

ARE SERIES

# ARE-ARM1 プログラミング

Scratch2.0 で制御するロボットアーム



Ability

## はじめに

当マニュアルは、Scratch（※1）の拡張機能を使用して、ARE-ARM1（Ability robot education-Arm1）ロボットアーム制御をおこなうためのマニュアルです。

「ARE-ARM1 プログラミング準備マニュアル(Windows10 Scratch2.0 対応版)」で、環境設定がおこなわれていること、下記サイトのアームの調整がおこなわれていることを前提としています。

<https://robot.ability-evolves.com/are-arm1-electronic-work/>

また、ロボットアーム制御に特化して解説しますので、Scratch2.0 の基本的な操作方法やプログラミングについては、様々なサイトで解説されていますので、割愛させていただきます。

それでは Scratch2.0 のプログラミングでロボットアームを動かしてみましょう。

### ※1 Scratch

Scratch(スクラッチ)は、ブロック型のビジュアルプログラミング言語及びその開発環境で、Scratch 財団が、マサチューセッツ工科大学メディアラボ ライフロングキンダーガーデングループ (MIT Media Lab Lifelong Kindergarten Group) と共同開発しました。

基本的な操作方法やプログラミングについては、以下の公式サイトなどを参照してください。

<https://scratch.mit.edu/>

日付	Ver	内容	担当者
2021/06/01	1.0	新規	吉田

## 目次

はじめに .....	2
§1. ロボットアーム制御用のブロック .....	5
1. ブロック機能一覧.....	5
2. 可動域 .....	6
§2. プログラミング .....	7
1. 基礎編 .....	7
1. Arm1 拡張機能の読み込み .....	7
2. LED を ON/OFF する .....	9
3. グリッパーを開閉する .....	10
4. アームの各軸を動かす .....	11
5. ブロックを移動する .....	13
6. アーム動作を PC のテンキーに割り付ける .....	15
2. 応用編 動作指示ファイルの作成と実行プログラム .....	16
1. サンプルプログラムの読み込み .....	16
2. サンプルプログラムの内容について .....	17
§2. 特記事項 .....	21

