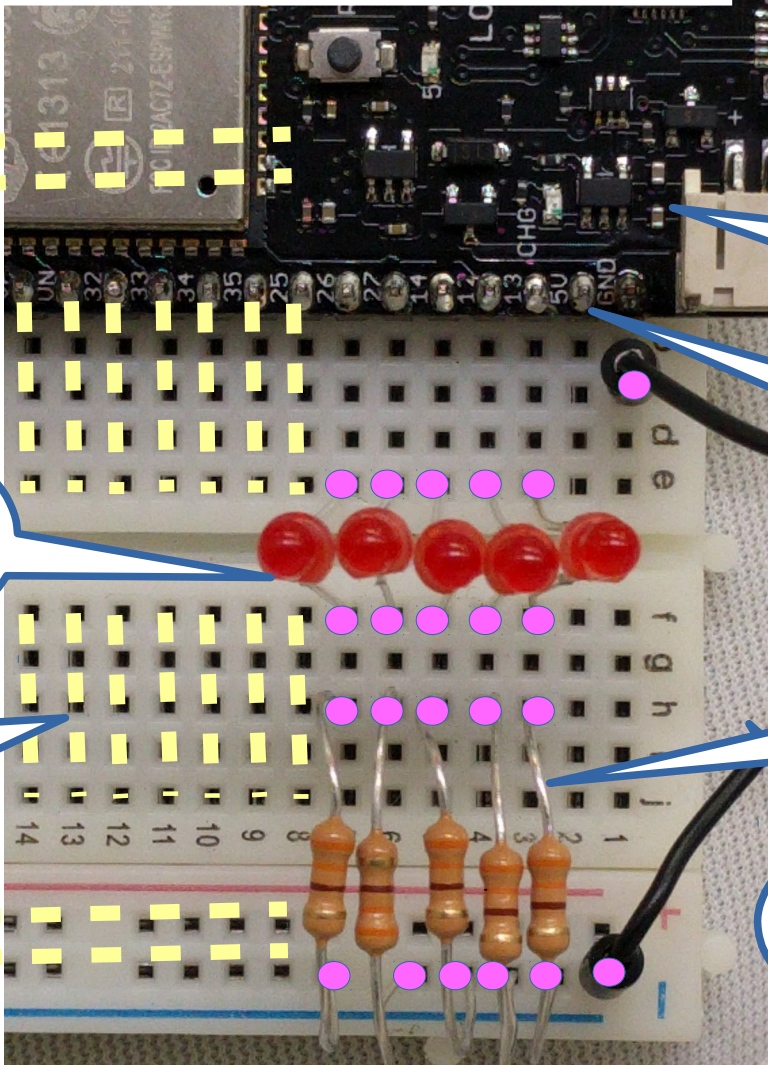
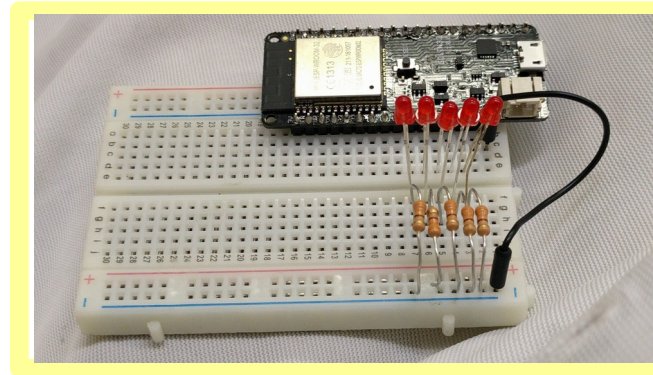
発光ダイオード  
(LED)



- ・光る
- ・光の流れをつくる

# ハードウェア工作

- ①最初に先生が部品(抵抗、発光ダイオード、電線)を取り付けます。よく見ていてください。
- ②つぎに写真を参考にして、部品を取り付けてください。



発光ダイオード  
(LED)  
長い足がプラス側

ブレッドボード  
黄色のように  
つながっています

電子基板  
ソフトウェアを保存し、  
実行する

ピン  
ソフトウェアで指定した  
電気を出力する

抵抗  
電気を流れ  
にくくする

電線  
電気を通す



# 「デモ1」 1つ点滅

ソフトウェア1

実行ボタン  
ソフトウェアをパソコンから  
電子基板に送り、電子基板  
上で実行します。

ArduBlock untitled \*

らしいのをつくる

ほぞん

なまえをつけてほぞん

まえにつくったのをひらく

Arduinoにアップロード

シリアルモニター

Tasks

といきよ

ピン

らべる

いさんする

うとていすう

なみ

esp32

うしん

トレージ

ネットワーク

シリアルコード

The image shows the ArduBlock code editor interface. A blue 'loop' block contains the following code blocks:

