ポコロボスタジオ 操作ガイド

バージョン: v1.1
製造元: 株式会社ロボットスポーツゲームズ

目次

クイックスタート：最初にやる3つ

本体のセットアップが完了していることを確認してから始めてください。（まだの場合は「ポコロボ 本体セットアップ・安全ガイド」を先に読んでください）

ステップ1：スタジオが開いているか確認する

1. Chrome のアドレス欄に `http://192.168.20.1` と入力してスタジオを開きます
2. 画面右側の操作パネルに **「接続済み」** と表示されていれば準備完了です
   - 「再接続中...」の場合 → ケーブルと電源を確認してください（「ポコロボ 本体セットアップ・安全ガイド」§8 参照）

ステップ2：LEDを光らせてみる（約3分）

1. 左のブロック置き場から「**ずっと繰り返す**」ブロック（緑）をドラッグしてワークスペースに置きます
2. その中に「**LED**」ブロック（青紫）を入れます
3. LED ブロックの赤・緑・青の数字をクリックして好きな値（0〜100）に変更します
4. 操作パネルの **「▶ 実行」** をクリックします
5. ポコロボのLEDが光れば成功です

> 「▶ 実行」を押しても光らない場合 → §6「よくある質問」を参照してください

ステップ3：次にやること

- モーターを動かす → 付録C「例題② モーターを動かそう」
- ゲームパッドで操縦する → 付録C「例題③ ゲームパッドで操縦しよう」
- ブロックの使い方を調べる → 付録B「ブロック一覧早見表」

---

1. このガイドについて

このガイドで分かること

このガイドは、**ポコロボスタジオ**（Webブラウザで動くプログラミング画面）の操作方法を説明します。

次のことが分かります：

- スタジオをブラウザで開く手順
- 画面の各部の名前と役割
- ブロックを置いてプログラムを組み立てる基本操作
- 各ブロックの意味・使い方・入力できる値の範囲
- LEDを光らせる・モーターを動かす・ゲームパッドで操縦するなどの例題

前提：本体の準備ができていること

このガイドを使う前に、以下の準備が完了していることを確認してください。

- ☐ ロジック用電源（単4電池×2本）がセットされている
- ☐ 必要であればパワー用電源も接続されている（モーター・サーボを使う場合）
- ☐ モーターやサーボなど使う周辺機器が接続されている

上記の準備方法は **「ポコロボ 本体セットアップ・安全ガイド」** をご覧ください。

> 分からない単語が出てきたら、巻末の「7. 用語集」を参照してください。

このガイドの読み方

- **初めて使う方**：「2. スタジオを開く」から順番に読んでください
- **特定の操作を知りたい方**：「5. 各ブロックの説明」または「6. よくある質問」を直接参照してください
- **例題をやってみたい方**：「付録C: 作ってみよう」から始めてください

---

2. スタジオを開く

必要なもの

- ポコロボ本体（電源ON済み）
- USB Type-Cケーブル（通信対応のもの。充電専用ケーブルは使えません）
- パソコン（**Chrome 推奨**）

USB接続でスタジオを開く

1. ポコロボの電源スイッチを **ON** にします
2. **通信対応の** USB Type-Cケーブルでポコロボとパソコンをつなぎます（充電専用ケーブルでは接続できません）
3. パソコンでChromeブラウザを開きます
4. アドレス欄に `http://192.168.20.1` と入力して **Enterキー** を押します
5. スタジオの画面が表示されたら完了です