## §1. ロボットアーム制御用のブロック

### 1. ブロック機能一覧

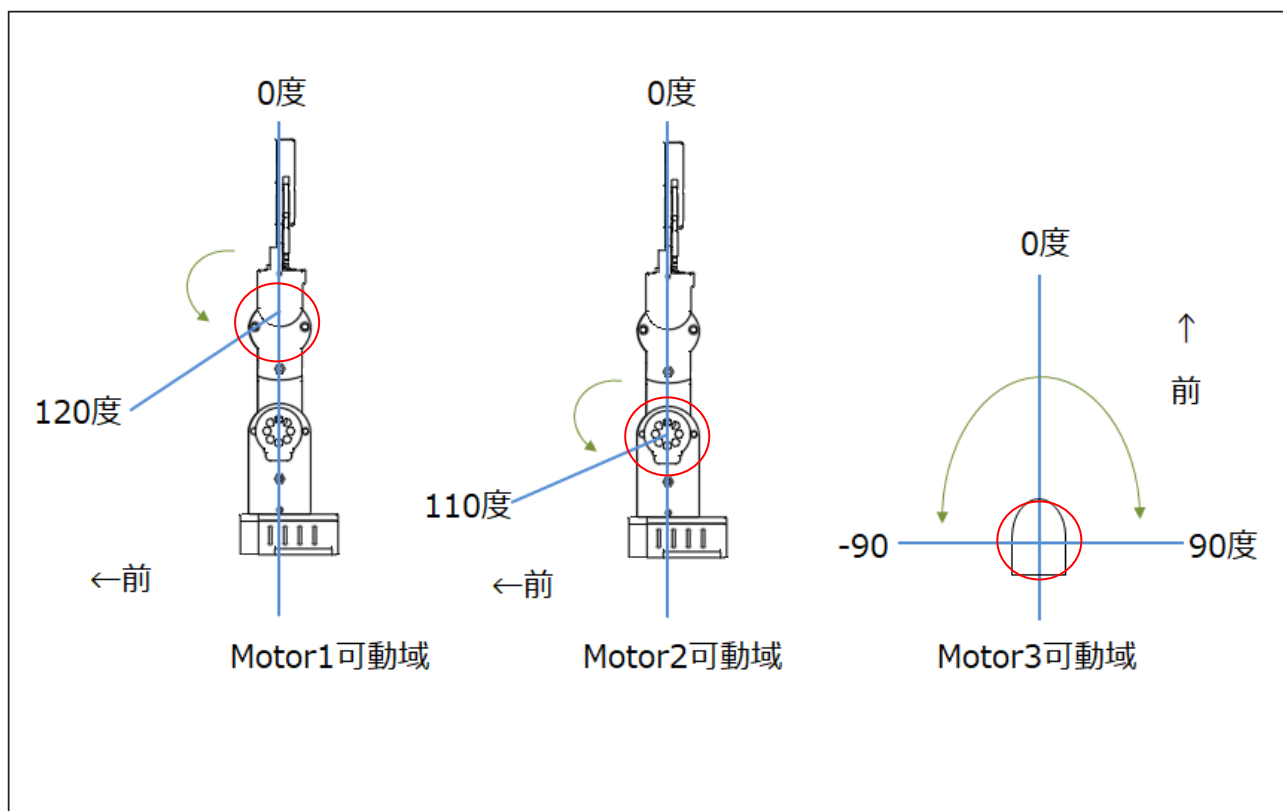
ARE-ARM1 ロボットアームを動かすためのブロック一覧です。

No	ブロック	機能内容	備考
1		LED の ON/OFF を選択	ON : 点灯 OFF : 消灯
2		グリッパーの速度を選択	1 (早い)~5(遅い)
3		グリッパーの開閉位置を入力	10(開)~100(閉)の範囲内で、 グリッパー調整時の開閉の値を 入力します ※閉値は 70~100 くらい グリッパーにより異なります
4		各軸モーターの現在位置を原点 に設定	各アーム軸の△マークを合わせ て、アームを一直線の形状にして から原点を設定してください
5		全軸モーターの速度を選択	1 (早い)~5(遅い)
6		指定モーターの現在の角度に、入 力した角度を加算して移動	Motor1~Motor3 相対移動
7		指定モーターを入力した角度に 移動	Motor1~Motor3 絶対移動
8		各モーターの現在の角度に、入力 した角度を加算して移動	Motor1~Motor3 相対移動
9		各モーターを入力した角度に移 動	Motor1~Motor3 絶対移動
10		モーターの角度表示有無をチェ ック	ON : ステージエリアに表示 OFF : ステージエリアに非表示
11		グリッパーの位置表示有無を チェック	ON : ステージエリアに表示 OFF : ステージエリアに非表示

## 2.可動域

各軸モーターの可動域と稼働方向は以下のとおりです。

モーター	場所	可動域
Motor1	アーム中モーター	0～120 度
Motor2	アームボディモーター	0～110 度
Motor3	アーム台モーター	-90～90 度