- create array ( ESP32 renap data )**: A light blue block with 'へんすう' (variable) set to 'D', 'size' set to '10', and 'D array contents' set to '13, 12, 14, 27, 26, 25, 35, 34, 33, 32'.
- デジタルピンをセット**: A red block with 'ピン#' (pin number) set to 'D # 1' and a purple block below it labeled 'ぶらすのでんき' (wait for power).
- まつ ミリびょう**: An orange block with a value of '1000'.
- デジタルピンをセット**: A red block with 'ピン#' (pin number) set to 'D # 1' and a purple block below it labeled 'まいなすのでんき' (wait for power).
- まつ ミリびょう**: An orange block with a value of '1000'.

Three arrows point to the code blocks: a yellow arrow points to the 'loop' block, a red arrow points to the first 'デジタルピンをセット' block, and a green arrow points to the 'create array' block.

おまじない  
ちょっと長いけど、電子基板の  
ピン番号を順番に並べ直すためのもの。  
もし、DevKitCを使っている場合は  
13,12,14,27,26,25,33,32,35,34に直そう。

Save as image...

Web サイト

v 20170528 (for KIDS B)

# 「デモ2」 光の流れ (発光ダイオード5本)

D#5 D#4 D#3 D#2 D#1

ソフトウェア2

PIN

5

色が違うよ!  
気をつけて!

待ち時間  
時間を1000から減らすと  
流れが速くなる

The screenshot shows the ArduBlock software interface for an ESP32. The main workspace contains a program with the following blocks:

- Loop:** A blue loop block containing the following sequence:
  - create array ( ESP32 renap data ):** A light blue block with 'size' set to 10 and 'D array contents' set to 13, 12, 14, 27, 26, 25, 35, 34.
  - デジタルピンをセット:** A red block with 'ピン#' set to D # PIN and 'ぶらすのでんき' set to 5.
  - まつ ミリびょう:** An orange block with a value of 30.
  - デジタルピンをセット:** A red block with 'ピン#' set to D # PIN and 'まいなすのでんき' set to 5.
  - まつ ミリびょう:** An orange block with a value of 30.

Yellow arrows point from the 'D#5' to 'D#1' labels to the corresponding pins on the hardware board. A speech bubble points to the 'デジタルピンをセット' block, and another points to the 'まつ ミリびょう' block.