> 初めてつないだとき、WindowsパソコンはUSBネットワークドライバーを自動でインストールします。少し待ってからアクセスしてください。

スタジオが開かない場合の確認事項

本体や接続まわりのトラブルは **「ポコロボ 本体セットアップ・安全ガイド」** の「困ったときは」を参照してください。

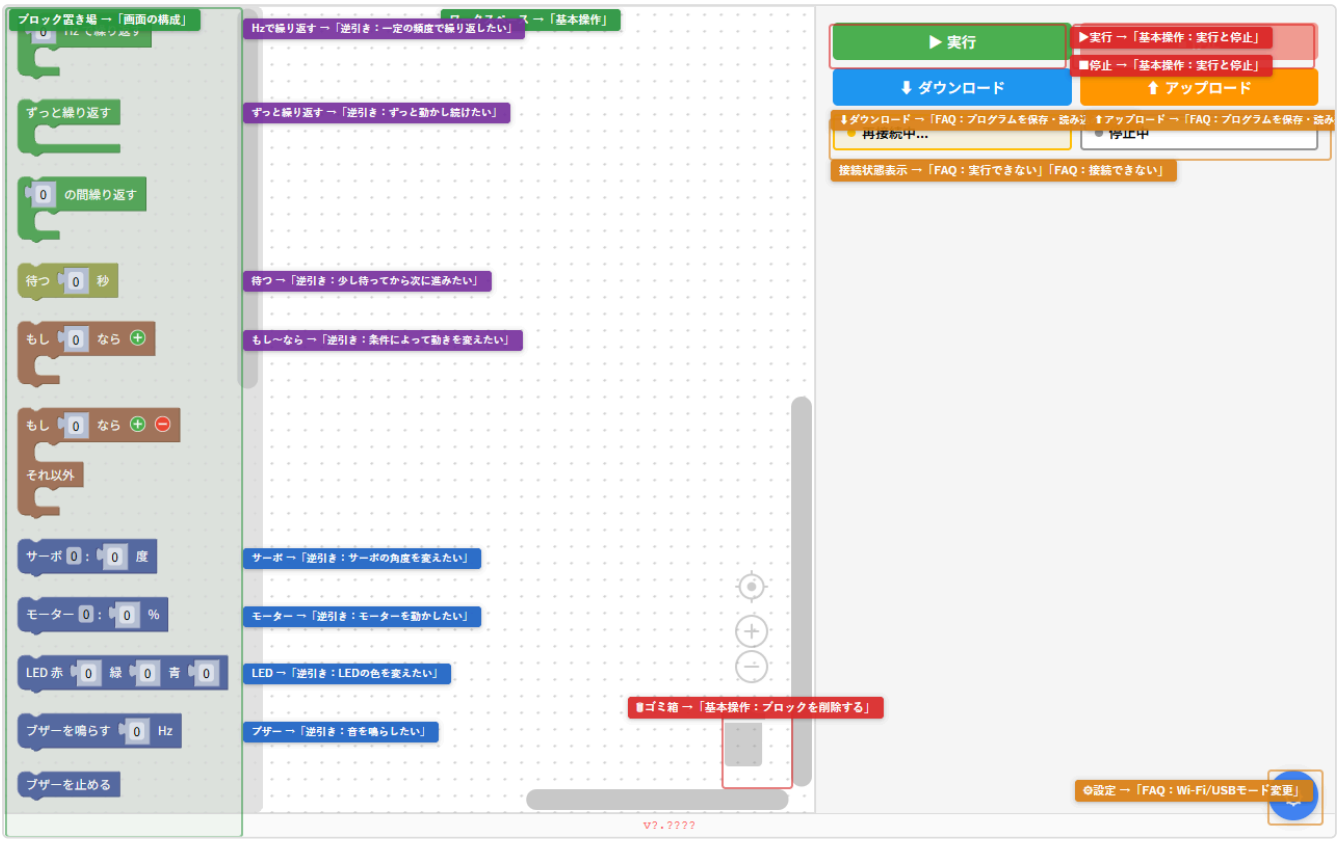
主な原因として次が挙げられます：

- ポコロボの電源が入っていない
- USBケーブルがしっかり刺さっていない
- アドレスが間違っている（`https://` ではなく `http://` で始まることを確認）
- Chrome 以外のブラウザを使っている（Chrome 推奨。他のブラウザは動作確認していません）

---

3. 画面の見方

スタジオを開くと、次のような画面が表示されます。



画面は大きく5つのエリアに分かれています。

番号	エリア名	説明箇所
❶	ブロック置き場	→ 3.2節
❷	ワークスペース	→ 3.3節
❸	操作パネル	→ 3.4節
❹	ゴミ箱	→ 3.5節
❺	設定ボタン	→ 3.6節

---

❶ ブロック置き場（左側）

使えるブロックがすべてここに並んでいます。ここからドラッグしてワークスペースに持っていきます。上下にスクロールすると全ブロックを見ることができます。

ブロックは色でカテゴリ分けされています。

色	カテゴリ	主な内容
緑・茶色	制御	繰り返し・条件分岐・待機
青紫	ロボット	モーター・サーボ・LED・ブザー

色	カテゴリ	主な内容
紫	計算・論理	四則演算・比較・AND/OR
水色	入力	ゲームパッドのボタン・スティック
ピンク	変数	値を名前付きで記憶する

❷ ワークスペース（中央）

ブロックを配置してプログラムを組み立てる場所です。ブロックを上から下へつなげた順番に実行されます。

マウスホイールで拡大縮小できます。ワークスペースの空いている部分をドラッグすると表示位置を移動できます。

❸ 操作パネル（右側）

接続状態の表示：操作パネルにポコロボとの接続状態が表示されます。**実行前に必ず確認してください。**

表示	意味	対処
接続済み	ポコロボと通信できています	実行できます
再接続中...	ポコロボと通信できていません	ケーブルと電源を確認（SETUP_GUIDE §8 参照）

ボタン	機能
▶ 実行	プログラムをポコロボに送って実行する
■ 停止	実行中のプログラムを停止する

> 実行中はブロックの編集ができなくなります。編集したい場合は「■ 停止」を押してください。

ボタン	機能
⬇ ダウンロード	プログラムを JSON（データ保存用のファイル形式）ファイルとして保存する
⬆ アップロード	保存したJSONファイルからプログラムを読み込む

