学習日：

Lesson 3 ピクさんアニメーション

ピクさんの動画を作ってみよう。

1 ピクさんアニメーションを作ろう

☑ アニメーションさせてみよう

今から示す①～③の 3 つのプログラムを実行してみましょう。
そして、その 3 つの違いを観察しましょう。

①
```
R LUA -140
```

②
```
R LUA -140 1
```
間はスペースキーで空白を入力

③
```
R LUA -140 1 2
```

得られる結果

左の①～③を実行すると、いずれもこのような姿勢になりますが、何かしらの違いがあります。Lesson3 ではその違いがとても重要なポイントです。

①は Lesson2 でやりましたね。左手を上げているピクさんになります。

②は 左手を上げているピクさんになりますが、少し様子が違います。
徐々に手が上がりました。これはアニメーションです。
1 秒間かけて左上腕を関節点を中心に反時計回りに -140 度、
つまり時計周りに 140 度回転するという意味です。

③は プログラムの実行は改行キーが押された瞬間から始まりますが、
この場合 しばらくは回転せず、改行キーを押して 2 秒後から
1 秒かけて反時計回りに 140 度回転しました。
実は命令 R は最大 4 つまで引数を持つことができます。

命令の様式	処理
R 引数 1 引数 2 引数 3 引数 4	引数 4 秒後に 引数 1 で指定される体の部位を反時計回りに 引数 2 度だけ 引数 3 秒かけて支点を中心に等速回転する。引数 4 が省略された時はに 0 が、引数 3 、引数 4 の両方が省略された時はいずれも 0 が入力されているものとして取り扱う。

- 引数 1：体の部位
- 引数 2：反時計回りに回す角度
- 引数 3：数値（秒数）かける（省略時は自動的に 0）
- 引数 4：数値（秒数）後に動く（省略時は自動的に 0）

☑ タイミングを考えたアニメーション

では次に以下の 3 行からなるプログラムを記述して実行します。

例 アニメーションの例 1

```
R LUA -140 1   // 左腕をあげる
R RUA  140 1   // 右腕をあげる
R RLL  25  1   // 右足をあげる
```

1 秒間かけて両腕と右脚を同時に上げるアニメーションが再生されます。では、両腕を上げてから右脚を上げるようにするにはどうすれば良いでしょう。そのために RW(Rotate Wait: 回転待ち) 命令というのがあります。

先ほどのプログラムの命令名 R のところを RW に変えて実行してみてください。つまり以下の通りです。

例 アニメーションの例 2

```
R  LUA -140 1     // 両手を同時にあげる
RW RUA  140 1
R RLL  25  1      // 足をあげる
```

RW 命令の仕様を下に示します。ポイントは「回転が終了するまで次の命令は実行されない。」という点です。

命令の様式	処理
RW 引数 1 引数 2 引数 3	引数 1 で指定される体の部位を反時計回りに 引数 2 度だけ 引数 3 秒かけて支点を中心に等速回転する。回転が終了するまで次の命令は実行されない。

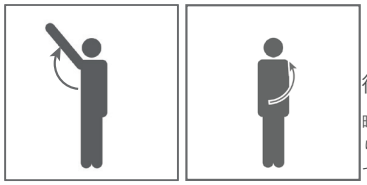
- 引数 1：体の部位
- 引数 2：反時計回りに回す角度
- 引数 3：数値 (秒数) かける（省略時は自動的に 0）

☑ 同じ部位のアニメーション命令が 同時に実行される場合の注意

R 命令は、「次の命令も同時に実行される。」されます。そのため、R 命令では両腕が同時に上がりました。
R 命令は、「次の命令も同時に実行される。」ということを注意しなければいけません。

例 手をあげる ...?

```
R  LUA -140 1
RW LUA  140 1
```



得られる結果

時計回りに腕を回転、反時計回り腕を回転する。同時に行うので打ち消していますね。

この命令を実行してもピクさんは何も反応しません。
これは、左上腕 (LUA) を 140 度 1 秒間で回転するのと
左上腕 (LUA) を -140 度 1 秒間で回転する命令が同時に
実行されるため、実際には両方が相殺されるからです。

例 手をあげる（3 つの命令を同時に）
40(度) 手をあげる命令を 3 つ実行します。

```
R RUA 40 1
R RUA 40 1
R RUA 40 1
```

この命令を実行するとどうなるでしょう？これは `R LUA 120 1` と同一になります。

R 命令と RW 命令をうまく組み合わせることでピクさんの様々なアニメーションが可能となります。

☑ プログラムを制御するボタン

3 つのボタンを改めて紹介します。

プログラム最初から実行

プログラムを最初のから実行します。

プログラム一旦停止

プログラムの実行を一旦停止します。
このボタンを押した瞬間の状態でピクさんは状態で停止します。

プログラム一旦停止箇所からスタート

一旦停止した状態から再び実行します。

2 繰返し

例 手をあげる

```
R LUA -140 1
```

と入力して改行キーを押すと左手があがります。

例 手を振る

手を振ってみましょう。友達と別れる時を想像してみてください。

```
RW LUA -140 1
RW LLA -60  0.3  // 1 往復分 手を振る
RW LLA  60  0.3
RW LLA -60  0.3  // 1 往復分 手を振る
RW LLA  60  0.3
RW LLA -60  0.3  // 1 往復分 手を振る
RW LLA  60  0.3
```