# 問題に挑戦しよう。

- (1) 光の流れを逆方向にする
- (2) 光の流れを折り返す
- (3) 発光ダイオードを1つ増やす
- (4) その他

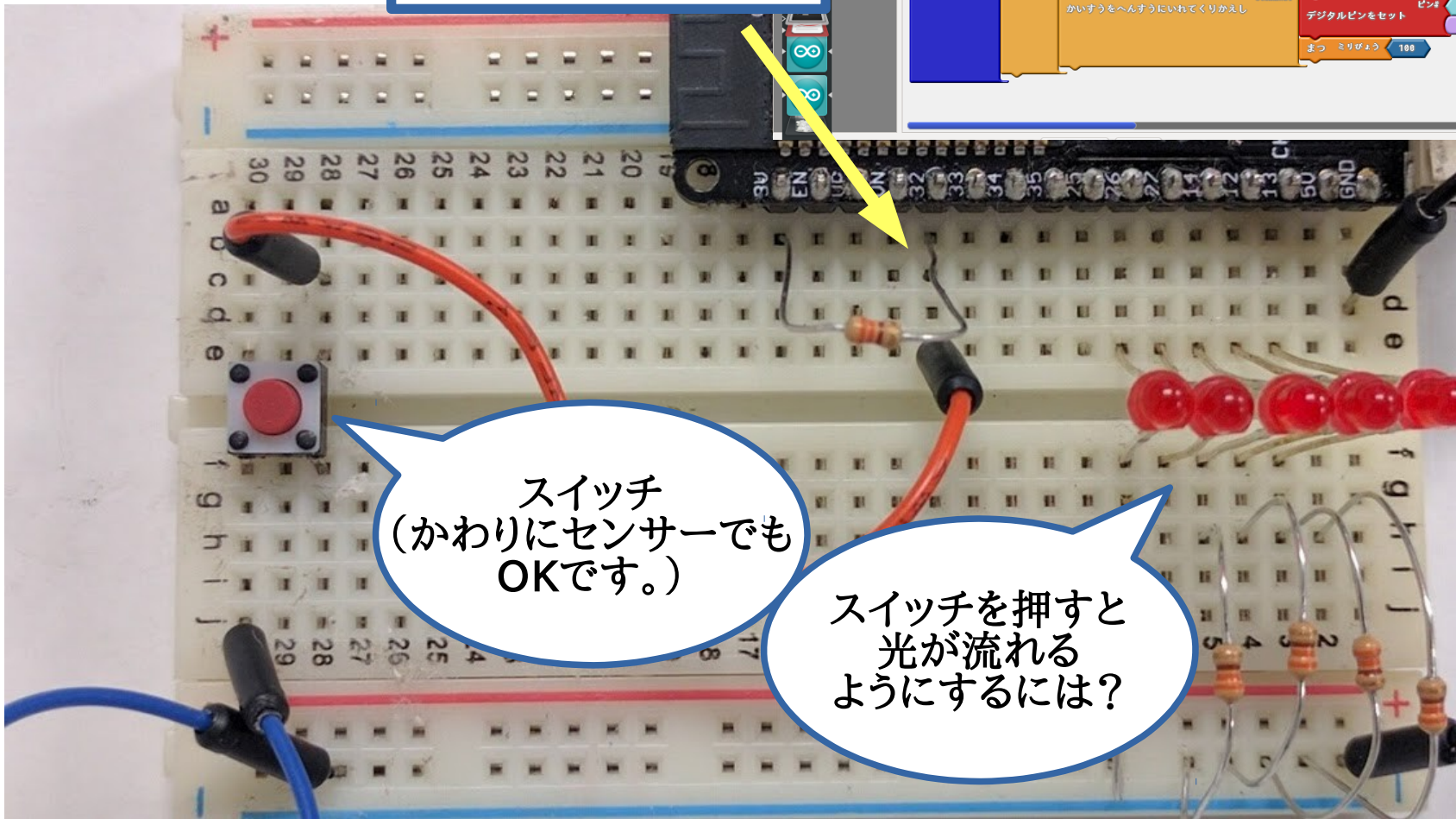
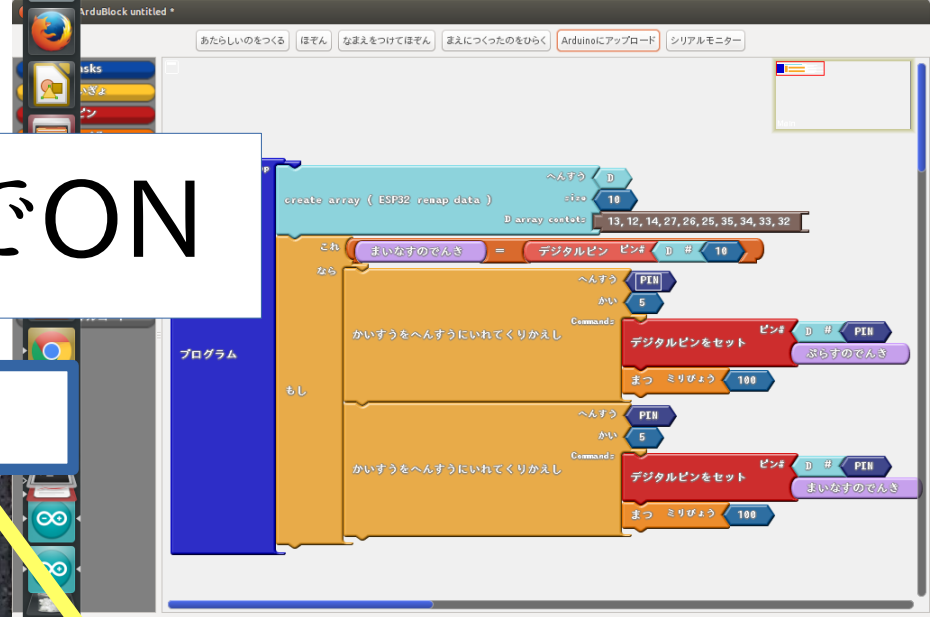
問題をとくアイデアは？

# 「おまけ」スイッチでON

32(D#10)

スイッチ  
(かわりにセンサーでも  
OKです。)

