

原　秀一　（株式会社ict4e）
http://res.ict4e.jp/progrun

CC BY-NC © 原 秀一

progrun API 説明書：β3版

この説明書では、ブラウザ上でお手軽プログラミング環境“progrun” でプログラミングを行う時の、機能要素の説明を行っています。

主に、指導する大人や、より詳しい事を知りたい基礎を理解した学生向けの内容です。

Let’s Programming for creating / enhancement “Wheels for the Mind”.

# 目次

[目次 1](#_Toc387483222)

[図形を描く 3](#_Toc387483223)

[色を指定する 4](#_Toc387483224)

[制御 4](#_Toc387483225)

[三角関数 5](#_Toc387483226)

[値を得る 5](#_Toc387483227)

[演算 6](#_Toc387483228)

[変数 7](#_Toc387483229)

[付録：真と偽の判定 7](#_Toc387483230)

[付録：stamp命令の2進数・16進数の対応 8](#_Toc387483231)

[付録：progrunの特徴 10](#_Toc387483232)



# 図形を描く

|  |
| --- |
| **rect ( x, y, scale\_x, scale\_y )** |
| 引数 | x | 四角を描く始点の x軸の値 |
| y | 四角を描く始点の y軸の値 |
| width | 始点から四角の幅（x軸方向の増分） |
| height | 始点から四角の高さ（y軸方向の増分） |
| 説明 | 四角を描いて、塗りつぶします。 |
| 関連 | 塗りつぶす色は、事前に color() で指定してください。 |
| 記述例 | rect(10,10,20,5)rect(30,10,-20,5) |

|  |
| --- |
| **circle ( x, y, radius )** |
| 引数 | x | 円を描く中心の x軸の値 |
| y | 円を描く中心の y軸の値 |
| radius | 円の中心から、半径　　　　[正の数] |
| 説明 | 円を描いて、塗りつぶします。 |
| 関連 | 塗りつぶす色は、事前に color() で指定してください。 |
| 記述例 | circle(25,25,10)circle(0,0,1) |

|  |
| --- |
| **line ( x1, y1, x2, y2 )** |
| 引数 | x1 | 直線の始点 x軸の値 |
| y1 | 直線の始点 y軸の値 |
| x2 | 直線の終点 x軸の値 |
| y2 | 直線の終点 y軸の値 |
| 説明 | 直線を、始点(x1, y2)から終点（x2, y2） に向かって引きます。 |
| 関連 | 塗りつぶす色は、事前に color() で指定してください。 |
| 記述例 | line( 10, 10, 20, 10 )line( 0, 0, 50, 50 ) |

|  |
| --- |
| **stamp ( s, x, y )** |
| 引数 | s | スタンプのパターンの16進数 (0～f) 文字列 |
| x | スタンプを配置する始点のx軸の値 |
| y | スタンプを配置する始点のy軸の値 |
| 説明 | スタンプ（ドット絵）を描きます。サイズは 8ドット × 8ドット （4ドット x 4ドット を 4つ組み合わせ）。 |
| 関連 | この命令の次に描く図形から、座標が変更されます。※付録の二進数表を参照※ |
| 記述例 | stamp("1248842184211248",10,10) stamp("505554c00075717",0, 0)stamp("109151300075744",8, 0) |

# 色を指定する

|  |
| --- |
| **color ( n )** |
| 引数 | n | 色番号　（ 0　～　15　） |
| 説明 | 図形を描く色を変更します。 |
| 関連 | この命令の次に描く図形から、指定した色を使います。0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15droppedImage.tiff |
| 記述例 | color( 0 )color( 15 ) |

|  |
| --- |
| **rgb ( r, g, b ) / rbga ( r, g, b, a )** |
| 引数 | r |  |
| g |  |
| b |  |
| a | αチャネル値で透明度を指定 |
| 説明 | 図形を描く色を変更します。 |
| 関連 | この命令の次に描く図形から、指定した色を使います。 |
| 記述例 |  |

|  |
| --- |
| **hsv ( h, s, v ) / hsva ( h, s, v, a )** |
| 引数 | h |  |
| s |  |
| v |  |
| a | αチャネル値で透明度を指定 |
| 説明 | 図形を描く色を変更します。 |
| 関連 | この命令の次に描く図形から、指定した色を使います。 |
| 記述例 |  |