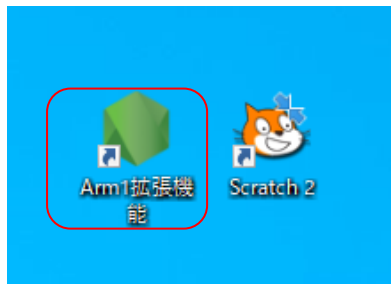
## §2.プログラミング

### 1.基礎編

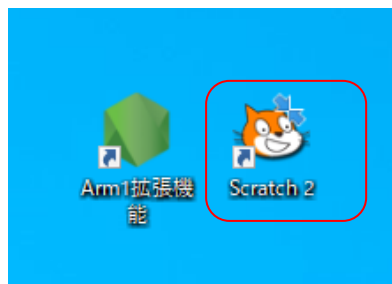
#### 1.Arm1 拡張機能の読み込み

まず scratch と ARE-ARM1 の通信をサポートする「Arm1 拡張機能」を使用できるようにします。

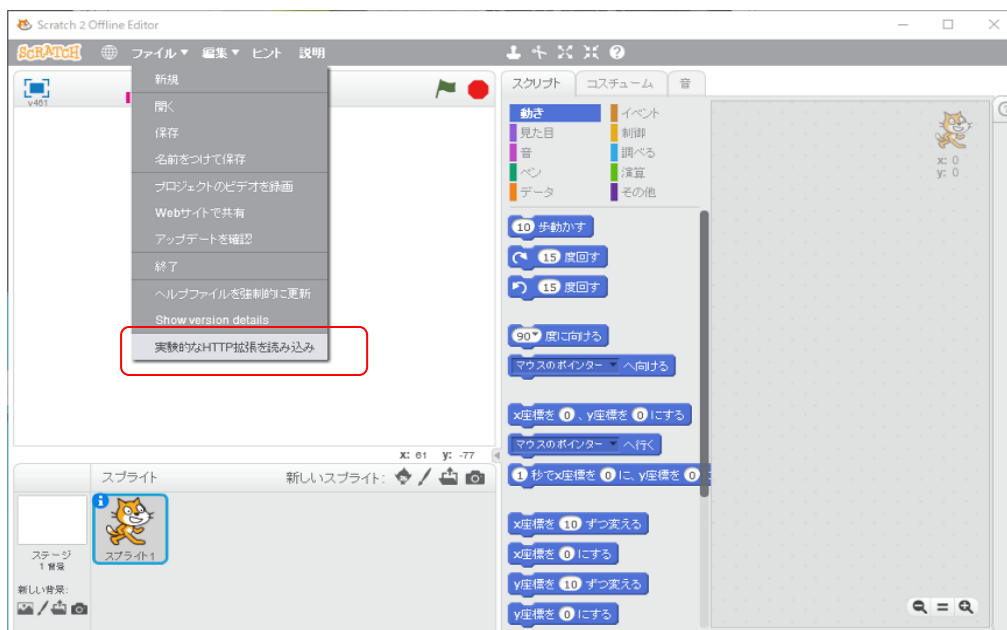
①デスクトップの「Arm1 拡張機能」をダブルクリックして起動します。



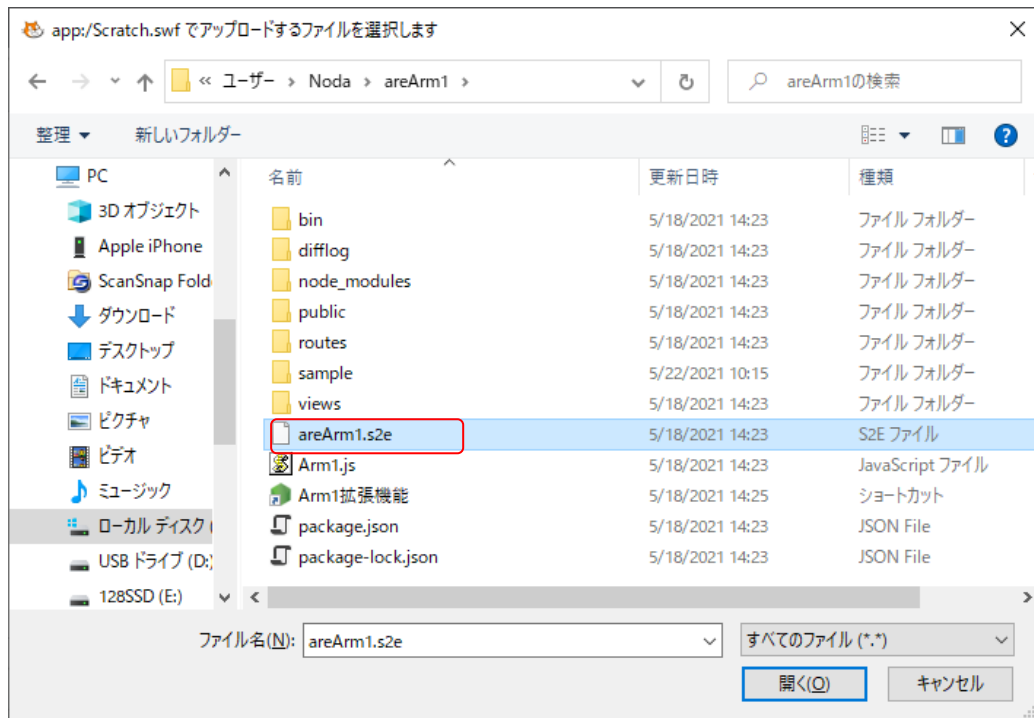
②次に「Scratch2」をダブルクリックして起動します。



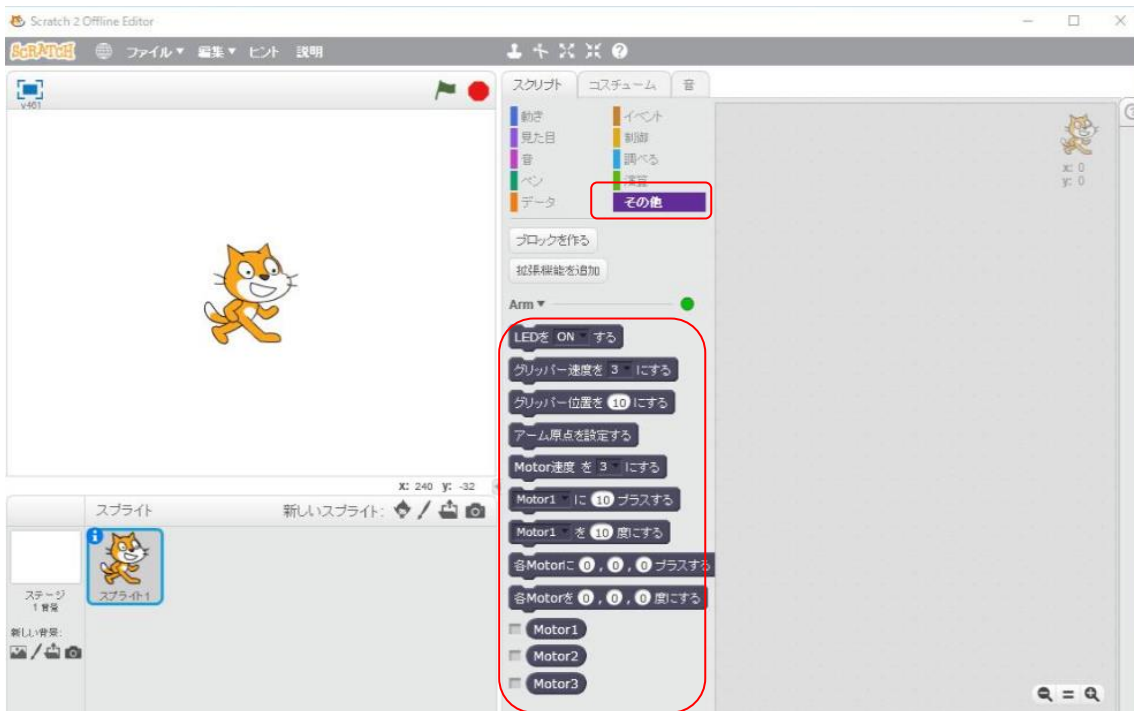
③SHIFT キーを押しながら、メニューバーの「ファイル」をクリックし、「実験的な HTTP 拡張を読み込み」を選択します。