スイッチを押すと  
光が流れる  
ようにするには？

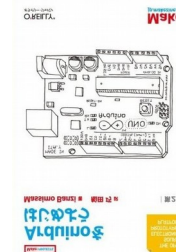


# もっと遊びたい

## 1. 本

Arduinoをはじめよう 第3版 (Make:PROJECTS) ¥2,160

入手先:アマゾン <http://www.amazon.co.jp/> など



## 2. ハードウェア (つぎのいずれか)

(1) 本テキストで使用した電子基板、部品 「AKBONE」 価格は問い合わせください

入手先: ロボット部 <http://groups.google.com/group/robot-android-group-japan-akb>

(2) 「Arduinoで利口なガジェット」用実験キット ¥1500

入手先: アマゾン、秋葉原 若松通商など

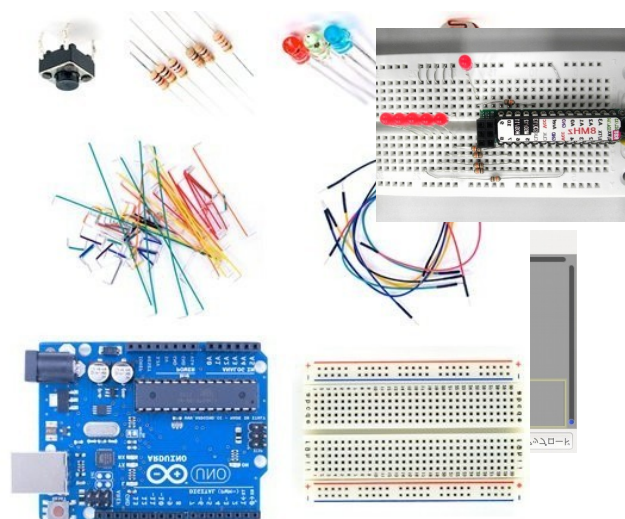
## 3. ソフトウェア開発ソフト (本テキストで使用したソフト)

**Arduino IDE** 無料 開発環境

**ArduBlock** 無料 ビジュアルプログラミング環境

ダウンロードやインストール方法は、ロボット部の「プロジェクトAKBONE」などを参考にしてください。

<<http://sourceforge.jp/projects/akbone>>

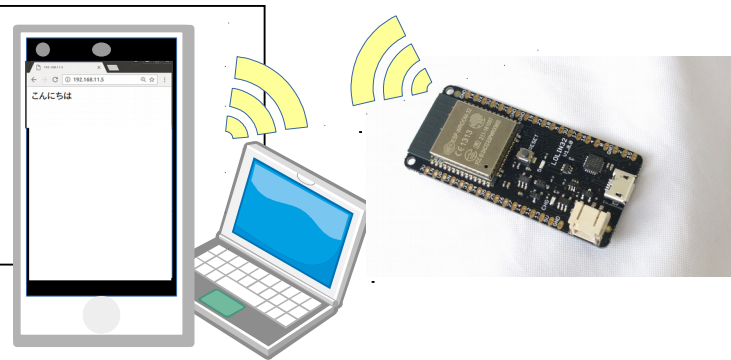


## ワークショップ その2

### 「ネットワーク」と「インターネット」

# IoTの実験

秘密結社オープンフォース



ESP32を使った小さな電子基板を使って、ネットワークやインターネットと連携するプログラムを作ります。

協力: [GEEKLAB.NAGANO](https://www.geeklab.nagano.ac.jp/)