# 制御

|  |
| --- |
| **origin ( x, y )** |
| 引数 | x | キャンパス座標の原点を動かす、x軸方向の増分 |
| y | キャンパス座標の原点を動かす、y軸方向の増分 |
| 説明 | キャンパス座標の原点を、指定した (x, y) 増分だけ、ずらします。 |
| 関連 | この命令の次に描く図形から、座標が変更されます。 |
| 記述例 | origin( 10, 10 )origin(-10, -10 ) |

# 三角関数

|  |
| --- |
| **sin ( n )** |
| 引数 | n | 数値 |
| 説明 | 正弦関数：サイン sin(n) を計算した値が返ります。 |
| 関連 |  |
| 記述例 | sin( 10 )sin( -2 ) |

|  |
| --- |
| **cos ( n )** |
| 引数 | n | 数値 |
| 説明 | 余弦関数：コサイン cos(n) を計算した値が返ります。 |
| 関連 |  |
| 記述例 | cos( 10 )cos( -2 ) |

|  |
| --- |
| **tan ( n )** |
| 引数 | n | 数値 |
| 説明 | 正接関数：タンジェント tan(n) を計算した値が返ります。 |
| 関連 |  |
| 記述例 | tan( 10 )tan( -2 ) |

# 値を得る

|  |
| --- |
| **rnd ( n )** |
| 引数 | n | 数値 |
| 説明 | ランダムな数字　（0以上 nより小さい） が返ります。 |
| 関連 |  |
| 記述例 | color( rnd( 10 ) )circle( 10, 10, rnd( 5 ) ) |

|  |
| --- |
| **abs ( n )** |
| 引数 | n | 数値 |
| 説明 | 絶対値　（正の値は、そのまま。負の値は、正の値に。） が返ります。 |
| 関連 |  |
| 記述例 | abs( -20 )abs( sin( x ) ) |

|  |
| --- |
| **条件式 ? 真の時の値 : 偽の時の値** |
| 引数 | 条件式 | 条件判定のための式 |
| 真の時 | 数値 |
| 偽の時 | 数値 |
| 説明 | 条件式の結果によって、真の時の値 か 偽の時の値 が返ります。別名を三項（さんこう）演算子と呼び、IF文の役割をします。 |
| 関連 | ※付録：真と偽の判定　を参照※ |
| 記述例 | one ? 10 : 0t == 10 ? 1 : 0 |

# 演算

|  |
| --- |
| **四則演算子 ＋α**※見易さの為、全角で記述していますが、利用時は半角で指定 |
| ＋ | 足す。加算。 |
| － | 引く。減算。 |
| ＊ | 掛ける。乗算。　（　×　と　同じ意味） |
| ／ | 割る。割算。　　　（　÷　と　同じ意味） |
| ％ | 余りを求める。　　例）　7 % 4 = 3 |
| （　） | 括弧。四則演算の順番を決める。 |

|  |
| --- |
| **比較演算子　＋α**※見易さの為、全角で記述していますが、利用時は半角で指定 |
| ＜ | 小なり。less than。 |
| ＜＝ | 小なりイコール。less than or equal。 |
| ＞ | 大なり。greater than。 |
| ＞＝ | 大なりイコール。greater than or equal。 |
| ＝＝ | 等しい。equal。 |
| ＆＆ | 論理積。and。　両辺が共に TRUE の時に TRUE。 |
| ｜｜ | 論理和。or。　両辺のいずれかが TRUE の時に TRUE。 |
| ！ | 否定。not。　続く値がTRUE では無いときに TRUE。 |

|  |
| --- |
| **代入演算子**※見易さの為、全角で記述していますが、利用時は半角で指定 |
| ＝ | 左側の変数に、右側の値などを代入します。プログラムでは = 記号は、等しいを意味せず、代入を意味します。 |

# 変数

|  |
| --- |
| **変数** |
| a～su～z | 変数。数字をいれておける、入れ物。 |

|  |
| --- |
| **特殊な変数** |
| t | 0から1ずつ自動で増え続ける特殊な変数。リセットボタンを押すと、0に戻る。 |
|  |  |
| one | 「１」ボタンを押すと 1 、離すと0になる特殊な変数。 |
| two | 「２」ボタンを押すと 1 、離すと 0 になる特殊な変数。 |
| up | 「↑」ボタンを押すと 1 、離すと 0 になる特殊な変数。 |
| down | 「↓」ボタンを押すと 1 、離すと 0 になる特殊な変数。 |
| left | 「←」ボタンを押すと 1 、離すと 0 になる特殊な変数。 |
| right | 「→」ボタンを押すと 1 、離すと 0 になる特殊な変数。 |
|  |  |
| cx | 左右ボタンに対応する特殊な変数。「←」ボタンを押すと -1, 「→」ボタンを押すと +1する |
| cy | 上下ボタンに対応する特殊な変数。「↑」ボタンを押すと -1, 「↓」ボタンを押すと +1する。 |