④ C:\Users\【ユーザー名】\areArm1\areArm1.s2e を選択し「開く」を押下します。



⑤ 「スクリプト」タブの「その他」を選択して Arm ブロックが読み込まれていることを確認します。




これで scratch から ARE-ARM1 にブロック命令を出せます。




## 2.LED を ON/OFF する


では、はじめに ARE-ARM1 の基板の LED を 3 回点滅させるプログラムを作って実行してみましょう。

①「スクリプト」タブの「イベント」の  を、scratch 画面右のスクリプトエリアにドラッグ・アンド・ドロップ（クリックしたまま移動して放す）します。

以後は、scratch 画面右のスクリプトエリアにドラッグ・アンド・ドロップすることを、配置すると表記します。

②「スクリプト」タブの「制御」を開き、 3 回に変更して配置します。

③「制御」の  を、「3 回繰り返し」ブロック内に配置します。（以下④～⑦もブロック内に配置）  
Scratch と ARE-ARM1 の ESP32-DevKitC との通信速度を考慮して、待ち時間を設定します。

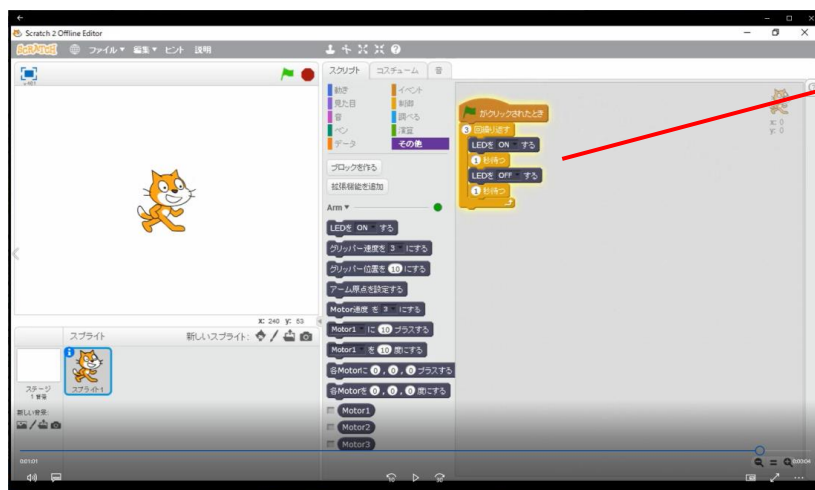
④「その他」の  を配置します。

⑤「制御」の  を配置します。

⑥「その他」の  を OFF に変更し配置します。

⑦「制御」の  を配置します。


⑧下図のようにスクリプトエリアにブロックが配置出来たら  をクリックして実行します。




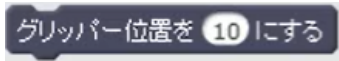
ARE-ARM1 の LED が 3 回点滅します。

### 3.グリッパーを開閉する

グリッパーを開閉してみましょう。開閉速度を変えて2回開閉します。

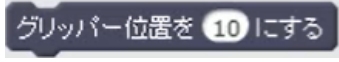
①「スクリプト」タブの「イベント」を開き、 を配置します。

②「その他」の  を、1(早い)に変更して配置します。

③「その他」の  を配置します。 値は10(全開)です。

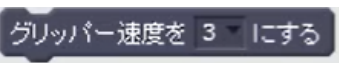
④「制御」の  を配置します。

グリッパーの動作速度を考慮して、待ち時間を設定します。


⑤「その他」の  を 80(全閉)に設定して配置します。

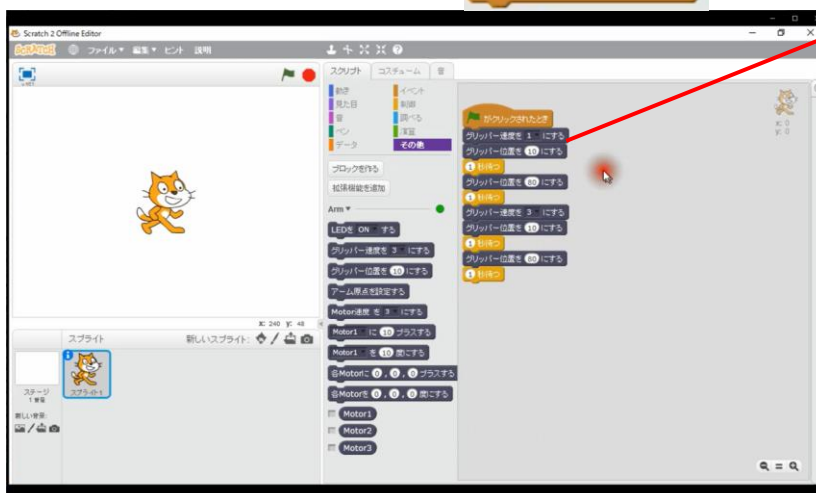
※全閉の値はグリッパーによって違います。ご自身のグリッパーの開値を入力してください。

⑥「制御」の  を配置します。

⑦「その他」の  を3(普通)のままで配置します。

⑧④～⑥を繰り返します。

⑨下図のようにブロックが配置出来たら  をクリックして実行します。



グリッパーが速度を変えて2回開閉します。

#### 4.アームの各軸を動かす

アームの各軸を動かしてみましょう。

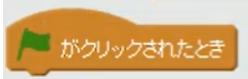
最初にアームの原点を設定し、ハードとソフトの原点を揃えます。

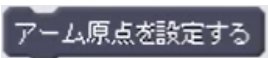
Motor1～3 を順番に 10 度ずつ加算（相対値指定）して、3 回繰り返し動かします。