❹ ゴミ箱

ブロックをここにドラッグすると削除できます。

❺ 設定ボタン

画面右上の歯車マークのボタンです。USBモード設定・その他のシステム設定を変更できます。

4. 基本操作

ブロックを置く・繋げる

1. ❶ブロック置き場から使いたいブロックを **ドラッグ** して、❷ワークスペースに持っていきます
2. ブロックを離すと配置されます
3. ブロックの **凸（出っ張り）と凹（くぼみ）** を合わせると接続されます

4. 上から下へつなげた順番に実行されます

ブロックを削除する

方法は2つあります：

- ブロックを **❹ゴミ箱までドラッグ** して離す
- ブロックを **右クリック** →「ブロックを削除」を選択する

繋がっているブロックをまとめて削除したい場合は、一番上のブロックをゴミ箱へドラッグしてください。繋がった下のブロックも一緒に削除されます。

数値を変更する（シャドウブロック）

ブロックの中にある **灰色の数字** をクリックすると、直接数値を入力して変更できます。

この灰色の数字ブロックを「　　シャドウブロック　　」と呼びます。シャドウブロックは削除できませんが、計算ブロックや変数ブロックを上から重ねるとその値に置き換えることができます。重ねたブロックを外すと、元の灰色ブロックに戻ります。

初期値が入った編集可能な灰色ブロック

ブロックをコピーする

ブロックを **右クリック** →「ブロックをコピー」を選択すると、同じブロックが複製されます。繋がっているブロックごとコピーされます。

ワークスペースを移動・拡大縮小する

- **移動**：ワークスペースの空いている部分をドラッグする
- **拡大縮小**：マウスホイールを回す
- **全体を表示**：右下のズームアウトボタンを繰り返し押す

プログラムを実行する

1. ❸操作パネルの 「**▶ 実行**」 ボタンをクリックします
2. プログラムがポコロボに送信され、実行が始まります
3. 実行中はブロックの編集ができなくなります

プログラムを停止する

操作パネルの 「**■ 停止**」 ボタンをクリックします。実行が止まり、ブロックの編集ができるようになります。

プログラムを保存する

操作パネルの 「**⬇ ダウンロード**」 ボタンをクリックすると、プログラムが **JSON ファイル**（データ保存用のファイル形式）としてパソコンにダウンロードされます。保存先は通常、ブラウザの「ダウンロード」フォルダです（ブラウザの設定で変更可能）。

プログラムを読み込む

操作パネルの 「**⬆ アップロード**」 ボタンをクリックし、保存済みのJSONファイルを選択します。ワークスペースのプログラムが読み込んだ内容で置き換わります。

> **注意**：読み込むと現在のプログラムは上書きされます。必要なプログラムは先に保存しておいてください。

---

5. 各ブロックの説明

このセクションでは、ポコロボスタジオで使えるすべてのブロックをカテゴリごとに説明します。

このセクションの見方

各ブロックの説明は次の形式で書かれています：

1. **ブロックの画像**（実際の見た目）
2. **機能の説明**
3. **入力できる値の範囲**
4. **使い方の例または注意点**

制御ブロック（緑・茶色）

ずっと繰り返す

中に入れたブロックを **無限に繰り返し** 実行します。

ブロックは上から一度だけ実行されて終わります。モーター制御のような **継続的に動かし続けたい処理** は、必ずこのブロックの中に入れる必要があります。囲まないとプログラムが一瞬で終了してしまいます。

ロボット制御では、基本的にすべての動作をこのブロックで囲む使い方が標準的です。

> **よくある間違い**：「ずっと繰り返す」の**外**にモーターや LED ブロックを置くと、プログラムが一瞬で終了してしまいます。継続的に動かし続けたい処理は必ずこのブロックの**中**に入れてください。

> **複数置いた場合**：ワークスペースに「ずっと繰り返す」ブロックを2つ以上置くと、それぞれが交互に実行されます（並列処理しているように見えますが、内部では1つずつ順番に処理しています）。慣れないうちは1つにまとめることを推奨します。

～Hzで繰り返す

ヘルツ：1秒間の繰り返し回数

1秒間に指定した回数だけ繰り返します。　　Hz　　（ヘルツ）で頻度を指定します。

値	意味
1 Hz	1秒に1回
50 Hz	1秒に50回（モーター制御に適した頻度）
100 Hz	1秒に100回

「ずっと繰り返す」より細かくタイミングを制御したい場合に使います。

～の間繰り返す

条件の部分の値が **0以外の間**、中のブロックを繰り返します。値が **0になると停止**します。

「もし〜なら」と組み合わせて使うことで、特定の条件が続く間だけ動作させることができます。

もし〜なら

条件の部分にはめたブロックの値によって動作が変わります。

- 値が **0** → 中を **実行しない**（条件が満たされていない）
- 値が **0以外**（1でも、50でも、-3でも） → 中を **実行する**（条件が満たされている）