3 回左前腕 (LLA) を左右に振ることができました。では手を振ることを 10 回や 50 回にするにはどうしますか。同じ命令をたくさん記述することで実現できますが、プログラムがとても長くなってしまいますし、プログラムを見て何回振っているのかをすぐ確認することも難しくなります。そこで繰返しの処理を書く方法を学んでみましょう。「○○回やって」ということと同じです。使う命令は以下の 2 つです。

命令の様式	処理
REPEAT 引数 1	対応する *END* までの命令を 引数 1 回繰返す。
END	繰返しの終了。

```
REPEAT ●
 ▨
END
```

例 手を振る（繰返しを使って）

上の「手を振る」の例を、REPEAT を使って書き直します。

```
RW LUA -140 1
REPEAT 3       // 手を振るのを 3 回繰返す
RW LLA -60 0.3   // 1 往復分 手を振る
RW LLA 60 0.3
END
```

さらに、前回学んだ変数の定義と組み合わせると、手を振る角度や回数を
変数で設定できるようになり、より応用性の高いプログラムを書くことができます。

例 手を振る（変数を使って）

```
RW LUA -140 1
SET :ANGLE 60          // 曲げる前腕の角度
SET :NUM 3       // 手を振る回数
REPEAT :NUM
RW LLA -:ANGLE 0.3
RW LLA :ANGLE 0.3
END
```

ここまでに紹介した命令の一覧

命令の様式	処理
R 引数1 引数2 引数3 引数4	引数4 秒後に 引数1 で指定される体の部位を反時計回りに 引数2 度だけ 引数3 秒かけて支点を中心に等速回転する。 引数4 が省略された時はに 0 が、 引数3 、 引数4 の両方が省略された時はいずれも 0 が入力されているものとして取り扱う。
RW 引数1 引数2 引数3	引数1 で指定される体の部位を反時計回りに 引数2 度だけ 引数3 秒かけて支点を中心に等速回転する。回転が終了するまで次の命令は実行されない。
FR	ピクさんを正面向き (初期状態) にする。 Front の意味。
SD	ピクさんを側面向きにする。 Side の意味。
C	ピクさんの状態を直立状態 (初期状態) にする。Clear の意味。

命令の様式	処理
SET : 引数1 引数2	変数 引数1 に 引数2 を代入する。 引数1 の前には : をつける。
REPEAT 引数1	対応する *END* までの命令を 引数1 回繰返す。
END	繰返しの終了。
W 引数1	引数1 秒間、何もせずに待つ。待ちが終了するまで次の命令は実行されない。

MEMO

ピクさんからの挑戦状

3 回目のピクさんからの挑戦状です。ピクさんからの挑戦状は、ピクさんからの挑戦状は、あなたがやりたいと思うところからやってください。ピクチャレ 1 から順に取り組む必要はありません。また、全てのピクチャレに取り組む必要もありません。あなたがやりたいピクチャレに取り組んでください。

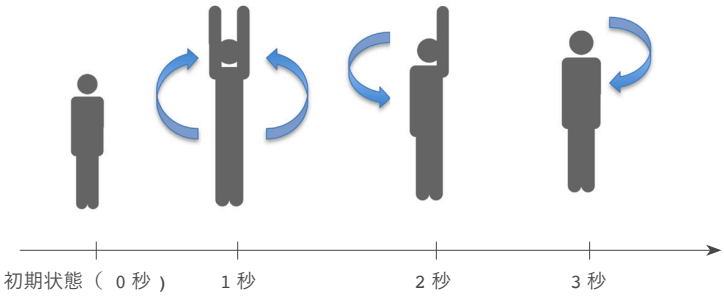
さあ、ピクさんからの挑戦状「ピクチャレ」に挑戦してみましょう。

ピクチャレ 1 ピクさんに体操をさせよう

次のようなピクさんアニメーションを作成してください

0 秒目から 1 秒かけて 両腕を 上げる。
1 秒目から 1 秒かけて 左腕を 下ろす。
2 秒目から 1 秒かけて 右腕を 下ろす。

参考画像



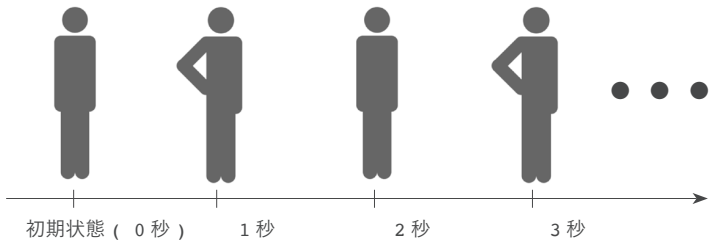
ピクチャレ 2 ピクさんに体操をさせよう 2

次のようなピクさんアニメーションを作成してください。

ただし REPEAT 命令を用いること。

1 秒かけて左手を曲げて、続けて 1 秒かけて再び左手を伸ばす
ということを 10 回 繰り返す。

参考画像



ピクチャレ 3 オリジナルピクさんを作ろう

繰返しを使った自由作品を作成してみましょう。

MEMO