アームの動作速度を考慮して、待ち時間を設定します。

1 回目で 10 度の位置、2 回目で 20 度位置、3 回目で 30 度の位置に移動します。

最後に絶対値指定で原点に戻ります。

①「スクリプト」タブの「イベント」を開き、 を配置します。


②「その他」の  を配置します。

アーム軸の△マークを合わせて一直線にし、各軸を原点位置にしてください。


原点位置の調整の詳細は、下記サイトの、「3.アームの調整」の①を参照してください。

<https://robot.ability-evolves.com/are-arm1-electronic-work/>


③「スクリプト」タブの「制御」を開き、 3 回に変更して配置します。

④「その他」の  を配置します。

⑤「制御」の  を配置します。

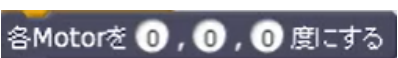
⑥「その他」の  を、Motor2 に変更して配置します。

⑦「制御」の  を配置します。

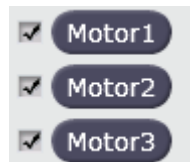
⑧「その他」の  を、Motor3 に変更して配置します。

⑨「制御」の  を配置します。

⑩④～⑨は③の繰り返しブロックの中に配置します。

⑪「その他」の  を配置します。

⑫「その他」の

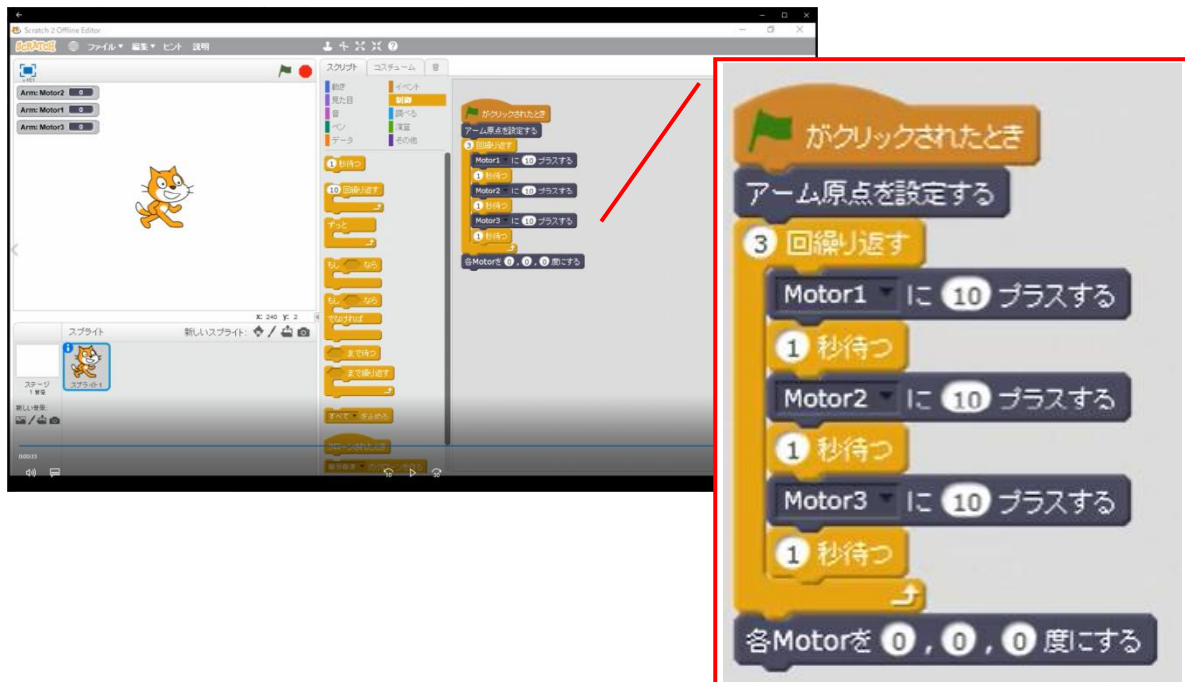


のチェックを ON します。

⑬下図のようにブロックが配置出来たら



をクリックして実行します。



Motor1～3 を順番に 10 度ずつ加算（相対値指定）して、3 回繰り返し動かします。

最後に絶対値指定で原点に戻ります。

ステージエリアに Motor1～3 の現在角度が表示されます。

※アーム移動が完了する前に次の命令を受信した場合、取りこぼす場合があります。

アームの動作速度を考慮して、待ち時間を設定してください。

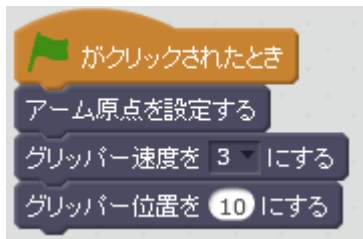
## 5.ブロックを移動する

ブロックを移動するプログラムを作ってみましょう。

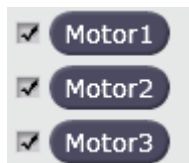
まずお約束の原点位置を設定し、ブロックをグリッパーで掴める位置の各軸角度と、移動先の位置の各軸角度を調べます。移動命令ブロックに調べた値を設定します。