ほとんどの場合、ボタンや比較ブロックの結果（0か1）をはめるので、「**1なら動く、0なら動かない**」 と考えればOKです。

ブロック右上の ⊕ **ボタン** をクリックすると「でなければ」の枠が追加されます。⊖ **ボタン** で削除できます。

もし（複数条件）

3つ以上の条件で処理を分けたいときに使います。⊕ **ボタン** で条件を追加、⊖ **ボタン** で削除できます。

「それ以外」の枠は、どの条件にも該当しなかったときに実行されます。

〜秒待つ

指定した秒数だけ処理を止めて待ちます。**0以上の値**を指定できます（小数も可）。負の値を指定するとエラーになります。

入力値	待ち時間
1	1秒
0.5	0.5秒（500ミリ秒）
0.1	0.1秒（100ミリ秒）

モーターをある時間だけ動かして止めたいときなどに使います。

> **よくある間違い**：「ずっと繰り返す」の中に「〜秒待つ」を入れると、待っている間もループが止まります。ゲームパッド操作のようにリアルタイムで反応させたい処理と同じループに「〜秒待つ」を入れると、意図した動作にならない場合があります。

ロボットブロック（青紫）

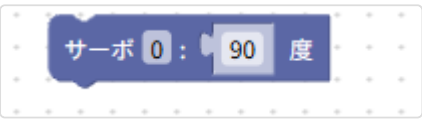
モーター

指定した番号のモーターを、指定した速度で動かします。

入力	範囲	説明
番号	0〜3	本体のコネクタ（M0〜M3）に対応（4以上は未接続）
速度	-100〜100（%）	正の値で正転、負の値で逆転、0で停止

> モーターを動かすには**パワー用電源（VHコネクタ）** が必要です。ロジック用電源だけでは動きません。

サーボ



指定した番号のサーボを、指定した角度に動かします。

入力	範囲	説明
番号	0〜3	本体のコネクタ（S0〜S3）に対応（4以上は未接続）
角度	0〜180（度）	0°＝左端、90°＝中央、180°＝右端

LED

本体内蔵のフルカラーLEDの色を設定します。赤・緑・青の3色を組み合わせて自由な色を作れます。

入力	範囲	説明
赤（R）	0〜100	0で消灯、100で最大輝度
緑（G）	0〜100	0で消灯、100で最大輝度
青（B）	0〜100	0で消灯、100で最大輝度

色の組み合わせ例：

色	赤	緑	青
赤	100	0	0
緑	0	100	0
黄	100	100	0

色	赤	緑	青
白	100	100	100
消灯	0	0	0

ブザーON

ヘルツ：1秒間の振動回数

指定した周波数で音を鳴らします。　　Hz　　（ヘルツ）で音の高さを指定します。

周波数	音程の目安
262 Hz	ド（中央C）
330 Hz	ミ
392 Hz	ソ
440 Hz	ラ（A音）

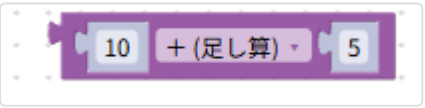
> 音を止めるには「ブザーOFF」ブロックを使います。

ブザーOFF

鳴っているブザーを止めます。「ブザーON」で鳴らした後、このブロックで停止させます。

計算・論理ブロック（紫）

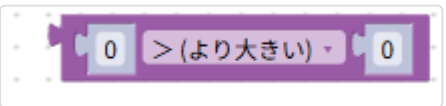
計算（＋ − × ÷）



2つの値を計算します。＋（**足し算**）、−（**引き算**）、×（**掛け算**）、÷（**割り算**） から選択できます。

他のブロックの数値入力にはめ込んで使います。例えばスティックの値（-1.0〜1.0）を100倍してモーターの速度（-100〜100）に変換するときに使います。

比較（＝ ≠ ＜ ≤ ＞ ≥）



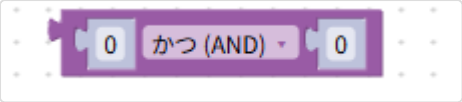
2つの値を比較します。条件が **成立すると 1**、**不成立のとき 0** を返します。

記号	意味
＝	等しい
≠	等しくない

記号	意味
＜	左が小さい
≤	左が小さいか等しい
＞	左が大きい
≥	左が大きいか等しい

「もし〜なら」ブロックの条件部分にはめ込んで使います。

論理演算（AND / OR）



2つの条件を組み合わせます。

演算	意味	結果が1になる条件
AND（かつ）	両方が真のとき	左も右も 0 以外
OR（または）	どちらかが真のとき	左か右の少なくとも一方が 0 以外

「ボタンAを押している **かつ** スティックが前に倒れている」のような複合条件を作るときに使います。

NOT（論理否定）

値の真偽を反転します。

入力	出力
0（偽）	1（真）
0以外（真）	0（偽）

「ボタンを押して **いない** とき」のような否定条件を作るときに使います。

入力ブロック（水色）

入力ブロックはゲームパッドの操作を読み取るブロックです。

ゲームパッドの2つの接続方法

接続先	用途	備考
ポコロボの USB Type-A（大きい方）	ロボットが直接ゲームパッドを読む	USB Type-C（PC接続）中は自動的に無効になる
パソコンの USB ポートに直接接続	パソコン経由でスタジオを通じて操作する	USB Type-C接続中でも使用可能