# 「IoTの実験」ワークショップ

## 概要

目的: 「ネットワーク」と「インターネット」につなげる

概要: スマホやパソコンやクラウドと連携

対象: 小学生以上

コース時間: 60分～

コース方式: 個別指導

## 時間

(1) コース説明 5分

(2) WiFiにつなげる 10分

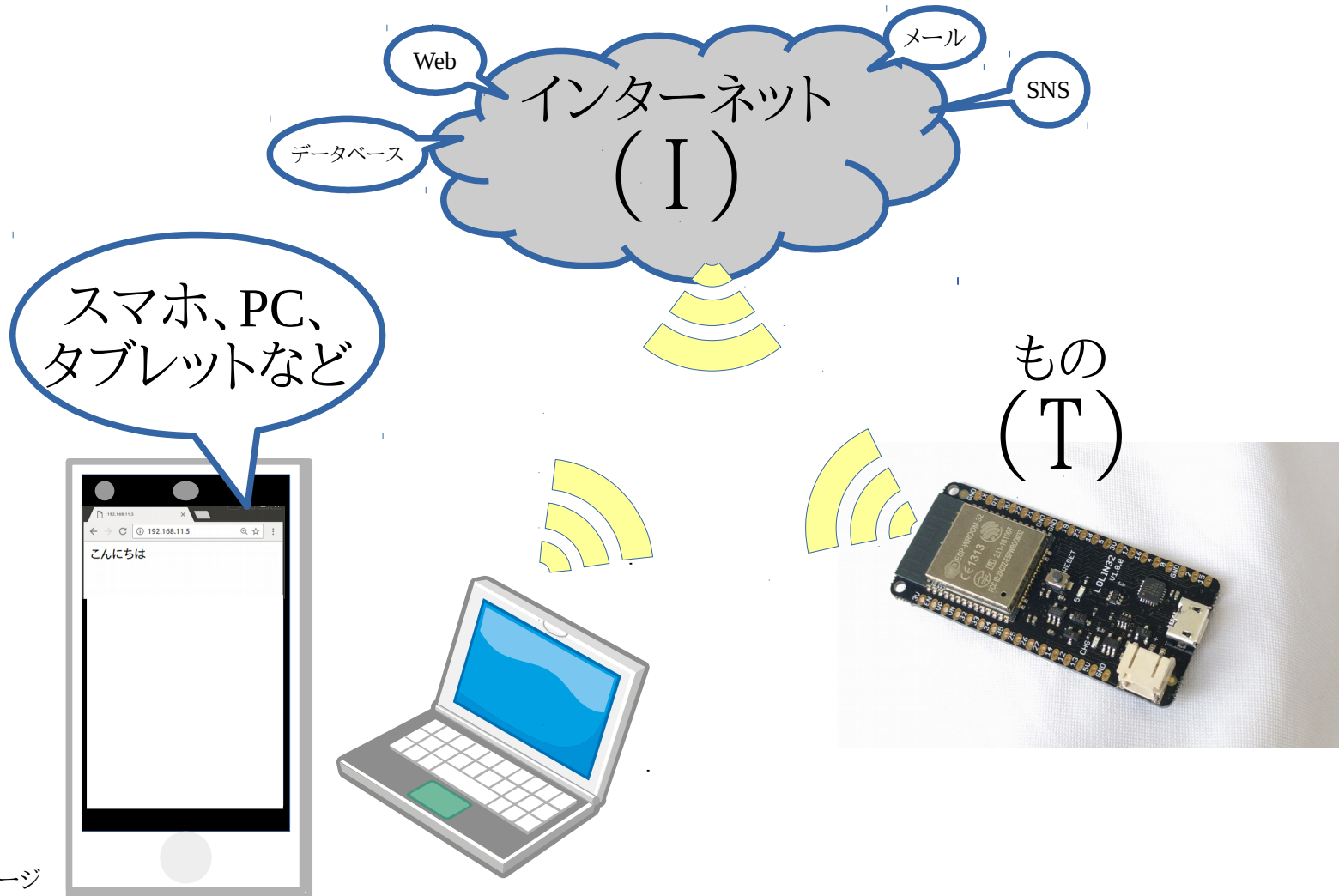
(3) トラブルシューティング(休憩) 10分

(4) 事例デモ(Webサーバ、メール、クラウド連携) 10分

(5) Webサーバにする 10分

(6) 自由課題(インターネットに繋げるなど) 15分

# 全体のつながり





# 「動作チェック」 WiFiにつなぐ

ソフトウェア1

実行ボタン  
ソフトウェアをパソコンから  
電子基板に送り、電子基板  
上で実行します。

ESSIDと  
PASSPHASEは  
ルーターに  
合わせます

The screenshot shows the ArduBlock software interface. On the left is a sidebar with various task categories: Tasks, せいぎよ (Seigyō), ピン (Pin), くらべる (Kureru), けいさんする (Keisan suru), へんすうとていすう (Hensū to teisū), なみ (Nami), esp32, つうしん (Tūshin), ストレージ (Storage), ネットワーク (Network), and スペシャルコード (Special Code). The main workspace contains a code block with the following blocks: 'ESP32 Serial Begin 115200 9600', 'ESP32 WiFi Access Point' (with a WiFi icon) containing 'ESSID Your ESSID' and 'PASSPHASE Your passphrase', and 'ESP32 Web Server' (with 'HTTP!' text) containing 'Web Port 80' and 'Contents HTML contents: こんにちは'. A blue box highlights the 'loop' section of the code block. A red box highlights the 'Main' monitor area on the right. At the bottom, there are buttons for 'Save as image...', 'Web サイト', and 'v 20170528 (for...)'.

日本語は  
コピー&貼り付け  
で入力

# IPアドレスを調べる

①  
シリアルモニタを  
起動しておきます

②  
ここを9600にします

③  
このようになったら  
書込OKです

④  
192.168.11.5  
が  
IPアドレスです



# Webブラウザで見てみる



IPアドレスを入力します

← → ↻ 192.168.11.5

表示できたらOK

こんにちは

文字化けするときには  
下のように変更する

Contents HTML contents こんにちは

Contents HTML contents <meta charset="utf-8"/>こんにちは

# トラブルシューティング!!

IPアドレスを決めたい時

The screenshot shows the Arduino IDE interface with several configuration blocks for an ESP32 WiFi module. A yellow arrow points to the 'ESP32 WiFi IPAddress' block. The configuration details are as follows:

Parameter	Value
IP Address	192.168.11.100
Subnet Mask	255.255.255.0
Default Gateway	192.168.11.1
DNS	8.8.8.8

## PING

OKなとき

```
nanbuwks@LATITUDE: ~  
nanbuwks@LATITUDE:~$ ping 192.168.11.5  
PING 192.168.11.5 (192.168.11.5) 56(84) bytes of data.  
64 bytes from 192.168.11.5: icmp_seq=1 ttl=255 time=1.54 ms  
64 bytes from 192.168.11.5: icmp_seq=2 ttl=255 time=2.84 ms  
64 bytes from 192.168.11.5: icmp_seq=3 ttl=255 time=1.82 ms
```

ダメなとき

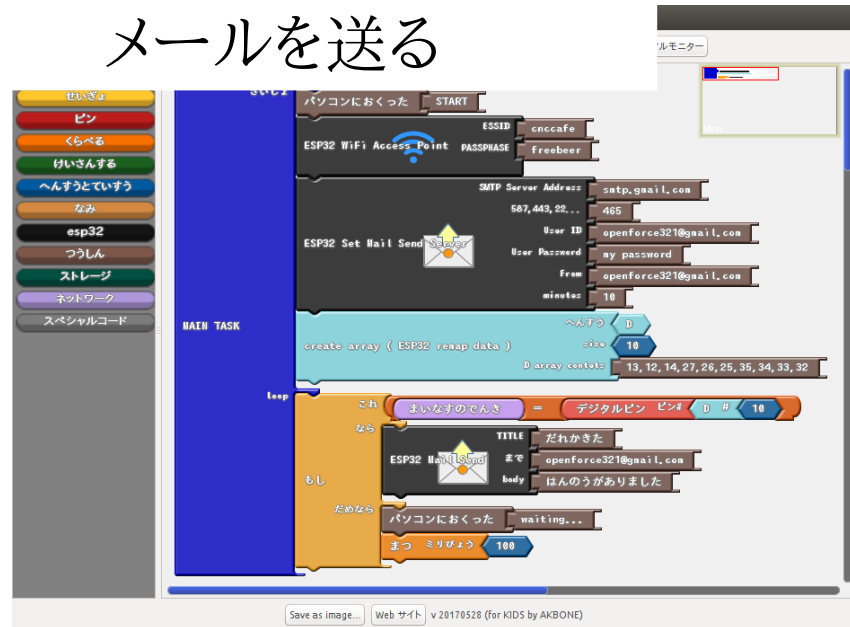
```
nanbuwks@LATITUDE: ~  
nanbuwks@LATITUDE:~$ ping 192.168.11.5  
PING 192.168.11.5 (192.168.11.5) 56(84) bytes of data.  
From 192.168.11.8 icmp_seq=9 Destination Host Unreachable  
From 192.168.11.8 icmp_seq=13 Destination Host Unreachable  
From 192.168.11.8 icmp_seq=14 Destination Host Unreachable
```

# 「デモ」

## WebでON/OFF



## メールを送る



クラウドに接続して  
TwitterやLineに送る



# 「Webサーバ」ブラウザでOn/Off

ソフトウェア1

The image shows a screenshot of the ArduBlock IDE interface. On the left, there is a sidebar with a "Tasks" menu containing various categories like "せいぎょ" (Control), "ピン" (Pin), "くらべる" (Compare), "けいさんする" (Calculate), "すうとていすう" (Number and Text), "なみ" (Name), "esp32", "つうしん" (Communication), "ストレージ" (Storage), "ネットワーク" (Network), and "スペシャルコード" (Special Code). The main workspace displays a program for an ESP32. The program starts with "ESP32 Serial Begin" at 9600 baud. It then sets up WiFi with ESSID "cnccafe" and PASSPHASE "freebeer". A "create array ( ESP32 renap data )" block is followed by a "Loop" block. Inside the loop, there are two conditional blocks. The first block checks for a "GET /ON" request. If true, it sets a digital pin (pin # 1) to "ぶらすのでんき" (High). The second block checks for a "GET /OFF" request. If true, it sets the same digital pin to "まいなすのでんき" (Low). A green box highlights a connector labeled "A" in the top right, which is connected to the "へんすう" (Variable) input of the "デジタルピンをセット" blocks. Other colored arrows (red, yellow, black, grey) point from the "Tasks" menu to various parts of the program.