①アーム本体の△マークを合わせて一直線にしておきます。

②スクリプトエリアに以下ブロックを配置し、実行します。



③「その他」の Motor1～Motor3 のチェックを ON し、各モーターの現在角度を表示します。



④「その他」の下図ブロックをクリックしてグリッパーでブロックを挟める位置に移動します。



それぞれのブロックを個別にクリックして、  
ブロックの掴める位置に移動します。

⑤ブロックを掴める位置に移動したら、各モーターの角度がステージエリアに表示されていますので



に角度を設定して配置します。

各軸の角度



⑥同様にしてブロックの移動先の位置に移動したら、その時の各軸角度を設定して配置します。  
以下のようなプログラムになります。



scratch と ESP32-DevKitC との通信速度や、  
アームの移動速度を考慮して、  
待ち時間を設定します

この角度の位置に移動させます



をクリックして実行してみましょう。

以下サイトの動画も参考にしてください。

<https://robot.ability-evolves.com/working-on-electronics/arm1/programming-Scratch2/>

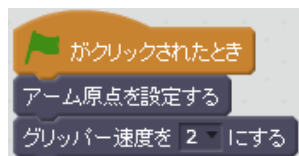
#### 1.基本編

- 1.ブロック移動プログラム作成
- 2.ブロック移動プログラム実行

## 6. アーム動作を PC のテンキーに割り付ける

アーム動作を PC のテンキーに割り付け、アームをキーボードから動かしてみましょう。

①アームの原点を設定して実行しておきます。



②テンキーに動作を割り付けます。下図の内容でキーが押されたときのブロックを配置します。

Num Lock	/	*	—
7 グリッパを開(10)に	8 Motor2に-10度プラス	9 グリッパを閉(60)に	+
4 Motor3に-10度プラス	5 Motor1に-10度プラス	6 Motor3に10度プラス	
1	2 Motor1に10度プラス	3	Enter
0 Motor2に10度プラス		.	

スクリプトエリアのキー割付ブロック



割り付けたキーを押すことで、アームが動作します。

プラスする値を加減することで、もっと細かくも、もっと大きくも動かすことができます。

グリッパの閉値は、掴みたい物のサイズに合わせて値を変更してください。



## 2.応用編 動作指示ファイルの作成と実行プログラム

アームをキーボード操作で動かし、その動作を保存して再現するサンプルプログラムを用意しました。

C:\Users\【ユーザー名】\areArm1\sample\arm1InstructionFile.sb2 です。

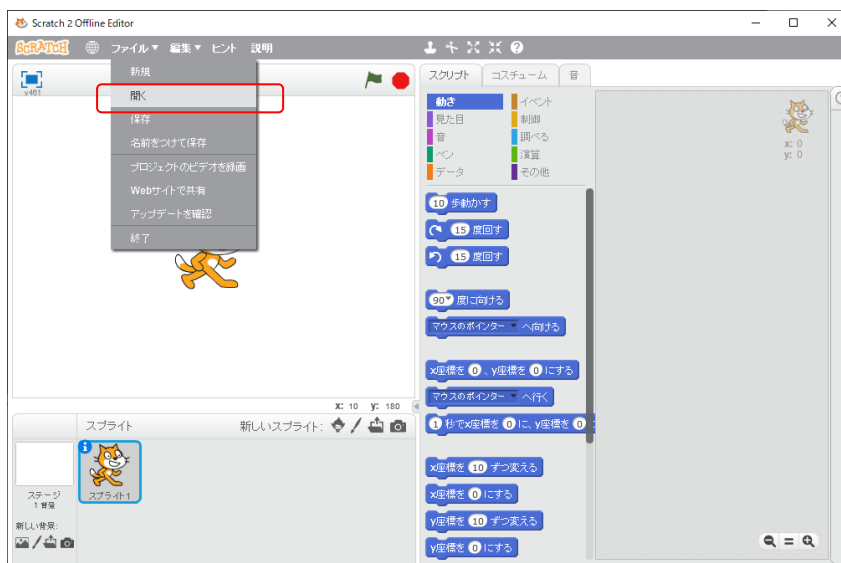
サンプルプログラムですので、エラー処理等は省略しています。ご了承ください。

arm1InstructionFile.sb2 の内容を編集する場合は、万一、プログラムが動かなくなった時に備えて、arm1InstructionFile.sb2 のコピーを取っておくと元に戻せて安心です。