> **パソコンに直接つないだ場合**：ゲームパッドをパソコンに接続すると、Chrome がゲームパッドの入力を読み取り、スタジオを通じてポコロボに送ります。ポコロボの Type-A ポートにつながなくても動作します。

入力ブロックには2種類あります：

- **〜を押してたら**（茶色の枠付き）：押している間だけ中のブロックを実行する

- **〜の値**（丸い形）：押している間は 1、離していると 0 を返す。「もし〜なら」や計算ブロックにはめ込んで使う

ボタンを押してたら（A/B/X/Y）

A・B・X・Y のいずれかのボタンを押している間、中のブロックを実行します。

> ブロック右上の ⊕ **ボタン** をクリックすると「でなければ」の枠が追加されます。ボタンを押していないときの動作を指定できます。⊖ **ボタン** で削除できます。

十字キーを押してたら

十字キー（↑ ↓←→）のいずれかを押している間、中のブロックを実行します。

LR1/LR2ボタンを押してたら

LB・RB・LT・RT のいずれかを押している間、中のブロックを実行します。

ボタンの値（A/B/X/Y）

ボタンを押している間は **1**、離していると **0** を返します。「もし〜なら」や比較ブロックにはめ込んで使います。

十字キーの値

十字キー（↑ ↓←→）の押下状態を 0 または 1 で返します。押している間は **1**、離していると **0** です。

スティックの軸の値

スティックの傾き具合を **-1.0〜1.0** の数値で返します。左X・左Y・右X・右Y から選択できます。

方向	値の目安
前（奥）に倒す（左Y / 右Y）	マイナス（-1.0に近づく）
後ろ（手前）に引く（左Y / 右Y）	プラス（1.0に近づく）
左に倒す（左X / 右X）	マイナス（-1.0に近づく）
右に倒す（左X / 右X）	プラス（1.0に近づく）
中央（何も触らない）	0

> **注意**：スティックを**前に倒すと値がマイナス（-1.0方向）**になります。「前倒し → モーター前進」にしたい場合は、スティックの値にマイナスを掛けてください（計算ブロックで × -100 など）。これはゲームパッドの標準仕様によるものです。

変数（ピンク）

値に名前を付けて記憶する仕組み

変数は、**値を入れたり書き換えたりできる名前付きの箱**です。プログラムの実行中に中身が変化します。

なぜ変数を使うのか？ — 同じ値を複数の場所で使いたいときに役立ちます。例えばスティックの値を2つのモーターに使いたい場合、変数に一度記憶させて使い回せます。変数を使わずに同じ計算を2回書くと、ブロックが増えて読みにくくなるうえ、片方だけ修正し忘れるミスが起きやすくなります。

変数を作る

ブロック置き場の一番下にある「**変数を作成**」ボタンをクリックします。名前を入力すると変数が作成され、使えるブロックが追加されます。

> 現在、作成した変数の名前変更と削除はできません。

変数を設定する

変数に値を記憶させます。右側の入力に、記憶させたい値または計算ブロックをはめ込みます。

変数を読み出す

記憶されている値を取り出します。モーターブロックやLEDブロックの数値入力にはめ込んで使います。

---

6. よくある質問

ブロックが見つからない

ブロック置き場は上下にスクロールできます。ブロックの色（カテゴリ）を覚えておくと見つけやすいです。→「3.2節」のカテゴリ表を参照してください。

ブロックを削除したい

❹ゴミ箱にドラッグ、または右クリック →「ブロックを削除」で削除できます。

プログラムを保存・読み込みたい

- **保存**：「⬇ ダウンロード」ボタンをクリックするとJSONファイルがダウンロードされます
- **読み込み**：「⬆ アップロード」ボタンをクリックして保存済みのJSONファイルを選択します

実行できない・エラーが出る

接続状態の表示が「再接続中...」の場合、ポコロボと通信できていません。

1. ポコロボの電源が入っているか確認します
2. USB Type-Cケーブルがポコロボとパソコンにしっかり刺さっているか確認します
3. ポコロボを再起動してから（電源OFF→ON）、再度アクセスします

本体・接続まわりのトラブルは「ポコロボ 本体セットアップ・安全ガイド」の「困ったときは」を参照してください。

モーターが動かない

ポコロボには2つの電源が必要です。ロジック用電源（単4×2本）だけでは画面やLEDは動きますが、モーターには電気が足りません。**パワー用電源**（VHコネクタに接続する外部電源）が必要です。