# 付録：真と偽の判定

プログラミング学習を進めていくと、真（True）と偽（False）の判定を行う場面が出てきます。ここでは少し簡略化して、その考え方を説明します。

1. one ? 10 : 0

「１」ボタンを押している間は one が 真となり10が、離している間はoneが偽となり0が値として返ります。「１」ボタンを押している間、キャラクターをジャンプさせるなどに活用できます。

1. t %10== 9 ? 1 : 0

特殊な変数「t」との演算 t%10 が9になった時に条件判定が真となり1が、それ以外の場合は0が値として返ります。tはどんどん大きな値になって行きますが、% 演算子により余りを取っているので、t%11は0～9の値を繰り返し続けるため、10回に1回だけ1の値が変えることになります。

# 付録：stamp命令の2進数・16進数の対応

stamp命令では、縦8ドット横8ドットの絵がかけます。絵は、次の図のように4つのパートに分けて縦4ドット横4ドットずつ指定を行います。まず「１」の4x4の指定を行い、次の「２」という順番です。「４」までなくても指定されたところまでが描画されます。

例えば、次の絵を描く場合。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | **１** |  |  |  |  | **３** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  | **２** |  |  |  | **４** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

まず次の４パーツに分解して、１パーツずつ、一行単位で白・黒を2進数で表し、16進数で指定します。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

１パーツ目：0124　　「1行目：0000 / 2行目：0001 / 3行目：0010 / 4行目0100」
2パーツ目：8997　　「1行目：1000 / 2行目：1001 / 3行目：1001 / 4行目0111」
3パーツ目：0842　　「1行目：0000 / 2行目：1000 / 3行目：0100 / 4行目0010」
4パーツ目：199E　　「1行目：0001 / 2行目：1001 / 3行目：1001 / 4行目1110」

上記を連続して指定すれば完成です。

stamp("012489970842199E",0, 0)

（参考） 16進数と2進数の対応表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2進数 | 16進数 |  | 2進数 | 16進数 |
| 0000 | 0 |  | 1000 | 8 |
| 0001 | 1 |  | 1001 | 9 |
| 0010 | 2 |  | 1010 | A |
| 0011 | 3 |  | 1011 | B |
| 0100 | 4 |  | 1100 | C |
| 0101 | 5 |  | 1101 | D |
| 0110 | 6 |  | 1110 | E |
| 0111 | 7 |  | 1111 | F |

# 付録：progrunの特徴

プログラム(program) が自動的に実行(run) される事から progrun というネーミングが生まれました。他のプログラム言語では、　「１．書く」　「２．実行」　というように順序を踏んで使うことが多いので、自動的に実行されるというのはprogrunのユニークな点です。

次の特徴を覚えておくことが、活用する上では大切です。

1. コードは自動でループ
2. エラーが発生しない

### コードは自動でループ

　コード（書いたプログラム）は、自動でループされて、永遠に実行され続けます。他の言語を知っている方は、書いたコードの外側に while(1) で囲まれているとイメージしてください。そのため、繰り返しについては得意ですが、初期値（一番最初に使うときの値）の設定などには工夫が必要になってきます。

 例）変数x の初期値を1 にしたい場合
 x = t==0?1:x　　　※三項演算子を用い、tが0の時（リセット押下時） x を 1にする

### エラーが発生しない

　　progrunは命令や引数などに間違いがあってもエラーは発生しません。そのため、適時「リセットボタン」を押下する、デバッグコード（動作確認の為のプログラム）をいれるなどして検証をしてください。

 例）変数z の値を確認したい場合
 circle(25 + z, 25, 1)　※半径1の円を座標中心に配置し、x軸のz分の増分で確認

### その他

　progrun は プログラム言語JavaScript と HTML/CSSによって記述されています。多くの機能は、JavaScriptのCanvas を利用したものです。動作環境として、次を想定しております。

* Safari (iPad)
* Safari (MacOS X)
* Firefox, Chrome (Windows , MacOS X)

　これらの環境での不具合や、その他の環境での動作情報などあれば、ご連絡ください。

http://sabae.club/progrun/

toiawase\_progrun@ict4e.jp

progrun開発：jig.jp 福野泰介 / ドキュメント著者：ict4e 原 秀一