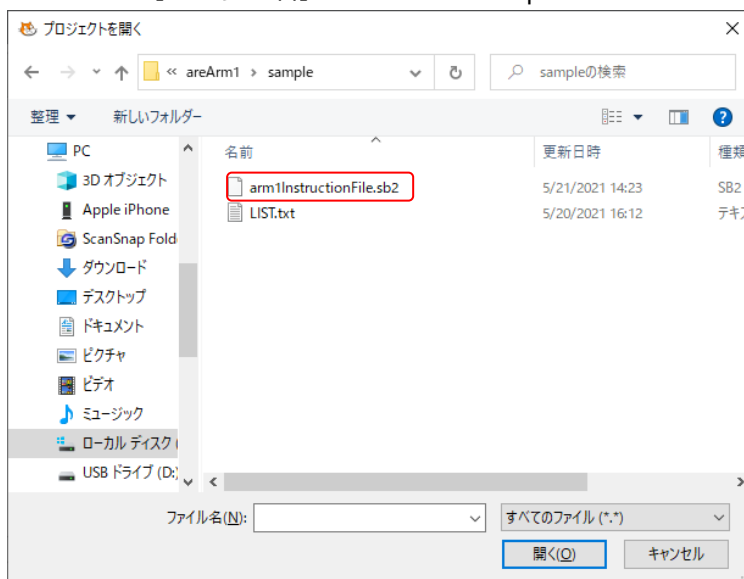
### 1. サンプルプログラムの読み込み

以下の手順で Scratch にサンプルプログラムを読み込みます。

メニューバーの「ファイル」をクリックし、「開く」を選択します。



C:\Users\【ユーザー名】\areArm1\sample\arm1InstructionFile.sb2 を選択し「開く」を押下します。



サンプルプログラムが Scratch に表示されます。



## 2. サンプルプログラムの内容について

サンプルプログラム(`arm1InstructionFile.sb2`)の機能について説明します。

### 1) アーム動作のテンキー割付

プログラミング基礎編の「6. アーム動作を PC のテンキーに割り付ける」と同様の機能です。

### 2) 動作指示ファイルの保存

動作指示データ（各軸の角度と、グリッパーの開閉値）をテキストファイルに保存する機能です。

キーの動作割付は下表のとおりです。

キー	処理
s	現在の位置情報（各軸の角度、グリッパーの開閉位置）を、スペース区切りで LIST に追加する  [例] Motor1 角度:10 Motor2 角度:20 Motor3 角度:30 グリッパー位置:60 の場合 "10 20 30 60"を LIST に追加
d	LIST の最後の位置情報を LIST から削除する

#### ■ 使用する変数

str 現在の位置情報を入れる

LIST 動作指示データを入れるリスト

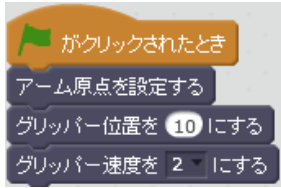
#### ■ スクリプトエリアのキー割付ブロック

Str に Motor1、スペース、 Motor2、スペース、Motor3、  
スペース、グリッパー位置を入れる



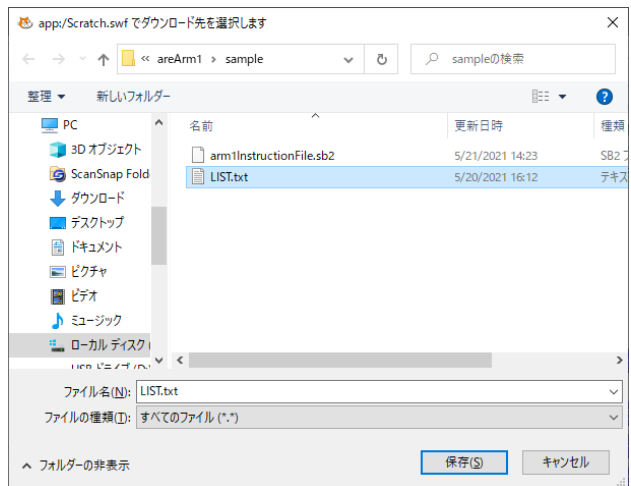
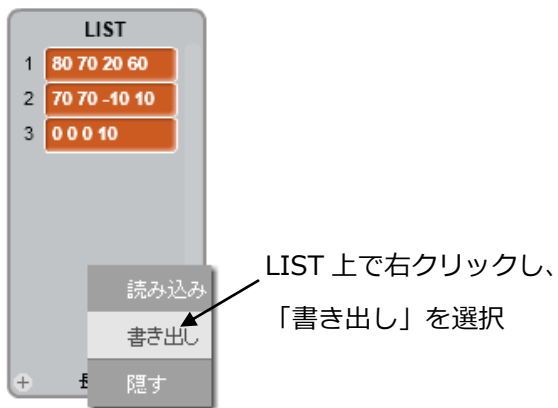
## ■操作方法

①アームの原点を設定して実行しておきます。



②テンキーを押してアームを移動し、覚えておきたいアームの位置で s キーを押します。

③LIST が出来たら、LIST 上で右クリックし、「書き出し」を選択します。



「保存」を押して動作指示ファイルを保存します。

### 3)動作指示ファイルの読み込みと実行

動作指示ファイルからデータを読み込んで実行する機能です。

キーの動作割付は下表のとおりです。

キー	処理
スペース	LIST から位置情報（各軸の角度、グリッパーの開閉位置）を読み込み、アームとグリッパーを動作させる

## ■使用する変数

da	処理対象の位置情報を入れる
i	LIST の i 番目の行
j	位置情報の j 番目
tmp	da から各軸角度、グリッパー位置を取得するときに使用
LIST	動作指示データリスト
data	1 番目: Motor1 の角度、2 番目: Motor2 の角度、3 番目: Motor3 の角度、4 番目: グリッパー位置 を入れる