# だれかきた..はんのうがありました のメールを送る

ArduBlock untitled \*

あたらしいのをつくる ほぞん なまえをつけてほぞん まえにつくったのをひらく Arduinoにアップロード シリアルモニター

Tasks  
せいぎょ  
ピン  
くらべる  
けいさんする  
へんすうとていすう  
なみ  
esp32  
つうしん  
ストレージ  
ネットワーク  
スペシャルコード

MAIN TASK

パソコンにおくった START

ESP32 WiFi Access Point  
ESSID cnccafe  
PASSPHASE freebeer

ESP32 Set Mail Send Server  
SMTP Server Address smtp.gmail.com  
587, 443, 22... 465  
User ID openforce321@gmail.com  
User Password ny password  
From openforce321@gmail.com  
minutes 10

へんすう D  
create array ( ESP32 renap data ) size 10  
D array contents 13, 12, 14, 27, 26, 25, 35, 34, 33, 32

Loop

これ  
まいなすのでんき = デジタルピン ピン# D # 10

ESP32 Mail Send  
TITLE だれかきた  
まで openforce321@gmail.com  
body はんのうがありました

もし  
だめなら  
パソコンにおくった waiting...  
まつ ミリびょう 100

# Web(クラウド)にデータを送る

ArduBlock untitled \*

あたらしいのをつくる ほぞん なまえをつけてほぞん まえにつくったのをひらく Arduinoにアップロード シリアルモニター

Tasks  
せいぎょ  
ピン  
くらべる  
けいさんする  
へんすうとていすう  
なみ  
esp32  
つうしん  
ストレージ  
ネットワーク  
スペシャルコード

MAIN TASK

さいしょ

パソコンにおくった START

ESP32 WiFi Access Point  
ESSID: cnccafe  
PASSPHASE: freebeer

へんすう D  
create array ( ESP32 renap data ) size: 10  
D array content: 13, 12, 14, 27, 26, 25, 35, 34, 33, 32

これなら

まいなすのでんき = デジタルピン ピン# D # 10

デジタルピンを設定  
ピン# D # 1  
ぶらすのでんき

ESP32 Web Client  
HOST: maker.ifttt.com  
path: /trigger/esp32 to twitter/with/key/dsDrejzIHnEiB...

もし

まつ ミリびょう 100000

デジタルピンを設定  
ピン# D # 1  
まいなすのでんき

だめなら

パソコンにおくった waiting...

まつ ミリびょう 100



# クラウドから送る

今回は、「IFTTT」という無料のサービスを使います。  
ワークショップでは、あらかじめ設定したキーを使います。

IFTTT → Gmail

[https://maker.ifttt.com/trigger/esp32\\_to\\_gmail/with/key/](https://maker.ifttt.com/trigger/esp32_to_gmail/with/key/)

IFTTT → twitter

[https://maker.ifttt.com/trigger/esp32\\_to\\_twitter/with/key/](https://maker.ifttt.com/trigger/esp32_to_twitter/with/key/)

自分で設定するには、別紙「IFTTTの設定.pdf」を参照してください

# 考えよう!

- (1) どうしてメールは10分に1つなの?
- (2) パスワードはどうして隠すの?

## 通信は相手がある

相手に迷惑をかけないようにするには?

# 「本格的に実験するには」

1. IFTTTの設定をする方法
2. GMailのメールサーバを使ってメールを送る方法
3. インストールするには(設定済み環境をインストールする)
4. インストールするには(自分で設定する)
5. 設定を整える
6. わからないことがあったとき

## 1. IFTTTの設定をする方法

ワークショップでは設定済みIFTTTの仕組みを使います。  
自分で設定をすることもっと詳しく設定できます。

やってみたい人は別紙資料「IFTTTの設定.pdf」を参照して下さい。

## 2. Gmailのメールサーバを使ってメールを送る方法

- Gmailを使っている場合はその仕組みを使うことができます。
- <https://support.google.com/a/answer/176600?hl=ja>  
この資料の「制限付き Gmail SMTP サーバーを使用する」設定を済ませておく必要があります。
- メールサーバのユーザIDとメールの送信元は同じにする。
- 自分のPCの置き場所と別の場所からメールを送ろうとしたときは、自分のPCに警告が出るので解除して下さい。



# 3. インストールするには (設定済み環境をインストールする)

プロジェクトAKBONEから一式をダウンロードできます  
<https://ja.osdn.net/projects/akbone/releases/>

ダウンロード→解凍→自分の好きなところに置いて、  
インストールして下さい

- ・Linux64環境は `install.sh`を実行して下さい。
- ・Windows環境／Linux64環境は `portable` 環境です。
- ・Mac OSX環境は通常バージョンです