詳しくは「ポコロボ 本体セットアップ・安全ガイド」の「困ったときは」内にある「モーターが動かない」を参照してください。

サーボが動かない

本体の接続に関するトラブルは「ポコロボ 本体セットアップ・安全ガイド」§8 を参照してください。

スタジオ側で確認する項目：

1. 「サーボ」ブロックの番号が、実際に接続したコネクタ（S0〜S3）と一致しているか確認します
2. 角度の値が **0〜180** の範囲内か確認します（範囲外は無効になる場合があります）
3. 操作パネルに「接続済み」と表示されているか確認します

ゲームパッドが反応しない

接続方法によって確認する内容が異なります。

パターン①：ゲームパッドをポコロボに直接つないでいる場合

1. ゲームパッド（またはUSBドングル）がポコロボの**大きい方のUSBポート（Type-A）** に接続されているか確認します
2. 動作確認済み機種を使用しているか確認します（エレコム JC-U3912TBK、Flydigi DUNE FOX）
3. スタジオの設定で「USBモード」が **auto** または **host** になっているか確認します
4. USB Type-C にパソコンをつないでいる場合は、Type-A が自動的に無効になります → パターン②へ

パターン②：ゲームパッドをパソコンに直接つないでいる場合

1. ゲームパッドがパソコンに正しく接続されているか確認します
2. Chrome のアドレス欄で `chrome://gamepad-internals` を開き、ゲームパッドが認識されているか確認します
3. 認識されていない場合は、ゲームパッドを抜き差しして再接続してください
4. それでも認識されない場合は、動作確認済み機種をお使いください

実行中にブロックを編集できない

仕様です。実行中はブロックの編集がロックされます。「■ 停止」ボタンを押すと編集できるようになります。

USBモードを変更したい

❺設定ボタン → システム設定 → USBモード選択。

モード	動作
auto（推奨）	自動切替（通常はこのモード）
host	常にゲームパッドモード
device	常にPC接続（NCM）モード

変数を削除したい・名前を変えたい

現在、変数の削除と名前変更はできません。不要な変数がある場合は、プログラムを作り直すか、実害がなければそのままにしてください。

---

7. 用語集

用語	意味
シャドウブロック	ブロック内に最初から入っている灰色の数値ブロック。クリックして直接編集でき、削除はできない
Hz（ヘルツ）	1秒間の繰り返し回数。50Hzなら1秒に50回
JSON	プログラムの保存に使われるデータファイル形式。拡張子は `.json`
ブラウザ経由のゲームパッド接続	ゲームパッドをパソコンに直接接続すると、Chrome がゲームパッドの入力を読み取り、スタジオを通じてポコロボに送る仕組み
条件分岐	条件に応じて異なる処理を実行すること。「もし〜なら」ブロックで実現する
変数	値に名前を付けて記憶する仕組み。実行中に中身を書き換えられる
正転・逆転	モーターが正方向・逆方向に回ること。モーターブロックの速度の正負で切り替わる
サーボ	指定した角度まで回転して保持するモーター。0〜180度で制御する

付録A: ゲームパッドボタン対応表

名前	番号	一般的なボタン名
A	0	A / ○
B	1	B / ×
X	2	X / △
Y	3	Y / □
LB / RB	4 / 5	L1 / R1
LT / RT	6 / 7	L2 / R2
Select / Start	8 / 9	Back / Menu
L3 / R3	10 / 11	スティック押込
↑ ↓ ← →	12〜15	十字キー
Home	16	ホームボタン

軸	番号	範囲
左X / 左Y	0 / 1	-1.0〜1.0
右X / 右Y	2 / 3	-1.0〜1.0

※ゲームパッドの機種によってボタン配置が異なる場合があります。

付録B: ブロック一覧早見表

「〜したい」から使うブロックを探せます。

やりたいこと	使うブロック	カテゴリ（色）
LEDの色を変えたい	LED	ロボット（青紫）
LEDを消したい	LED（赤0・緑0・青0）	ロボット（青紫）
モーターを動かしたい・止めたい	モーター	ロボット（青紫）
サーボの角度を変えたい	サーボ	ロボット（青紫）
音を鳴らしたい	ブザーON	ロボット（青紫）
音を止めたい	ブザーOFF	ロボット（青紫）
ずっと動かし続けたい	ずっと繰り返す	制御（緑・茶色）
一定の頻度で繰り返したい	〜Hzで繰り返す	制御（緑・茶色）
少し待ってから次に進みたい	〜秒待つ	制御（緑・茶色）
条件によって動きを変えたい	もし〜なら	制御（緑・茶色）
条件を3つ以上に分けたい	もし（複数条件）	制御（緑・茶色）
ゲームパッドのボタンで動かしたい	ボタンを押してたら	入力（水色）
十字キーで動かしたい	十字キーを押してたら	入力（水色）
スティックの傾きを取得したい	スティックの軸の値	入力（水色）
計算結果を使いたい	計算（＋ − × ÷）	計算（紫）
大小を比較したい	比較	計算（紫）
「かつ」「または」を使いたい	論理演算（AND/OR）	計算（紫）
値を記憶しておきたい	変数を設定 / 変数	変数（ピンク）