## ■スクリプトエリアのキー割付ブロック

スペース キーが押されたとき

i を 1 にする

i > LIST の長さ まで繰り返す

da を i 番目 ( LIST ) にする

データ取得

各 Motor を 1 番目 ( data ) , 2 番目 ( data ) , 3 番目 ( data ) 度にする

3 秒待つ

グリッパー位置を 4 番目 ( data ) にする

2 秒待つ

i を 1 ずつ変える

スペースキーが押されたとき  
i を 1 に  
i が LIST の行数を超えるまで繰り返す  
da を LIST から i 行目のデータに  
「データ取得」で取得した  
各 Motor 角度を設定  
「データ取得」で取得したグリッパー位置を設定  
次の LIST 情報を処理できるように i に 1 加算する

「データ取得」  
を呼び出す

定義 データ取得

tmp を にする

j を 0 にする

すべて 番目を data から削除する

j > da の長さ まで繰り返す

もし j 番目 ( da ) の文字 = なら

tmp を data に追加する

tmp を にする

でなければ

tmp を tmp と j 番目 ( da ) の文字 にする

j を 1 ずつ変える

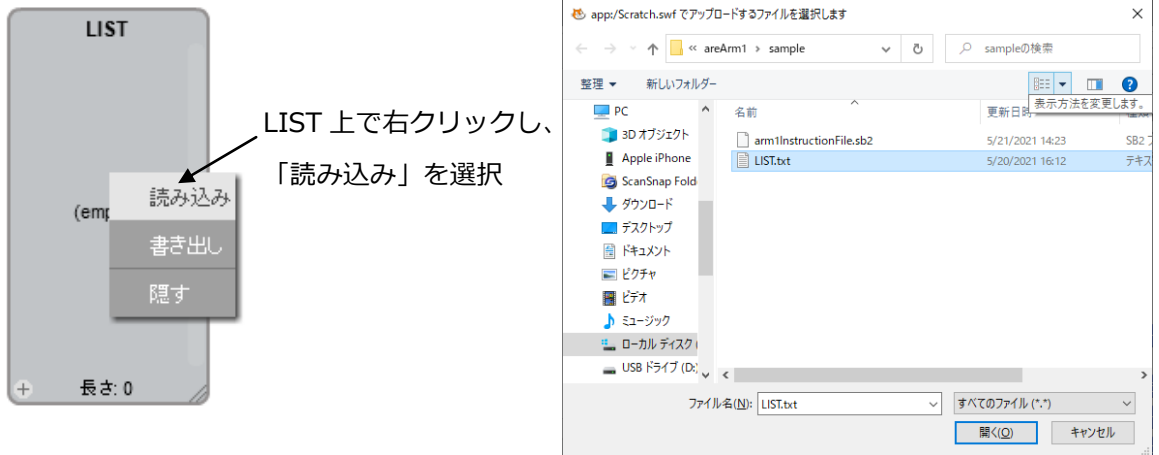
tmp を tmp と j 番目 ( da ) の文字 にする

tmp を data に追加する

tmp をスペースに  
j を 0 に  
data を全部削除  
j が da の文字数を超えるまで繰り返す  
j 番目の da の文字がスペースなら  
tmp を data に追加  
tmp をスペースに  
j 番目の da の文字がスペース以外なら  
Tmp に j 番目の da の文字を追加 (各モーターの角度)  
次の da の文字を処理できるように j に 1 加算する  
Tmp に j 番目の da の文字を追加 (グリッパー位置)  
tmp を data に追加

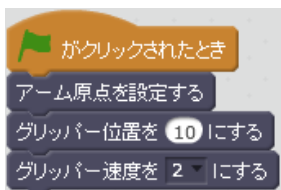
## ■操作方法

- ① LIST 上で右クリックし、「読み込み」を選択します。



「開く」を押して動作指示ファイルを読み込みます。

- ②アームの原点を設定して実行します。



- ③スペースキーを押すと、動作指示ファイルのデータのとおりアームを移動します。

以下サイトの動画も参考にしてください。

<https://robot.ability-evolves.com/working-on-electronics/arm1/programming-Scratch2/>

## 2.応用編

- 1.動作指示ファイル作成
- 2.動作指示ファイル実行

## §2.特記事項

当マニュアルで説明するプログラミングでは、モーターや電源などの電子部品を使用するため、操作によっては危険が伴う場合があります。

特にお子様が操作される場合は、大人のサポートのもとで安全等に配慮の上、操作をおこなってください。

また、モーター類は連続して稼働すると大変熱くなります。

モーターを停止する時間を設けるなどして発熱を抑制してください。

モーターにカバーなどして対策していますが、火傷に十分ご注意ください。

当マニュアルで説明する操作をおこなうことによって、万が一事故など起きても、Ability では一切の責任を負いません。予めご了承ください。

当マニュアルの内容について著作権法の定める範囲を超えて、Ability に無断で複写、複製、転載することはご遠慮ください。

当マニュアルに記載しているホームページアドレス、製品の仕様等は予告なく変更されることがあります。

当マニュアルの内容について運用した結果の影響については Ability では責任を負いかねます。

予めご了承ください。

Ability ホームページ

<https://robot.ability-evolves.com>