# 4. インストールするには(自分で設定する)

1.Arduino開発環境をダウンロード、インストールする

<http://www.arduino.cc>

2.ESP32環境をダウンロード、インストールする

<https://github.com/espressif/arduino-esp32>

から ZIP ファイルをダウンロード、

展開してできた arduino-esp3-master フォルダの名前を esp32に変更、

hardware フォルダ(\*)内に espressifフォルダを作りその中に移動(hardware/espressif/esp32となる)

その中のtoolsフォルダの中のインストーラを実行(Windowsはget.ext Mac/Linuxは python get.py)

3.Ardublock環境をダウンロード、インストールする

<https://ja.osdn.net/projects/akbone/downloads>

からardublock-all.jarをダウンロード、tools フォルダ(\*)の ArduBlockTool/tool フォルダを作って配置

4.Mailライブラリをダウンロード、インストールする

<http://kerikeri.top/posts/2017-04-08-esp32-mail/>

からmail.zipをダウンロードして展開、libraries フォルダ(\*)のMailerフォルダを作りそこにmailer.h配置

5.BLEライブラリをダウンロード、インストールする(エラーが出るとき)

<https://github.com/espressif/arduino-esp32/tree/master/libraries> の BLE @ cb1ab4e (名前が変わる場合があります)

のリンク先からZIPファイルをダウンロード、

展開してできたexamplesやsrcなどを 2.の hardware/espressif/esp32/libraries/BLE内にコピー

6.USBシリアルドライバをインストールする

<https://jp.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

「CP210x USB - UART ブリッジ VCP ドライバ」( Linux の多くはインストールの必要はありません)

hardwareフォルダ、toolsフォルダ、librariesフォルダの場所について  
・通常の場合はドキュメントフォルダのArduinoの中です。  
・portable環境の場合はportableフォルダを置いているところと同じ所にあります。

# 5. 設定を整える

「ファイル」-「環境設定」

The image shows the Arduino IDE interface with the 'Environment Settings' dialog box open. The dialog has tabs for 'Settings' and 'Network'. Under 'Settings', the 'Sketchbook location' is set to '/home/nanbuwks/Arduino'. The 'Language' is set to 'Follow system'. The 'Editor font size' is 12. The 'Interface scale' is set to 'Automatic' at 100%. The 'Show more detailed information' section has 'Compile' unchecked and 'Save' checked. The 'Compiler warnings' are set to 'None'. There are several checkboxes for compiler options: 'Show line numbers' (unchecked), 'Make code folding effective' (unchecked), 'Check save' (checked), and 'Use external editor' (unchecked).

The 'Tools' menu is open, showing options like 'Auto format', 'Archive sketch', 'Fix encoding', 'Serial monitor', 'Serial plotter', 'ArduBlock', 'WiFi101 Firmware Updater', 'Board: "WEMOS LOLIN32"', 'Flash Frequency: "80MHz"', 'Upload Speed: "921600"', 'Serial port: "/dev/ttyUSB0"', 'Get board information', and 'AVRISP mkII'. The 'Serial port' option is highlighted in orange, and the 'Serial port' dropdown menu is open, showing '/dev/ttyUSB0' selected with a checkmark.

Yellow arrows point from the text '「ファイル」-「環境設定」' to the 'File' menu and the 'Environment Settings' dialog. A blue speech bubble points to the 'Serial port' dropdown menu with the text: 'シリアルポート: ESP32ボードをつなげたら出現するポートを選びます'. Another blue speech bubble points to the 'Board: WEMOS LOLIN32' option with the text: 'ボード: WEMOS LOLIN32'. A third blue speech bubble points to the 'Check save' checkbox with the text: 'チェックつける'.

シリアルポート:  
ESP32ボードを  
つなげたら出現する  
ポートを選びます

ボード: WEMOS LOLIN32

チェックつける

# 6. もっとよく知りたい時

新版のソフトウェアや基板は <https://ja.osdn.net/projects/akbone/> にて公開しています。

勉強会やコミュニティに参加してみよう!



日本Androidの会秋葉原支部ロボット部

月例で勉強会を行なっています(東京都内)。  
興味のある方はML(メーリングリスト)にご登録ください。

<http://groups.google.com/group/robot-android-group-japan-akb>

MLでは勉強会のアナウンスや部員が興味をもった様々な話題がなされています。



秘密結社オープンフォース

ArduBlockのESP32バージョンの開発を行っています。

開発のトピックは <https://qiita.com/nanbuwks> などで公開しています。



GEEKLAB.NAGANO

長野市で、ITやテクノロジー関係に関する勉強会、イベント、セミナーを開催しています!  
平日 9:00~18:00 は無料開放しておりますので、作業場としてなどご利用いただけます。  
予約等は不要ですので、お気軽にお越しください!(土日祝日は問い合わせ要)

住所:長野県長野市南県町1003 県都ビル6F