付録C: 作ってみよう（例題）

ここからは実際にプログラムを組み立てる例題です。各ブロックの仕様は「5. 各ブロックの説明」、困ったときは「6. よくある質問」を参照してください。

例題① LEDを光らせよう

> 所要時間：約10分 ／ 使うブロック：LED

> **この例題を始める前に**
> - 前提：スタジオが開いていること、操作パネルに「接続済み」と表示されていること
> - 成功条件：実行ボタンを押すとポコロボのLEDが指定した色で光る
> - つまずきやすい点：「ずっと繰り返す」の中にLEDブロックを入れていない → 一瞬光って消える

ブロック置き場の青紫のブロックから「LED」をドラッグして配置し、赤の値を100に変更します。

「▶ 実行」を押すとLEDが赤く光ります。色の組み合わせを変えていろいろな色を試してみましょう。

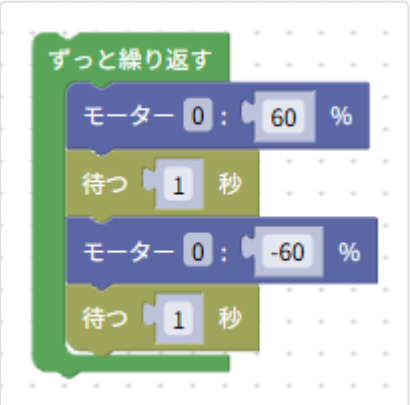
色	赤	緑	青
赤	100	0	0
緑	0	100	0
黄	100	100	0
白	100	100	100

例題② モーターを動かそう

> 所要時間：約15分 ／ 使うブロック：ずっと繰り返す、モーター、待つ

> **この例題を始める前に**
> - 前提：パワー用電源が接続されていること、モーターが M0〜M3 のいずれかに接続されていること
> - 成功条件：実行ボタンを押すとモーターが回転する
> - つまずきやすい点：パワー用電源が未接続 → モーターが動かない（SETUP_GUIDE §4 参照）

なぜ「ずっと繰り返す」で囲むのか？ — ブロックは上から一度だけ実行されて終わります。モーターの制御など継続的に動かしたい処理は、必ず「ずっと繰り返す」の中に入れる必要があります。囲まないとプログラムが一瞬で終了します。



- モーターの速度は **-100〜100%**（負の値で逆転、0で停止）
- 「待つ」ブロックの単位は **秒**（1 = 1秒、0.5 = 0.5秒）

> 動かない場合：モーターのコネクタと **パワー用電源** を確認してください。ロジック用電源だけでは動きません。

例題③ ゲームパッドで操縦しよう

> 所要時間：約20分 ／ 使うブロック：ずっと繰り返す、十字キーを押してたら、モーター

> **この例題を始める前に**
> - 前提：ゲームパッドがポコロボの Type-A またはパソコンに接続されていること
> - 成功条件：ゲームパッドのボタンを押すとモーターが反応する
> - つまずきやすい点：スティックを前に倒すと値がマイナスになる → 前進させたい場合は × -100 で反転する

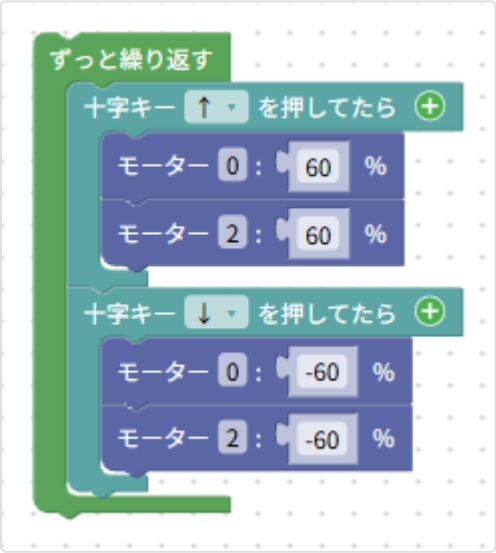
ゲームパッドをポコロボの大きい方のUSBポートに接続します（または、パソコンに直接接続する）。

> パソコンとUSB接続中はポコロボ側のUSBポートが使えません。その場合はゲームパッドをパソコンに直接接続してください（Chrome（ブラウザ）がゲームパッドを読み取り、スタジオを通じてポコロボに送ります）。

手順：

1. 緑色の「ずっと繰り返す」ブロックをワークスペースに配置する
2. 水色ブロックから「十字キーを押してたら」（↑を選択）を「ずっと繰り返す」の中にはめる
3. 青紫の「モーター」ブロックを2つ、「十字キー↑を押してたら」の中にはめる
4. 1つ目：モーター番号を **0**、速度を **60** に設定
5. 2つ目：モーター番号を **2**、速度を **60** に設定
6. 同様に「十字キー↓を押してたら」を追加し、モーター速度を **-60** にする

> **モーター番号について**：番号は本体のコネクタ（M0〜M3）に対応します。自分の配線に合わせて番号を変更してください。



応用：← → を追加して左右旋回、ボタンA/Bでサーボやブザーも試してみてください。

例題④ 条件分岐を使おう

> 所要時間：約20分 ／ 使うブロック：もし〜なら、ボタンの値、サーボ、LED

> **この例題を始める前に**

> - 前提：ゲームパッドが接続されていること、サーボが S0〜S3 のいずれかに接続されていること
> - 成功条件：ボタンAを押している間だけサーボが動き、離すと戻る
> - つまずきやすい点：「ずっと繰り返す」で囲んでいない → 一度だけ実行して終わる

「もし〜なら」ブロックは、条件の部分にはめたブロックの値によって動作が変わります。

- 値が **0** → 中を **実行しない**（条件が満たされていない）
- 値が **0以外**（1でも、50でも、-3でも） → 中を **実行する**（条件が満たされている）

ほとんどの場合、ボタンや比較ブロックの結果（0か1）をはめるので、**「1なら動く、0なら動かない」** と考えればOKです。

「ボタン A」の値ブロック 

は、押していると **1**、離していると **0** を返します。これを「もし〜なら」にはめると、ボタンを押している間だけ中が実行されます。

- ボタンA押下中 → サーボ90度 + LED緑
- 離している間 → サーボ0度 + LED赤

「もし〜なら」ブロックの右上にある ⊕ **ボタン** をクリックすると「でなければ」の枠が追加されます。⊖ **ボタン** で削除できます。3つ以上の分岐が必要な場合は、ブロック置き場にある「もし（複数条件）」ブロックを使ってください。

例題⑤ 変数を使おう

> 所要時間：約25分 ／ 使うブロック：変数を設定、変数、計算、スティック、モーター

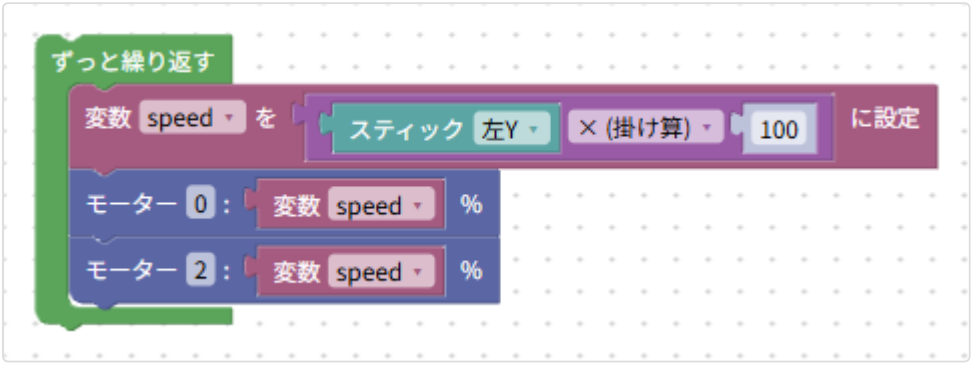
> **この例題を始める前に**
> - 前提：ゲームパッドが接続されていること、モーターが2台（M0・M2など）接続されていること
> - 成功条件：スティックの傾きに応じて2台のモーターが同じ速度で動く
> - つまずきやすい点：スティックを前に倒すと値がマイナスになるため、モーターが期待と逆方向に回る → × -100 で反転する

値に名前を付けて記憶する仕組み

変数 は、**値を入れたり書き換えたりできる名前付きの箱**です。

なぜ変数を使うのか？ — 計算結果を1箇所でしか使わないなら、計算ブロックを直接はめ込んでも構いません。しかし、**同じ値を複数の場所で使いたい場合**に変数が役立ちます。下の例では「スティック左Y × 100」の

結果をモーター0番と2番の両方に使いたいので、変数「speed」に一度記憶させて使い回しています。
ブロック置き場の一番下「変数を作成」ボタンから新しい変数を作れます。



スティックの値は **-1.0〜1.0** の範囲です。

- 前（奥）に倒す → **マイナス**（-1.0に近づく）
- 後ろ（手前）に引く → **プラス**（1.0に近づく）
- 中央（何も触らない） → **0**

これを100倍すると **-100〜100** になり、モーターの速度範囲にちょうど合います。

製造元: 株式会社ロボットスポーツゲームズ ／ **問い合わせ**: info@robotic-games.com